Project 1d1 - Food Delivery System: The Hungry Wolf

Pain Points in Using LLMs

The main challenge was the trade-off between creativity and precision.

- Zero-shot prompting generated diverse and out of the box ideas (e.g., eco-friendly delivery, fraud detection), but the outputs were often vague, incomplete, and inconsistently formatted.
- Careful prompting produced structured and aligned deliverables, but felt rigid and sometimes limited creative exploration.

Another pain point was scope control. ChatGPT stayed close to assignment instructions but lacked flexibility, while Claude risked overscoping by introducing technical or non-functional requirements not relevant to the task. Managing this balance required extra human effort.

Surprises

The biggest surprise was the divergent behavior of both LLMs:

- ChatGPT excelled at compliance and formatting, making it ideal for meeting our expectations.
- Claude added professional depth, such as system-level critiques and realism, though sometimes going beyond the scope.

This contrast showed that different LLMs have complementary strengths — one for precision, the other for depth.

What Worked Best

- Hybrid workflow: Brainstorming with zero-shot prompting, then refining with structured, careful prompts.
- Structured templates: Prompts specifying things, like, use cases should contain preconditions, main flows, subflows, and alternative flows, and ensure usable outputs.
- Model comparison: Using both ChatGPT and Claude helped balance “minimum viable” deliverables with a professional outcome.

What Worked Worst

- Relying on zero-shot prompts for final deliverables: outputs lacked completeness and needed heavy editing.
- Using Claude for strict rubric tasks: valuable insights, but not always aligned with assignment requirements.
- Depending on a single model: each had blind spots that made results less reliable without synthesis.

Pre and Post Processing

- Pre-processing: Designing structured prompts and feeding in prior deliverables (e.g., stakeholder lists) to ground new outputs.
- Post-processing: Synthesizing outputs from different LLMs, filtering out overscoped features, and ensuring consistent formatting across documents.

Best and Worst Prompting Strategies

- Best:
    - Structured, explicit prompts (“Write a use case with Preconditions, Main Flow…”).
    - Stage-based prompting (brainstorm → refine).
    - Comparative prompting across models.
- Worst:
    - Overly broad prompts (“List all possible use cases”) led to bloated, repetitive outputs.
    - Relying only on defaults led to skipped error handling and missed details.