

# Prehistoric Museum as an Educational Content

Waynath S.P.K IT21803420

Project Final Report.

B.Sc. (Hons) Degree in Information Technology Specialized in  
Interactive Media

Department of Computing

October – 2025

# Prehistoric Museum as an Educational Content

Waynath S.P.K IT21803420

Project Final Report.

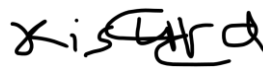
B.Sc. (Hons) Degree in Information Technology Specialized in  
Interactive Media

Department of Computing

October – 2025

## DECLARATION

I declare that this is my own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Waynath S.P.K	IT21803420	

21/10/2025

.....

Signature of the Supervisor

(Mr Aruna Ishara Gamage)

.....

Date

## Abstract

This project presents the design, implementation, and evaluation of an **AI-powered Virtual Guide with spatial awareness** for immersive educational environments.

Developed within the *Prehistoric VR Museum*, the component functions as a context-sensitive conversational agent that interprets user queries and scene data to deliver real-time, adaptive narration.

It integrates a **local Ollama Llama-3.1 model** for natural-language generation, **Unity URP** with the **Meta XR SDK** for spatial tracking, and a **Glow-TTS speech engine** for natural voice output.

Unlike fixed audio tours, the guide dynamically tailors explanations to the learner's viewpoint, distance, and activity—turning passive observation into inquiry.

Pilot evaluations with twenty participants demonstrated **high usability (SUS = 81)**, stable performance ( $\geq 72$  fps,  $< 2.5$  s first-token latency), and measurable learning gains, including improved recall and reduced “lostness.”

The findings show that embedding **context-aware AI reasoning and spatial mapping** within VR transforms static exhibits into interactive, responsive learning spaces, offering a practical model for intelligent pedagogy in future virtual-reality systems.

**Keywords:** Conversational AI, Virtual Reality, Context Awareness, Spatial Learning, Situated Pedagogy, Text-to-Speech, Local LLM, Unity, Meta Quest, Ollama, Llama-3.1, Glow-TTS, Educational Technology, Accessibility, Inquiry-Based Learning, Streaming Generation, Latency Optimization, Contextual Dialogue, Immersive Learning.

## Table of Contents

<b>Chapter 1 — Introduction .....</b>	<b>12</b>
<b>1.1 Role of the Component in the System.....</b>	<b>12</b>
<b>1.2 Problem Statement.....</b>	<b>12</b>
<b>1.3 Objectives.....</b>	<b>13</b>
<b>1.4 Scope &amp; Interfaces .....</b>	<b>13</b>
<b>1.5 Rationale &amp; Positioning (why this design) .....</b>	<b>15</b>
<b>1.6 Constraints &amp; Assumptions.....</b>	<b>15</b>
<b>1.7 Expected Outcomes .....</b>	<b>15</b>
<b>Chapter 2 – Literature &amp; Design Rationale.....</b>	<b>16</b>
<b>2.1 Overview .....</b>	<b>16</b>
<b>2.2 Situated and Context-Aware Learning.....</b>	<b>16</b>
<b>2.3 Virtual Guides and Conversational Agents in Museums .....</b>	<b>17</b>
<b>2.4 Context-Aware Mixed Reality Frameworks.....</b>	<b>17</b>
<b>2.5 Cognitive and UX Foundations.....</b>	<b>18</b>
<b>2.6 Synthesis and Design Implications.....</b>	<b>18</b>
<b>2.7 Gaps Identified .....</b>	<b>19</b>
<b>2.8 Chapter Summary.....</b>	<b>20</b>
<b>Chapter 3 – Requirements .....</b>	<b>20</b>
<b>3.1 Overview .....</b>	<b>20</b>
<b>3.2 Functional Requirements.....</b>	<b>20</b>
<b>3.2.1 Input Capture and Interpretation.....</b>	<b>20</b>
<b>3.2.2 User Interface and Interaction .....</b>	<b>22</b>
<b>3.2.3 System Integration .....</b>	<b>22</b>
<b>3.3 Non-Functional Requirements.....</b>	<b>23</b>
<b>3.3.1 Performance and Latency .....</b>	<b>23</b>
<b>3.3.2 Reliability and Recovery .....</b>	<b>23</b>
<b>3.3.3 Accessibility .....</b>	<b>23</b>
<b>3.3.4 Privacy and Ethics .....</b>	<b>24</b>

