Leveraging GenAI Models to Understand Business Risk from Unstructured Data

CS 700-Generative AI

Francis Kurian

Objectives:

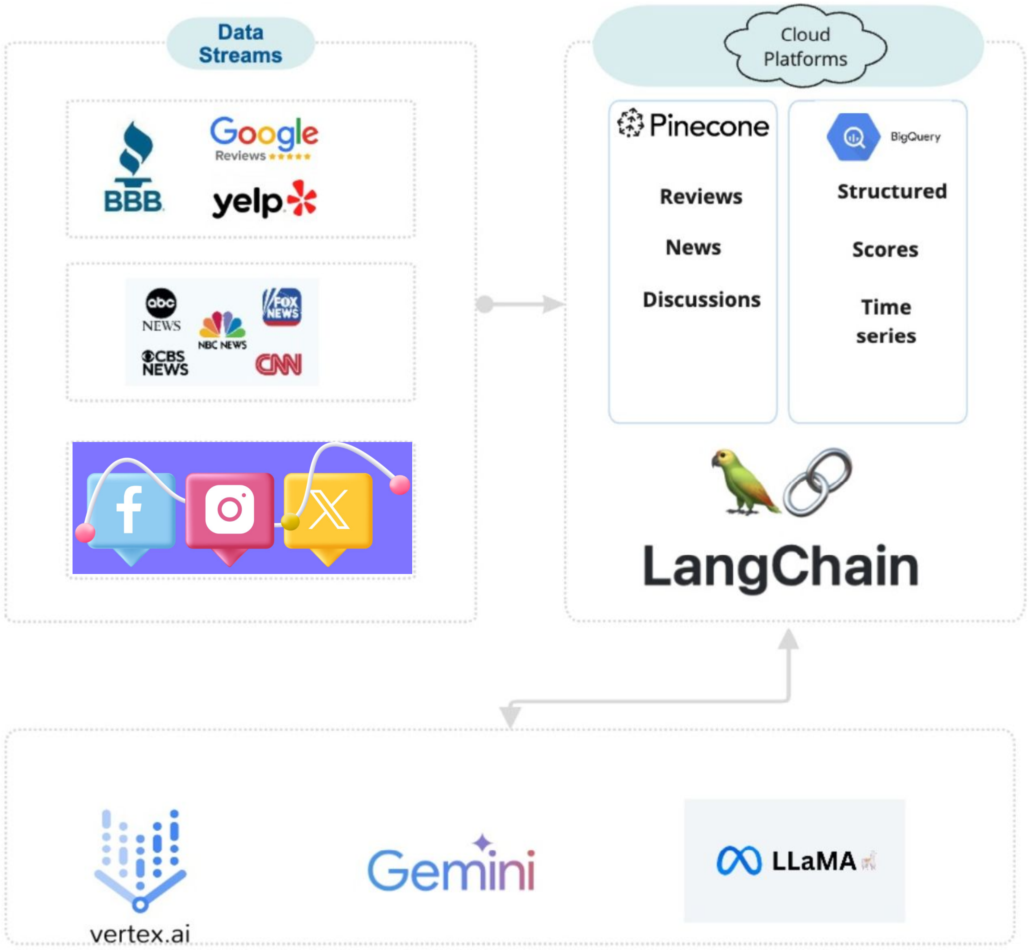
- Leverage LLM to analyze unstructured data to understand Risk associated with a Business.
- Build a time-series behaviour scorecard to quantify business risk from unstructured data.
- Apply prompt engineering techniques and retrieval-augmented generation (RAG) to blend structured and unstructured data to enhance data retrieval and analysis.
- Utilize LangChain, Hugging Face/TRANSFORMER, Vector Database, Google Cloud services(BigQuery, Vertex AI APIs,) and Colab Notebook, to build and deploy a data engineering pipeline for analytics.

Data:

Synthetic Data Generation: Sample of unstructured data for a list of businesses were generated using LLM. Dataset mimic reviews from review sites like Yelp, social media site like twitter, news articles, and discussion forums where a business name appear . A json file was generated with specific structure and was stored in a vector database.

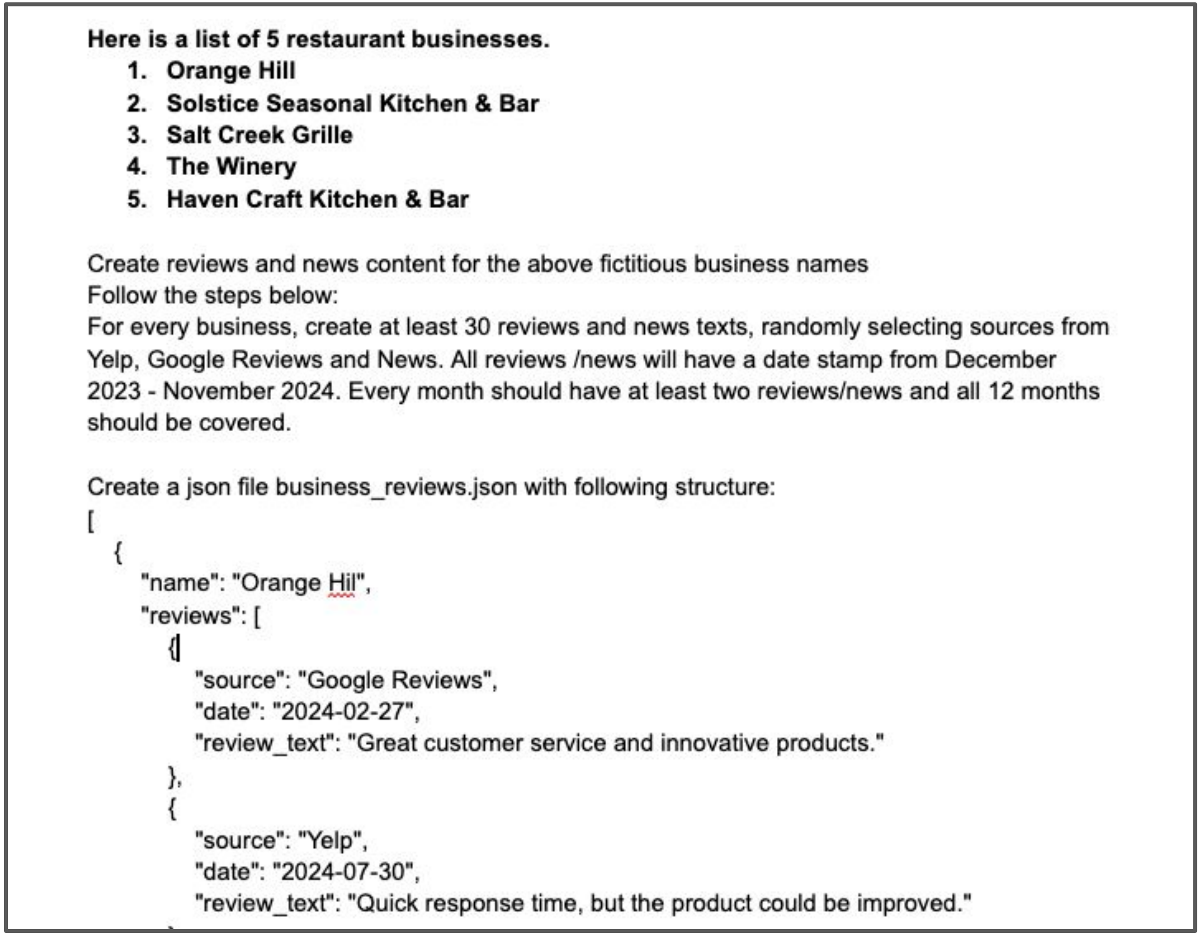
Structured Data: Classification algorithm was used to create a timeseries behaviour scorecard for each businesses. This scorecard was used as a proxy for other financial metrics.

