Project Template

PROJECT MARKO -

FEMMa Oracle Description: The research project "Future Electrified Mobile Machines" (FEMMa) focuses on developing electrified and autonomous mobile working machines to support the transition towards a sustainable and carbon-neutral future. As a customer, Marko wants a customized AI assistant that can response to user query about the project accurately and relevantly subjected to the data embedded. As a requisite for the Project’s user interface, the chatbot must have a UI similar to that of ChatGPT with “FEMMa’s log”. At initiating a new conversation, the user can either pre-select the prompts provided on the UI or input a new prompt in the text area. Please find attached the pre-defined prompts decided with the customer: Data Source: https://wiki.eduuni.fi/display/tunifemma/FEMMa+Home The site compromise all the information about the research project FEMMa necessary to be embedded as an external knowledge base on the LLM. It also have hyperlinks to publications in pdf format. Development Plan: INTERNAL DEADLINE FOR THE PROTOTYPE: Start of August, 2024. 1. Main Objectives: What were the main objectives of the research project? 2. Data Collection: How was the research data selected and collected? 3. Analysis Methods: What methods were used for data collection and analysis? 4. Project Timeline: What were the key phases and timeline of the project? 5. Key Findings: What were the main findings of the research? 6. Practical Applications: What are the most significant practical applications of the research findings? 7. Impact on Industry: How can the research findings impact industry practices or policies? 8. Future Utilization: How can the project results be utilized in the future? 9. Societal Impact: What are the societal impacts of the project? 10. Publications: What are the publications of the project? 11. Funding: Who were the funders? Data Collection: Export all the website’s data into multiple pdf files. Building Vector Database: Embedding of the data source into the vector database (handling addition, edition and removal of single/multiple pdf files from the database) Backend Development: Create an assistant, exploit best practice to format prompts with assistant’s instructions + context (and possibly the thread of conversation) to get most accurate response with the source of information. Frontend Development: Use Streamlit to develop the user interface as designed by the customer as attached at the end of this document. Deployment of MVP: Running the application on the University’s server

**2: Project Nick**

Aligning more than 500 different vendors and API solutions is difficult. The current system struggles to manage 500 suppliers, each with different vendors, making it difficult to centralize and compare prices. The complexity of tracking menu costs, resource costs, price changes, and different vendors creates significant challenges. A unified platform is needed to streamline this process, allowing for the extraction and analysis of data from various vendor PDFs. This platform would store the data in a vector format to provide better insights. Nick will provide more PDFs of different vendors to run the experiment smoothly. The current system struggles to manage 500 suppliers, each with different vendors, making it difficult to centralize and compare prices.

### **MVP (Minimum Viable Product)**

The MVP aims to develop a chatbot capable of real-time analysis, providing insights into supplier offerings. The final LLM (Large Language Model) assistant will help managers make informed decisions based on cost analysis, ensuring profitable deals. The first phase targets a release date of July 23rd.

### **Technical Approach**

1. **PDF Extraction:** Obtain PDFs from different vendors of each supplier and extract the content.
2. **Translation:** Translate the extracted data into English for internal processing.
3. **Data Structuring:** Convert the data into a structured format suitable for use with a Large Language Model (LLM).
4. **Storage:** Store the structured data in a vector database like MongoDB.
5. **LLM Integration:** Utilize the structured data to enable the LLM to generate accurate and relevant responses through well-crafted prompts or queries.
6. **Demo:** Aim to deliver a functional demo by the third week of July.

### **Use Cases or Query Prompts**

1. **Ingredient Requirement for Lasagna:** Determine the required ingredients and their quantities for making 10 dishes of lasagna, along with the current rates.
2. **Supplier Rate Comparison for Pasta:** Identify which supplier offers the best rates and deals for making pasta this month.
3. **Recipe and Cost Analysis for Lasagna:** Retrieve the recipe for lasagna, list all necessary ingredients, and calculate the cost of preparing 20 plates for the next day.

### **Future Prospects**

* **API Development:** Provide APIs for the team to upload PDFs and perform detailed insights on historical data and various offers.
* **Enhanced Data Analysis:** Enable deeper analysis of historical data and trends to optimize procurement and cost management.

This approach ensures a streamlined process for managing supplier data, facilitating better decision-making through advanced data analysis and LLM integration.

### **3: Project Description LIRA**

#### **Issue or Problem**

There are thousands of CVs that HR needs to manually analyze, which is time-consuming. AI can resolve this by using agents and RAG-based applications. HR can run a campaign, add a job description, and people will apply for that job. CVs will be filtered based on the criteria, and relevant CVs will be ranked accordingly.

#### **MVP (Minimum Viable Product)**

The MVP aims to develop a web application for parsing CVs, where HR can add a campaign and job criteria. Based on that criteria, CVs will be filtered and ranked. Additionally, an LLM chat assistant bot will provide real-time queries on the records.

#### **Prerequisites**

* Campaign setup
* Job description
* Job acceptance criteria

#### **Technical Approach**

1. **User Upload:** Users upload their CVs.
2. **Parsing:** Parse the CVs according to a base model and specified criteria.
3. **Data Structuring:** Save and return the parsed information in JSON format.
4. **Agent Processing:** Pass the CV output information to a second agent.
5. **Scoring:** The second agent scores the CVs according to the acceptance criteria.
6. **Storage:** Store all relevant and marked CVs in a vector-based database for the RAG-based assistant.
7. **Insights:** Generate insights based on the stored data.

#### **Use Cases**

1. **Top Applicants:** Identify the top five applicants against campaign number 1.

#### **Future Prospects**

* **Real-Time Assistance:** Provide more real-time assistance reports based on the chatbot's interactions.

This project aims to streamline the CV analysis process, making it more efficient and effective through advanced AI and RAG-based solutions.

**Project 4 Tampere**

Objective The goal of the Tampere City RAG (Retrieve and Generate) Application is to create an AI-driven chatbot that answers visitor questions about the Land Use and Construction Act. The chatbot will function in Finnish, both for input questions and output answers. An English version will be considered later. Data Sources The information will be sourced from the following websites, all in Finnish: 1) Link 1: Finnish legislation website containing the Land Use and Construction Act. 2) Link 2: City planning and zoning information for Tampere. 3) Link 3: Information on building permits, and information services in Tampere. Additionally, a relevant FAQ has been provided from the customer for understanding: • https://www.kouvola.fi/asuminen-ja-ymparisto/rakentaminen/rakennusvalvonta/useinkysyttya-pienet-pihalla-tehtavat-tyot-ja-rakennusprojektit/ Technical Approach ▪ Web Scraping: Extract data from the three specified websites. ▪ Translation: Translate the extracted data into English for internal processing (initially, focus on Finnish for the MVP). ▪ Data Structuring: Convert the data into a structured format suitable for use with a Large Language Model (LLM). ▪ Storage: Use a vector database like Pinecone or Qdrant to store the structured data. ▪ LLM Integration: Utilize the structured data to enable the LLM to generate accurate and relevant responses. ▪ Final Deployment: Deploy the AI chatbot on the official Tampere City website (e.g., Tampere.fi). Not now. Example User Questions • Voinko kaataa puun tontiltani?" (Can I cut down a tree from my plot?) • "Entä pensasaita? Tarvitseeko luvan?" (What about a hedge? Need a permit?)