CSC 510 - SE Date: 09/21/2025

Project 1d1

Harsha Puvvadi, Samarth Shah, Jai Ramani, Pratham Patel

1. **What are the pain points in using LLMs?**
   - **Incomplete or unstructured outputs:** In project 1a1, even after providing the output structure of the use cases, the LLMs couldn't stick with the structure when generating use cases. It often required a significant amount of manual restructuring.
   - **Bias toward certain stakeholders:** NotebookLM repeatedly overemphasized WIC as a central stakeholder, even though WIC was only listed as one of several regulators to monitor for compliance.
   - **Over-extension of scope:** Perplexity and DeepSeek sometimes generated futuristic or out-of-scope ideas, which, while creative, were not practical for an MVP design.
   - **Limited analysis:** Gemini struggled with external link analysis and required significant prompt scaffolding to stay useful. It limited the requirements design to only the key stakeholders, such as User, Staff, and Admin.

2. **Any surprises? E.g. different conclusions from LLMs?**
   - **RAG-based system contrast:** NotebookLM initially showed strong WIC bias, it took several attempts to redirect it to the main stakeholders, while the professor's RAG code provided on-point, health-focused stakeholder lists with far less bias.
   - **Different emphases across LLMs:** The most surprising part was Gemini, even with the "Pro" model, at best, it was providing mediocre results. Gemini cannot think in "out of the box" situations unless specifically prompted to do so. Even with "Deep Research" mode, it was a lot more verbose with redundant content.
   - **Structured & Chain-of-Thought prompting:** LLMs performed best when given a clear structure, which narrowed their focus and led to more relevant, usable outputs. Chain-of-Thought prompting further improved results by guiding the model step-by-step, allowing it to explore details deeply while staying on track.

3. **What worked best?**
   - **Chain-of-Thought prompting:** Step-by-step reasoning consistently turned shallow lists into structured, logical, and detailed outputs. It reduced bias toward single stakeholders and kept models like ChatGPT and Gemini focused on practical, in-scope requirements.
   - **Strategic scoping with role-based framing:** Breaking prompts into categories such as Customer, Staff, and Admin ensured balanced coverage across perspectives. Adding constraints kept outputs grounded, preventing models like DeepSeek from over-engineering solutions.
   - **Pre-processed context input:** Curating and summarizing external resources before feeding them to the LLM significantly improved relevance. This was especially helpful for models like Gemini, which struggled with raw, unstructured inputs.

4. **What worked worst?**
   - **Zero-shot prompting alone**: It yielded good results, but it had to be well defined. Without constraints it did not give out the expected results. Another problem with zero shot was it often hallucinated the results and this problem was prevalent with Perplexity.

- **Relying on model creativity without constraints**: DeepSeek produced highly imaginative but impractical use cases that added noise rather than clarity. The LLMs on their own had varying scopes which often did not align with the requirements of the project in the initial phase.
- **Unbounded web integration (Perplexity)**: Perplexity's web search is great when you want more generalized answers that don't stick to a certain fixed context, but it falls apart when you give it your own context. It includes way too many unnecessary details from random, and in many cases unverified sources. This does not change even if you try the many different models available in Perplexity's Pro version.

5. **What pre-post processing was useful for structuring the import prompts, then summarizing the output?**

   Pre-processing:
   - Dividing use cases into role-based categories before prompting.
   - Supplying structured context or data snippets when external link analysis was unreliable.
   - Explicit constraints (eg: "Stay within MVP scope," "Limit to three stakeholder categories").
   - Creating a structured prompt setup (prompting not just with the question, but with clear category distinctions, eg: giving some initial context first, then some examples (few shot), then giving the actual question prompt along with an expected output format).
   - **Preprocessing would greatly benefit from a tool such as "Prompt Review". This tool would review your prompts to LLMs and recommend changes to make the responses better. Basically using LLMs to review existing prompts and improve future LLM responses.**

   Post-processing:
   - Consolidating redundant use cases (e.g., folding "Process Payment" into "Checkout").
   - Summarizing outputs into stakeholder-focused categories for project planning.
   - Filtering futuristic or biased outputs to refine a practical set of use cases.
   - The best method of post-processing was validating the results of each model by comparing them with each other, and filling in the gaps.

6. **Did you find any best/worst prompting strategies?**

   Best:
   - **Chain-of-Thought prompting**: The most consistently effective approach, especially with ChatGPT and Claude, leading to expanded scope and detailed stakeholder insights. The LLMs were able to work back on the given scope and CoT gave more error room to work with getting exactly what we want from the prompt.
   - **Checklists and constraints:** Phrasing prompts with "should/must-not" reduced irrelevant or overly creative outputs. Furthermore defining the exact goal worked well as narrowing the strategy to reducing scope of hallucination. Eg: Provide the output in table format.
   - **Structured role-based prompts:** Explicitly asking for Customer, Staff, and Admin roles ensured balance and coverage. This helped in narrowing the scope and allowed prompting for niche cases that could be important but not mentioned.

   Worst:
   - **Pure zero-shot prompting:** Produced incomplete or poorly scoped results, requiring heavy manual revision. A great strategy only for brainstorming.
   - **Overly open-ended creative prompts:** Encouraged some models (e.g., DeepSeek) to generate out-of-scope ideas that were more distracting than useful. Led to more hallucinations as opposed to giving out important information.