Data Pipelines and infrastructure



- Generate synthetic data that mimic various platforms
- Develop a pipeline in Python within a Google Colab Notebook
- Utilizing Pinecone and BigQuery for data storage.
- Integrate Google Cloud services/Vertex AI to streamline data flow from collection to analysis.
- LangChain classes to integrate API calls/prompts
- Vertex AI/Hugging Face for LLM calls

Synthetic unstructured data: Json file creation prompt



Here is a list of 5 restaurant businesses.

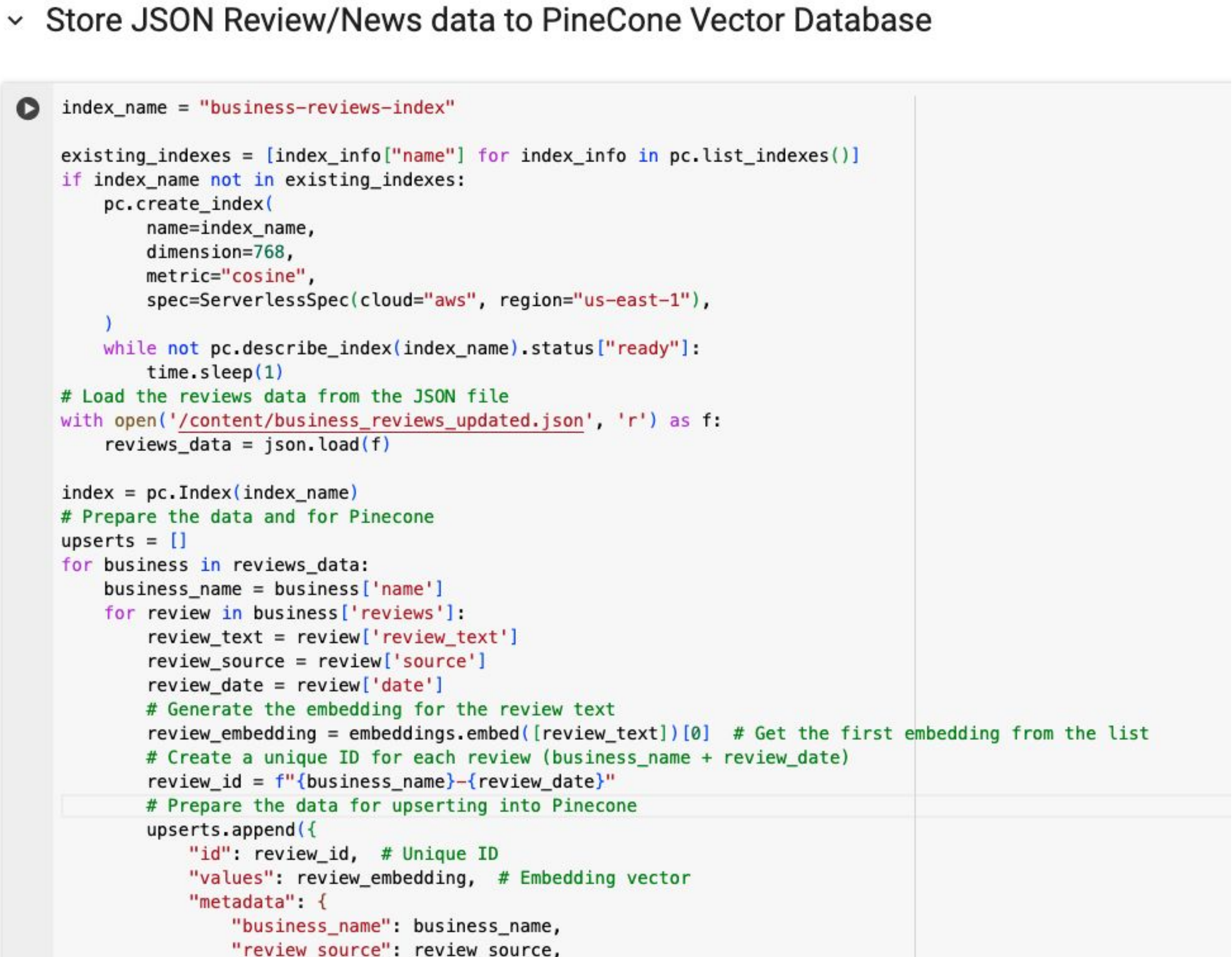
1. **Orange Hill**
2. **Solstice Seasonal Kitchen & Bar**
3. **Salt Creek Grille**
4. **The Winery**
5. **Haven Craft Kitchen & Bar**

Create reviews and news content for the above fictitious business names
Follow the steps below:
For every business, create at least 30 reviews and news texts, randomly selecting sources from Yelp, Google Reviews and News. All reviews /news will have a date stamp from December 2023 - November 2024. Every month should have at least two reviews/news and all 12 months should be covered.

Create a json file business_reviews.json with following structure:

```
[
  {
    "name": "Orange Hil",
    "reviews": [
      {
        "source": "Google Reviews",
        "date": "2024-02-27",
        "review_text": "Great customer service and innovative products."
      },
      {
        "source": "Yelp",
        "date": "2024-07-30",
        "review_text": "Quick response time, but the product could be improved."
```

