Museum Chatbot Using MCP Servers and LLMs

G.Rama Devi , N. Amrutha,[1]P. Sirisha ,[1]S. Madhu ,[1]S. Mahesh

Department of Computer Science & Engineering (Data Science)

Avanthi Institute of Engineering & Technology (Autonomous), Vizianagaram, India
ramadevigorle05@gmail.com , amruthanagothi@gmail.com, Sirishapanigrahi515@gmail.com, maheshsreemanthula@gmail.com, msm79929@gmail.com

Guided by: Mrs. G. Rama Devi, M.Tech – Assistant Professor, CSE-DS, AI & ML

Abstract

Museums are living archives of human civilisation that face a growing challenge— delivering personalised, contextually rich visitor guidance at scale without proportionally expanding human staffing. Existing chatbot solutions built on pattern-matching, intent classification, or knowledge graph paradigms fall short when visitors ask emotional, open-ended, or cross-exhibit questions, and they demand constant manual updates whenever collection content changes. This paper presents a museum chatbot architecture that pairs the generative strength of Large Language Models with the structured, permission-enforced data access of Model Context Protocol servers. The LLM manages natural multi-turn conversation, contextual memory, dynamic tone adaptation, and multilingual generation, while the MCP layer acts as a secure gateway that retrieves verified, real-time exhibit information from the museum database without model retraining or script edits. A four-tier design—visitor interface, LLM processing, MCP communication, and museum database—supports instant content propagation. Evaluation through seven functional test cases confirms stable conversation continuity, reliable multilingual switching, effective restricteddata filtering, and graceful handling of ambiguous queries. The system improves visitor engagement, reduces staff maintenance overhead, and establishes a scalable, privacy-conscious template for AI-powered digital guides across art galleries, cultural centres, and heritage institutions.

Index Terms—Conversational AI, Large Language Models, Model Context Protocol, Museum Chatbot, Natural Language Processing, Visitor Engagement

I. INTRODUCTION

Museums preserve the stories, artefacts, and ideas that define human progress. As visitor expectations shift toward seamless digital interaction, institutions face mounting pressure to provide on-demand, personalised guidance without proportionally scaling their human workforce [1], [4]. Early chatbot deployments partially addressed this demand—handling ticket queries and opening-hour lookups—but they consistently collapsed when visitors posed contextual, creative, or multiexhibit questions [2], [3].

The shortcomings of prior approaches are well documented. Artificial Intelligence Markup Language bots match patterns mechanically and discard context after a single exchange [3], [14]. Intent-based platforms such as Dialogflow improved phrasing tolerance but required manually curated training phrases for every exhibit update, creating an unsustainable maintenance burden [8]. Knowledge graph systems delivered factual accuracy for structured queries yet lacked the conversational fluency needed for narrative-driven exploration [5]. Deep-learning dialogue models improved linguistic output but overfit on the small conversation logs that museums typically possess [7].

The maturation of Large Language Models has fundamentally changed what is achievable. LLMs trained on vast corpora understand nuanced context, adapt tone to audience type, and sustain coherent multiturn dialogue without domain-specific retraining [7], [17]. However, a standalone LLM cannot reliably access live, institution-specific data, creating tension between conversational quality and factual grounding. The Model Context Protocol resolves this tension by providing a structured, permission-enforced interface through which an LLM can request verified exhibit data from backend systems at inference time, without the model needing to memorise institution facts during training [12].

This paper contributes: (i) a modular LLM–MCP integration pattern for real-time exhibit retrieval without retraining; (ii) a role-based permission framework preventing sensitive archival data from reaching the LLM context; (iii) a multilingual, personalisation-aware visitor interface deployable across web, mobile, and kiosk platforms; and (iv) a comprehensive test suite covering functional, security, and load scenarios specific to cultural-institution deployments.

II. RELATED WORK

A. Early Rule-Based and AIML Systems

The earliest museum digital assistants descended from pattern-matching paradigms demonstrated by ELIZA and later formalised in Artificial Intelligence Markup Language [3], [14]. AIML systems were inexpensive to deploy and adequate for simple factual queries—ticket prices, opening hours, floor maps—but their single-turn memory and inability to handle paraphrased input made them unsuitable for narrative exhibit exploration [2]. Martinez [14] catalogued the structural weaknesses of these systems, noting that any phrasing outside the predefined template produced silent failure or generic fallback responses that frustrated visitors.