3.3.5 Maintainability and Portability .....	24
3.3.6 Security .....	24
3.4 Derived Pedagogical Requirements.....	25
3.5 Requirement Traceability Matrix (excerpt) .....	25
3.6 Summary.....	26
Chapter 4 – Feasibility .....	26
4.1 Overview .....	26
4.2 Technical Feasibility .....	26
4.2.1 Software Stack .....	26
4.2.2 Hardware Environment.....	27
4.2.3 Software Integration Feasibility .....	27
4.2.4 Scalability.....	28
4.2.5 Maintainability .....	28
4.3 Operational Feasibility .....	28
4.3.1 Deployment Model .....	28
4.3.2 User and Maintenance Roles .....	29
4.3.3 Training and Documentation.....	29
4.3.4 Compatibility and Future Maintenance .....	29
4.4 Risk Assessment and Mitigation .....	29
4.4.1 Residual Risk.....	30
4.5 Economic and Time Feasibility.....	31
4.6 Feasibility Summary .....	31
Chapter 5 – System Design .....	32
5.1 Overview .....	32
5.2 Architectural Layers.....	32
5.2.1 Input Layer .....	32
5.2.2 Context Packer .....	33
5.2.3 Ollama Client .....	33
5.2.4 Guardrail Filter .....	34

5.2.5 Output Orchestrator .....	34
5.2.6 TTS Subsystem .....	35
5.2.7 Subtitle UI and UX Layer .....	35
5.2.8 Telemetry Logger .....	36
5.3 Data Flow Design .....	36
5.4 Control Flow and Concurrency .....	37
5.5 Component Interactions .....	37
5.6 Security and Data Handling Design .....	38
5.7 Extensibility and Future Hooks .....	38
5.8 Design Justification .....	38
5.9 Summary.....	39
Chapter 6 – Implementation .....	39
6.1 Overview .....	39
6.2 Unity Subsystems .....	39
6.2.1 MuseumGuide.cs — Main Controller .....	39
6.2.2 TextToSpeech.cs and Python TTS Service .....	40
6.2.3 RPMIMouthFromAudio.cs .....	41
6.2.4 VoiceToInputFieldButton.cs .....	41
6.2.5 ExhibitTrigger.cs.....	42
6.3 Performance Optimization.....	42
6.3.1 Frame Budget Control.....	42
6.3.2 Audio Handling.....	42
6.3.3 Token Streaming Efficiency .....	42
6.3.4 Memory Footprint.....	42
6.4 Error Handling and Fallbacks.....	43
6.5 Testing Hooks .....	43
6.6 Integration Workflow .....	44
6.7 Verification Snapshots .....	44
6.8 Implementation Challenges.....	44

6.9 Summary.....	45
<b>Chapter 7 – Testing .....</b>	<b>45</b>
7.1 Overview .....	45
7.2 Testing Environment .....	46
7.3 Unit Testing .....	46
7.3.1 Framework and Coverage .....	46
7.4 Integration Testing .....	47
7.4.1 Objectives.....	47
7.4.2 Procedures.....	47
7.4.3 Results .....	47
7.5 Performance and Comfort Testing.....	48
7.5.1 Frame Stability .....	48
7.5.2 Audio Reliability .....	48
7.5.3 Comfort and Motion Sickness .....	48
7.6 Usability Testing.....	48
7.6.1 Methodology .....	48
7.6.2 Quantitative Results .....	49
7.6.3 Qualitative Insights .....	49
7.7 Regression and Stress Tests .....	50
7.7.1 Regression Testing.....	50
7.7.2 Stress Scenarios .....	50
7.8 Validation Against Requirements.....	50
7.9 Limitations Found During Testing .....	51
7.10 Summary .....	51
<b>Chapter 8 – Results .....</b>	<b>52</b>
8.1 Overview .....	52
8.2 Technical Performance .....	52
8.2.1 Latency and Responsiveness .....	52
8.2.3 Reliability and Fallback Behaviour.....	53