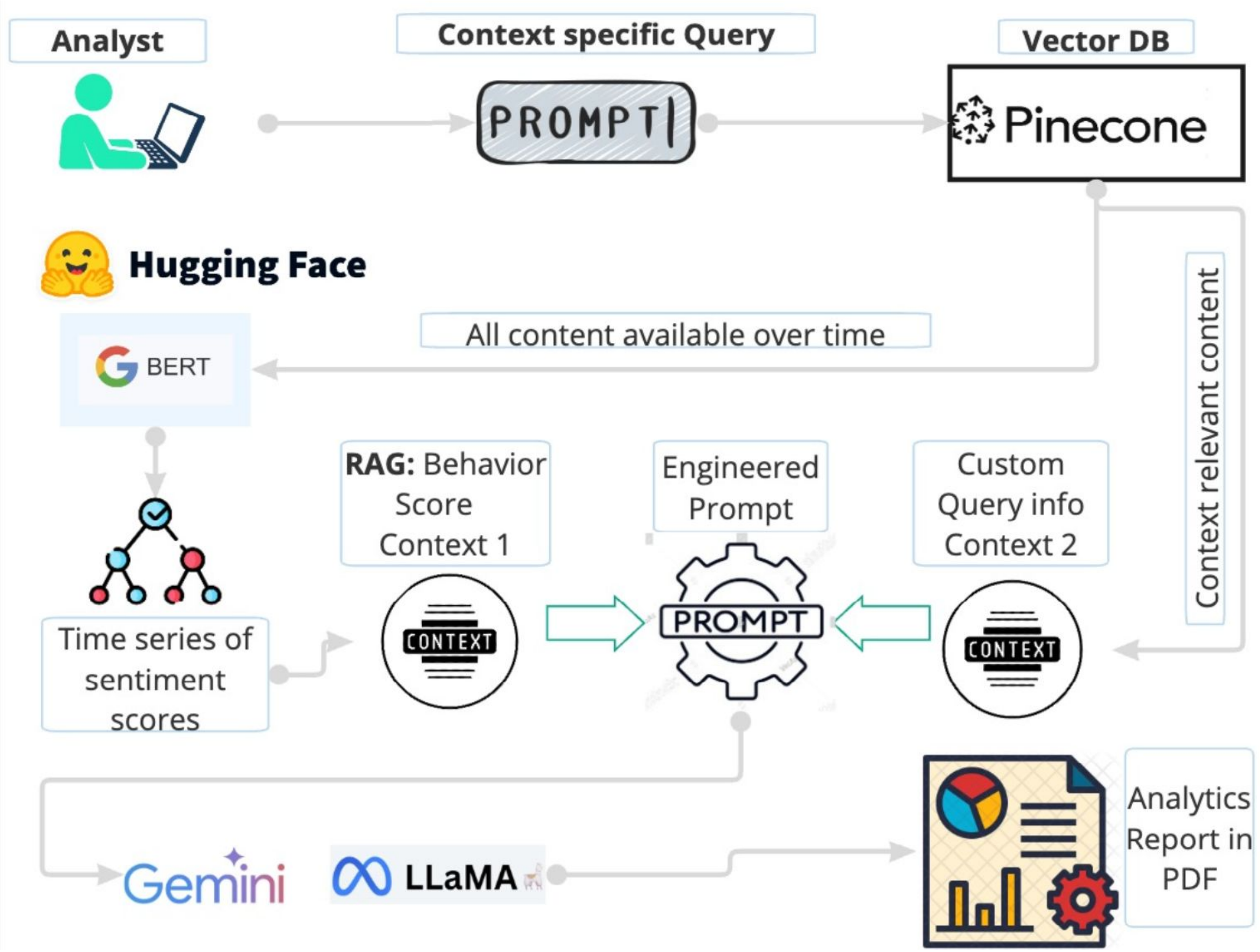
- **Realistic and Diverse Data Simulation**:Generative AI can produce natural-sounding reviews and news articles closely mimicking real-world user-generated content. This makes the synthetic data highly realistic and suitable for testing or analysis.
- **Customizable Trends and Patterns**:The model can follow specified patterns (e.g., declining sentiment over time, specific issues like customer service or chef changes) to simulate realistic scenarios.
- **Rapid and Cost-Effective Data Generation**: Generating synthetic data eliminates the need for manual data collection and annotation, reducing both time and costs.



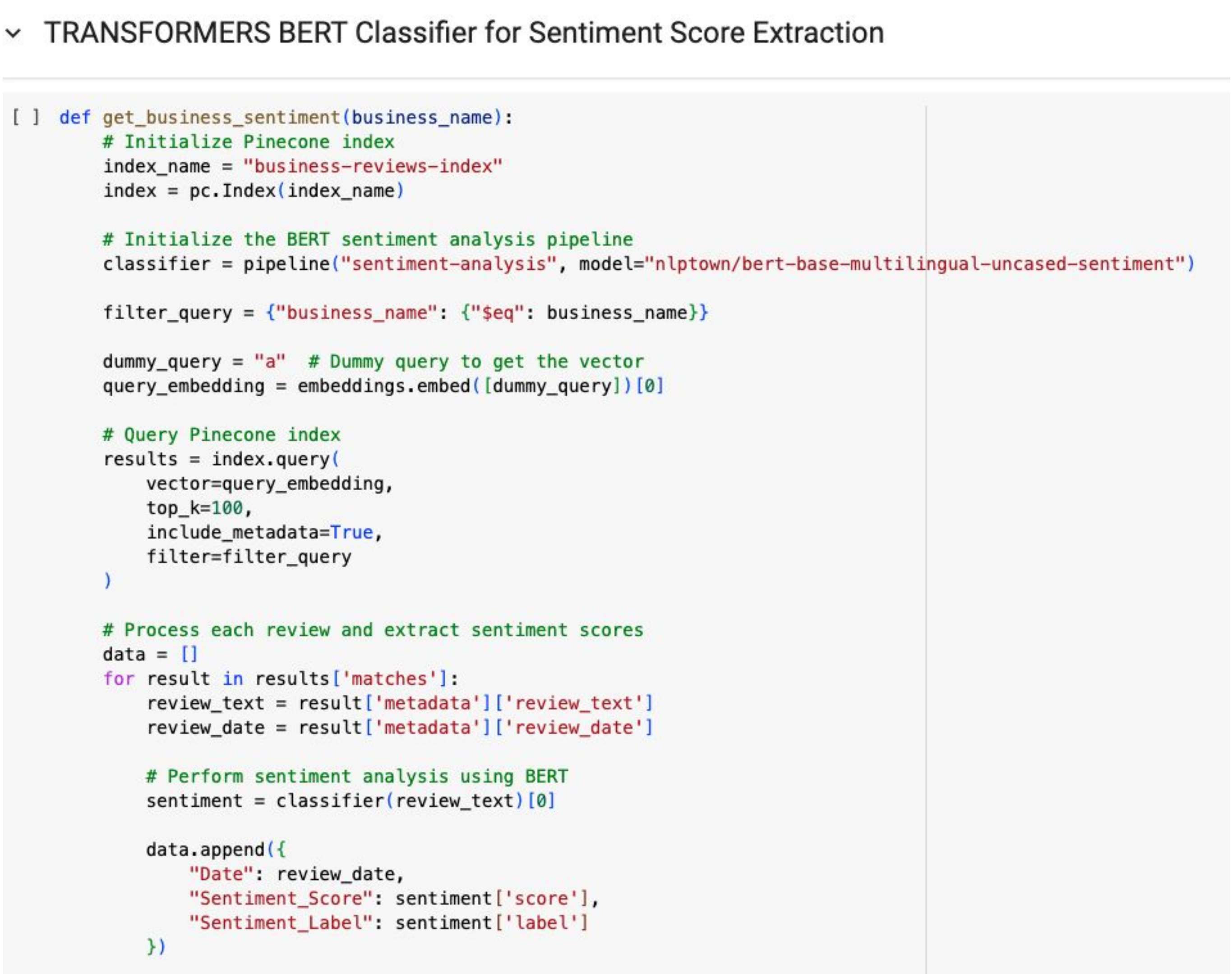
Store JSON Review/News data to PineCone Vector Database

```python
index_name = "business-reviews-index"

existing_indexes = [index_info["name"] for index_info in pc.list_indexes()]
if index_name not in existing_indexes:
    pc.create_index(
        name=index_name,
        dimension=768,
        metric="cosine",
        spec=ServerlessSpec(cloud="aws", region="us-east-1"),
    )
    while not pc.describe_index(index_name).status["ready"]:
        time.sleep(1)
# Load the reviews data from the JSON file
with open('/content/business_reviews_updated.json', 'r') as f:
    reviews_data = json.load(f)

index = pc.Index(index_name)
# Prepare the data and for Pinecone
upserts = []
for business in reviews_data:
    business_name = business['name']
    for review in business['reviews']:
        review_text = review['review_text']
        review_source = review['source']
        review_date = review['date']
        # Generate the embedding for the review text
        review_embedding = embeddings.embed([review_text])[0]  # Get the first embedding from the list
        # Create a unique ID for each review (business_name + review_date)
        review_id = f"{business_name}-{review_date}"
        # Prepare the data for upserting into Pinecone
        upserts.append({
            "id": review_id,  # Unique ID
            "values": review_embedding,  # Embedding vector
            "metadata": {
                "business_name": business_name,
                "review_source": review_source,
```

