3 Large Language Model (LLM) Business Applications

1. *Language Translation*
2. *Product Recommendation*
3. *AI-Powered LLMs*

LLM Application 1: Language Translation

Application: Google Translate (Google Cloud Translation API)

Company: Google (Alphabet Inc.)

Industry: Technology / Global Business Services

Title, References Summary of the Use Case

Title: Google Cloud Translation API (LLM-Enhanced) for Enterprise Communication

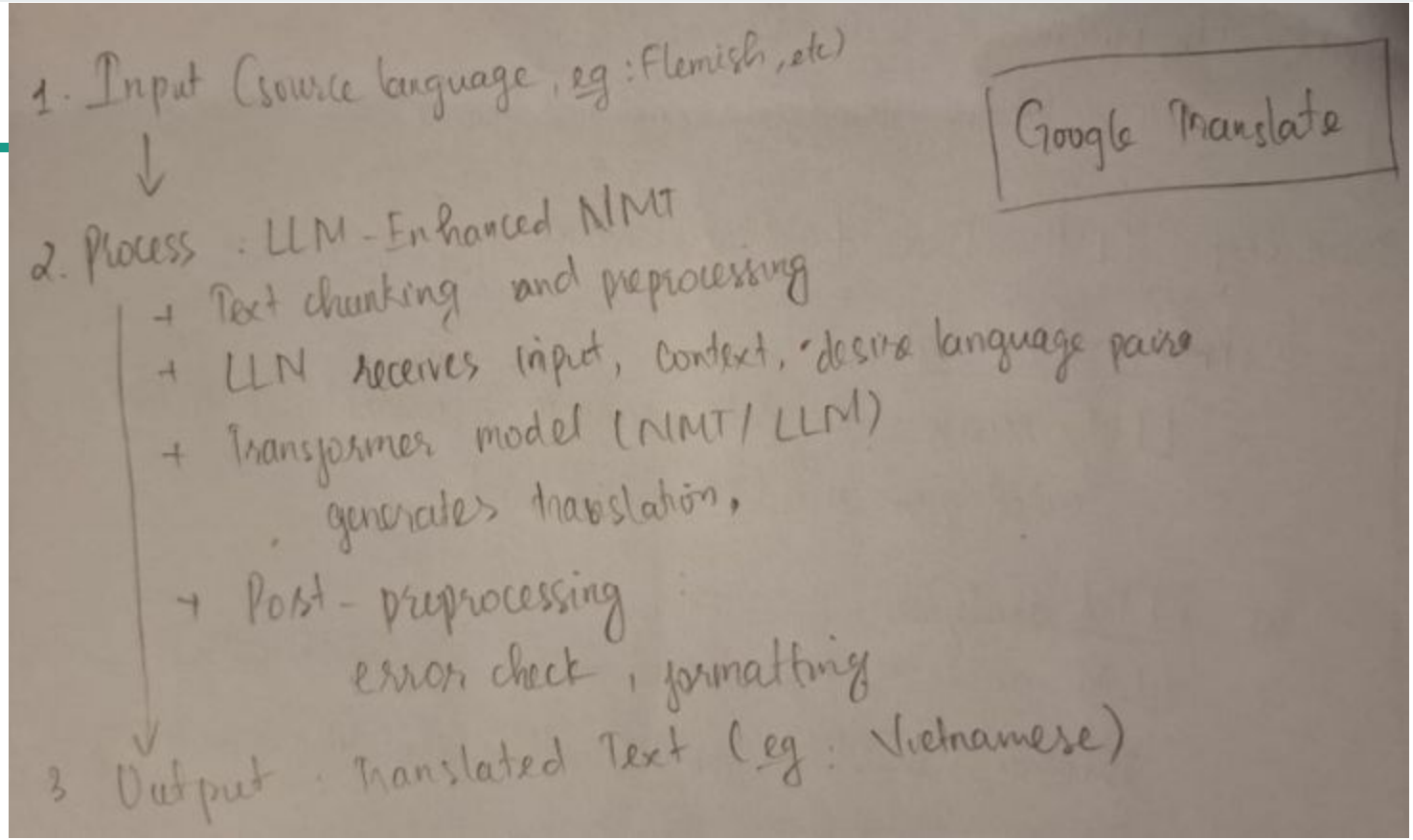
References (URLs): - Google Cloud Translation API Documentation (e.g., Cloud Translation - Google Cloud)
- Industry articles discussing LLM integration in NMT (e.g., academic papers, tech blogs)

- **What it does:** Provides highly accurate, real-time, and batch translation services for over 100 languages. LLMs enhance the core Neural Machine Translation (NMT) with better contextual awareness, fluency, and the ability to retain document formatting.
- **The problem it solves:** Breaking down global language barriers for businesses, enabling real-time multilingual customer support, localizing content at scale (websites, documents), and streamlining international legal or financial document review.
- **Key business value/impact:** Facilitates global market expansion, improves customer satisfaction (through real-time multilingual support), and significantly reduces the cost and time of human translation/localization.

