Entertainment Planning Assistant Agent

Course Name: Agentic AI

Institution Name: Medicaps University – Datagami Skill Based Course

Student Name(s) & Enrolment Number(s):

Sr no	Student Name	Enrolment Number
1	PRABAL SHUKLA	EN22CS301713
2	PRATIK KOUSHAL	EN22CS301747
3	PRIYANSH CHOUHAN	EN22CS301759
4	PUSHPRAJ SINGH CHOUHAN	EN22CS301774
5	RAJAT MALVIYA	EN22CS301783

Group Name: Group 11D7

Project Number: AAI-334

Industry Mentor Name: Aashruti Shah

University Mentor Name: Ajeet Singh Rajput

Academic Year: 2025-2026

Table of Contents

1. Problem Statement & Objectives

1.1 Problem Statement

In today's fast-paced digital environment, users frequently rely on online platforms to plan entertainment activities such as watching movies in theatres. Information related to movies—including theatre availability, show timings, ticket price ranges, ratings, and reviews—is scattered across multiple platforms. Users must manually browse, compare, and validate this information, which is inefficient and often leads to incomplete or outdated decisions.

The key problem addressed in this project is the absence of a unified intelligent system capable of autonomously planning and executing entertainment-related tasks. Existing systems lack dynamic planning, multi-step reasoning, and dependency management based on high-level user goals.

To address this issue, the Entertainment Planning Assistant Agent is designed as a sophisticated Planner Agent that decomposes high-level objectives into actionable tasks, validates resources through tool-based interfaces, and generates structured, optimized outputs.

1.2 Project Objectives

The main objectives of the project are:

- To design and implement an AI-powered Planner Agent for entertainment planning.

- To enable autonomous task decomposition and multi-step reasoning.

- To integrate tool-based reasoning using the Groq API.

- To generate structured and user-friendly responses using a Large Language Model.

- To develop a backend system using FastAPI.

- To create an interactive frontend using Streamlit.

1.3 Scope of the Project

The scope of this project includes the development of a locally hosted AI assistant focused on movie-related entertainment planning. The system processes user inputs such as movie name and city to generate intelligent recommendations.

The scope covers:

- City-based movie and theatre search
- Retrieval of show timings, ticket price ranges, and ratings
- AI-generated summaries and recommendations
- Planner Agent-based task orchestration
- Backend–frontend integration

Features such as online ticket booking, user authentication, payment processing, and cloud deployment are not included in the current scope and are considered future enhancements.

2. Proposed Solution

2.1 Key Features

- AI-powered Planner Agent
- Autonomous task decomposition
- Multi-step reasoning and dependency handling
- Tool-based reasoning using Groq API
- LLM-based intelligent summarization
- Structured and transparent outputs
- Interactive Streamlit-based interface

2.2 Overall Architecture / Workflow

This section explains the high-level architecture and execution flow of the **Entertainment Planning Assistant Agent**. The system follows a **layered and agent-based architecture** to support autonomous planning, reasoning, and tool integration using the Groq API.

To represent the system design and workflow clearly, the following UML diagrams are used:
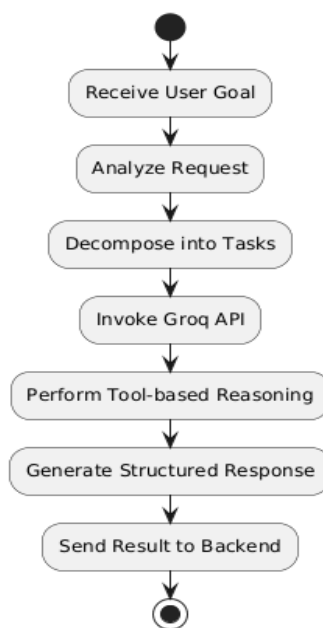
- **System Architecture Diagram:** Shows the interaction between the frontend, backend, Planner Agent, Groq API, and the Large Language Model.

System Architecture - Entertainment Planning Assistant Agent
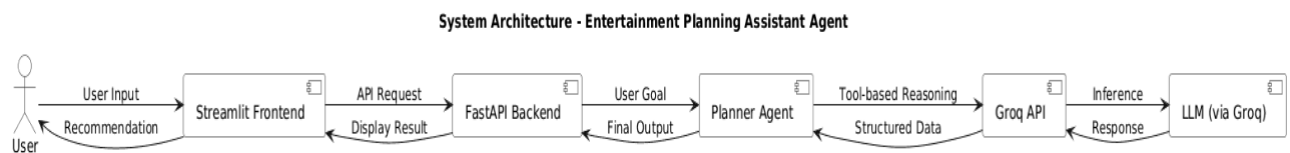
(Figure 2.1)

- **Planner Agent Workflow Diagram:** Illustrates the step-by-step reasoning and task decomposition performed by the Planner Agent.