RAG and Prompt Engineering

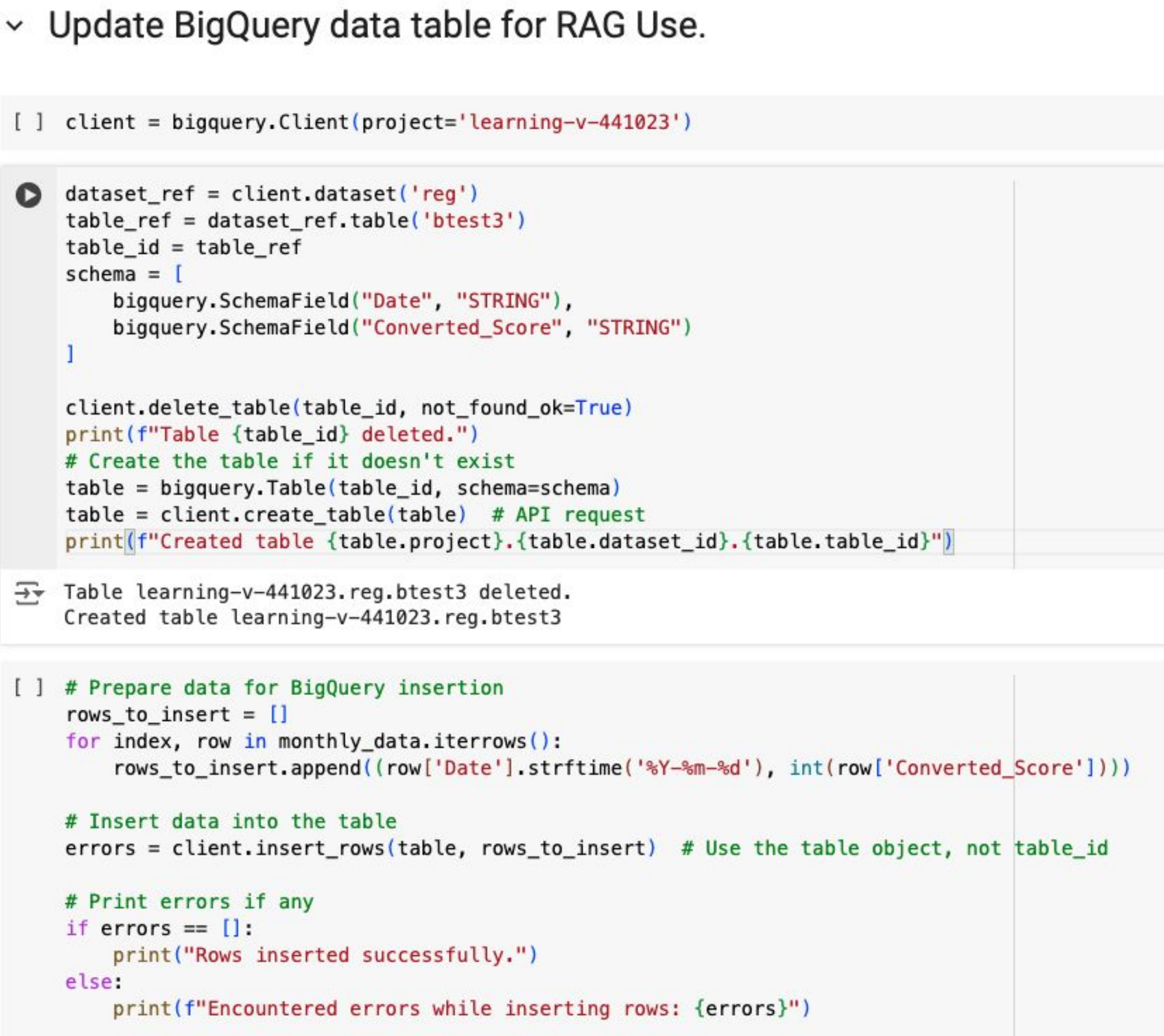


- User asks specific questions about the business to understand the risk: How this business is doing in last 12 months ?
- All data around that business is extracted and send to BERT for sentiment score creation.
- Context specific content is extracted and fed into engineering prompt
- Time Series data extracted and embedded in the prompt engineering process



TRANSFORMERS BERT Classifier for Sentiment Score Extraction

```
def get_business_sentiment(business_name):
    # Initialize Pinecone index
    index_name = "business-reviews-index"
    index = pc.Index(index_name)

    # Initialize the BERT sentiment analysis pipeline
    classifier = pipeline("sentiment-analysis", model="nlptown/bert-base-multilingual-uncased-sentiment")

    filter_query = {"business_name": {"$eq": business_name}}

    dummy_query = "a"  # Dummy query to get the vector
    query_embedding = embeddings.embed([dummy_query])[0]

    # Query Pinecone index
    results = index.query(
        vector=query_embedding,
        top_k=100,
        include_metadata=True,
        filter=filter_query
    )

    # Process each review and extract sentiment scores
    data = []
    for result in results['matches']:
        review_text = result['metadata']['review_text']
        review_date = result['metadata']['review_date']

        # Perform sentiment analysis using BERT
        sentiment = classifier(review_text)[0]

        data.append({
            "Date": review_date,
            "Sentiment_Score": sentiment['score'],
            "Sentiment_Label": sentiment['label']
        })
```



Update BigQuery data table for RAG Use.