My Guess of What the Key Prompt(s) Look Like

System Prompt Guess (for a Custom Model Fine-Tune): "You are an expert financial services translator. Your task is to translate sensitive legal and financial documents from Spanish to English. You must retain all original formatting (e.g., bullet points, table structure) and use specific terminology found in the provided glossary. Ensure a highly formal, professional tone." **User Prompt Guess (API Call): (Input text to translate)**

But I need to pay for the google cloud setup so I have not tried in real life.


1. Input (source language, eg: Flemish, etc)
Google Translate
2. Process : LLM - Enhanced NMT
+ Text chunking and preprocessing
+ LLM receives input, context, desire language pairs
+ Transformer model (NMT / LLM)
generates translation,
+ Post - preprocessing
error check, formatting
3 Output : Translated Text (eg : Vietnamese)

LLM Application 2: Product Recommendation

Application: The Amazon Store's AI-Powered Recommendation Engine

Company: Amazon

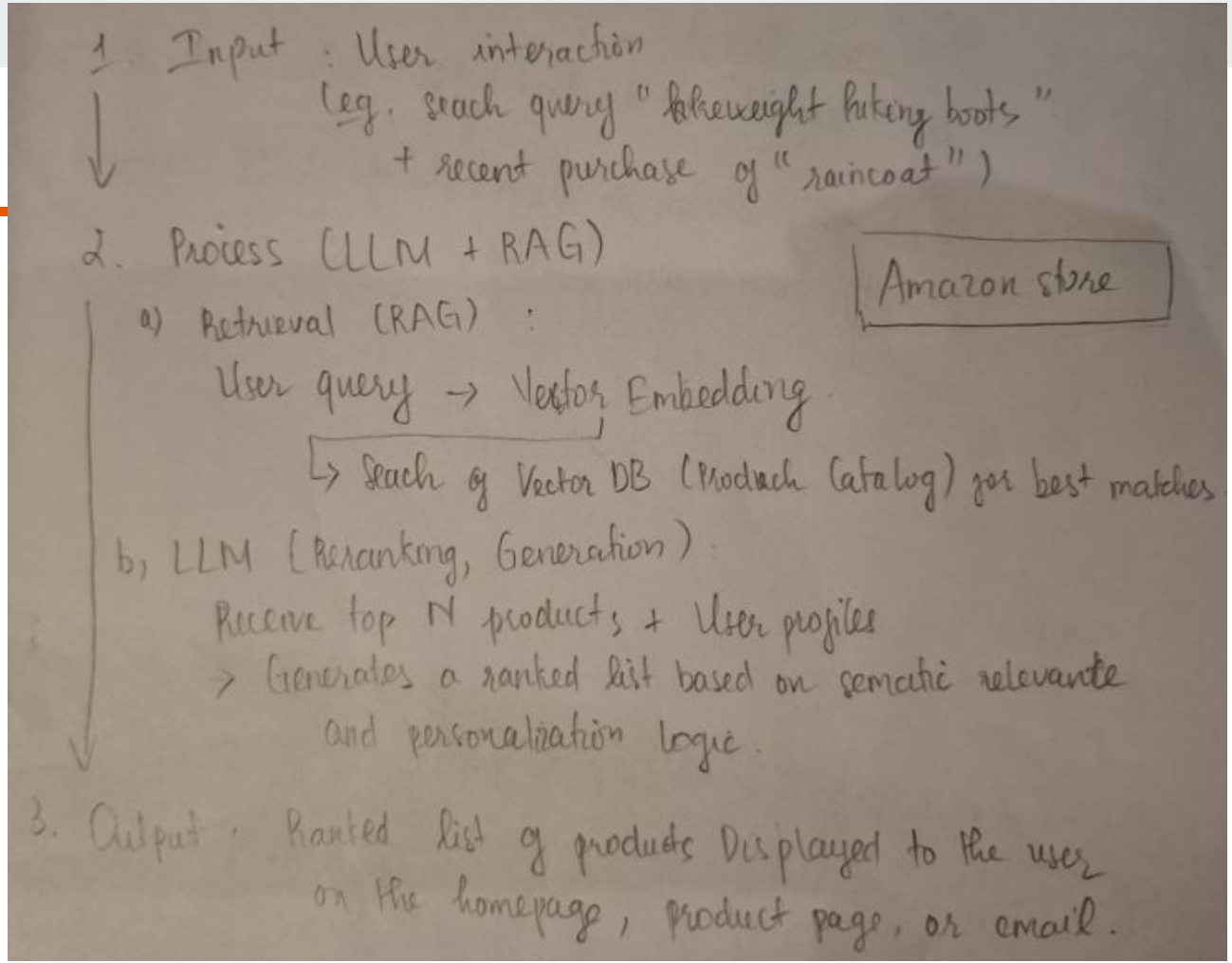
Industry: E-commerce / Retail

Title, References Summary of the Use Case

Title: Amazon's LLM-Enhanced Product Discovery and Recommendation Engine

References (URLs): - AWS Machine Learning Blog (e.g., Amazon Personalize and Generative AI discussions) - Articles on Amazon's recommendation system and RAG/LLM integration (e.g., tech papers, industry analysis) - Amazon.com product page UI screenshot for recommendation placement

- **What it does:** The system analyzes a user's past behavior (purchases, views, searches) and combines it with product attributes (from text descriptions, reviews) to predict and display highly relevant product suggestions ("Customers who bought this also bought..."). LLMs enhance this by understanding nuanced queries and complex product descriptions, improving *why* a product is recommended.
- **The problem it solves:** Information overload and low conversion rates in a massive catalog. It solves the "paradox of choice" by presenting a curated, personalized selection.
- **Key business value/impact:** Drives significant revenue (estimated to be over 35% of sales), increases Average Order Value (AOV), and boosts customer loyalty and engagement.


1. Input : User interaction
(eg. seach query "lightweight hiking boots"
+ recent purchase of "raincoat")
2. Process (LLM + RAG)
Amazon store