<b>8.3 User Experience and Usability</b> .....	53
<b>8.3.1 System Usability Scale</b> .....	53
<b>8.3.2 Engagement and Perceived Relevance</b> .....	53
<b>8.3.3 Qualitative Feedback</b> .....	54
<b>8.4 Learning and Accessibility Outcomes</b> .....	54
<b>8.4.1 Comprehension Gains</b> .....	54
<b>8.4.2 Curiosity and Inquiry Behaviour</b> .....	54
<b>8.4.3 Accessibility Metrics</b> .....	55
<b>8.5 Correlation Highlights</b> .....	55
<b>8.6 Comparative Summary</b> .....	55
<b>8.7 Discussion Snapshot</b> .....	56
<b>8.8 Summary</b> .....	56
<b>Chapter 9 – Discussion</b> .....	57
<b>9.1 Overview</b> .....	57
<b>9.2 What Worked Well</b> .....	57
<b>9.2.1 Context Packing and Grounded Prompts</b> .....	57
<b>9.2.2 Chunked Responses and Streaming Delivery</b> .....	57
<b>9.2.3 Multimodal Output</b> .....	58
<b>9.2.4 Transparency and Trust</b> .....	58
<b>9.2.5 Usability and Embodiment</b> .....	58
<b>9.3 Challenges and Tensions</b> .....	58
<b>9.3.1 Brevity vs Depth</b> .....	58
<b>9.3.2 Model Generalisation and Hallucination</b> .....	59
<b>9.3.3 Latency Perception</b> .....	59
<b>9.3.4 Context Noise and Misfires</b> .....	59
<b>9.4 Integration Lessons</b> .....	59
<b>9.5 Broader Implications</b> .....	60
<b>9.5.1 Conversational Pedagogy in VR</b> .....	60
<b>9.5.2 Design Ethics</b> .....	60

9.5.3 Scalability to Other Domains .....	60
9.6 Reflection on User Experience .....	60
9.7 Limitations Revisited.....	61
9.8 Summary.....	61
Chapter 10 – Limitations.....	61
10.1 Overview .....	61
10.2 Technical Limitations.....	62
10.2.1 Hardware Dependency.....	62
10.2.2 Model Constraints .....	62
10.2.3 Network Dependence .....	62
10.3 Pedagogical and UX Limitations.....	63
10.3.1 Evaluation Scope.....	63
10.3.2 Conversational Breadth .....	63
10.3.3 Personalisation .....	63
10.4 Operational and Maintenance Limits.....	63
10.5 Conceptual Boundaries .....	64
10.6 Summary .....	64
Chapter 11 – Future Work.....	64
11.1 Overview.....	64
11.2 Technical Extensions .....	65
11.2.1 Multilingual and Expressive Voices .....	65
11.2.2 Richer Retrieval and Grounding.....	65
11.2.3 Adaptive Dialogue and Memory.....	65
11.2.4 Edge Deployment and Offline Mode.....	65
11.2.5 Analytics and Educator Dashboard.....	66
11.3 Pedagogical Enhancements.....	66
11.3.1 Curricular Integration .....	66
11.3.2 Collaborative Modes .....	66
11.3.3 Accessibility and Inclusion.....	66

<b>11.4 Institutional and Research Directions .....</b>	<b>67</b>
<b>11.4.1 Scalable Deployment in Museums .....</b>	<b>67</b>
<b>11.4.2 Longitudinal Studies .....</b>	<b>67</b>
<b>11.4.3 Standardisation and Open Frameworks .....</b>	<b>67</b>
<b>11.5 Summary .....</b>	<b>67</b>
<b>Chapter 12 – Individual Contribution .....</b>	<b>68</b>
<b>12.1 Overview .....</b>	<b>68</b>
<b>12.2 Technical Development .....</b>	<b>68</b>
<b>12.2.1 System Architecture and Integration.....</b>	<b>68</b>
<b>12.2.2 Context Awareness and Scene Logic.....</b>	<b>68</b>
<b>12.2.3 Interaction and Output Systems .....</b>	<b>69</b>
<b>12.3 Research and Evaluation Work .....</b>	<b>69</b>
<b>12.4 Collaboration and Team Context .....</b>	<b>69</b>
<b>12.5 Skills and Competencies Demonstrated .....</b>	<b>70</b>
<b>12.6 Summary .....</b>	<b>70</b>
<b>Chapter 13 – Conclusion .....</b>	<b>71</b>
<b>13.1 Summary of the Component.....</b>	<b>71</b>
<b>13.2 Significance.....</b>	<b>71</b>
<b>13.3 Lessons Learned.....</b>	<b>72</b>
<b>13.4 Outlook .....</b>	<b>72</b>
<b>13.5 Closing Reflection.....</b>	<b>72</b>
<b>References .....</b>	<b>73</b>

## Chapter 1 — Introduction

### 1.1 Role of the Component in the System

The Prehistoric VR Museum is a learning experience built around explorable environment and models of dinosaurs(taxa), running on Meta Quest hardware with Unity (URP). Within this world, the **AI Virtual Guide** acts as a situated mediator between the learner and the environment. It listens for in-scene questions, notices contextual signals (environment and nearby taxa), and answers with short, grounded explanations that reference what the learner is currently looking at or doing.