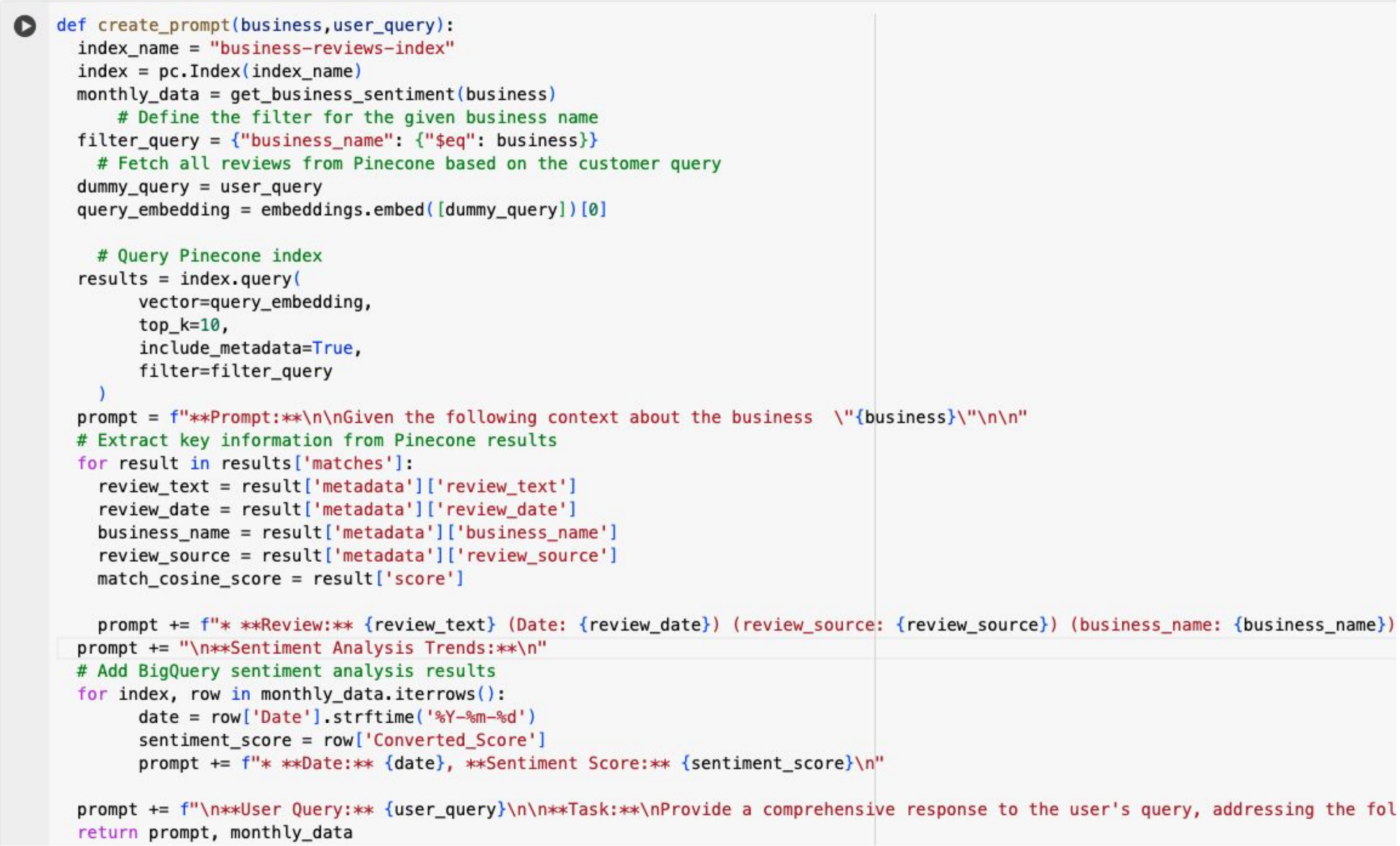
```
[ ] client = bigquery.Client(project='learning-v-441023')
```

```
dataset_ref = client.dataset('reg')
table_ref = dataset_ref.table('btest3')
table_id = table_ref
schema = [
    bigquery.SchemaField("Date", "STRING"),
    bigquery.SchemaField("Converted_Score", "STRING")
]

client.delete_table(table_id, not_found_ok=True)
print(f"Table {table_id} deleted.")
# Create the table if it doesn't exist
table = bigquery.Table(table_id, schema=schema)
table = client.create_table(table)  # API request
print(f"Created table {table.project}.{table.dataset_id}.{table.table_id}")
```

```
Table learning-v-441023.reg.btest3 deleted.
Created table learning-v-441023.reg.btest3
```

