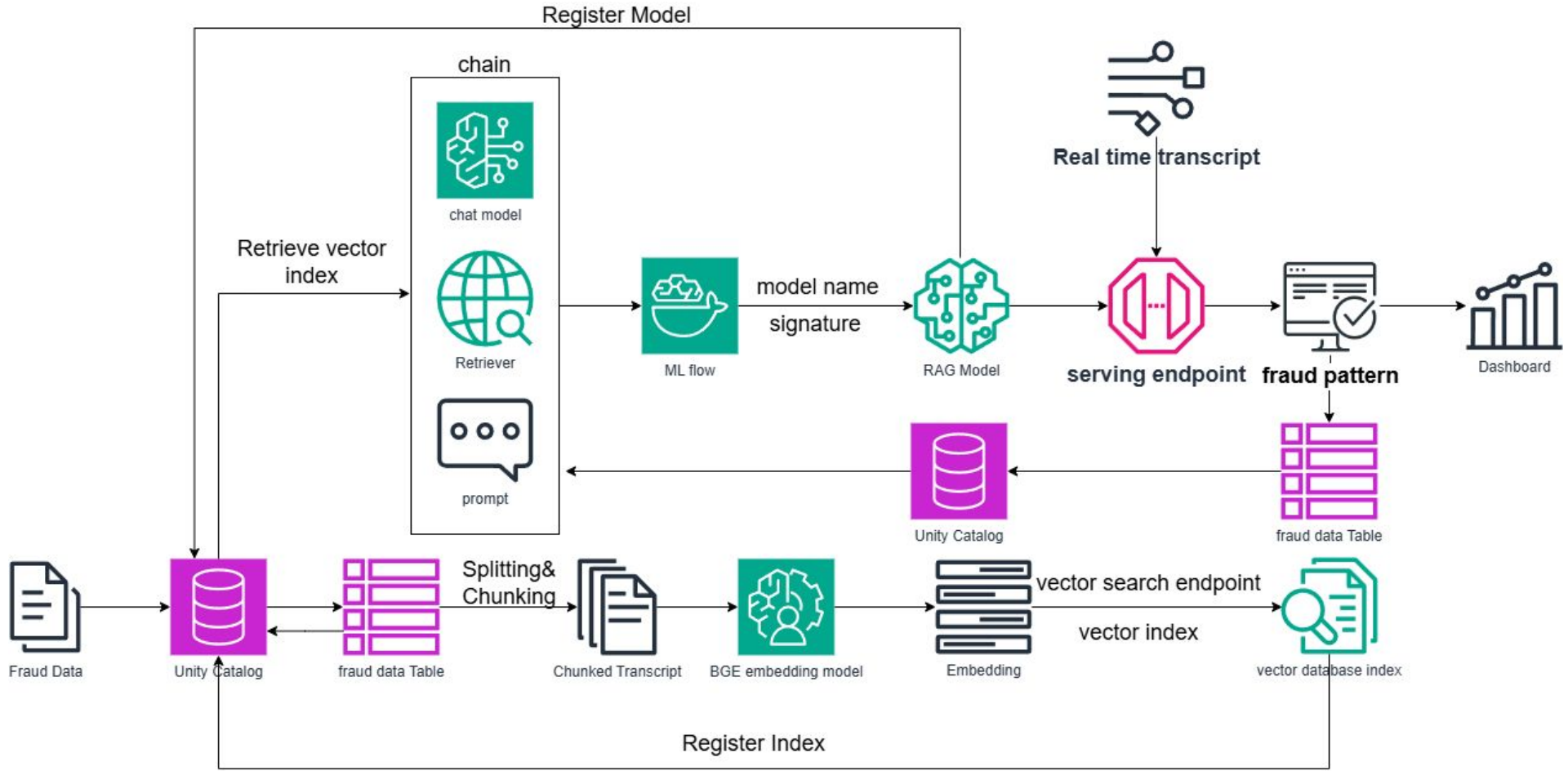
Chanllenges

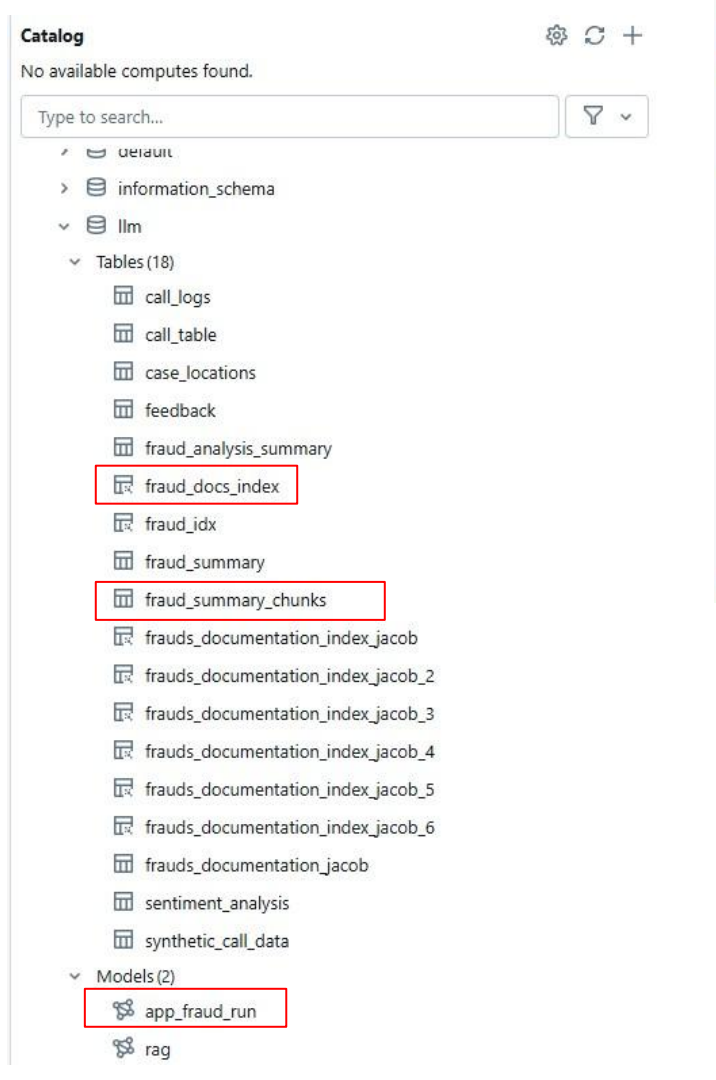
- Accuracy and coherence (lack of domain knowledge)
- Corner/Borderline case (Human/Expert feedback)
- No updated knowledge base
- Deployment

Solutions

- Prompt Engineering + Retrieval-Augmented Generation (RAG)
- Databricks: vector index+unity catalog+mlflow+serving endpoint


Register Model
chain
chat model
Retriever
prompt
Retrieve vector index
ML flow
model name
signature
RAG Model
Real time transcript
serving endpoint
fraud pattern
Dashboard
Unity Catalog
fraud data Table
Fraud Data
Unity Catalog
fraud data Table
Splitting&
Chunking
Chunked Transcript
BGE embedding model
Embedding
vector search endpoint
vector index
vector database index
Register Index

RAG




```python
# Create or update serving endpoint
from databricks.sdk import WorkspaceClient
from databricks.sdk.service.serving import EndpointCoreConfigInput, ServedModelInput, ServedModelInputWorkloadSize
# from databricks.sdk.service.serving import EndpointCoreConfigInput, ServedEntityInput


serving_endpoint_name = "fraud_app_rag_endpoint_dev"
# latest_model_version = get_latest_model_version(model_name)
latest_model_version = '3'

w = WorkspaceClient()
endpoint_config = EndpointCoreConfigInput(