Planner Agent Workflow Diagram

Receive User Goal → Analyze Request → Decompose into Tasks → Invoke Groq API → Perform Tool-based Reasoning → Generate Structured Response → Send Result to Backend

(Figure 2.2)

- **Sequence Diagram:** Represents the order of interactions among the user, frontend, backend, Planner Agent, Groq API, and LLM.
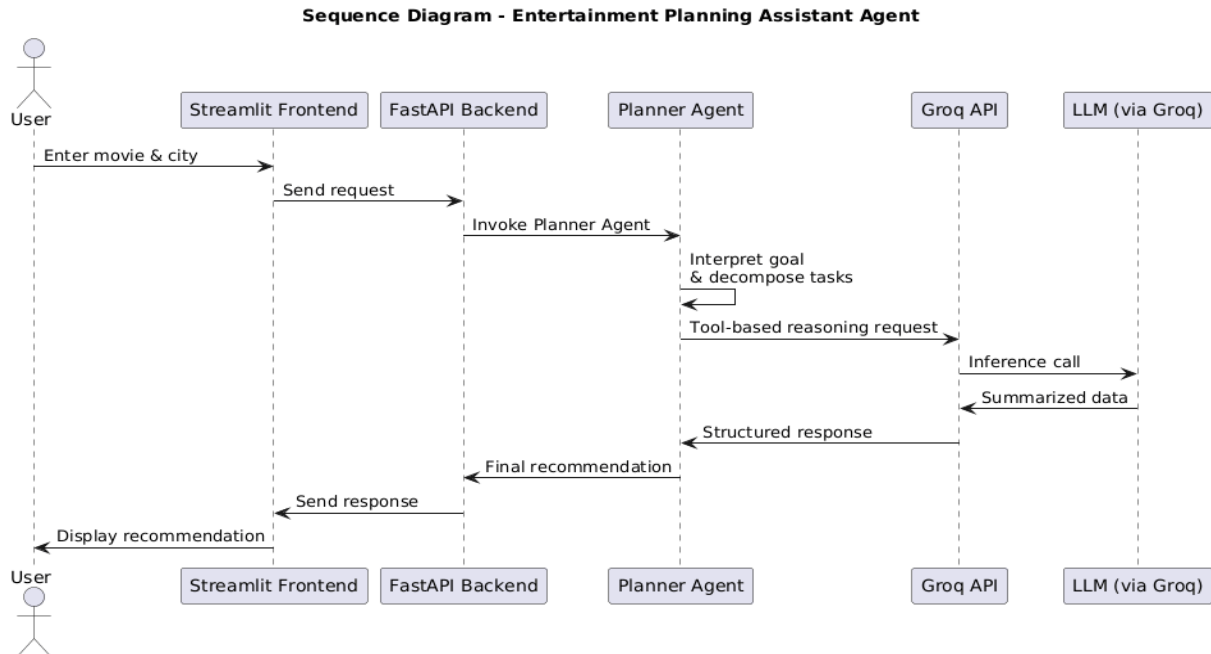
(Figure 2.3)