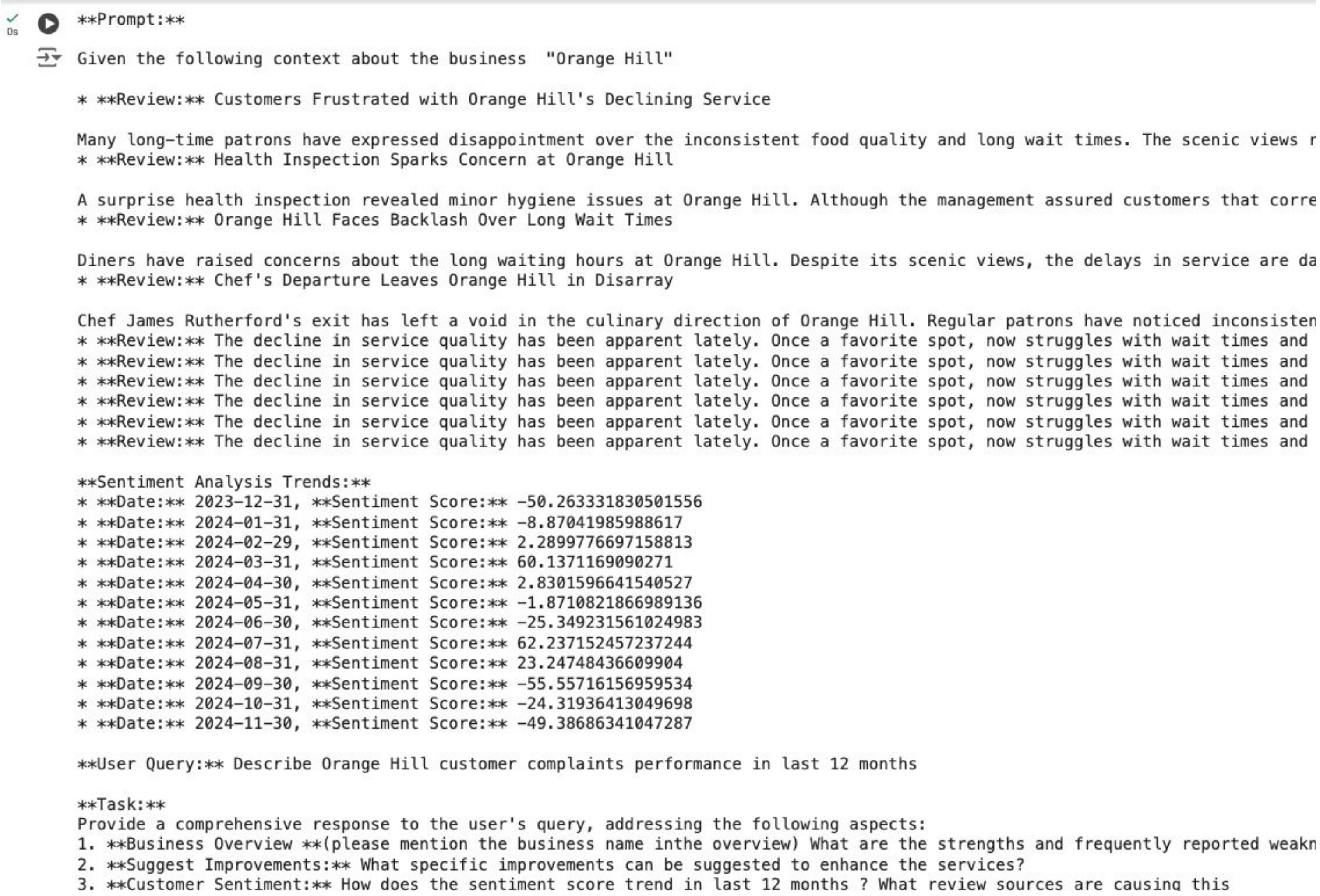
```
[ ] # Prepare data for BigQuery insertion
rows_to_insert = []
for index, row in monthly_data.iterrows():
    rows_to_insert.append((row['Date'].strftime('%Y-%m-%d'), int(row['Converted_Score'])))

# Insert data into the table
errors = client.insert_rows(table, rows_to_insert)  # Use the table object, not table_id

# Print errors if any
if errors == []:
    print("Rows inserted successfully.")
else:
    print(f"Encountered errors while inserting rows: {errors}")
```

Build Context Relevant Prompt

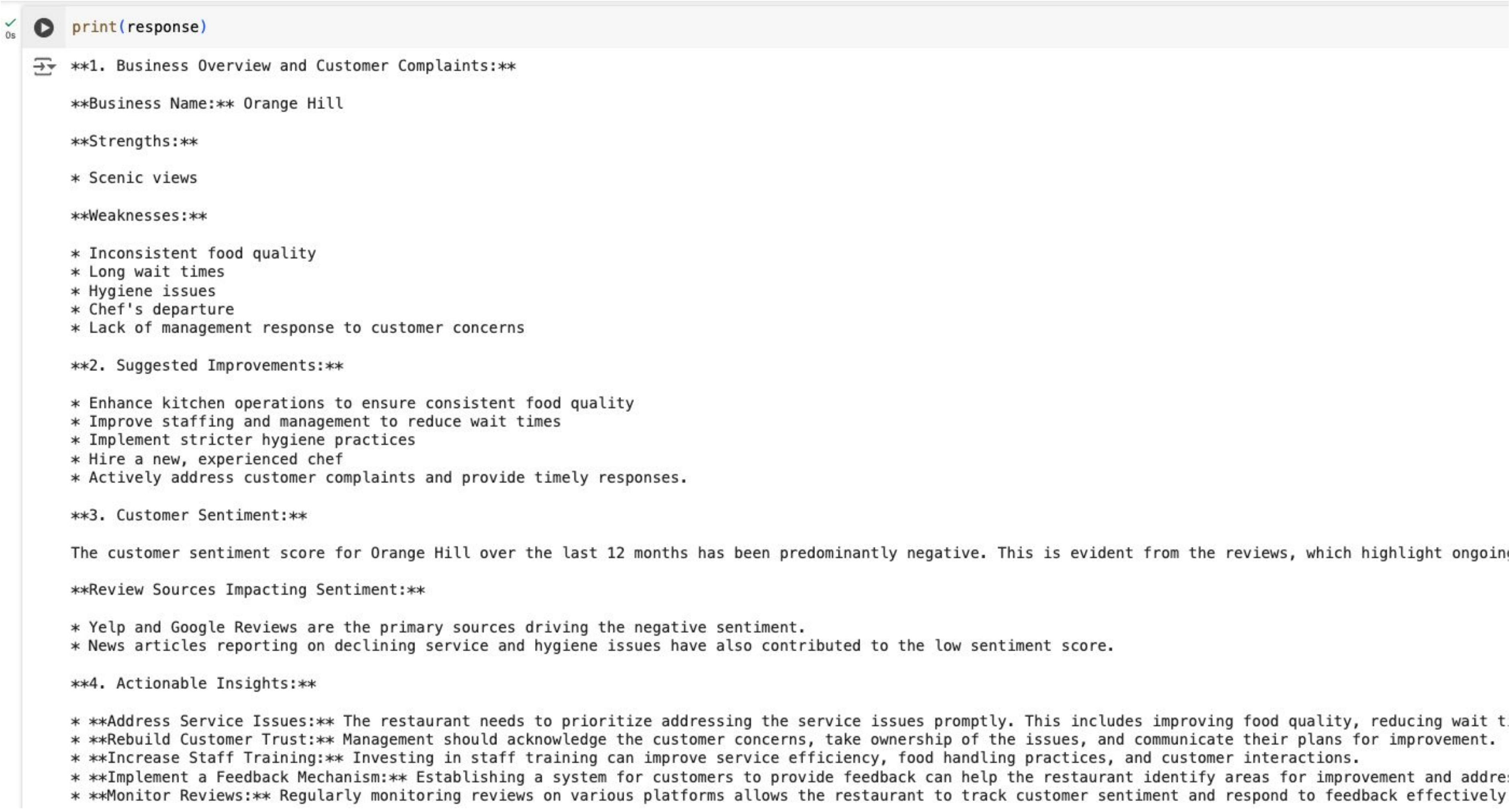


```
def create_prompt(business,user_query):
 index_name = "business-reviews-index"
 index = pc.Index(index_name)
 monthly_data = get_business_sentiment(business)
   # Define the filter for the given business name
 filter_query = {"business_name": {"$eq": business}}
  # Fetch all reviews from Pinecone based on the customer query
 dummy_query = user_query
 query_embedding = embeddings.embed([dummy_query])[0]

  # Query Pinecone index
 results = index.query(
    vector=query_embedding,
    top_k=10,
    include_metadata=True,
    filter=filter_query
  )
 prompt = f"**Prompt:**\n\nGiven the following context about the business  \"{business}\"\n\n"
 # Extract key information from Pinecone results
 for result in results['matches']:
  review_text = result['metadata']['review_text']
  review_date = result['metadata']['review_date']
  business_name = result['metadata']['business_name']
  review_source = result['metadata']['review_source']
  match_cosine_score = result['score']

  prompt += f"* **Review:** {review_text} (Date: {review_date}) (review_source: {review_source}) (business_name: {business_name})
 prompt += "\n**Sentiment Analysis Trends:**\n"
 # Add BigQuery sentiment analysis results
 for index, row in monthly_data.iterrows():
    date = row['Date'].strftime('%Y-%m-%d')
    sentiment_score = row['Converted_Score']
    prompt += f"* **Date:** {date}, **Sentiment Score:** {sentiment_score}\n"

 prompt += f"\n**User Query:** {user_query}\n\n**Task:**\nProvide a comprehensive response to the user's query, addressing the fol
 return prompt, monthly_data
```