    name=serving_endpoint_name,
    served_models=[
        ServedModelInput(
            model_name=model_name,
            model_version = latest_model_version,
            workload_size=ServedModelInputWorkloadSize.SMALL, # Use string for workload size
            scale_to_zero_enabled=True,
            environment_vars={
                "DATABRICKS_TOKEN": os.environ['DATABRICKS_TOKEN']  # <scope>/<secret> that contains an access
                token
            }
        )
    ]
)
```

```python
import requests
import json
import os

host = "https://" + spark.conf.get("spark.databricks.workspaceUrl")
os.environ['DATABRICKS_TOKEN'] = dbutils.notebook.entry_point.getDbutils().notebook().getContext().apiToken().get
()

local_endpoint = 'https://dbc-15e7860d-511f.cloud.databricks.com/serving-endpoints/fraud_app_rag_endpoint_dev/
invocations'
headers = {
    "Authorization":  f"Bearer {os.environ['DATABRICKS_TOKEN']}",
    "Content-Type": "application/json"
}

payload = {
    "inputs": [{"query": question}]
}
response = requests.post(local_endpoint, headers=headers, data=json.dumps(payload))
response_json = response.json()
print(response_json)
```

Future Work

- Consolidate fraud patterns
- Fraud pattern across the industries
- Build knowledge base on real-world data