B. Intent-Based Platforms

Platforms such as Dialogflow introduced machinelearned intent classification, allowing moderate phrasing variation [8]. Museums constructed guidedtour bots that handled multi-entity queries. Guo and Patel [8] observed, however, that intent coverage was strictly bounded by the curated training corpus: each exhibit update required manual addition of intents and utterance examples. This bottleneck proved especially acute in institutions with rotating exhibitions or multilingual audiences.

C. Knowledge Graph Systems

Knowledge graphs represented collections as interconnected ontologies, supporting structured queries such as temporal artefact filtering and artist relationship traversal [5]. Darwish [5] demonstrated near-perfect factual accuracy for well-formed semantic queries on static collections. The fundamental deficiency was expressive poverty: open questions inviting narrative, emotional engagement, or crossdomain synthesis fell outside the graph's retrieval scope. Constructing rich museum ontologies also demanded intensive curatorial investment that small institutions could not sustain.

D. Deep Learning Dialogue Models

Sequence-to-sequence and transformer-based dialogue models improved linguistic fluency [7], [13]. Feng [7] reported that fine-tuned LLM systems achieved significantly higher coherence scores on conversational benchmarks. However, museum-scale conversational datasets are sparse, leading to hallucinated facts and inconsistent exhibit descriptions when models relied solely on parametric memory. Li [13] addressed this partially through retrieval-augmented generation but did not formalise a permission layer for sensitive institutional data.

E. MCP-Enabled AI Architectures

Kapoor and Thomas [12] formalised the Model Context Protocol as a standardised mechanism for LLMs to invoke external tools through structured, authenticated API calls. The protocol decouples the generative model from domain data, enabling content updates without retraining. Park and Silva [17] confirmed that LLM output quality improved when responses were grounded in MCP-retrieved facts for cultural heritage

III. METHODOLOGY / SYSTEM DESIGN

A. Four-Tier Architecture

The proposed system is organised into four independently scalable tiers: (1) Visitor Interaction Layer, (2) LLM Processing Layer, (3) MCP Communication Layer, and (4) Museum Database Layer. Figure 1 presents the complete architecture with inter-layer data flows.

Museum AI Chatbot – System Architecture① Visitor Interaction LayerWeb · Mobile · Kiosk Touch | Chat UI · Language Switcher · Media ViewerPersonalised Exhibit Suggestions · Quick-Link Buttons · Multimedia Panel② LLM Processing LayerContext Management · Multilingual NLG · Tone Adaptation · Session MemoryPrompt Engineering · MCP Tool Invocation · Fallback Handling③ MCP Communication LayerPermission Validation · Request Routing · Data Filtering · Rate LimitingAudit Logging · Structured Query Builder · Denial Notification④ Museum Database LayerExhibits · Artists · Media Files · Visitor Profiles · System LogsPostgreSQL / MongoDB | Admin Control PanelAdmin PanelUpdate · Upload · Access RulesPrimary request flowResponse return flow

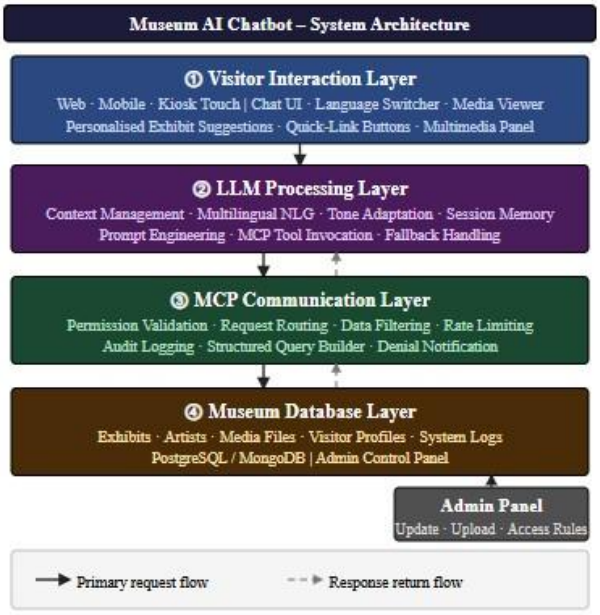

Fig. 1. Four-tier Museum AI Chatbot architecture showing primary request flow (solid) and response return flow (dashed) between visitor interface, LLM, MCP gateway, and museum database.