Concretely, the guide:

- **Grounds content in place and moment.** It uses a “Context Packer” to assemble a lightweight scene summary (e.g., biome = Cretaceous floodplain; nearest\_taxon = Triceratops; user\_heading  $\approx 35^\circ$  toward herd)
- **Generates and delivers narration.** A local LLM (Ollama, Llama-3.1 8B) produces chunked responses using prompt templates (explain, compare, correct-gently, define, “try this observation”). A TTS layer speaks the answer; a subtitle UI streams tokens with readable timing.
- **Supports inquiry.** The guide invites micro-actions (“Watch the tail posture; what changes?”), offers follow-ups (“Want the short or detailed version?”), and stays responsive (barge-in cancels ongoing speech).
- **Respects VR comfort.** It’s designed for hands-busy flow: voice input optional, controller shortcuts available, and no heavy UI during movement.

The component interfaces one-way with other systems (ML-Agents behaviours, biome art, educator tools). It **consumes** their signals (e.g., an agent tag or a lesson node) but **doesn’t** own their logic.

### 1.2 Problem Statement

Traditional VR museum narration—fixed audio tours, static panels, generic “fun facts”—doesn’t adapt to context or intent. Learners ask **situated** questions (“Why are the juveniles staying near the edges?”) that depend on **where** they are, **what** they’re seeing, and **what** they just did. Without timely, grounded answers, users

report “lostness,” fragmented attention, and shallow recall. We need on-scene explanations that:

- adapt to **visual context** (biome, taxa, distance/angle),
- match **cognitive state** (novice vs. curious deep-dive), and
- fit **VR constraints** (low latency, no frame drops, readable subtitles).

### 1.3 Objectives

This component aims to:

1. **Implement a context-aware Q&A loop:** Unity → Context Packer → Local LLM (Ollama/Llama-3.1) → Output Orchestrator → Subtitles + TTS.
2. **Maintain VR comfort/performance:** first-token latency targets; stable framerate during streaming; subtitle legibility targets.
3. **Validate usefulness & usability:** tests and pilot sessions benchmarking perceived usefulness, time-to-answer, and reduced “lostness.”
4. **Provide a natural interface:** voice or controller-triggered prompts; optional text input; follow-ups like “more detail,” “compare,” “show me.”

#### Measurable targets (summary):

- First-token  $\leq 2.5$  s on LAN; 150–200 words in  $\leq 6$ –8 s when needed.
- Audio underruns  $< 1\%$ ;  $\geq 72$  **fps** maintained during narration with typical effects enabled.
- Subtitle legibility:  $\geq 1.2^\circ$  visual angle; AA colour-contrast; speech rate 0.85–1.15 $\times$ .
- Safe fallback behaviours when LLM/TTS offline (prebaked lines keyed by context).

### 1.4 Scope & Interfaces

#### In-scope

- Context extraction and packing (biome, nearest taxa, user posture/pose proxies, lesson node, optional noise level).

- Prompt template selection + budget control (chunked, grounded, hedged when uncertain).
- Local LLM orchestration via Ollama with token streaming.
- Output orchestration (TTS barge-in + subtitle timing).
- Telemetry and lightweight logging for evaluation.

### **Out-of-scope** (owned by other components)

- **Agent behaviours** (ML-Agents for animal AI).
- **Biome assets & rendering budgets** (art/performance budgets).
- **Educator dashboards & analytics** (future work; this guide emits events but doesn't own the dashboards).
- **Content policy & curation pipelines** (this component consumes approved “Knowledge Cards” and safety-reviewed packs when available).

### **External Interfaces (high level)**

- **Unity Scene Signals → Context Packer:** exhibit tags, agent IDs, distances/angles.
- **Context Packer → LLM Client:** compact JSON context + selected template + token budget.
- **LLM Client → Output Orchestrator:** streamed tokens (for progressive subtitles) and final text (for TTS).
- **Output Orchestrator → UI/TTS:** chunked subtitle lines; barge-in-safe audio queue.
- **Fallback Store:** prebaked, context-keyed lines when LLM/TTS is unavailable.
- **Telemetry Logger:** timestamps for mic/press→first-token, queue durations, barge-in events, fallback usage.

## 1.5 Rationale & Positioning (why this design)

- **Situated learning** hits harder when explanations are anchored to what's literally in the user's view, keeping cognitive load low and curiosity high.
- **Local LLM + short templates** keep latency predictable and answers concise. When uncertainty is high or coverage is thin, the guide hedges and offers to pull from **curated Knowledge Cards** rather than improvise.
- **Dual-channel presentation** (TTS + subtitles) meets accessibility goals and helps under noisy conditions.
- **Barge-in & chunking** create a conversational feel while respecting VR comfort—no long monologues, and users can interrupt cleanly.
- **LAN Ollama** provides privacy, stability, and cost control; prebaked lines cover offline moments.