Engineered Prompt

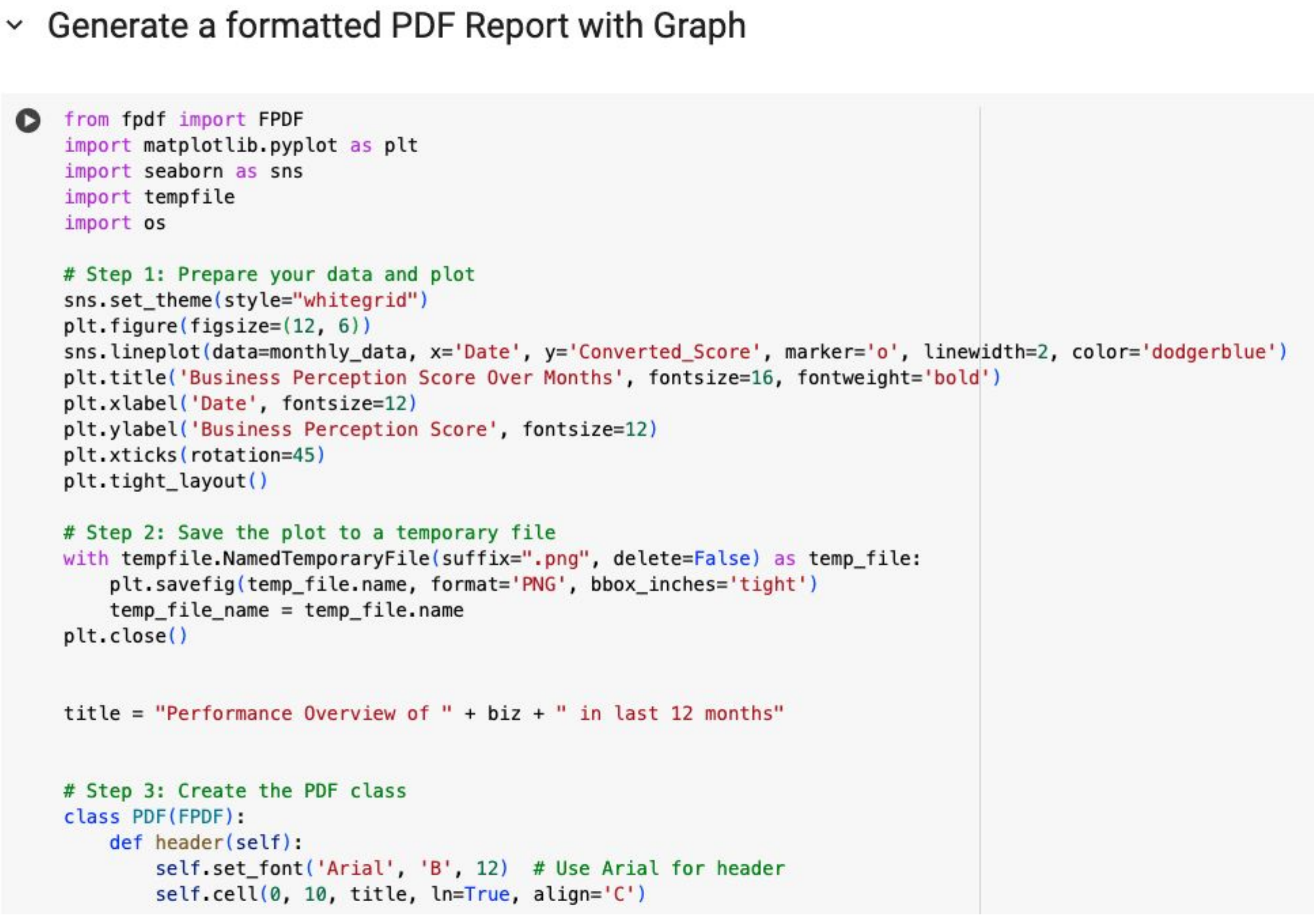


```
**Prompt:**

Given the following context about the business  "Orange Hill"

* **Review:** Customers Frustrated with Orange Hill's Declining Service

Many long-time patrons have expressed disappointment over the inconsistent food quality and long wait times. The scenic views r
* **Review:** Health Inspection Sparks Concern at Orange Hill

A surprise health inspection revealed minor hygiene issues at Orange Hill. Although the management assured customers that corre
* **Review:** Orange Hill Faces Backlash Over Long Wait Times

Diners have raised concerns about the long waiting hours at Orange Hill. Despite its scenic views, the delays in service are da
* **Review:** Chef's Departure Leaves Orange Hill in Disarray

Chef James Rutherford's exit has left a void in the culinary direction of Orange Hill. Regular patrons have noticed inconsisten
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and
* **Review:** The decline in service quality has been apparent lately. Once a favorite spot, now struggles with wait times and

**Sentiment Analysis Trends:**
* **Date:** 2023-12-31, **Sentiment Score:** -50.263331830501556
* **Date:** 2024-01-31, **Sentiment Score:** -8.87041985988617
* **Date:** 2024-02-29, **Sentiment Score:** 2.2899776697158813
* **Date:** 2024-03-31, **Sentiment Score:** 60.1371169090271
* **Date:** 2024-04-30, **Sentiment Score:** 2.8301596641540527
* **Date:** 2024-05-31, **Sentiment Score:** -1.8710821866989136
* **Date:** 2024-06-30, **Sentiment Score:** -25.349231561024983
* **Date:** 2024-07-31, **Sentiment Score:** 62.237152457237244
* **Date:** 2024-08-31, **Sentiment Score:** 23.24748436609904
* **Date:** 2024-09-30, **Sentiment Score:** -55.55716156959534
* **Date:** 2024-10-31, **Sentiment Score:** -24.31936413049698
* **Date:** 2024-11-30, **Sentiment Score:** -49.38686341047287

**User Query:** Describe Orange Hill customer complaints performance in last 12 months

**Task:**
Provide a comprehensive response to the user's query, addressing the following aspects:
1. **Business Overview **(please mention the business name inthe overview) What are the strengths and frequently reported weakn
2. **Suggest Improvements:** What specific improvements can be suggested to enhance the services?
3. **Customer Sentiment:** How does the sentiment score trend in last 12 months ? What review sources are causing this
```



```python
print(response)
```

