IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Tournament Management Agent Using Agentic Artificial Intelligence

Hemalatha S V[1], Aishwarya Lakshmi A[1], Kundan D R[1], Ranjith K C[2]

[1] Students, Department of CSE–AIML, MIT Mysore
[2]Associate Professor, Department of CSE–AIML, MIT Mysore

Abstract: This paper presents a detailed design and implementation of an AI-powered Tournament Management Agent based on Agentic Artificial Intelligence. Traditional tournament management relies on manual coordination or static software systems, leading to inefficiencies, data inconsistency, and limited scalability. The proposed system introduces an autonomous conversational agent capable of understanding natural language queries and executing tournament management tasks such as creation, updating, deletion, and retrieval of tournament information. The system integrates FastAPI for backend services, SQLite for lightweight persistent storage, and Google Gemini for natural language understanding and reasoning. By combining CRUD-based automation with agentic decision-making, the system significantly reduces human effort and improves usability. Extensive testing and analysis demonstrate that the proposed approach enhances operational efficiency and provides a scalable solution suitable for academic institutions, sports clubs, and event organizations.

Index Terms - Agentic AI, Tournament Management System, Artificial Intelligence, FastAPI, SQLite, Natural Language Processing, AI Agents, Gemini

Introduction

Sports tournaments are an integral part of academic institutions, professional organizations, and community-level events. Organizing such tournaments involves multiple activities including participant registration, fixture preparation, scheduling, venue allocation, result recording, and coordination among organizers. As the number of participants and events increases, managing these activities manually becomes difficult and prone to errors.

In many cases, tournament management is still carried out using paper records, spreadsheets, or basic software tools. These methods are often inefficient, lack real-time updates, and depend heavily on human intervention. Manual handling of data can lead to duplication, scheduling conflicts, and communication gaps, especially in medium to large-scale tournaments.

Recent advancements in Artificial Intelligence have enabled the development of intelligent systems that go beyond static automation. Agentic Artificial Intelligence focuses on autonomous agents capable of understanding user intent, reasoning over tasks, and performing actions without continuous supervision. Such systems are more adaptive and user-friendly compared to traditional rule-based applications.

This project applies agentic AI principles to tournament management by introducing an intelligent Tournament Management Agent. The system allows users to interact using natural language commands to create, update, retrieve, or delete tournament information. By integrating FastAPI, SQLite, and Google Gemini, the proposed solution transforms a conventional management system into a conversational and autonomous platform that improves efficiency and usability.

Motivation And Need For The System

The development of an AI-powered Tournament Management Agent is motivated by the practical difficulties faced by organizers in educational institutions and local sports bodies. Tournament coordination is often handled by individuals who already have academic or administrative responsibilities, making manual management time-consuming and stressful.

Most existing tournament management tools rely on complex user interfaces and repetitive data entry. These systems require users to follow fixed workflows and possess technical knowledge, which limits accessibility for non-technical users. Additionally, they lack intelligent assistance, forcing users to manually search for information and update records.

Another major limitation is the absence of interactive support. Organizers must navigate through multiple menus or datasets to access simple details such as tournament schedules or venues. A conversational AI-based interface simplifies this process by allowing users to obtain information instantly through natural language queries.

Agentic AI addresses these challenges by enabling autonomous task execution. The agent understands the user's intent, plans the required actions, and performs them automatically. This reduces manual workload, minimizes errors, and enhances the overall experience of tournament management.

Problem Statement

Several studies have focused on the digital transformation of tournament management systems. Early research by Chitroda et al. (2015) introduced a digital platform aimed at replacing manual paper-based tournament records. While the system improved data organization, it did not incorporate intelligent automation or adaptive decision-making.

Raof (2018) proposed an automated scheduling system based on predefined rules to generate match fixtures. Although effective for basic tournaments, rule-based systems lack flexibility when handling dynamic changes such as rescheduling or participant withdrawal.

Rutjanisarakul and Jiarasuksakun (2017) applied genetic algorithms to tournament scheduling to reduce conflicts and improve fairness. Their approach achieved better optimization results but required manual configuration and did not support user-friendly interaction.