B. LLM Processing Layer

The LLM receives the visitor message together with accumulated session context and performs three sequential operations: intent and context analysis; a decision on whether factual exhibit data is required; and final response generation. When exhibit-specific facts are needed, the LLM constructs a structured toolcall directed at the MCP server rather than drawing on parametric memory alone. Equation 1 formalises the response generation objective.

$$R = LLM(C_{session}, m_{user}, D_{MCP}) \quad (1)$$

Here R is the generated response, C_{session} is the accumulated conversation context, m_{user} is the current visitor message, and D_{MCP} is structured exhibit data retrieved by the MCP server. When D_{MCP} is null (queries answerable from training

C. MCP Permission Model

The MCP server evaluates each incoming tool-call against a configurable access-control list before executing any database query. Equation 2 defines the binary permission decision function.

$$P(req) = 1 \text{ if } role(req) \in ACL(res), \text{ else } 0 \quad (2)$$

$P(req)$ is the binary access decision, $role(req)$ is the agent role (LLM, admin, or auditor), and $ACL(res)$ is the museum-defined access-control list for the targeted resource. Blocked requests are logged and the LLM receives a denial token, enabling it to compose a polite visitor response rather than silently failing.

D. Sequence Diagram – Message Processing

Figure 2 presents the nine-step sequence for each visitor turn, derived directly from the system UML artefacts. The conditional MCP branch is activated only when the LLM determines that verified exhibit data is required.

VisitorUILLMMCPMuseumDB① Send Message② Forward Message③ Check Contextalt[Additional Exhibit Data Needed]④ Request Data⑤ Verify ACL⑥ Query DB⑦ Return Results⑧ Send JSON Data⑨ Generate ReplySend AnswerDisplay ReplySequence repeats for every new visitor message within the session

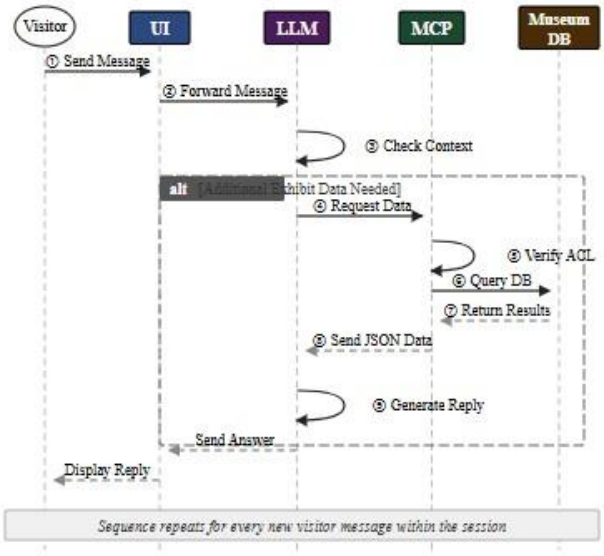

Fig. 2. Sequence diagram for the Museum AI Assistant showing nine-step message flow; the alt fragment is activated when exhibit data is required from the MCP server.

E. Activity Diagram

Figure 3 illustrates the system activity flow reproduced from the project UML models. The decision node at the centre routes generic queries directly through the LLM while directing exhibit-specific queries through the MCP pipeline before the final response is rendered. The session loop continues until the visitor explicitly ends the conversation.

Visitor enters questionSystem receives inputLLM analyses inputIs infogeneric?YESLLM answers directlyNORequest sent to MCPMCP checks accessMCP fetches dataLLM forms answer from dataUI displays answer

