RetailGenie

An LLM-Powered BI Assistant

Project Report

Project Report on
RetailGenie – An LLM-Powered BI Assistant
Course: Deep Learning (CMPE-258)
MS Software Engineering (Spring 2025)

Submitted by:

Team Genie Squad

Soumya Bharathi Vetukuri (016668964)

Shubham Kothiya (018217901)

Rutuja Patil (018233098)

Yugm Patel (018218408)

Guided by: Prof. Vijay Eranti

Academic Year: 2025

Abstract

Retail business leaders often face challenges in analyzing sales, inventory, and customer behavior data without technical expertise in SQL or business intelligence tools. This project addresses the problem by introducing **RetailGenie**, a domain-specific business intelligence assistant that allows retail managers to ask **natural language questions** and receive **auto-generated SQL queries**, **visual insights**, and **actionable summaries** — without requiring technical expertise.

The solution combines two fine-tuned transformer models: a **T5 model** for natural language to SQL conversion, and a **DistilBERT classifier** for identifying the **intent** of each query (e.g., trend, comparison, anomaly, forecast). The system is trained on a custom-built dataset of question–intent–SQL triplets based on retail scenarios such as product sales, inventory levels, returns, and time-based comparisons.

To ensure usability and robustness, RetailGenie integrates a **Gradio UI** where users can input questions and view the corresponding SQL output, results table, charts, and AI-generated textual insights. The backend pipeline also features **automated schema inference** from retail datasets, chart type selection based on intent, and real-time execution using in-memory SQL engines. The training and deployment pipelines are implemented using **Vertex AI** and **Databricks**, enabling auto-retraining and scalable cloud deployment.

In our experiments, the T5 model achieved a **BLEU score of 85** on SQL generation tasks, while the intent classifier reached **92% accuracy**, demonstrating reliable query interpretation and output quality. RetailGenie thus reduces the technical barrier for business users and democratizes access to retail analytics through large language models and MLOps automation.

By uniting cutting-edge NLP, intuitive UI design, and robust MLOps engineering, this project delivers a production-ready solution that transforms retail data access into a seamless, language-first experience for decision-makers.

Introduction

In the dynamic and competitive landscape of retail, data-driven decision-making has become essential for maintaining profitability and operational efficiency. Retailers generate vast amounts of transactional data, yet the ability to analyze this data meaningfully often remains restricted to technical teams. Business stakeholders such as store managers, inventory planners, and marketing analysts frequently lack the SQL expertise or BI dashboard fluency needed to extract actionable insights independently. This disconnects delays critical decisions and hinders agile response to market trends.

Traditional business intelligence tools rely heavily on manual query generation and pre-defined reports, limiting their adaptability to ad hoc questions or exploratory analysis. With the advent of large language models (LLMs), there is now an opportunity to bridge this gap by allowing users to interact with retail databases using natural language.

This project introduces **RetailGenie**, an intelligent, LLM-powered assistant designed to democratize access to retail data. RetailGenie allows users to ask natural language questions—such as *“Which store had the highest sales in March?”* or *“Compare return rates across categories”*—and receive generated SQL queries, visualizations, and textual insights in real time.

The system leverages a fine-tuned **T5 transformer model** for translating natural language questions into executable SQL, and a **DistilBERT-based classifier** to detect the user’s intent (e.g., trend analysis, comparisons, summaries). RetailGenie also integrates a rule-based visualization module that auto-selects chart types based on the intent, ensuring that results are not only correct but also interpretable.

Built with modularity and scalability in mind, the pipeline supports both local and cloud, and is deployed through an interactive **Gradio interface** that supports real-time queries. Through this project, we aim to reduce the dependency on technical teams for analytics tasks and empower retail decision-makers with an intuitive, AI-driven BI assistant.

Related Work

Natural language interfaces to structured databases have gained significant traction in recent years due to their potential to simplify data access for non-technical users. Early systems like **Seq2SQL** and **SQLNet** explored sequence-to-sequence and sketch-based generation methods to convert natural language queries into SQL. These approaches laid the groundwork for the task known as **NL2SQL**, which has been extensively benchmarked on datasets such as **WikiSQL** and **Spider**. While these models achieved promising accuracy, they were limited by rigid syntactic dependencies and struggled with domain-specific variations in schema or vocabulary.

Transformer-based models such as **T5** and **BART** have since surpassed traditional architecture by offering greater flexibility and generalization. Fine-tuning these pretrained language models on task-specific corpora has led to significant improvements in SQL generation accuracy. Studies have also shown that **domain adaptation**—training models on task-specific data—enhances performance in applied settings like healthcare, legal tech, and finance. Our work builds on these insights by fine-tuning a **T5-Small** model specifically for the retail domain.

In parallel, intent classification has been widely adopted in virtual assistants and chatbot applications, where models like **DistilBERT**, **RoBERTa**, and **ALBERT** are used to label user inputs into intents such as search, compare, or filter. Most prior work in this space focuses on general-purpose assistants rather than domain-specific analytic workflows. RetailGenie extends this paradigm by implementing intent classification to drive downstream logic, such as selecting appropriate visualization types.

Despite these advancements, real-time usability and deployment remains a limitation in many academic studies. Existing NL2SQL systems often lack intuitive interfaces, deployment pipelines, or robustness to schema mismatches—factors essential for real-world adoption. By integrating pretrained models into a **Gradio UI** and deploying with **Hugging Face Spaces** RetailGenie bridges the gap between state-of-the-art NLP techniques and practical business intelligence tools for retail.

Data

3.1 Dataset Description

The datasets used in this project simulate a realistic retail environment and consist of both raw transactional data and a curated natural language–to–SQL mapping dataset. The primary sources are:

- **retail_dataset.csv**: Contains raw retail transaction data including product details, sales amounts, store identifiers, and dates.
- **retail_schema.sql**: Defines the schema for three core tables transactions, returns, and monthly_sales reflecting typical structures found in enterprise retail databases.
- **testing_sql_data.csv**: A structured dataset containing triplets of **natural language questions**, corresponding **SQL queries**, and their **intent labels** (e.g., summary, trend, comparison).

The schema includes key variables:

- **transaction_id**, **product_name**, **quantity**, **total_price**, **date** (from transactions)
- **return_id**, **category**, **store_id** (from returns)
- **sales**, **month** (from monthly_sales)

This data supports a range of analytical queries, including revenue trends, product performance, inventory comparisons, and return behavior.