More recent work by Li et al. (2024) explored agent-based scheduling models where autonomous agents coordinate and adapt to constraints dynamically. These studies demonstrate the potential of agentic AI but are not directly applied to practical tournament management scenarios.

From the existing literature, it is clear that limited research combines conversational AI with autonomous task execution in tournament management. This project addresses this gap by integrating agentic AI with a real-world tournament management application.

Objectives Of The Project

The primary objective of this project is to design and implement an intelligent Tournament Management Agent that simplifies the process of managing tournaments through AI-assisted automation. The system aims to reduce manual effort, improve data accuracy, and enhance user interaction by leveraging agentic AI principles.

The specific objectives of the project are as follows:

• To automate tournament data management using CRUD operations

• To provide a clean and intuitive web-based user interface

• To integrate an AI-powered conversational agent for natural language queries

• To ensure seamless communication between frontend, backend, database, and AI agent

• To design a modular and scalable architecture for future enhancements

• To improve efficiency and reduce errors in tournament organization

Literature Survey

Several researchers have explored the digitization and automation of tournament management systems. Chitroda et al. (2015) proposed a basic digital tournament management platform aimed at replacing manual paper-based systems. Their work demonstrated that digitization improves data consistency and reduces paperwork; however, the system lacked intelligent automation and decision-making capabilities.

Raof (2018) introduced a rule-based automated scheduling system for sports tournaments. The system generated fair match schedules using predefined rules and constraints. While effective for simple tournaments, rule-based systems struggle with dynamic conditions such as last-minute changes or complex scheduling constraints.

Rutjanisarakul and Jiarasuksakun (2017) applied genetic algorithms to tournament scheduling. Their approach demonstrated improved optimization and reduced scheduling conflicts. However, the system required manual configuration and did not provide a user-friendly or conversational interface.

Recent studies by Li et al. (2024) focused on agent-based scheduling systems, where autonomous agents negotiate constraints and adapt schedules dynamically. This research highlights the potential of agentic AI in handling complex coordination tasks. However, these systems were not specifically designed for real-world tournament management applications.

From the literature, it is evident that while significant progress has been made in automation and optimization, limited work integrates **conversational AI and agentic decision-making** into tournament management. This research aims to bridge that gap.

System Architecture

The Tournament Management Agent is designed using a modular three-layer architecture to ensure scalability, flexibility, and ease of maintenance. Each layer is responsible for a specific set of functions, allowing independent development and future enhancements.The Presentation Layer provides the user interface and enables interaction through both traditional forms and a conversational chat interface. It is developed using HTML, CSS, and JavaScript, offering dashboards and input forms for managing tournaments.

The Application Layer is implemented using FastAPI and handles the core business logic. This layer processes incoming requests, validates inputs, manages API routing, and communicates with both the database and the AI agent. It also acts as an intermediary that ensures secure execution of operations initiated by the AI agent.

The Data Layer uses SQLite for storing tournament information in a structured format. This layer ensures persistent storage, efficient data retrieval, and reliable CRUD operations. The separation of layers enhances system reliability and supports future expansion.

Fig. 1 shows the overall system architecture of the proposed Tournament Management Agent. The presentation layer allows users to interact with the system through a web-based interface. User requests are forwarded to the application layer, where FastAPI processes the requests and manages the core business logic. The AI agent understands natural language queries and communicates with the backend to perform the required operations. All tournament-related information is securely stored and managed using an SQLite database.

Agentic Ai Architecture

The Agentic AI component is the most critical and innovative part of the proposed system. Unlike traditional chatbot systems that simply generate responses, the Tournament Management Agent is designed as an **autonomous agent** capable of reasoning, planning, and executing tasks based on user intent.

The agent follows a structured lifecycle consisting of the following stages:

1. **Input Interpretation**

The user provides input in natural language through the chat interface. The agent receives this input and preprocesses it to identify key entities such as tournament name, date, location, or action type.

2. **Intent Recognition**

Using Google Gemini, the agent determines the intent behind the user's query, such as creating a tournament, retrieving tournament details, updating records, or deleting an entry.