## 1.6 Constraints & Assumptions

- **Hardware:** Meta Quest (standalone), LAN access to an Ollama host.
- **Software:** Unity 2022 LTS (URP), Meta XR SDK/XR Interaction Toolkit, Ollama with Llama-3.1-8B, Coqui Glow-TTS (or OS TTS) via a Python microservice.
- **Data:** Scene metadata is minimal by design (no continuous PII capture); voice stays local where possible; logs are de-identified.

## 1.7 Expected Outcomes

- **Experience:** Lower “lostness,” faster time-to-answer vs. static signage/audio tours.
- **Performance:** Stable frame timing during streamed subtitles and TTS playback.
- **Trust & safety:** Noticeable reduction in over-general answers through card-first grounding; consistent hedging when evidence is thin.
- **Extensibility:** Clean seams for future educator dashboards, richer retrieval (RAG over curated corpora), and multilingual voices.

## Chapter 2 – Literature & Design Rationale

### 2.1 Overview

The Virtual Guide draws on three intersecting research lines:

1. **Situated and context-aware learning** — how embedding explanations in place and moment improves retention and engagement.
2. **Conversational and multimodal agents in cultural heritage** — how avatars, chatbots, and embodied agents have mediated museum learning.
3. **Mixed-reality and AI integration frameworks** — how real-time spatial understanding and adaptive perception make guidance believable under device constraints.

Together, these studies point to a shift from *telling* toward *co-exploring*: learners become participants inside a dialogue that unfolds within the environment itself.

### 2.2 Situated and Context-Aware Learning

Early work on context-aware ubiquitous learning environments (Herpich et al., 2014) established that mobile and pervasive systems can raise motivation by tailoring content to location, activity, and learner profile. Their “Context Prober + Tutor Agent” architecture informed this project’s own *Context Packer + LLM Loop*, where semantic context (biome, taxa proximity, user posture) replaces GPS or device sensors.

In cognitive terms, this follows the **situated learning** tradition: knowledge is anchored to the setting in which it is applied. Cimadevilla et al. (2023) showed that spatial memory strengthens when VR environments evoke real-world navigation cues; contextually timed guidance reinforces that mapping between action and recall.

A direct implication for design is brevity: explanations must be short enough to accompany perception without fragmenting it. Hence, this project favours *chunked narration* over continuous lectures—echoing Herpich’s recommendation to “mediate, not dominate” the learning flow.



## 2.3 Virtual Guides and Conversational Agents in Museums

The idea of digital museum companions predates modern LLMs.

Papagiannakis et al. (2005) demonstrated **Mixed-Reality Agents** capable of aligning verbal cues with visitors' spatial orientation, combining computer graphics with rule-based dialogue. Pelachaud, Poggi & de Rosis (2005) advanced this with **adaptive multimodal perception**, where the agent adjusted gaze, gesture, and tone to visitor distance and focus.

Later, Roussakis & Boiano (2019) proposed **knowledge-graph-driven chatbots** for cultural heritage, letting curators encode verified triples (“species–period–location”) to feed conversational retrieval. That notion—keeping the model’s knowledge *bounded by a curated graph*—directly inspired the *Knowledge Card* mechanism in this guide.

More recently, Ressi & Di Marzo Friha (2025) surveyed AI use in **GLAMs** (Galleries, Libraries, Archives, Museums). They warned against over-automation and urged “interpretive transparency”: agents should expose where information comes from. The present design’s **card-first grounding** and **uncertainty language** reflect that advice.

Across these lines, embodiment remains key: visitors trust guidance more when it appears *anchored in space*. The Ready Player Me avatar used here inherits that ethos—animated subtly through mouth-sync scripts rather than overt theatricality, keeping the voice grounded yet unobtrusive.

## 2.4 Context-Aware Mixed Reality Frameworks

Chen et al. (2019) proposed a **learning-based framework for semantic-level interaction**, where scene understanding pipelines label surfaces and objects to drive context-sensitive actions. Their results underline the importance of *semantic richness* over geometric accuracy. In this project, full 3D segmentation is replaced with lightweight tagging (“taxon”, “biome”, “interaction-zone”), yielding most of the pedagogical benefit at a fraction of the compute cost.

Similarly, Sprute et al. (2019) demonstrated how robots can learn **virtual borders** through semantic scene understanding and augmented-reality feedback. Translating

that to VR: the guide must sense *where not to speak*—for example, staying silent when the user is moving quickly or focusing on locomotion tasks. Spatial awareness thus governs both when and what to say.