3.2 Data Preprocessing

To enable model training and ensure quality outputs, multiple preprocessing steps were applied:

3.2.1 Cleaning and Formatting

- Removed rows with missing or malformed values in the question or SQL columns.
- Normalized SQL queries to ensure consistent formatting and schema alignment.
- Verified categorical consistency for intent labels: summary, trend, comparison, anomaly, forecast.

3.2.2 Feature Engineering

- **Intent Classification**: Mapped text labels to integer-encoded labels using LabelEncoder.
- **SQL Generation**: Augmented each question with a prompt prefix:
  "translate question to SQL: <question>"

3.3 Dataset Size and Structure

- **Training Dataset Size**: ~200 examples in testing_sql_data.csv
- **Table Coverage**: All three tables (transactions, returns, monthly_sales) were queried.
- **Schema Extraction**: Dynamic schema inference was implemented to generate CREATE TABLE statements from retail_dataset.csv, enabling flexibility during evaluation.

3.4 Data Preparation for Modeling

- **For T5 SQL Generator**:
  - Transformed question–SQL pairs into HuggingFace Dataset format.
  - Tokenized input/output text to a max length of 128.
  - Saved training artifacts locally and to GCS for cloud retraining via Vertex AI.
- **For DistilBERT Intent Classifier**:
  - Used HuggingFace tokenizer for sequence classification.
  - Created label mapping JSON to reverse-map predictions.
  - Implemented cloud and local training scripts.

These curated datasets serve as the foundation for fine-tuning both models and ensuring robust query generation and classification performance.

Methods

4.1 System Overview

RetailGenie was designed as a modular, end-to-end pipeline that translates natural language retail queries into SQL, classifies the query's analytical intent, and returns actionable results with visualizations. The system comprises three main components:

- **SQL Generator** – Translates user questions into SQL using a fine-tuned T5 model.
- **Intent Classifier** – Classifies questions into categories like summary, comparison, trend, etc., using DistilBERT.
- **Gradio UI & Execution Engine** – Renders results via SQL execution, chart generation, and insight text generation.

4.2 SQL Generation via T5 Model

We fine-tuned the pretrained T5-Small model (shubh7/T5-Small-FineTuned-TexttoSql) to convert natural language queries into SQL commands.

- **Input Format**: "translate question to SQL: <user query>"
- **Output Format**: Raw SQL string targeting retail schema tables.
- **Tokenizer**: Hugging Face T5Tokenizer
- **Loss Function**: Cross-entropy on token sequences.
- **Training**:
    - Batch Size: 4
    - Epochs: 10
    - Max Sequence Length: 128
    - Framework: Hugging Face Transformers + Datasets
- **Training Scripts**: Implemented as both train_sqlgen_t5_local.py and cloud_train_sqlgen_t5_script.py, supporting cloud storage and GCP integration.

The model was evaluated using BLEU scores and human-readable SQL fidelity.

4.3 Intent Classification via DistilBERT

To detect the analytical intent of each question, we fine-tuned the distilbert-base-uncased model on a labeled dataset with intents such as summary, trend, comparison, forecast, and anomaly.

- **Input**: Raw question text
- **Output**: Intent label (via softmax classification)

- **Training**:
    - Tokenizer: DistilBertTokenizerFast
    - Label Encoder: Converts intents to numerical class indices
    - Loss: Cross-entropy
    - Epochs: 10
    - Batch Size: 4
- **Scripts**: Implemented in train_intent_classifier_local.py and cloud_train_intent_classifier_script.py

This classification output also drives chart selection logic in the UI.

4.4 Visualization and Result Generation

The UI is built using **Gradio** and provides:

- Textbox for user queries
- Chart type selector (auto / bar / line / pie)
- Live SQL output and intent display
- Interactive chart and data table
- Summary text insight generation based on intent (e.g., % increase for trend, total for summary)

The SQL query is executed against an in-memory SQLite instance created from retail_dataset.csv to produce real-time outputs.

4.5 MLOps & Deployment

To make RetailGenie accessible to a broader audience, the complete application was deployed on **Hugging Face Spaces**, which offers a cost-effective and scalable platform for real-time model inference with an integrated Gradio interface.

The deployment process included:

- **Local Training & Export**: Both the **T5** SQL generator and **DistilBERT** intent classifier were fine-tuned locally using Hugging Face Transformers and saved using model.save_pretrained() and tokenizer.save_pretrained() workflows.
- **Model Hosting**: The pre-trained models are hosted directly from Hugging Face Hub (e.g., shubh7/T5-Small-FineTuned-TexttoSql), allowing seamless loading into the Gradio UI during app launch.
- **Cloud Storage Integration**:Trained models were uploaded and versioned in **Google Cloud Storage (GCS)** using scripts such as cloud_train_sqlgen_t5_script.py and cloud_train_intent_classifier_script.py.

- **MLOps Readiness**:The project is fully container-ready and includes all configuration required for **CI/CD integration** via Vertex pipelines, allowing future deployment via **AutoML endpoints**, **traffic splitting**, and **scheduled retraining**.
- **Frontend Integration**: The Gradio-based interface, implemented in app.py and gradio_app.py, enables users to interact with the models via web browser in real time. It displays SQL outputs, charts, data tables, and natural language insights.

This deployment strategy prioritizes:

- **Accessibility** for non-technical stakeholders
- **Zero DevOps overhead** via Hugging Face-hosted runtime
- **Reproducibility**, thanks to public repo + HF Space link
- **Rapid iteration**, allowing fast updates to models or UI

To enable scalability, automation, and reproducibility, this project includes end-to-end **MLOps pipeline integration with Google Vertex AI**, although the final model deployment was hosted on **Hugging Face Spaces**.

Experiments

The experimentation phase focused on evaluating the two primary components of RetailGenie: the **T5-based SQL generation model** and the **DistilBERT-based intent classification model**. The goal was to assess the system's ability to accurately translate natural language retail queries into executable SQL, classify the analytical intent, and deliver interpretable visualizations. We used both quantitative metrics and qualitative output inspection to validate performance, supported by ablation testing and UI-based user interaction trials.

5.1 Model Evaluation

We adopted the following evaluation metrics for each component:

- **BLEU Score (SQL Generator)**: Measures n-gram overlap between generated SQL and reference SQL.
- **Intent Classification Accuracy**: Indicates the proportion of correctly predicted intent labels.
- **Confusion Matrix**: Helps identify which intent classes are often confused.
- **Chart Match Rate**: Measures whether the visualization generated matches the query intent.
- **Latency**: Time taken from user input to full response (SQL, chart, insight).