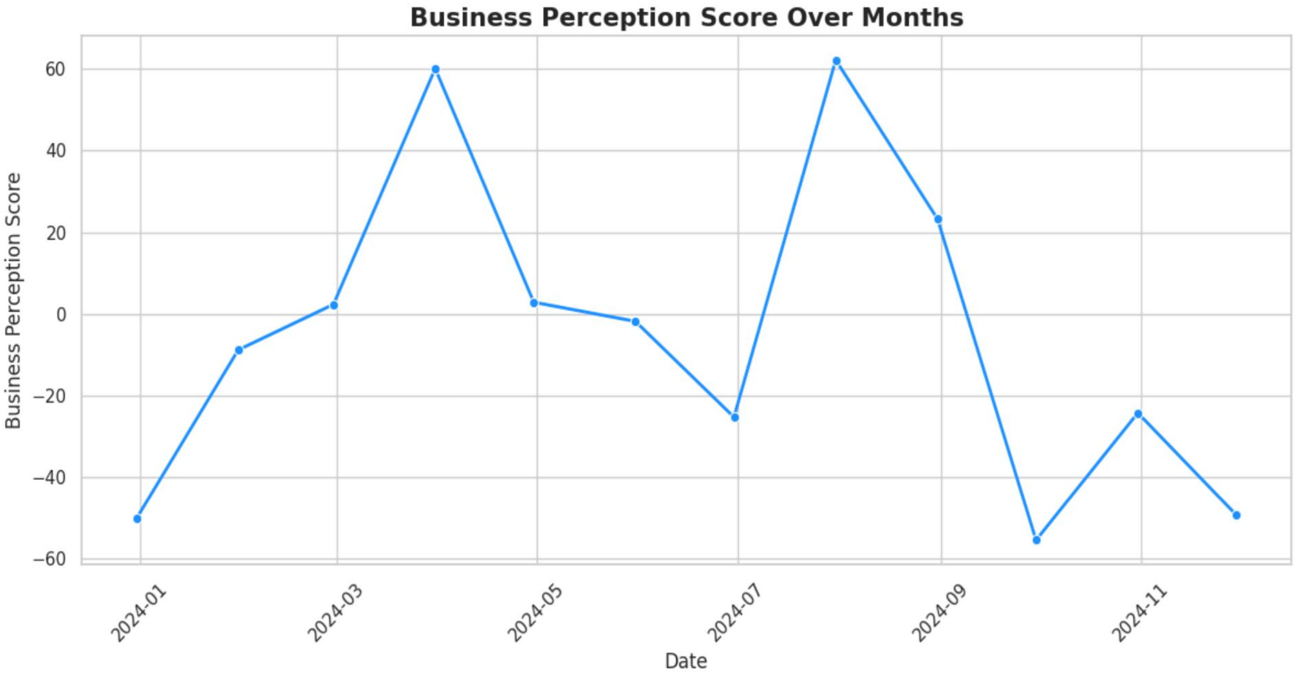
```
**1. Business Overview and Customer Complaints:**

**Business Name:** Orange Hill

**Strengths:**

* Scenic views

**Weaknesses:**

* Inconsistent food quality
* Long wait times
* Hygiene issues
* Chef's departure
* Lack of management response to customer concerns

**2. Suggested Improvements:**

* Enhance kitchen operations to ensure consistent food quality
* Improve staffing and management to reduce wait times
* Implement stricter hygiene practices
* Hire a new, experienced chef
* Actively address customer complaints and provide timely responses.

**3. Customer Sentiment:**

The customer sentiment score for Orange Hill over the last 12 months has been predominantly negative. This is evident from the reviews, which highlight ongoin

**Review Sources Impacting Sentiment:**

* Yelp and Google Reviews are the primary sources driving the negative sentiment.
* News articles reporting on declining service and hygiene issues have also contributed to the low sentiment score.

**4. Actionable Insights:**

* **Address Service Issues:** The restaurant needs to prioritize addressing the service issues promptly. This includes improving food quality, reducing wait t
* **Rebuild Customer Trust:** Management should acknowledge the customer concerns, take ownership of the issues, and communicate their plans for improvement.
* **Increase Staff Training:** Investing in staff training can improve service efficiency, food handling practices, and customer interactions.
* **Implement a Feedback Mechanism:** Establishing a system for customers to provide feedback can help the restaurant identify areas for improvement and addre
* **Monitor Reviews:** Regularly monitoring reviews on various platforms allows the restaurant to track customer sentiment and respond to feedback effectively
```



Generate a formatted PDF Report with Graph

```python
from fpdf import FPDF
import matplotlib.pyplot as plt
import seaborn as sns
import tempfile
import os

# Step 1: Prepare your data and plot
sns.set_theme(style="whitegrid")
plt.figure(figsize=(12, 6))
sns.lineplot(data=monthly_data, x='Date', y='Converted_Score', marker='o', linewidth=2, color='dodgerblue')
plt.title('Business Perception Score Over Months', fontsize=16, fontweight='bold')
plt.xlabel('Date', fontsize=12)
plt.ylabel('Business Perception Score', fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()

# Step 2: Save the plot to a temporary file
with tempfile.NamedTemporaryFile(suffix=".png", delete=False) as temp_file:
    plt.savefig(temp_file.name, format='PNG', bbox_inches='tight')
    temp_file_name = temp_file.name
plt.close()


title = "Performance Overview of " + biz + " in last 12 months"


# Step 3: Create the PDF class
class PDF(FPDF):
    def header(self):
        self.set_font('Arial', 'B', 12)  # Use Arial for header
        self.cell(0, 10, title, ln=True, align='C')
```

Performance Overview of Orange Hill in last 12 months



1. Business Overview and Customer Complaints:

Business Name: Orange Hill

2. Suggested Improvements:

* Enhance kitchen operations to ensure consistent food quality

* Improve staffing and management to reduce wait times

* Implement stricter hygiene practices

* Hire a new, experienced chef

* Actively address customer complaints and provide timely responses.

3. Customer Sentiment:

The customer sentiment score for Orange Hill over the last 12 months has been predominantly negative. This is evident from the reviews, which highlight ongoing issues with food quality, wait times, and hygiene. The sentiment score trend shows a sharp decline in recent months, indicating a