

---

PROMPT ENGINEERING — LAB 1

---

📅 Due date: Oct 24th, 2024 before EOD 📅

## Contents

<b>1 Introduction</b>	<b>3</b>
<b>2 Project Scope</b>	<b>3</b>
<b>3 Platform Requirements</b>	<b>3</b>
<b>4 High Level Requirements</b>	<b>3</b>
<b>5 Implementation Steps</b>	<b>4</b>
5.1 Allowed Techniques . . . . .	4
5.2 Preparing the Datasets . . . . .	5
5.3 Create Database Tables . . . . .	5
5.4 Implement an LLM Client . . . . .	5
5.5 Few shot Example Use Cases for the Chatbot . . . . .	6
5.6 Example of CoT Prompting . . . . .	6
5.7 Build a GUI based Chatbot . . . . .	7
5.8 Generate SQL Query . . . . .	7
5.9 Implement Runtime . . . . .	7
5.10 Optional Feature: ReAct . . . . .	8
5.11 Generate the final output . . . . .	8
5.12 Develop the Orchestrator . . . . .	8
5.13 Integrate and Test . . . . .	8
<b>6 Evaluation Criteria</b>	<b>8</b>



# 1 Introduction

The Indian General Elections are among the largest democratic exercises in the world, involving millions of voters and 543 parliamentary constituencies. Analyzing election data can offer powerful insights into voter behavior, party performance, and constituency dynamics. Maharashtra, one of India's key political states, is scheduled to hold its next Assembly elections in November 2024. As the political landscape evolves, parties are looking for actionable insights from previous elections to develop data-driven strategies.










This hackathon focuses on creating a conversational AI-powered chatbot, code named **MahaNeta**, using a Large Language Model (LLM). The chatbot should assist political strategists by analyzing historical data from both the General Elections (2019, 2024) and the 2019 Maharashtra Assembly Elections to generate deep insights and compute relevant analytics. The goal is to help strategists from a political party gain an edge by understanding voting patterns, candidate performance, and electorate behavior.

## 2 Project Scope

The scope of this hackathon includes using any prompt engineering techniques including ReAct and building the end to end workflow involving SQLite database, runtime development and GUI.

The scope doesn't include using other LLM programming techniques such as RAG, Agents or Finetuning. You are not required to implement chat history in version 1, we can extend this to support this feature in the next version.

## 3 Platform Requirements

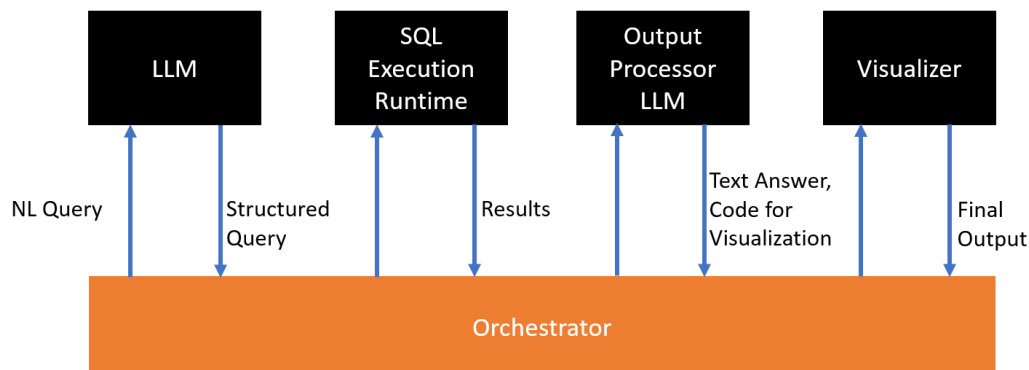
-  **Operating system:**  Windows 11,  Linux, **or**  macOS
-  **GPU:** For macOS, M2 and above silicon, or NVIDIA CUDA supported GPUs for Windows with > 12 GB RAM. You can also use Google Colab. Based on memory and speed requirements, obtain either T4 or A100 GPU.
-  **IDE:** PyCharm, VS Code, or any other Python development system with Anaconda environment.
-  **Frameworks:** Ollama, LMStudio.
-  **LLM:** Llama 3 Family, Google gemma, Microsoft Phi 3 Family.
-  **Web Browser:** Google Chrome.

## 4 High Level Requirements

Students are tasked with developing an LLM-powered chatbot that can provide insights, analytics, and recommendations by using three datasets:

1. **Indian General Elections Dataset for 2019** (including Maharashtra)
2. **Indian General Elections Dataset for 2024** (including Maharashtra, only parliamentary constituency level granule is available)

# General Architecture (Non Agentic, Chain)



Can we use [ReAct prompting](#) here?