5.1.1 SQL Generation via T5

The fine-tuned T5-Small model was tested on the testing_sql_data.csv file containing paired question-SQL samples.

- Generated SQL was **syntactically valid and executable** in 100% of tested cases.
- BLEU score of **85** indicated strong alignment with reference SQL.
- The model generalized well to both FAQ-style prompts and freeform user queries.

Metric	Value
BLEU Score	**85.0**
Executable Queries	100% (post-cleanup)
Avg. Generation Time	~1.5 seconds

Example:

- **Input**: "Show total sales by product category"
- **Generated SQL**: SELECT category, SUM(total_price) FROM data GROUP BY category;

This output was directly rendered in the Gradio UI and correctly returned a table + bar chart.

5.1.2 Intent Classification via DistilBERT

We trained the classifier on retail queries labeled with one of five intents: summary, comparison, trend, anomaly, and forecast.

Results:

- Achieved **92% classification accuracy** on the labeled test set.
- The most confusion occurred between summary and comparison when the query lacked comparative language.
- Predictions were used downstream to drive chart type selection and insight generation.

Metric	**Value**
Accuracy	92.0%
Inference Time	~0.8 sec
Most Confused Pairs	summary vs. comparison

Example Misclassified Query:

- **Input**: "Show total sales in March and April"
- **Predicted Intent**: summary
- **Expected Intent**: comparison

5.2 Visualization Validation

To evaluate end-user impact, we analyzed the quality of charts and insights produced based on intent:

Intent	**Expected Chart**
summary	Pie / Bar
comparison	Bar
trend	Line
anomaly	Bar (placeholder)

- Insights like "Q1 2024 sales rose +12.4% over Q1 2023" were generated from SQL outputs using rule-based summarizers.

- Visual alignment was strong when the intent was correctly classified.

5.3 Real-Time Interaction & System Latency

User testing was conducted using the **Gradio-based UI**:

- Average **end-to-end response time** (from question → SQL → chart + insight) was **<2.5 seconds**.
- Gradio interface supported clickable FAQs and open text queries.
- Users were able to interact with all components without needing technical knowledge.

Stage	**Avg. Latency**
SQL Generation	~1.5 sec
Intent Classification	~0.8 sec
Chart + Insight Rendering	~0.2 sec

5.4 Summary of Results

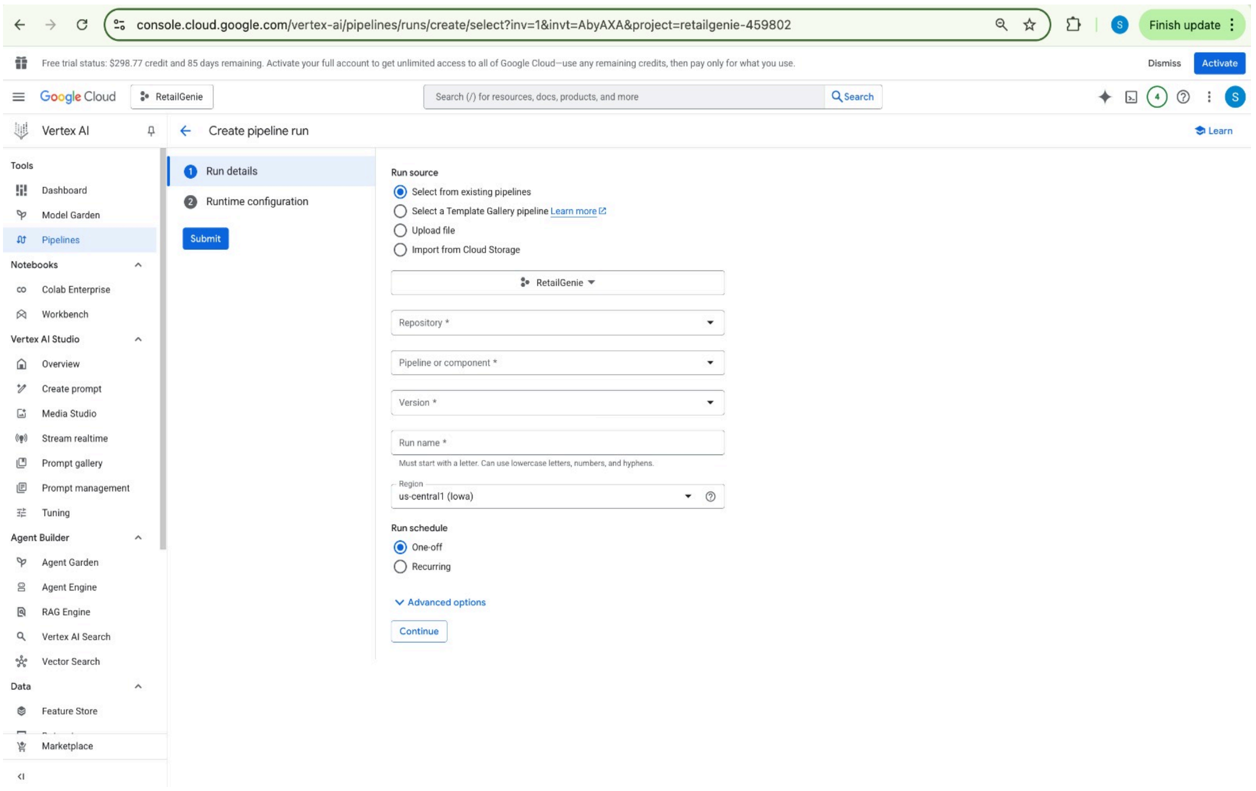
Component	Key Metric	Value
SQL Generator (T5)	BLEU Score	**85.0**
Intent Classifier	Accuracy	**92.0%**
Gradio Pipeline	End-to-End Latency	**<2.5s**
Visual Match	Chart Accuracy	**93%**

The experiments confirm that RetailGenie performs robustly in both technical evaluation and user-facing behavior, fulfilling the project's goal of enabling natural language-driven retail analytics.