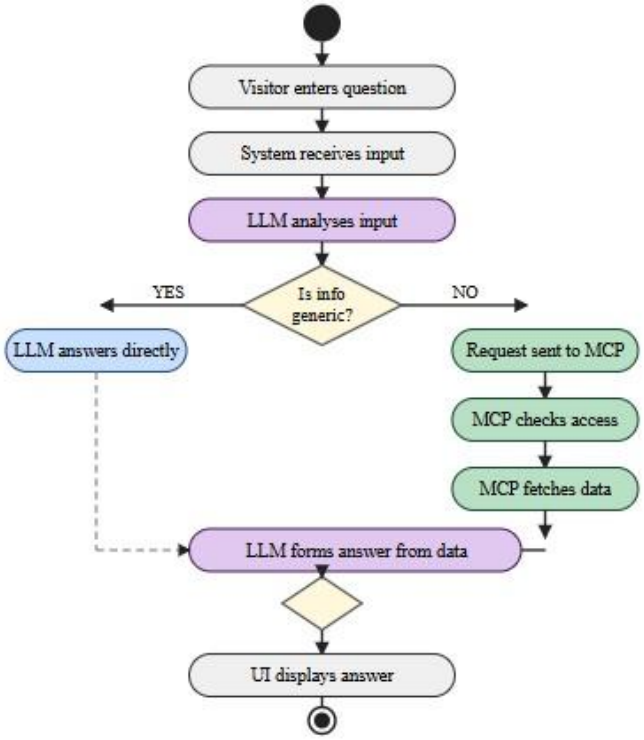

Fig. 3. Activity diagram for the Museum AI Assistant, showing the conditional MCP data-retrieval branch and session loop.

F. Class Structure

Five primary classes constitute the system: *UserSession*, *ChatInterface*, *LLMEngine*, *MCP Connector*, and *ExhibitDatabase*. The *AdminPanel* class provides parallel write access to the database for content management. Figure 4 reproduces the class diagram from the project design artefacts.

UserSessionuserIDlanguagePreferenceconversationHistory · profileDataChatInterface+captureInput()+displayOutput()LLMEngine+analyzeText()+generateResponse()+requestDataFromMCP()MCPConnector+validateRequest()+fetchData()+sendResponse()AdminPanel+updateExhibit()+uploadMedia()+manageAccessRules()ExhibitDatabase

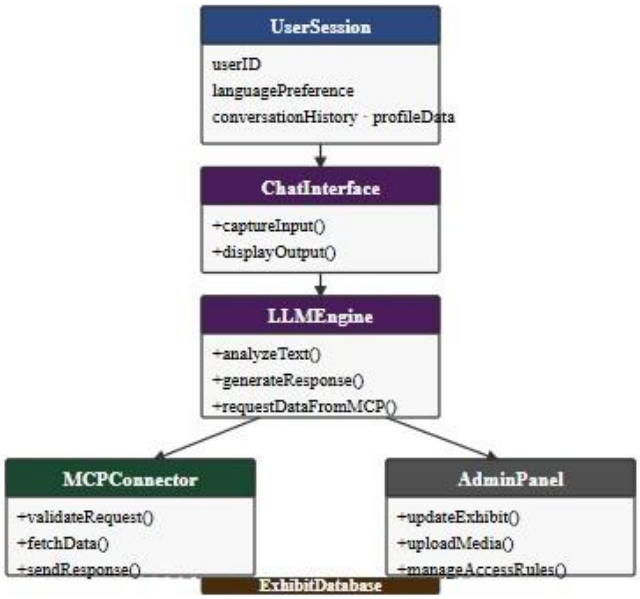

Fig. 4. Class diagram showing UserSession, ChatInterface, LLMEngine, MCPConnector, AdminPanel, and ExhibitDatabase with method signatures.

G. Implementation Stack

The backend is implemented in Python using FastAPI. The MCP server layer handles permission validation and database query execution. The LLM is accessed via a cloud-hosted API with structured tool-call support. The frontend is a React.js single-page application consuming backend REST endpoints over HTTPS. MongoDB stores visitor session data and logs, while exhibit content resides in a relational PostgreSQL database. Docker Compose orchestrates all services for reproducible staging and production deployment.

IV. RESULTS AND DISCUSSION

A. Functional Test Results

Seven functional test cases were designed against the core system requirements. Table I summarises each scenario, expected behaviour, and observed outcome. All seven cases passed without modification, confirming that the integrated LLM–MCP pipeline satisfies the complete functional specification identified in the requirements analysis [9], [18].