3. **Decision Making and Planning**

Once the intent is identified, the agent determines the sequence of actions required to complete the task. For example, creating a tournament may involve validating dates, checking for missing information, and invoking database insertion functions.

4. **Tool Selection and Execution**

The agent selects the appropriate backend function (CRUD operation) and executes it autonomously. If required information is missing, the agent asks clarifying questions instead of failing silently.

5. **Response Generation**

After executing the task, the agent generates a human-friendly response and presents it to the user. This architecture allows the system to behave like an intelligent assistant rather than a static application, significantly improving usability and efficiency.

• Tournament ID (Primary Key)
• Tournament Name
• Sport Type
• Location
• Start Date
• End Date

Each record uniquely represents a tournament. CRUD operations are implemented to manage these records efficiently. Indexing is used to improve query performance and ensure fast data retrieval.

Functional Requirements

Functional requirements describe the specific operations and services that the Tournament Management Agent must provide to users. These requirements define the expected behavior of the system and ensure that all essential tournament-related activities are supported effectively.

The system must allow users to **create new tournaments** by providing essential details such as tournament name, sport type, location, start date, and end date. This operation can be performed either through a traditional form-based interface or through natural language commands processed by the AI agent.

The system must support **viewing tournament details**, including listing all tournaments and retrieving specific tournament information based on user queries. Users should be able to request information such as upcoming tournaments, tournaments at a specific location, or tournaments conducted during a particular period.

The system must enable **updating tournament information**, allowing organizers to modify details such as venue, dates, or sport type. The AI agent should be capable of understanding update requests expressed in natural language and applying the changes automatically.

The system must support **deleting tournaments** that are canceled or completed. This operation ensures that outdated data does not clutter the database.

Additionally, the system must provide **AI-based query handling**, where users can interact conversationally with the system. The AI agent should interpret user intent, validate requests, execute appropriate actions, and return meaningful responses.

Non-Functional Requirements

Non-functional requirements define the quality attributes of the system and determine how well it performs under various conditions.

Performance Requirements:

The system should respond to user requests within a short time frame. API calls for CRUD operations should complete within one to two seconds under normal load. AI query responses should also be generated with minimal latency to ensure smooth user interaction.

Usability Requirements:

The system must be easy to use for both technical and non-technical users. The user interface should be intuitive, with clearly labeled forms and navigation elements. The conversational interface should allow users to interact naturally without needing to learn specific commands.

Reliability Requirements:

The system should operate consistently without crashes or data loss. Database transactions must ensure data integrity even in the event of unexpected failures.

Scalability Requirements:

The architecture should support future expansion, such as adding scheduling modules, team management, and analytics features, without major redesign.

Security Requirements:

The system must validate all user inputs and securely manage sensitive information such as API keys. Unauthorized access and malicious inputs should be prevented.

10. DATABASE DESIGN AND DATA MODELING

The database design plays a crucial role in ensuring accurate and efficient data management within the system. SQLite is selected due to its lightweight nature, ease of integration, and suitability for small to medium-scale applications.

The primary database table stores essential tournament details, including Tournament ID, Tournament Name, Sport Type, Location, Start Date, and End Date. The Tournament ID serves as the primary key to uniquely identify each record.

Normalization techniques are applied to minimize redundancy and maintain data consistency. Indexing is implemented on frequently queried attributes to improve performance. Secure, parameterized queries are used for all CRUD operations to prevent SQL injection and ensure data integrity.

The database structure is designed to be extensible, allowing additional tables for teams, players, and match results to be added in future versions.

11. SYSTEM IMPLEMENTATION DETAILS

The system implementation is divided into backend, frontend, and AI integration components. The backend is implemented using FastAPI, which provides high-performance asynchronous request handling and automatic API documentation.

RESTful API endpoints are created for each core operation, including creating, retrieving, updating, and deleting tournament records. Input validation ensures that incorrect or incomplete data is rejected with meaningful error messages.

The frontend is developed using HTML, CSS, and JavaScript. It provides dashboards for viewing tournaments, forms for adding and editing records, and a chat interface for interacting with the AI agent. JavaScript functions handle API calls and dynamically update the user interface based on responses.