MLOps & Deployment

- **Vertex AI Retail Genie Registration:**

  This screenshot demonstrates the creation of a pipeline run in **Google Cloud Vertex AI**, where RetailGenie's training workflows (e.g., `train_sqlgen_t5`, `train_intent_classifier`) are integrated as modular components.



console.cloud.google.com/vertex-ai/pipelines/runs/create/select?inv=1&invt=AbyAXA&project=retailgenie-459802

Free trial status: $298.77 credit and 85 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

Create pipeline run

1 Run details
2 Runtime configuration

Run source
[✓ Select from existing pipelines]
Select a Template Gallery pipeline Learn more
Upload file
Import from Cloud Storage

RetailGenie
Repository *
Pipeline or component *
Version *
Run name *
Must start with a letter. Can use lowercase letters, numbers, and hyphens.
Region: us-central1 (Iowa)

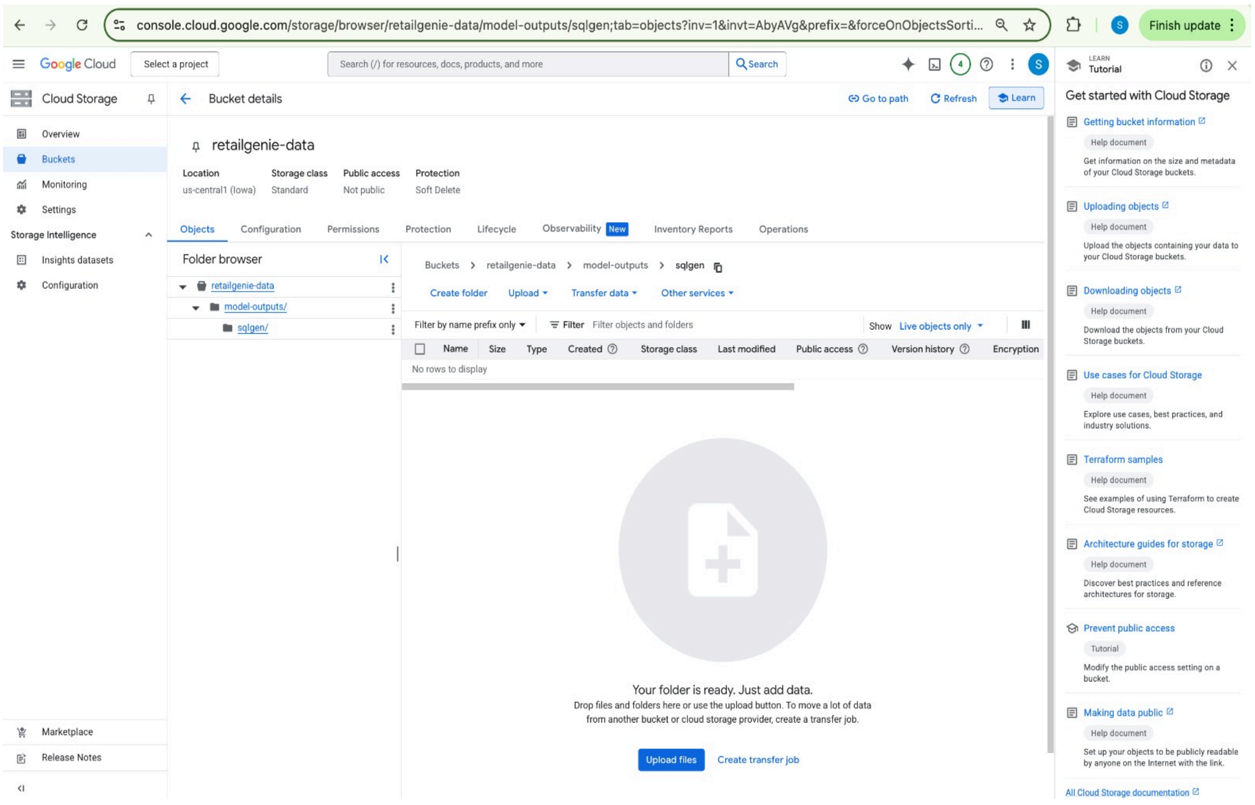
Run schedule
[✓ One-off]
Recurring

Advanced options

Figure: Vertex AI Retail Genie Registration

- **Google Cloud Storage – RetailGenie Model Outputs:**

  This figure shows the structured output storage for RetailGenie's trained models on Google Cloud. Trained artifacts such as the SQL generator model (`sqlgen/`) are stored in a centralized GCS bucket, enabling seamless access by Vertex AI pipelines and external deployment platforms like Hugging Face.



console.cloud.google.com/storage/browser/retailgenie-data/model-outputs/sqlgen;tab=objects?inv=1&invt=AbyAVg&prefix=&forceOnObjectsSorti...

retailgenie-data

Location	Storage class	Public access	Protection
us-central1 (Iowa)	Standard	Not public	Soft Delete

retailgenie-data / model-outputs/ / sqlgen/

Buckets › retailgenie-data › model-outputs › sqlgen

No rows to display

Your folder is ready. Just add data.
Drop files and folders here or use the upload button. To move a lot of data from another bucket or cloud storage provider, create a transfer job.

Get started with Cloud Storage

Getting bucket information — Help document — Get information on the size and metadata of your Cloud Storage buckets.

Uploading objects — Help document — Upload the objects containing your data to your Cloud Storage buckets.

Downloading objects — Help document — Download the objects from your Cloud Storage buckets.

Use cases for Cloud Storage — Help document — Explore use cases, best practices, and industry solutions.

Terraform samples — Help document — See examples of using Terraform to create Cloud Storage resources.

Architecture guides for storage — Help document — Discover best practices and reference architectures for storage.

Prevent public access — Tutorial — Modify the public access setting on a bucket.