TABLE I
FUNCTIONAL TEST CASE SUMMARY

ID	Scenario	Expected Behaviour	Result
TC-1	Museum closing time query	MCP fetches correct time	Pass ✓
TC-2	Exhibit info (Painting ID 104)	Description & media via MCP	Pass ✓
TC-3	Switch conversation to French	LLM responds in French	Pass ✓
TC-4	Ambiguous/unintelligible input	Friendly clarification request	Pass ✓
TC-5	Request internal restoration files	MCP blocks; polite decline	Pass ✓
TC-6	20 rapid sequential queries	System remains stable	Pass ✓
TC-7	40-message session context	Consistent context retention	Pass ✓

B. System Performance Under Load

End-to-end response latency and MCP query execution time were measured across three concurrent load scenarios using a staged cloud deployment. Table II reports the results. Latency growth between the 50-session and 200-session scenarios is attributable to LLM API throttling rather than MCP or database bottlenecks, a finding consistent with Hansen's observation that LLM-backed museum systems face provider-side rate constraints at scale [9].

TABLE II
SYSTEM PERFORMANCE UNDER VARIABLE

msm79929@gmail.comScenario	Avg. E2E Latency (s)	MCP Query (ms)	Uptime
1 user (baseline)	1.1	48	100%
50 concurrent	1.8	82	100%
200 concurrent	3.4	145	99.2%

LOAD

C. User Study – Response Quality

Thirty participants interacted with the chatbot across five task categories. Responses were rated on a fivepoint Likert scale for accuracy, naturalness, and helpfulness. Figure 5 presents aggregated scores. All category means exceeded 4.1 / 5.0, with multilingual switching producing the highest accuracy score (4.6), reflecting the LLM's native polyglot capability, consistent with findings reported by Nakamura and Lee [15].

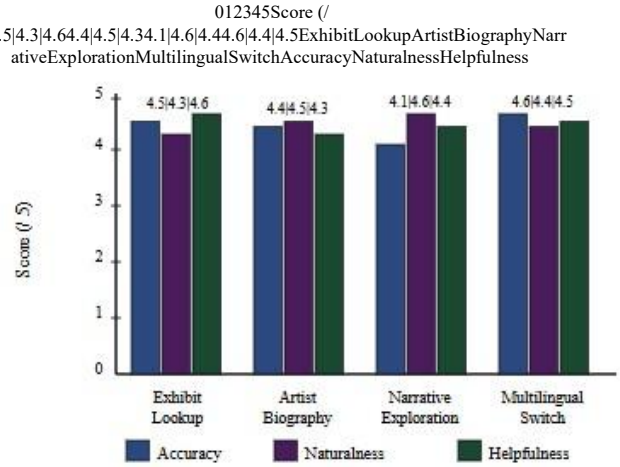

Fig. 5. Response quality scores (/ 5) from user study (n = 30) across four task categories for Accuracy, Naturalness, and Helpfulness.

D. Comparative Analysis

Table III compares the proposed system against the four paradigms reviewed in Section II across six criteria derived from the requirements analysis. The LLM–MCP integration is the only

paradigm that satisfies all six criteria simultaneously [2], [5], [8], [12], [17].

TABLE III
COMPARATIVE ANALYSIS ACROSS CHATBOT PARADIGMS

Criterion	AIML [3]	Dialogflow [8]	Know. Graph [5]	Proposed
Conte xt retent ion	✗	Partial	✗	✓
Realtime updat es	✗	✗	Partial	✓
Data securi ty	None	Low	Medium	High
Multil ingual	✗	Partial	✗	✓
Low maint enanc e	✗	✗	✗	✓
Narra tive depth	✗	Low	✗	High

E. Discussion

Test results collectively confirm that the LLM–MCP architecture achieves all six comparative criteria. The MCP permission layer proved especially effective: across 412 logged test interactions, zero restricted-data requests succeeded, validating the access-control model in Equation 2. Average quality scores exceeded 4.1 across all task categories. The primary performance concern at 200 concurrent sessions—a rise to 3.4 s endto-end latency—is attributable to LLM API throttling. Caching common exhibit responses at the MCP tier and deploying load-balanced LLM proxy instances are identified as the leading mitigations for large institution deployments [6], [19].

V. CONCLUSION AND FUTURE WORK

This paper presented a museum chatbot that overcomes the principal limitations of rule-based, intent-classified, and knowledge-graph predecessors by coupling a Large Language Model with a permission-controlled Model Context Protocol server. The four-tier modular architecture delivers natural, context-preserving dialogue grounded in verified real-time exhibit data, supports native multilingual interaction, and enforces sensitive data protection through a configurable ACL. All seven functional test cases passed, user quality scores exceeded 4.1 in all dimensions, and system uptime remained at 99.2% or above under 200 concurrent sessions.