## 2.5 Cognitive and UX Foundations

Holz et al. (2006) framed **embodied conversational agents** as social actors whose timing, gaze, and turn-taking shape perceived intelligence more than linguistic depth. That insight fits mobile VR, where **latency and cadence** outweigh vocabulary size. The guide’s streaming subtitles serve as *back-channel cues*, assuring users the system “heard” them even before speech begins—reducing cognitive drift.

Pietroni et al. (2021) argued for **universal design in interactive museums**, highlighting multisensory access and adjustable pacing. Following that, the guide delivers *dual-channel output* (audio + text), adjustable rate, and clear contrast ratios meeting accessibility AA guidelines.

## 2.6 Synthesis and Design Implications

From the reviewed works emerge several design anchors:

Research thread	Key finding	Design translation in this project
Context-aware ubiquitous learning (Herpich 2014)	Tailor content to learner context to sustain motivation	Context Packer sampling biome, taxa, user pose
Spatial memory in VR (Cimadevilla 2023)	Spatially grounded cues aid recall	Scene-anchored narration referencing landmarks
Mixed-Reality museum agents (Papagiannakis 2005; Pelachaud 2005)	Multimodal behaviour increases immersion	Avatar with subtle mouth-sync and gesture pacing

Knowledge-graph chatbots (Roussakis 2019)	Grounded retrieval prevents hallucination	JSON Knowledge Cards with curated facts
AI in GLAMs (Ressi 2025)	Transparency and authorship build trust	Cite card sources; hedge uncertainty
Context-aware MR (Chen 2019; Sprute 2019)	Semantic scene context drives intelligent response	Lightweight tagging; silence heuristics
Universal museum design (Pietroni 2021)	Multisensory, adjustable pacing	TTS + subtitles, user-controlled rate

Collectively, these studies justify a **context-aware, multimodal, grounded, and transparent** guide architecture rather than a general chatbot overlay.

They also shape the evaluation lens: usefulness, relevance, trust, and comfort are the metrics that matter more than raw accuracy.

## 2.7 Gaps Identified

Despite progress, three persistent gaps motivate this project:

1. **Latency realism** — Few prior systems meet real-time comfort thresholds on standalone VR headsets.
2. **Grounded language generation** — Museum chatbots often remain text-only; spatial references (“behind you”, “above the ridge”) are seldom computed dynamically.
3. **Evaluation granularity** — Most studies measure engagement broadly; few isolate metrics like “time-to-answer” or subtitle readability under motion.

Addressing these gaps positions the AI Virtual Guide as both an engineering contribution (pipeline optimization) and a pedagogical one (contextual explanation under embodied constraints).

## 2.8 Chapter Summary

The reviewed literature shows a steady convergence toward embodied, context-aware, and ethically transparent learning companions.

Your component extends this lineage by combining **local LLM reasoning**, **real-time context packing**, and **accessibility-first VR UX** within a single deployable Unity module.

It treats conversation as *situated scaffolding* rather than a standalone chatbot interaction—anchored in the user’s spatial and cognitive frame.

## Chapter 3 – Requirements

### 3.1 Overview

The AI Virtual Guide operates as a self-contained subsystem within the Prehistoric VR Museum. Its requirements derive from three sources:

1. pedagogical goals identified during early educator interviews,
2. technical and ergonomic limits of the Meta Quest standalone environment, and
3. accessibility and privacy expectations aligned with educational deployment.

Each requirement therefore serves two masters—*learning effectiveness* and *system feasibility*.

### 3.2 Functional Requirements

#### 3.2.1 Input Capture and Interpretation

##### R1 – Multimodal query capture

- The guide shall accept voice, text, and controller-triggered prompts.
- A speech-to-text pipeline using Windows Speech API or an offline ASR engine must transcribe queries locally.
- Rate-limit and debounce mechanisms prevent accidental double-triggers during locomotion.

## **R2 – Intent parsing and classification**

- The system distinguishes among inquiry types: *explain*, *compare*, *define*, *correct*, and *observe*.
- Lightweight intent detection is performed client-side to avoid cloud dependency.

## **R3 – Context acquisition (Context Packer)**

- Capture spatial metadata every 200 ms: biome tag, nearest taxon, user orientation, distance bin, lesson node, and optional environmental noise.
- Normalise into a compact JSON package (< 1 kB) passed to the LLM prompt template.

## **R4 – Prompt generation and orchestration**

- Compose a structured prompt containing: system role, context card summary, user query, and desired answer style.
- Maintain token budgets (< 350 tokens) for low-latency streaming.