The AI agent is integrated into the backend and communicates with the database through controlled interfaces. This integration allows the agent to execute operations autonomously while maintaining system security and reliability.

12. ALGORITHMS AND LOGICAL FLOW

The Tournament Management Agent uses a combination of traditional algorithms and AI-based logic to perform operations efficiently.

The **CRUD Algorithm** handles database operations by validating input, executing SQL queries, and returning structured responses.

The **Data Retrieval Algorithm** fetches tournament records based on user-defined criteria and formats the results for display.

The **AI Query Processing Algorithm** interprets natural language input, identifies user intent, extracts relevant entities, and maps the request to backend operations.

The **Error Handling Algorithm** ensures that invalid requests are handled gracefully by prompting users for clarification instead of terminating execution.

These algorithms collectively ensure accurate, efficient, and reliable system behavior.

13. TESTING METHODOLOGY

Testing was carried out to validate the functionality, performance, and reliability of the Tournament Management Agent.

Unit testing was conducted on individual components such as API endpoints and database functions to verify correct behavior. Integration testing ensured seamless communication between the frontend, backend, database, and AI agent.

System testing evaluated the complete application under realistic usage scenarios, including multiple user interactions and AI-based queries.

Acceptance testing involved users assessing the system's usability, accuracy, and responsiveness. The feedback confirmed that the system met the defined objectives.

14. RESULTS AND DISCUSSION

The results demonstrate that the proposed Tournament Management Agent significantly improves efficiency compared to traditional systems. Users were able to manage tournaments using natural language commands with minimal effort.

The AI agent accurately interpreted user intent and executed backend operations without manual intervention. This reduced the time required to perform administrative tasks and minimized human errors.

The discussion highlights that agentic AI adds substantial value by transforming static software into an interactive and intelligent assistant. The system provides faster access to information and improves the overall user experience.

15. PERFORMANCE ANALYSIS

Performance analysis was conducted to evaluate system responsiveness and scalability. API response times were measured under normal operating conditions and found to be within acceptable limits.

Database operations demonstrated efficient performance due to optimized queries and indexing. The AI agent processed user queries with reasonable latency, ensuring smooth conversational interaction.

The modular architecture ensures that system performance can be maintained even as additional features are integrated in future versions.

16. LIMITATIONS OF THE SYSTEM

Despite its advantages, the system has certain limitations. Advanced scheduling algorithms and predictive analytics are not included in the current implementation. The system also depends on continuous internet connectivity for AI-based query processing.

Additionally, while SQLite is suitable for small to medium-scale applications, larger deployments may require more robust database solutions.

17. FUTURE SCOPE AND ENHANCEMENTS

Future enhancements may include automated match scheduling using optimization algorithms, team and player management modules, real-time score tracking, and notification services.

Integrating analytics dashboards and AI-driven insights can further enhance decision-making. Cloud deployment and mobile application support can improve accessibility and scalability.

18. CONCLUSION

This project successfully demonstrates the application of agentic AI in tournament management. By integrating FastAPI, SQLite, and Google Gemini, the system provides an intelligent, efficient, and user-friendly solution.

The Tournament Management Agent reduces manual workload, improves accuracy, and enhances user interaction through natural language communication. The proposed approach is suitable for academic institutions, sports clubs, and event organizations, and provides a strong foundation for future enhancements.

REFERENCES

[1] D. Chitroda, V. Patel, and K. Shah, “Study on Tournament Management System,” IJTRD, 2015.

[2] R. A. A. Raof, “Sport Tournament Automated Scheduling System,” MATEC Web of Conferences, 2018.

[3] K. Devriesere et al., “Tournament Design: A Review from an Operational Research Perspective,” arXiv, 2024.

[4] B. Li, L. Li, M. Li, and R. Zhang, “Public Event Scheduling with Busy Agents,” IJCAI, 2024.

[5] T. S. Yashwanth et al., “A Multi-Agent Pokémon Tournament for Evaluating Strategy Learning,” arXiv, 2025.

[6] T. Rutjanisarakul and T. Jiarasuksakun, “A Sport Tournament Scheduling by Genetic Algorithm with Swapping Method,” arXiv, 2017.