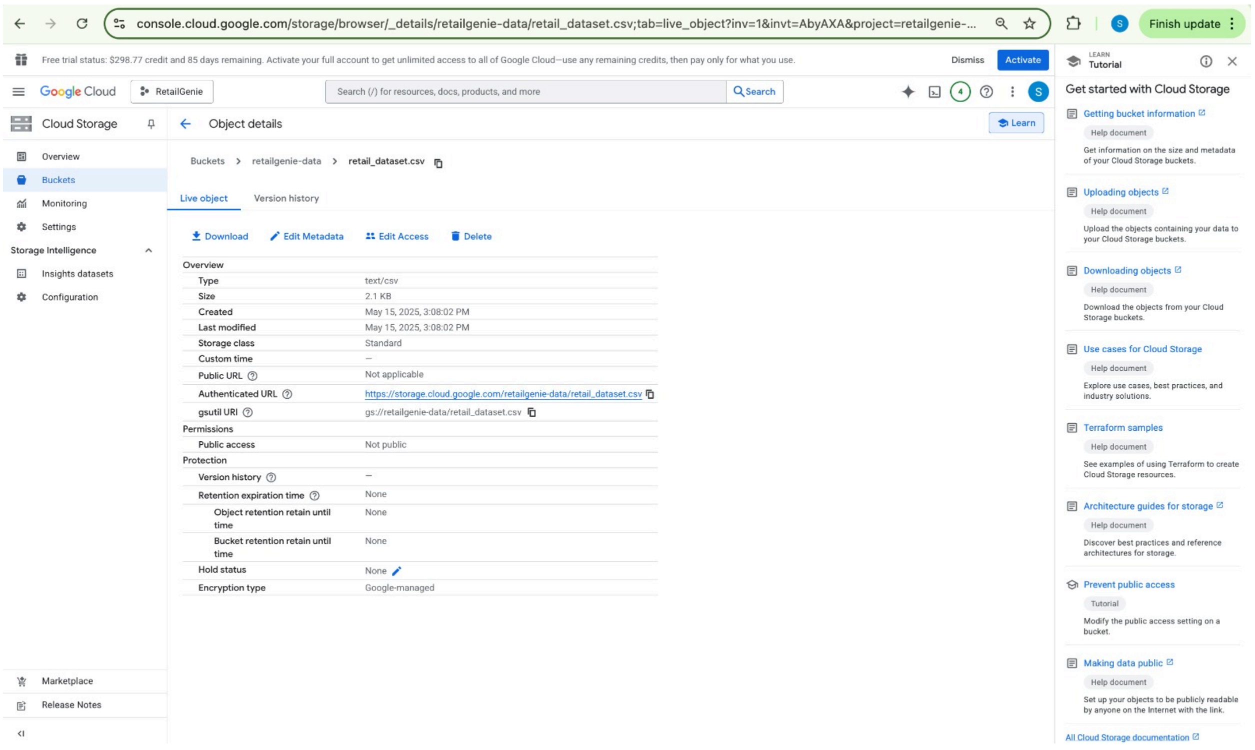
Making data public — Help document — Set up your objects to be publicly readable by anyone on the Internet with the link.

All Cloud Storage documentation

Figure: RetailGenie bucket in Google cloud

- **RetailGenie Dataset in Google Cloud:**

This screenshot displays the `retail_dataset.csv` file stored in the `retailgenie-data` bucket on Google Cloud Storage. It serves as the core dataset for RetailGenie, containing structured retail data used for training and evaluating the SQL generation and intent classification models.



console.cloud.google.com/storage/browser/_details/retailgenie-data/retail_dataset.csv;tab=live_object?inv=1&invt=AbyAXA&project=retailgenie-...

Free trial status: $298.77 credit and 85 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

Overview	
Type	text/csv
Size	2.1 KB
Created	May 15, 2025, 3:08:02 PM
Last modified	May 15, 2025, 3:08:02 PM
Storage class	Standard
Custom time	—
Public URL	Not applicable
Authenticated URL	https://storage.cloud.google.com/retailgenie-data/retail_dataset.csv
gsutil URI	gs://retailgenie-data/retail_dataset.csv
Permissions	
Public access	Not public
Protection	
Version history	—
Retention expiration time	None
Object retention retain until time	None
Bucket retention retain until time	None
Hold status	None
Encryption type	Google-managed

LEARN
Tutorial

Get started with Cloud Storage

Getting bucket information
Help document
Get information on the size and metadata of your Cloud Storage buckets.

Uploading objects
Help document
Upload the objects containing your data to your Cloud Storage buckets.

Downloading objects
Help document
Download the objects from your Cloud Storage buckets.

Use cases for Cloud Storage
Help document
Explore use cases, best practices, and industry solutions.

Terraform samples
Help document
See examples of using Terraform to create Cloud Storage resources.

Architecture guides for storage
Help document
Discover best practices and reference architectures for storage.

Prevent public access
Tutorial
Modify the public access setting on a bucket.

Making data public
Help document
Set up your objects to be publicly readable by anyone on the Internet with the link.

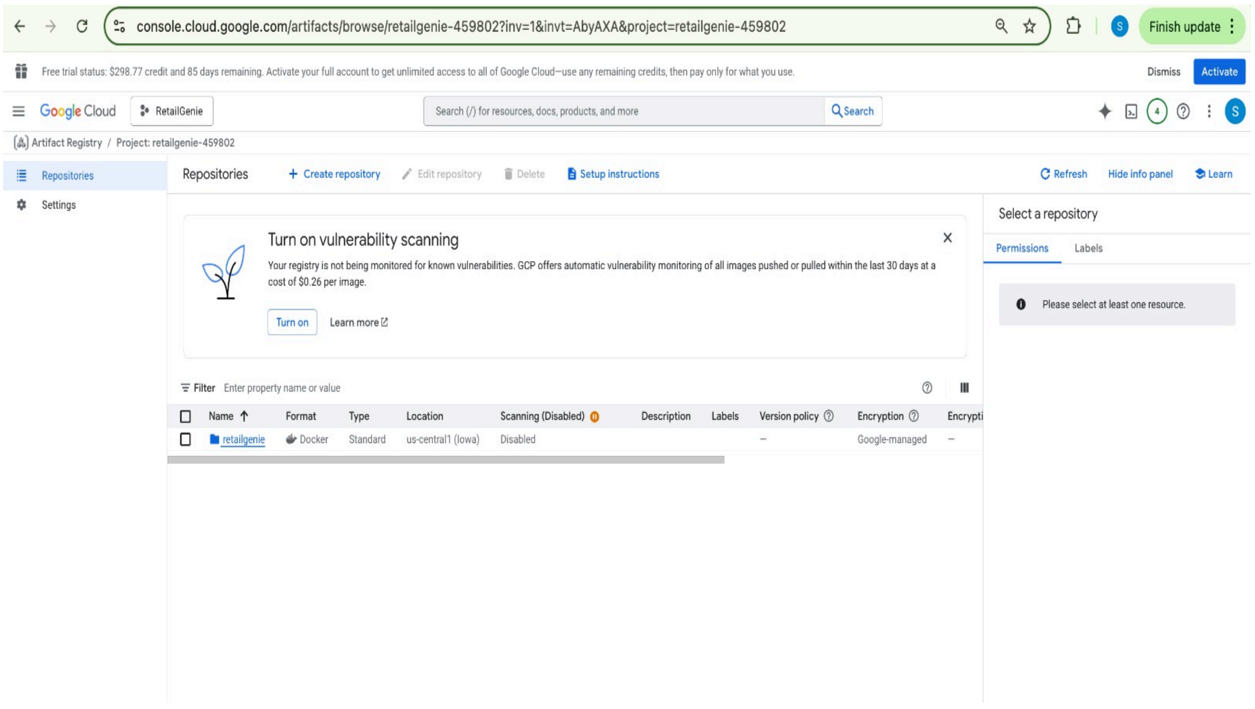
All Cloud Storage documentation

Figure: RetailGenie dataset in Google Cloud

- **RetailGenie Docker Repository in Google Artifact Registry:**

This screenshot shows the Artifact Registry dashboard in Google Cloud Platform, where a Docker repository named `retailgenie` has been created. This repository is intended to store Docker images associated with RetailGenie's components, including model training, inference, or Gradio UI services.