a) Retrieval (RAG) :
User query → Vector Embedding
↳ Seach of Vector DB (Product Catalog) for best matches
b) LLM (Reranking, Generation) :
Receive top N products + User profiles
→ Generates a ranked list based on semantic relevante
and personalization logic.
3. Output : Ranked list of products Displayed to the user
on the homepage, product page, or email.

LLM Application 3: AI-Powered LLMs

Application: AI-Powered Customer/Driver Support and Metadata Generation

Company: Grab

Industry: Mobility / Delivery / Financial Services (Super-App)

Title, References Summary of the Use Case

Title: Grab's LLM-Kit for Enterprise Data Classification and Customer Support Automation

References (URLs): - Grab Tech Blog (e.g., "Supercharging LLM application development with LLM-Kit") - Articles discussing LLM use in Super-Apps for data management or customer service - ZenML LLMOps Database (Grab: LLM-Powered Data Classification System for Enterprise-Scale Metadata Generation)

- **What it does:** Grab uses its internal LLM framework (LLM-Kit) to accelerate the deployment of many AI applications. One key application is **LLM-Powered Data Classification**, where the LLM analyzes massive, unstructured data (e.g., logs, documents) to automatically generate metadata, flagging sensitivity (PII) and classification, ensuring compliance and data governance. It also powers customer/driver service chatbots.
- **The problem it solves:** Manual classification of Petabyte-scale, sensitive data is inefficient, inconsistent, and a compliance risk. Automating this frees up human effort and ensures data safety at scale.
- **Key business value/impact:** Enhances data governance and security compliance; significantly increases the efficiency of engineers and data scientists (via LLM-Kit); and lowers operational costs by automating support tasks.

1. Input : Unstructured Data Stream
(new database Table / Field, internal Doc)

2. Processing : LLM - Powered Classification

Grab

a) Prompt / Tool Call :
- LLM receives the data sample n context via a tool called from a LLM - framework.

b) LLM analysis.
- LLM analyzes text based on an enforced output schema (JSON) for the attribute like sensitivity, Domain, and PII presence.

c) Structured Output :
- LLM generates a structured JSON output w the classification

3. Output : Metadata stored in Data Catalog / Vault, triggering security and access controls.