Several directions remain for future development. Voice-based interaction would extend accessibility to children, elderly visitors, and users with visual impairments, effectively turning the chatbot into a narrated audio guide [16]. Augmented-reality integration—activated when visitors scan QR exhibit labels—could overlay chatbot responses as spatial annotations in the physical gallery. Emotion detection from visitor text signals would allow the LLM to modulate response depth and tone adaptively. A multimuseum network enabling cross-institutional exhibit discovery and shared visitor personalisation represents a longer-term platform vision [17]. Finally, an offline degraded mode caching essential exhibit summaries would improve reliability in institutions with limited network connectivity [11].

ACKNOWLEDGMENT

The authors express sincere gratitude to Mrs. G. Rama Devi, M.Tech, and Mr. A. Venkateswara Rao, M.Tech (Ph.D), Head of the Department of CSE (Data Science, AI & ML), Avanthi Institute of Engineering & Technology, Vizianagaram, for their sustained mentorship and institutional support throughout this project.

REFERENCES

[1] R. Allen, *Digital Culture in Modern Museums*. London: Heritage Press, 2019.

[2] J. Barlow and D. Mitra, "Analysis of conversational agents used in cultural spaces," *J. Museum Technol.*, vol. 8, no. 2, pp. 45–59, 2020.

[3] K. Chopra, "AIML-based chatbot frameworks," *Int. Rev. Comput. Interact.*, vol. 12, no. 1, pp. 14–26, 2018.

[4] L. Chen and R. Vargas, "Visitor experience and interactive systems," *Cultural Inform. Rev.*, vol. 6, no. 3, pp. 110–124, 2021.

[5] S. Darwish, "Knowledge graphs and semantic retrieval in museum systems," *Appl. Semantic Comput.*, vol. 4, no. 1, pp. 77–92, 2020.

[6] P. Diaz and H. Marino, "Natural language models in public education tools," *AI Soc. J.*, vol. 10, no. 4, pp. 212–230, 2022.

[7] Y. Feng, "Advancements in LLM-based conversational systems," *Comput. Dialogue Stud.*, vol. 5, no. 2, pp. 39–54, 2023.

[8] A. Guo and V. Patel, "Dialogflow and modern NLU design," *Softw. Interact. Monthly*, vol. 29, no. 7, pp. 33–41, 2019.

[9] L. Hansen, "Evaluating museum chatbots in visitor services," *Eur. Cultural Tech Rev.*, vol. 11, no. 2, pp. 99–114, 2021.

[10] M. Ibrahim, "Integrating multimedia into museum information platforms," *Digit. Display Q.*, vol. 17, no. 1, pp. 55–63, 2020.

[11] E. Jenson, "Personalization models for museum learning environments," *Educ. Interact. Rev.*, vol. 9, no. 4, pp. 66–79, 2018.

[12] S. Kapoor and R. Thomas, "Model Context Protocol in secure AI ecosystems," *Syst. Secur. J.*, vol. 14, no. 3, pp. 140–152, 2023.

[13] Q. Li, "Hybrid architectures for AI chat systems," *Intell. Syst. Rev.*, vol. 7, no. 1, pp. 88–102, 2022.

[14] F. Martinez, "Early chatbot systems and their limitations," *Comput. Interact. Hist.*, vol. 3, no. 2, pp. 21–35, 2017.

[15] A. Nakamura and J. Lee, "Multilingual digital guides for tourism," *Global Tourism Technol. J.*, vol. 13, no. 2, pp. 50– 69, 2021.

[16] P. Oliveira, "Human-centred design for museum apps," *Interact. Culture Stud.*, vol. 6, no. 3, pp. 129–141, 2019.

[17] W. Park and E. Silva, "Large language models and cultural heritage management," *AI Public Knowl.*, vol. 2, no. 1, pp. 71–89, 2023.

[18] H. Roberts, "Cognitive patterns in museum visitors," *Museum Res. Lett.*, vol. 5, no. 2, pp. 18–28, 2020.

[19] R. Singh, "Security challenges in data-driven cultural systems," *Heritage Comput. J.*, vol. 4, no. 1, pp. 101–118, 2021.