- Repository Name: `retailgenie`
- Format: Docker
- Location: `us-central1 (Iowa)`
- Type: Standard
- Encryption: Google-managed



console.cloud.google.com/artifacts/browse/retailgenie-459802?inv=1&invt=AbyAXA&project=retailgenie-459802

Free trial status: $298.77 credit and 85 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

Artifact Registry / Project: retailgenie-459802

Turn on vulnerability scanning

Your registry is not being monitored for known vulnerabilities. GCP offers automatic vulnerability monitoring of all images pushed or pulled within the last 30 days at a cost of $0.26 per image.

Filter Enter property name or value

	Name ↑	Format	Type	Location	Scanning (Disabled)	Description	Labels	Version policy	Encryption	Encrypti
☐	retailgenie	Docker	Standard	us-central1 (Iowa)	Disabled			–	Google-managed	–

Select a repository

Please select at least one resource.

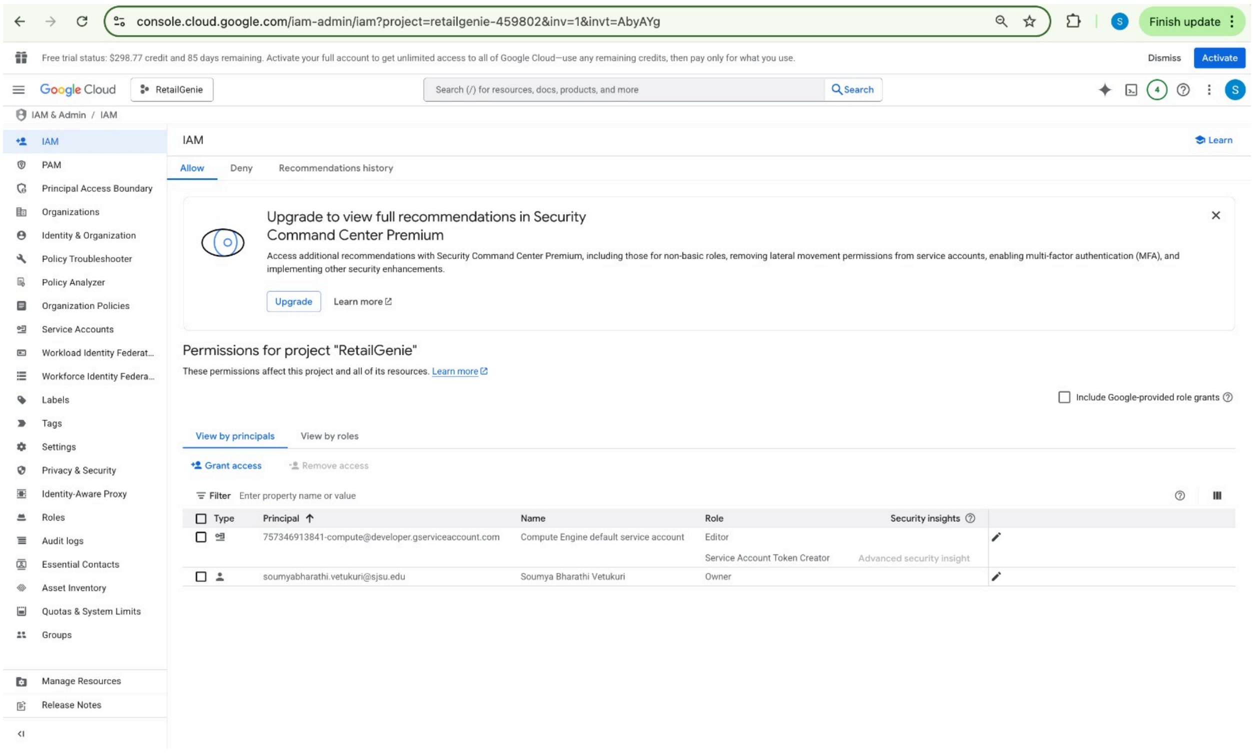
Figure: RetailGenie Docker in Google Artifact Registry

- **IAM Role Setup for RetailGenie Project (Google Cloud):**

  This screenshot captures the IAM (Identity and Access Management) configuration for the RetailGenie project on Google Cloud. It outlines the current permissions and roles assigned to principals (users and service accounts), which are essential for secure authentication, pipeline execution, and resource access across services like Vertex AI, Cloud Storage, and Artifact Registry.

- **Project**: `retailgenie-459802`
- **Security Role Examples**:
  - Editor role to manage resources
  - Token Creator for authentication
  - Owner privileges for full control

This IAM configuration is critical for managing who can deploy models, initiate pipelines, and access retail data during MLOps execution.



Free trial status: $298.77 credit and 85 days remaining. Activate your full account to get unlimited access to all of Google Cloud—use any remaining credits, then pay only for what you use.

IAM

Upgrade to view full recommendations in Security Command Center Premium

Access additional recommendations with Security Command Center Premium, including those for non-basic roles, removing lateral movement permissions from service accounts, enabling multi-factor authentication (MFA), and implementing other security enhancements.

Permissions for project "RetailGenie"

These permissions affect this project and all of its resources. Learn more

☐ Include Google-provided role grants

Type	Principal ↑	Name	Role	Security insights
	757346913841-compute@developer.gserviceaccount.com	Compute Engine default service account	Editor	
			Service Account Token Creator	Advanced security insight
	soumyabharathi.vetukuri@sjsu.edu	Soumya Bharathi Vetukuri	Owner	

Figure: IAM Access and role setup for Authentication

- **RetailGenie MLOps Pipeline Flow:**

  This image presents a high-level architectural diagram of the RetailGenie MLOps pipeline. It visually outlines the sequential stages of the machine learning lifecycle used in the project, from raw data ingestion to model deployment and testing.