Figure 1: High Level Architecture of **MahaNeta**

## 3. Maharashtra 2019 Assembly Elections Dataset

The chatbot should be able to answer questions, provide statistical insights, and make predictions based on historical data, enabling a political strategist to make informed decisions for the upcoming elections.

After completing basic features that constitute an MVP, please focus on handling nuances like differences in political alliances. For example, in 2019 the MVA was not formed and Shivaseena was a single party.

## 5 Implementation Steps

Please refer the Figure 1 that provides the block diagram for your implementation.

### 5.1 Allowed Techniques

To implement **Mahaneta** you use one or more of the following LLM prompt engineering techniques:

- **Zero-shot or Few-shot Learning:** Guide the LLM using minimal or no examples to understand and answer election-based queries.
- **Chain of Thought (CoT):** Break down complex questions into smaller, manageable reasoning steps for better accuracy.
- **Tree of Thought (ToT):** Evaluate different pathways for reasoning to explore multiple answers and insights.

- **ReAct:** Combine reasoning and actions together to perform analytical steps, calculations, and data interpretations.

## 5.2 Preparing the Datasets

- You are provided with following datasets:
  1. **details\_of\_assembly\_segment\_2019.csv:** This csv is already a preprocessed form of the dataset downloaded from ECI website. You can note down the fields and rename is you need them to be. Additional data cleaning may not be required.
  2. **10-Detailed Results.xls:** This file needs to be preprocessed. First save it in xlsx form. Convert to CSV. Please check for any missing or empty fields and fill them using Pandas. Please refer the sample code provided. Change the field names as needed and bring it to a uniform format that can be ingested in to a SQL database. Make sure that the excel sheet looks a strict plain 2 dimensional table without merged cells, extra rows, extra headers and so on. You can delete these manually or can write a simple script.
  3. **eci\_data\_2024.csv:** This dataset contains the data on 2024 general elections for each candidate across all parliamentary constituency. Details at assembly level is not available. You may need to perform some data cleaning that removes extra information in the constituency name, e.g., "Amalapuram (SC) - 7"
- Review the datasets mentioned above, clean them as required, bring them to a form where they can be directly ingested in to a SQL database.

## 5.3 Create Database Tables

- Write a function to save the 2019 General Elections dataset as a sqlite3 db. Name the db as "elections" and "elections" and table name as "elections\_2019"
- Similarly form a schema for 2024 dataset and save them in another table: elections\_2024
- Ingest the dataset of Maharashtra assembly elections results as another table, say, maha\_2019.

## 5.4 Implement an LLM Client

(Can be done in parallel while dataset preparation is going on)

- Run the LLM server using LMStudio as discussed during the earlier hands on. You can use Ollama or Google Colab as well based on your choice. If you use Ollama on Google Colab, you should use ngrok as we need to build a local GUI client using Streamlit.

- Develop the client code `get_completion(prompt)` that takes a prompt as input and returns the output returned by the server. You can reuse the code you might have developed or use the code given to you.
- Test the code by sending some test prompts and checking the results. You can write a suitable prompt that instructs the LLM to behave like a political strategist for a given party (e.g. the ruling party) and ask it to generate say 25 questions that you may want to ask in order to strategize. Your prompt needs to be elaborate and can include few shot, chain of thought examples. You can start with simple examples pertaining to one table and then go on to forming complex questions using the examples given in the next subsection.

## 5.5 Few shot Example Use Cases for the Chatbot

The chatbot should support a political strategist by offering answers and insights into questions such as:

- **Voter Turnout Analysis:** What was the voter turnout in a given assembly constituency in 2019 compared to the parliamentary election turnout?
- **Party Performance:** How did a specific party perform in a particular assembly constituency in the 2019 elections, and how does this compare with the party's performance in the corresponding parliamentary election segment?
- **Demographic Insights:** What is the distribution of voters by category (SC/ST/-General) in key constituencies, and how did different categories vote for the party in the last two elections?
- **Candidate Insights:** How did candidates of a specific age group or gender perform in assembly constituencies won by a particular party? What are the success rates of candidates under different demographic profiles?
- **Swing Analysis:** Which constituencies have seen the biggest swing in vote share between the parliamentary and assembly elections for a specific party?
- **Strategic Forecasting:** Based on historical voting patterns, which assembly constituencies should the party focus on to improve its chances in the upcoming 2024 elections?
- **Winning Margins:** In which constituencies were the winning margins tightest in the 2019 Assembly elections, and how does this compare with the 2019 parliamentary election results?
- **Party Loyalty:** What percentage of voters who supported the party in the parliamentary elections also voted for it in the assembly elections within the same segment?