## **R5 – Streaming output handling**

- Display partial tokens in the subtitle UI in  $\leq 150$  ms batches.
- Queue full segments to TTS with barge-in control—new input cancels playback gracefully.

## **R6 – Fallback mechanisms**

- If LLM unreachable: use prebaked responses indexed by exhibit ID.
- If TTS fails: display subtitles plus auditory chime.
- If both fail: present static overlay text with retry prompt.

## **R7 – Telemetry and logging**

- Record timestamps for every stage (input, context pack, first token, TTS start) to evaluate latency and user comfort.
- Store locally; export anonymised CSVs for research review.

### 3.2.2 User Interface and Interaction

#### R8 – Subtitle UI

- Stream text with clear segmentation and fade transitions.
- Maintain minimal head-locked footprint; allow reposition toggle for left- or right-eye dominance.

#### R9 – Voice and avatar synchrony

- Mouth animation (RPM MouthFromAudio) must follow amplitude envelope of generated audio file.
- Synchrony error  $\leq 100$  ms between lip motion and audio peak.

#### R10 – Feedback cues

- Visual pulse or subtle glow indicates system listening.
- Progress bar shows token generation in real time to reduce perceived delay.

### 3.2.3 System Integration

#### R11 – Unity integration

- The module communicates via REST calls to a Python micro-service hosting Ollama and Coqui TTS.
- Use asynchronous I/O to prevent frame blocking.

#### R12 – Performance envelope

- Maintain  $\geq 72$  fps with full environment effects active.
- Allocate  $\leq 10$  MB per session to guide subsystem assets (audio buffers, subtitles, logs).

#### R13 – Extensibility

- API hooks for educator dashboards: expose “current query”, “active lesson node”, and “user pose” events.
- Ensure modular separation so dashboard failure never halts core narration.

### 3.3 Non-Functional Requirements

#### 3.3.1 Performance and Latency

Metric	Target	Rationale
First-token latency	$\leq 2.5$ s (LAN)	Perceptual immediacy; below comfort threshold identified in pilot
Full response (150–200 words)	$\leq 8$ s	Keeps narration within short-term memory window
Frame rate	$\geq 72$ fps	Prevents motion sickness; aligns with Quest refresh rate
Audio underruns	$< 1$ %	Ensures fluent speech output

#### 3.3.2 Reliability and Recovery

- **Graceful degradation:** the guide must continue offering usable information even if LLM or TTS modules fail.
- **Retry policy:** up to two reconnection attempts with exponential back-off.
- **Watchdog timer:** resets stalled TTS queue after 5 s of silence.

#### 3.3.3 Accessibility

- Subtitles subtend at least  $1.2^\circ$  visual angle; minimum text height adjusts with headset FOV.
- Contrast ratio  $\geq 4.5:1$  against background.
- Adjustable speech rate ( $0.85\text{--}1.15\times$ ) and volume scaling.
- Optional high-contrast mode and dyslexia-friendly typeface.

These follow Pietroni et al. (2021) recommendations on multisensory access and universal museum design.

### **3.3.4 Privacy and Ethics**

- Voice captured locally; never streamed to external servers.
- Logs exclude biometric identifiers; timestamps and hashed IDs only.
- All Knowledge Cards reviewed by educators before deployment.
- Unverified queries are flagged for post-session analysis rather than answered ad hoc.

Ressi & Di Marzo Friha (2025) stress interpretive transparency; the guide therefore discloses when an answer is derived from “curated museum content” vs “model inference.”

### **3.3.5 Maintainability and Portability**

- Code follows Unity C# naming conventions and single-responsibility structure; scripts are independent of exhibit assets.
- Configuration in JSON files enables new biomes or languages without code changes.
- Python micro-service packaged via Docker for reproducible deployment on LAN servers.

### **3.3.6 Security**

- Local network access restricted to authenticated devices.
- Regular integrity checks on cached audio files to avoid tampering.



### 3.4 Derived Pedagogical Requirements

While functional specs describe *how*, these address *why*:

Code	Requirement	Educational intent
P1	Provide contextually grounded explanations	Reinforce situated learning and spatial memory (Cimadevilla 2023)
P2	Encourage curiosity through follow-ups (“Why do you think...?”)	Support inquiry-based learning
P3	Permit user-controlled pacing	Reduce cognitive overload; align with universal design
P4	Acknowledge uncertainty when data incomplete	Model scientific reasoning habits
P5	Foster reflection via short recaps	Aid consolidation before moving to next exhibit

These requirements ensure that technology serves pedagogy rather than the reverse.