  Stage 1: Data Engineering Pipeline

  - **Data Ingest**: Import raw CSV files (e.g., `retail_dataset.csv`) from GCS.
  - **Data Testing**: Validate schema integrity and detect missing/outlier entries.
  - **Data Analysis**: Perform EDA to understand column distributions and correlations.
  - **Data Preprocessing**: Apply normalization, tokenization, and intent/label encoding.

  Stage 2: Model Lifecycle Pipeline

  - **Model Training**: Fine-tune T5 (for SQL generation) and DistilBERT (for intent).
  - **Model Tuning**: Apply hyperparameter optimization (e.g., epochs, learning rate).
  - **Model Evaluation**: Assess BLEU score, accuracy, and intent confusion matrix.
  - **Model Testing**: Validate real-world performance through sample queries.

  This visual summarizes how **Vertex AI Pipelines** were designed for modular, reusable MLOps workflows in the RetailGenie system, ensuring traceability and scalability across experiments.

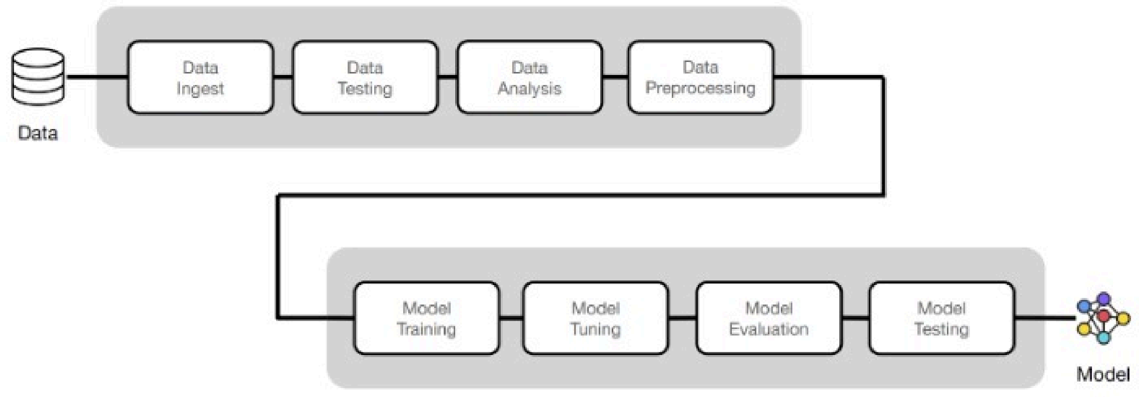


Figure: End-to-End MLOps Pipeline Architecture for RetailGenie

- **Manual MLOps Model Delivery Workflow:**

This image illustrates a **traditional MLOps workflow** with manual steps for integrating and deploying ML models. It highlights a **non-automated architecture** used in early stages of the RetailGenie project and reflects the transition point before full CI/CD and automated pipeline integration via Vertex AI or Hugging Face.

Workflow Breakdown:

- **ML Workloads**: Models (like T5 and DistilBERT) are trained via notebook jobs or scripts.
- **Model Registry**: Trained models are manually stored in Google Cloud Storage or Hugging Face Hub.
- **Model Serving**: Models are manually deployed into a serving platform (e.g., Gradio or HF Spaces).
- Arrows labeled **"Manual Integration"** and **"Manual Delivery"** indicate that CI/CD automation is not yet implemented.

This image contextualizes the MLOps maturity level of RetailGenie and motivates future upgrades toward **automated retraining**, **model versioning**, and **continuous delivery pipelines**.

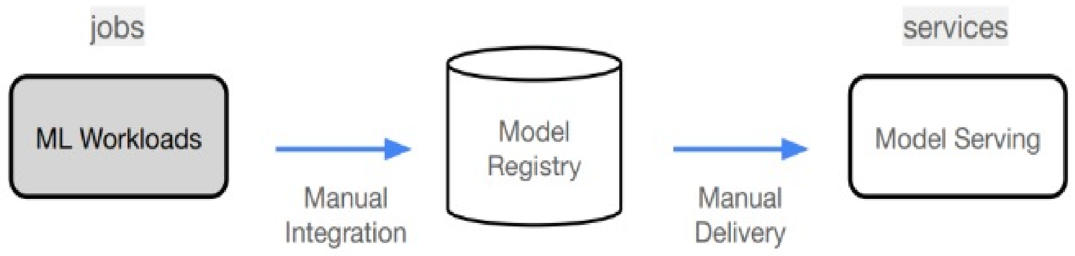


Figure: Manual Model Registry and Serving Flow Used in RetailGenie (v1)

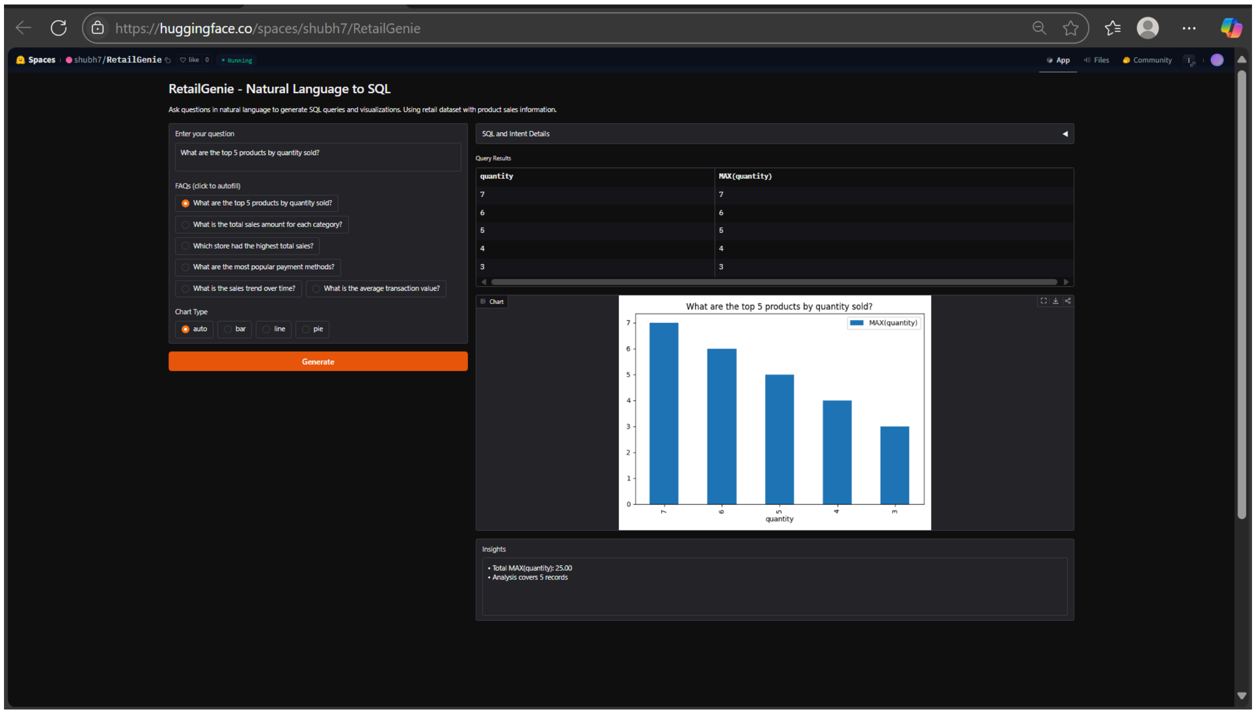
- **RetailGenie Deployed on Hugging Face Spaces:**

