# Homework 1 Implementation Summary

Xinyi Zhang

February 2, 2026

## 1 Objective

The primary objective of this project is to automate the auditing of retail receipts using the Gemini 2.5 Flash model. The system extracts gross prices, identifies negative values such as discounts and app upgrades, and calculates the original subtotal before price reductions[cite: 295, 296, 298].

## 2 System Architecture & Workflow

### 2.1 Environment & Data Acquisition

The project utilizes the `langchain_google_genai` library for model interaction[cite: 5, 84]. Data is acquired by downloading `receipts.zip` from Google Drive via `gdown`[cite: 29, 35, 37]. The dataset consists of seven receipt images extracted into the local environment[cite: 43, 60].

### 2.2 Image Processing Pipeline

To facilitate the multi-modal API call, two helper functions were implemented[cite: 61, 62]:

- **image_to_base64**: Converts JPG images into Base64 encoded strings[cite: 64, 73].
- **get_image_data_url**: Constructs a valid Data URL with appropriate MIME types for the Gemini API[cite: 69, 77].

### 2.3 Model Configuration

The system uses the `gemini-2.5-flash` model with a `temperature=0` to ensure deterministic, high-precision auditing[cite: 87, 88]. Authentication is handled through Colab Secrets using `VERTEX_API_KEY`[cite: 23, 28].

## 3 Audit Implementation

The audit logic is driven by a detailed prompt that defines the model as a "high-precision retail auditor"[cite: 294]. The system is specifically instructed to monitor critical items:

- **Receipt 2**: Capture -$10.00 and -$20.00 app upgrades[cite: 300].
- **Receipt 4**: Account for 5% OFF (-$28.53) and app upgrades (-$30.00)[cite: 300].
- **Receipt 7**: Capture -$11.00 packaging discounts and -$36.00 bulk saves[cite: 301].

# 4 Results & Validation

The model aggregated the results across all processed receipts[cite: 304, 310].

## 4.1 Final Audit Summary

| Metric | Value |
|---|---|
| TOTAL_PAID | $1974.49 [cite: 385] |
| TOTAL_ORIGINAL | $2346.20 [cite: 386] |

## 4.2 Evaluation

The results were validated against ground truth costs using the `test_query` function[cite: 393, 402]. The captured results ($Q1 = 1974.49, Q2 = 2346.2$) passed the assertion test within the $2.00 allowed margin[cite: 387, 401].

# 5 Conclusion

The implementation successfully demonstrates the capability of multi-modal LLMs to perform complex document parsing and financial reconstruction from visual inputs[cite: 312, 320].