Hackapizza 2025

Team - LSF

Summary

- Problem Description
- Solution Description
  - Design Principles
  - High Level Architecture
  - Offline Phase: Knowledge Base Creation
  - Online Phase: Knowledge Base Querying
- Results & Improvements

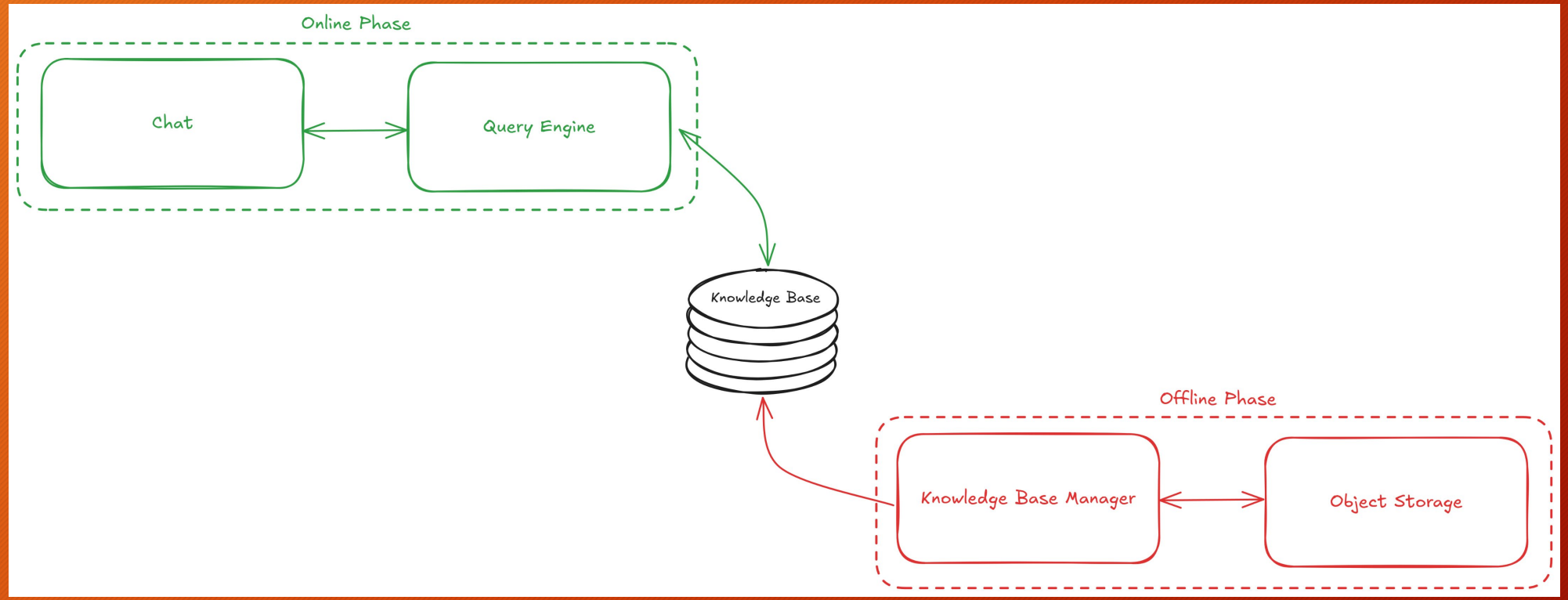
Problem Description

- ***Goal:*** develop a “intergalactic food advisor” that can provide the user with a list of suitable dishes based on a question in natural language, which implicitly contains some constraints.
- ***Basic Setting:*** in the galaxy there are 30 different *restaurants* scattered around 10 different *planets*, each with its *chef* and its unique set of *dishes* (i.e., menu). Every *chef* has their own *cooking* licenses, and every *dish* is prepared with specific *ingredients* and *cooking techniques*.
- ***Additional Info:*** every *cooking technique* requires a set of *licenses* to be performed, some *ingredients* that contain weird *substances* cannot exceed a certain thresholds, and there are three different *orders* to take care of.

Solution Description – Design Principles

- *Make the System as Predictable and Controllable as Possible.*
  - Combine LLMs with Rule-Based Parsing.
  - Map Questions to Predefined Sequences of Actions.
- *Make the System Deployable On-Premises with Limited Resources.*
  - Use “Small“ Open Weights Large Language Models (LLMs).
- *Focus on Data Quality.*
  - Define a Precise Data Model that Suits the Application.

Solution Description – High Level Architecture



Solution Description – Offline Phase (1/2)

- ***Goal:*** creating the Knowledge Base.
- ***Description:*** the *Knowledge Base Manager* (i) reads *menus* and supporting documents (e.g., *cooking manual*, *intergalactic code of conduct*, *dish mappings*, *planets distances*, …) from an **object storage**, (ii) generates for each *dish* a JSON descriptor that represents it, (iii) and saves it in a **document database**.
- ***Notes:*** in this exemplified setting both the **object storage** and the **document database** are folders in our repo, but the switch to acutal tools such as *MinIO* and *MongoDB*, for instance, is simple.

Solution Description – Offline Phase (2/2)

- ***Data Model:*** for each *dish*, we create a descriptor with the following information.
  - **Restaurant**
    - Name
    - Planet
  - **Chef**
    - Name
    - Licenses (Name, Code, Level)
  - **Dish**
    - Code
    - Name
    - Ingredients
    - Techniques (Name, Category, Subcategory)
  - **Orders Info**
    - *Andromeda* Info
    - *Armonisti* Info
    - *Naturalisti* Info

Solution Description – Online Phase (1/2)

- ***Goal:*** querying the Knowledge Base.
- ***Description:*** the *Query Engine* (i) receives a *question* from the user, (ii) "understands" it, (iii) and transforms it in a sequence of filters to apply on the *Knowledge Base* so to extract the *dishes* that respect the question's constraints.
- ***Notes:*** in this exemplified setting we made some assumptions about the types of questions the system can handle. Nevertheless, with some work, one can make the whole process more general and robust.

Solution Description – Online Phase (2/2)

- ***Question Understanding:*** each question is processed to produce a descriptor that strictly parallel that of a dish by applying the following steps.
  - Extract *ingredients* and *techniques* (LLM-Based).
  - Extract *restaurant* and *planet* (Rule-Based).
  - Extract *licenses* (LLM-Based).
  - Extract *orders* (Rule-Based).
- ***Question Execution:*** the content of the *Knowledge Base* is recursively filtered based on the *question* descriptor extracted in the previous step to produce a list of suitable *dishes*.

Solution Description – Results & Improvements

- The results are promising, considering the limited resources we relied upon. In particular, we obtained a **72.5** score with our best submission using the approach described in this presentation.
- Here's some potential improvements:
  - Refine the question understanding and execution mechanisms to produce more accurate questions representations and more reliable sequences of action.
  - Use a more powerful model to answer the question by passing it to a more powerful model with the whole *Knowledge Base* and leveraging the Data Model.