NLP Class Projects (Team of 3–4 Students)

Overview

This document provides step-by-step instructions for the NLP class project. Students must work in teams of 3–4 and follow all steps carefully. The final project should be professional, well-documented, and suitable for GitHub submission.

Step 1 — Make a Group

- Form a group of 3–4 students.
- Choose ONE project from the following list:
  1. Local Embedding + Ollama (gemma3)
  2. OpenAI Embedding + Ollama (gemma3)
  3. Local Embedding + vLLM (gemma3)
  4. OpenAI Embedding + vLLM (gemma3)
  5. AI Agent via MCP
  6. GraphRAG + vLLM (gemma3)
  7. KAG

Step 2 — Understand the Data Source

- Use only ONE PDF document: UET Prospectus.
- The chatbot must answer ONLY department-related questions.
- If a user asks anything else, the chatbot must reply:
  "I only answer department information."

Step 3 — Draw the Architecture Diagram (Before Coding)

Create a simple architecture diagram showing the following components or additional based on your architecture:

1. PDF → Preprocessing
2. Data Storage (Vector Database / Graph / Knowledge Base)
3. Retriever (find relevant text)
4. LLM (gemma3 via Ollama or vLLM)
5. Guardrail (check if the question is department-related)
6. API Layer (FastAPI)
7. GUI (Chat Interface)

You may use draw.io or any similar diagramming tool.

Step 4 — Divide Tasks (Team Responsibilities)

Create a simple task division plan, for example:

- Person 1: PDF reading, cleaning, and chunking
- Person 2: RAG / Graph / Agent logic

- Person 3: Backend API development (FastAPI)
- Person 4: GUI development, testing, and video recording

Step 5 — Build REST API

Develop the backend using FastAPI.
Required endpoint:

- POST /chat → Takes user message and returns an answer with citations.

Step 6 — Build GUI Chatbot

Create a simple chat interface using one of the following:

- Streamlit
- Gradio
- React etc.

The GUI must display:

- User messages
- Chatbot responses

Step 7 — Organize your Project folder structure

Your repository must include:

- requirements.txt
- README.md
- Proper folder structure
- Architecture diagram file

Recommended folder structure:
backend/
frontend/
data/
diagrams/
tests/
requirements.txt
README.md

Step 8 — Testing

Prepare at least 20 test questions:

- 10 department-related questions
- 5 tricky questions
- 5 out-of-scope questions

test the system and save them in:
tests/test_queries.json

Step 9 — Record Video and Presentation

Record a 6–10 minute video covering:

1. Architecture diagram explanation
2. Team task division
3. Live demo with 3 questions:
   - Department-related question
   - Tricky question
   - Out-of-scope question
4. Project repository structure

Add the video link to the README.md file.