- **Swimlane Activity Diagram:** Displays the end-to-end process flow while clearly defining the responsibilities of each system component.
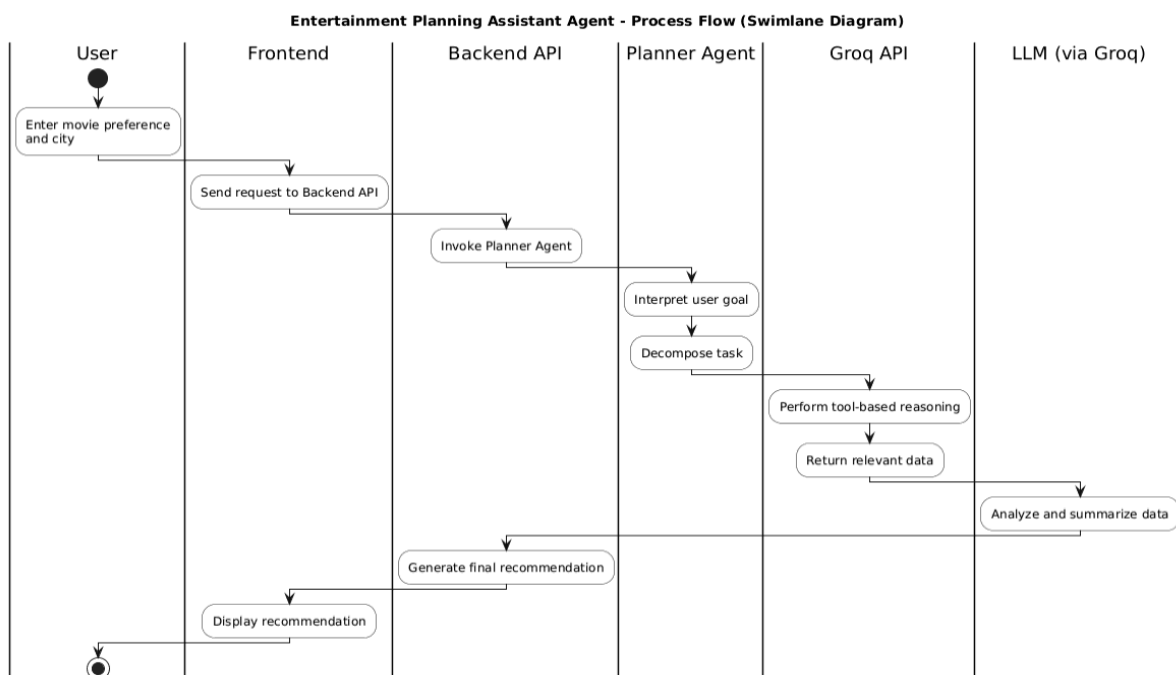
(Figure 2.4)

Workflow Summary:

1. The user submits a high-level entertainment query through the Streamlit interface.

2. The request is forwarded to the FastAPI backend.

3. The Planner Agent decomposes the goal into actionable subtasks.

4. Tool-based reasoning and inference are performed using the Groq API.

5. The Large Language Model generates a structured and concise response.

6. The final output is displayed to the user through the frontend interface.

2.3 Tools & Technologies Used

- Python – Core programming language

- FastAPI – Backend API framework

- Streamlit – Frontend user interface

- Groq API – Tool-based reasoning and LLM inference

- Large Language Model (via Groq)

- Uvicorn – ASGI server

- Dotenv – Environment variable management

3. Results & Output

This section presents the outcomes obtained after implementing and executing the Entertainment Planning Assistant Agent. It highlights the system outputs, intermediate results, and overall achievements of the project.

3.1 Screenshots / Outputs

This subsection presents the visual outputs of the Entertainment Planning Assistant Agent to demonstrate the working and effectiveness of the system. The screenshots validate the successful execution of the frontend interface,

backend processing, Planner Agent reasoning, and AI-generated recommendations.

Figure 3.1 shows the **Streamlit-based frontend interface** of the system. It allows the user to enter the movie or event name along with the selected city and generate an entertainment plan. This interface acts as the primary interaction point between the user and the system.
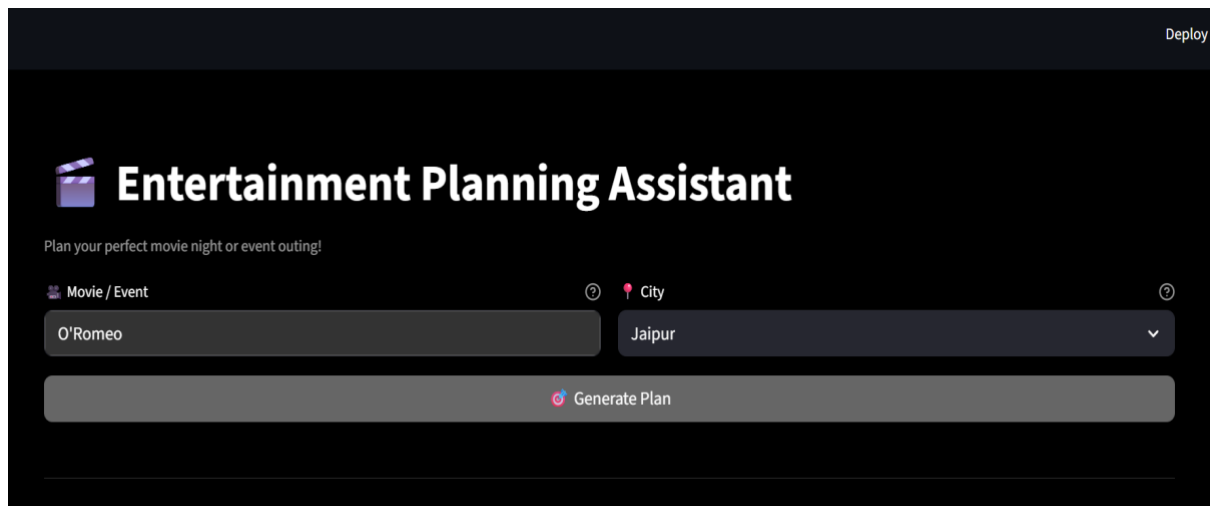