## 5.6 Example of CoT Prompting

Consider for example **Voter Turnout Analysis** for an assembly constituency Akkalkuwa. We would need the answer for "What was the voter turnout in Akkalkuwa in 2019 compared to the parliamentary election turnout?". We can form a chain of thought prompt using a high level reasoning as below:

1. Let us break this up in to step by step

2. I need to find out the votes polled by NOTA for each instance of Akkalkuwa in the parliamentary elections 2019.
3. Sum all the NOTA votes.
4. I need to find similarly aggregate votes polled by each candidate.
5. Sum up NOTA votes and candidates aggregate votes.
6. Obtain the TURNOUT for Akkalkuwa from assembly elections table.
7. Compare them and provide an analysis.

Note that the CoT example as above provides a general approach to the LLM to solve problems that have a similar pattern but are different. For example, consider a question like "how did BJP perform in Shivajinagar assembly constituency in the 2019 elections and how does this compare with the party's performance in the corresponding parliamentary election segment?". Note that this question has a similar pattern to the earlier question on voter turnout but still very different as it refers to a different assembly constituency and refers to a specific political party. The similarity is more in the nature of analysis needed where we are comparing parliamentary and assembly performance in 2019.

## 5.7 Build a GUI based Chatbot

In this step, you are required to implement a chatbot using Streamlit library. Sample code is provided. You are required to integrate Streamlit frontend with the LLM client backend. This allows the user to ask a question through a GUI interface and get the response from the LLM, display the results in graphical form (charting) as needed.

Refer: <https://github.com/streamlit/llm-examples/blob/main/Chatbot.py> and <https://docs.streamlit.io/develop/tutorials/llms/build-conversational-apps>

Also refer the starter code provided and modify as appropriate.

## 5.8 Generate SQL Query

You are required to accept a natural language question from the user through the GUI chatbot and generate a SQL query from the LLM. You are required to develop suitable prompts that can generate the queries accurately. Since these queries are autogenerated, care must be exercised before executing them. Make sure that these queries do not corrupt or delete records in the database.

## 5.9 Implement Runtime

The purpose of the runtime is to accept a SQL query generated by the LLM and execute it against the database we created already. Implement and test this runtime. In the first version, we will support executing only one query at a time. We will not support natural language queries that result in more than one SQL statement or involve very complex joins.

## 5.10 Optional Feature: ReAct

Extending this product using ReAct technique provides many powerful features. For example, for certain questions from the user, for example, "What are the top headlines pertaining to Maharashtra elections 2024?", the response can be provided by using Tavily search using ReAct framework.

## 5.11 Generate the final output

The results of the runtime is usually a structured output, generated by a SQL query. It is necessary to turn this in to a natural language before it can be presented to the user. You are required to feed the query results to the LLM, prompt it appropriately to generate the required outputs.

## 5.12 Develop the Orchestrator

Now that we have all the subsystems required for our product, integrate them using an orchestrator function. This should support end to end workflow, including any invocation of ReAct tools.

## 5.13 Integrate and Test

Complete the end to end workflow: starting from questions, generating prompts to the LLM, getting SQL code, running it, getting results from database, post processing and visualizing the results. You are required to develop and modify your prompts such that you get accurate SQL code out of the LLM. Evaluate against 25 test cases and report the results. Upload your work in the shared folder.

# 6 Evaluation Criteria

Submissions will be evaluated based on the following criteria:

- **Accuracy:** The chatbot's ability to provide correct insights and perform accurate calculations based on the datasets.
- **Relevance:** How well the chatbot's answers align with the needs of a political strategist.
- **Creativity in Prompt Engineering:** Innovative and effective use of prompt engineering techniques like Zero-shot, Few-shot, CoT, ToT, or ReAct.
- **User Experience:** The chatbot's ability to engage users, provide clear explanations, and handle follow-up questions.
- **Efficiency:** The chatbot's ability to handle complex queries while minimizing errors and maintaining the flow of conversation.

## Deliverables

Participants must submit:



- A working LLM-powered chatbot capable of interacting with the datasets and providing political strategy insights.
- A report documenting the prompt engineering techniques applied and how they were used to optimize the chatbot's performance.
- A demonstration of the chatbot answering several political strategy-related questions and performing analytics on the datasets.

## 7 Project Rubric

Please use the rubric given below to evaluate your work:

<b>Task</b>	<b>Completion %</b>
Dataset cleaned, schema for all tables formed	
Database with required tables created	
LLM Client implemented, sample questions generated using prompts	
LLM generates accurate SQL	
Streamlit based GUI implemented and is used to provide inputs and view the outputs	
Runtime implemented, generated SQL runs successfully on the runtime	
Results of runtime feedback to LLM, suitable prompts written for visualization	
Final integration completed, demo'ed	

For the purposes of records for the L&D department, visibility of your BU and to aid the future hackathons you may optionally submit your work to a shared folder that the L&D department may create.