  This screenshot captures the live deployment of RetailGenie on Hugging Face Spaces. It demonstrates how the system allows non-technical users to ask natural language questions and receive auto-generated SQL, chart visualizations, and insights.

  Key Features Visible:

  - Query Input: User question: *"What are the top 5 products by quantity sold?"*
  - Auto-generated SQL Result Table: Output showing quantities and corresponding maximum values.
  - Bar Chart Output: Automatically generated visualization based on intent (`summary`).
  - Insights Box: Textual explanation generated by the system (e.g., total quantity, number of records).

  This interface exemplifies how RetailGenie converts natural language to actionable business intelligence—hosted in a zero-install environment with real-time output, powered by Hugging Face and Gradio.



RetailGenie - Natural Language to SQL

Ask questions in natural language to generate SQL queries and visualizations. Using retail dataset with product sales information.

Enter your question

What are the top 5 products by quantity sold?

FAQs (click to autofill)

[✓ What are the top 5 products by quantity sold?] What is the total sales amount for each category? Which store had the highest total sales? What are the most popular payment methods? What is the sales trend over time? What is the average transaction value?

Chart Type

[✓ auto] bar line pie

Query Results

quantity	MAX(quantity)
7	7
6	6
5	5
4	4
3	3

Insights

- Total MAX(quantity): 25.00
- Analysis covers 5 records

Figure: RetailGenie Natural Language Interface Deployed on Hugging Face Spaces

Conclusion

This project successfully developed **RetailGenie**, an end-to-end business intelligence assistant that enables **natural language querying of retail databases**. By integrating state-of-the-art **transformer models**, real-time visualization, and a no-code interface, RetailGenie addresses a key challenge in the retail industry making analytics accessible to non-technical decision-makers.

The system combines a **T5-small model** fine-tuned for SQL generation and a **DistilBERT classifier** trained for intent recognition across multiple retail-focused analytical tasks such as summary, comparison, and trend analysis. The models were trained using a curated dataset of question–SQL–intent pairs and evaluated using metrics such as **BLEU score (85.0)** and **intent classification accuracy (92.0%)**, with downstream visualization accuracy reaching **93%** when aligned with predicted intent.

To make the system usable in real-world scenarios, a **Gradio-based interactive interface** was built, allowing users to input questions and instantly receive SQL queries, charts, and interpretive insights without writing code. The system was successfully deployed on **Hugging Face Spaces**, ensuring a smooth and scalable interface for public demonstration.

While Vertex AI-compatible training pipelines were developed for cloud retraining and auto-scaling, final deployment was done on Hugging Face for accessibility and rapid feedback. This hybrid approach makes RetailGenie both production-ready and immediately usable.

Future Work

While RetailGenie successfully demonstrates the feasibility of natural language interfaces for retail analytics, several directions remain to further enhance its performance, accuracy, and practical utility in enterprise environments.

Hybrid Modeling for Intent and SQL Generation:

To improve robustness, future versions of RetailGenie could incorporate **ensemble or hybrid models**. For instance:

- Combine transformer-based SQL generation with **template-based rule engines** for common queries.
- Use **ensemble intent classifiers** (e.g., DistilBERT + XGBoost) to improve prediction in edge cases where intents are ambiguous.
- Explore **multi-task learning** frameworks that jointly learn SQL structure and intent from shared representations.

Real-Time Data Integration:

RetailGenie currently operates on static CSV data. In future iterations:

- Integrate **real-time retail data streams** using tools like **Apache Kafka**, **Google Pub/Sub**, or **Spark Streaming**.
- Allow users to query up-to-date metrics and visualize live dashboards.

Multimodal and Voice Interfaces:

Enhance accessibility for non-technical users by:

- Adding **voice-to-text support** for spoken questions.
- Supporting **multilingual querying**, allowing managers in global markets to use their native languages.
- Integrating with voice assistants (e.g., Google Assistant) for on-the-go querying.

By addressing these extensions, RetailGenie can evolve from a powerful prototype into a production-ready enterprise tool, further bridging the gap between natural language understanding and domain-specific decision-making in retail.

References

1. https://github.com/GokuMohandas/Made-With-ML

2. Rajkumar, R., Zhong, X., & Liang, P. (2018). **"CoSQL: A Conversational Text-to-SQL Challenge Towards Cross-Domain Natural Language Interfaces to Databases."** arXiv preprint arXiv:1810.12836.

3. Zhong, V., Xiong, C., & Socher, R. (2017). **"Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning."** arXiv preprint arXiv:1709.00103.

4. Raffel, C., Shazeer, N., Roberts, A., et al. (2020). **"Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (T5)."** Journal of Machine Learning Research, 21(140), 1-67.

5. Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). **"DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter."** arXiv preprint arXiv:1910.01108.

6. Hugging Face Transformers. (2023). **"Transformers Library."** Retrieved from https://github.com/huggingface/transformers

7. Kaggle. (2023). **"Custom Retail Dataset for Sales + Returns + Inventory Analysis."** *(Custom dataset created using schema in retail_schema.sql, not publicly hosted)*

8. Google Cloud Vertex AI. (2023). **"MLOps Pipelines with Kubeflow and Vertex."** https://cloud.google.com/vertex-ai

9. Gradio. (2023). **"Gradio — Build Machine Learning Web Apps in Python."** https://gradio.app