Figure 3.1

Figure 3.2 presents the **combined entertainment plan output**. This figure includes the successful plan generation confirmation along with detailed results such as event information, popular theatres, show timings, ticket price range, public ratings, nearby attractions, transportation options, restaurant recommendations, weather considerations, accommodation suggestions, and safety tips. It reflects the structured and contextual output generated by the Planner Agent using Groq-based reasoning.

Figure 3.2

Figure 3.3 displays the **final recommendation summary**, highlighting the estimated cost, expected duration, and overall rating of the entertainment plan. This compact summary provides the user with a clear, concise, and actionable recommendation.

Figure 3.3

3.2 Reports / Dashboards / Models

This project does not include traditional analytical dashboards. Instead, it functions as an **AI-driven planning and recommendation system**. The intelligence of the system is embedded within its core components rather than visual dashboards.

The core model of the system consists of:

- **Planner Agent logic** for goal interpretation, task decomposition, and reasoning

- **Groq-powered Large Language Model (LLM)** for intelligent response generation

- **Backend API workflow**, which coordinates the frontend, Planner Agent, and AI inference

The internal working of the system is represented using **UML diagrams and workflow models**, which explain the architecture, agent reasoning process, and interaction flow of the system.

3.3 Key Outcomes

The key outcomes achieved through this project include:

- Successful implementation of an **Agentic AI-based Planner Agent**

- Effective use of **Groq API** for tool-based reasoning and LLM inference

- End-to-end integration of **frontend, backend, agent, and AI layers**

- Generation of **structured, accurate, and user-friendly entertainment recommendations**

- Improved practical understanding of **agent workflows and real-world AI system design**

4. Conclusion

The **Entertainment Planning Assistant Agent** successfully demonstrates the practical application of **Agentic AI** in solving real-world entertainment planning problems. The project integrates a Planner Agent with a Groq-powered Large

Language Model to autonomously analyze user goals, perform multi-step reasoning, and generate structured, meaningful entertainment recommendations.

Through this project, a complete end-to-end system was developed using **Streamlit for the frontend**, **FastAPI for the backend**, and **Groq API for intelligent reasoning and inference**. The use of UML diagrams and structured

workflows helped in designing a modular, scalable, and easy-to-understand system architecture.

Key learnings from this project include a deeper understanding of **agent-based system design**, **tool-based reasoning**, **backend-frontend integration**, and the practical implementation of modern AI technologies. Overall, the project provided valuable hands-on experience in building an intelligent, real-world AI application.

5. Future Scope & Enhancements

The Entertainment Planning Assistant Agent can be further enhanced to improve functionality, scalability, and user experience. Several extensions can be implemented to make the system more robust and suitable for real-world deployment.

Future enhancements may include:

- **Online ticket booking integration** to allow users to reserve tickets directly from the application.

- **User authentication and personalization**, enabling the system to store preferences and provide customized recommendations.

- **Cloud deployment** to improve accessibility, scalability, and performance for multiple users.

- **Conversation memory** using databases or vector stores to maintain context across multiple user interactions.

- **Mobile application development** to extend usability on Android and iOS platforms.

- **Voice-based interaction** using speech-to-text and text-to-speech technologies.

- **OTT platform recommendations** to include movies and shows from streaming services.

- **Multi-language support** to make the system accessible to a wider audience.

These enhancements can transform the project into a comprehensive, intelligent entertainment planning platform capable of serving diverse user needs.