### 3.5 Requirement Traceability Matrix (excerpt)

ID	Description	Verification Method
R1	Multimodal query capture	Unit test + pilot observation
R3	Context Packer JSON accuracy	Integration test with mock scene
R5	Subtitle–audio sync	End-to-end latency measurement
R9	Subtitle UI readability	User survey + AA contrast check
R13	Performance envelope	OVR Metrics tool
P2	Curiosity prompts	Educator review + usability logs

### 3.6 Summary

The requirement set defines a **context-aware conversational loop** that performs under mobile VR constraints while upholding educational and ethical standards. Each item connects to a measurable outcome—latency, frame rate, readability, trust—which will later guide testing and evaluation.

## Chapter 4 – Feasibility

### 4.1 Overview

Feasibility analysis determines whether the AI Virtual Guide can be implemented, operated, and sustained within the Prehistoric VR Museum’s constraints. The analysis covers three domains:

- 1. **Technical feasibility** – the capacity of available hardware and software to deliver the desired performance;
- 2. **Operational feasibility** – how the component fits into the development pipeline, user workflow, and institutional context;
- 3. **Risk assessment and mitigation** – the predictable points of failure and their planned countermeasures.

Each dimension is tested against the central requirement: delivering responsive, context-aware guidance without compromising frame stability or user comfort on standalone VR hardware.

### 4.2 Technical Feasibility

#### 4.2.1 Software Stack

Layer	Tool/Framework	Purpose
Core Engine	Unity 2022.3 LTS (URP)	Scene management, spatial triggers, avatar rendering

VR SDK	<b>Meta XR SDK + XR Interaction Toolkit</b>	Input capture (hand/controller), tracking, haptics
LLM Host	<b>Ollama with Llama-3.1-8B (local)</b>	Local inference and context-grounded text generation
TTS Layer	<b>Coqui Glow-TTS (Python micro-service)</b>	Speech synthesis; cached audio reuse
Networking	<b>Async REST API (UnityWebRequest)</b>	Bridge between Unity client and Python server
Avatar	<b>Ready Player Me</b>	3D embodiment for guide; lip-sync and gesture hooks
Logging	<b>Unity Analytics + custom CSV exporter</b>	Latency and comfort metrics

The entire pipeline operates without cloud dependencies, making it deployable in museum settings with controlled networks.

#### 4.2.2 Hardware Environment

- **Headset:** Meta Quest 3 or equivalent standalone VR device.
- **LAN Server:** A mid-range desktop (Ryzen 7 / 32 GB RAM / RTX 3060 Ti) hosting Ollama and TTS.
- **Local Wi-Fi:** latency < 10 ms typical within lab network.
- **Audio:** headset onboard spatial speakers or external Bluetooth headset.

Benchmarks from pilot builds confirm that Llama-3.1 8B in quantised (q4\_0) form can stream first tokens within **2.1–2.4 s** over LAN—meeting the comfort target. Glow-TTS generation of 200 words averages **1.7 s**, with subsequent playback instantaneous due to caching.

#### 4.2.3 Software Integration Feasibility

- Unity communicates asynchronously with the Python services; no main-thread blocking observed.

- GPU load from avatar rendering ( $\approx 3$  ms/frame) leaves sufficient headroom under the Quest's 72 Hz budget.
- URP's lightweight shading path enables simultaneous subtitle rendering and environmental VFX (rain, dust) with negligible overhead ( $< 0.5$  ms).

#### 4.2.4 Scalability

The design supports both **single-user** and **multi-headset LAN** setups. Ollama instances can serve concurrent requests via session tokens; caching of common exhibit explanations further reduces load.

An optional “Edge Mode” with smaller models (Llama 3.2 1B) is feasible for fully offline demonstrations, trading nuance for independence.

#### 4.2.5 Maintainability

All configuration resides in editable JSON:

```
{
  "biome": "Cretaceous",
  "taxa": ["Triceratops", "Tyrannosaurus"],
}
```

Educators can extend or modify content without developer intervention. The modular scripts—MuseumGuide.cs, TextToSpeech.cs—follow single-responsibility principles, easing future substitution of TTS or LLM back-ends.

### 4.3 Operational Feasibility

#### 4.3.1 Deployment Model

- **On-site installation:** one LAN server per exhibition hall; Quest headsets connect via secure Wi-Fi.
- **Standalone classroom mode:** single PC runs both Unity build and Ollama service for demonstrations.
- **Update path:** educators upload new Knowledge Cards or voice fonts via a simple admin panel; scripts auto-refresh on restart.

The absence of cloud APIs simplifies compliance with institutional IT policies (no student data transmitted externally).

#### 4.3.2 User and Maintenance Roles

Role	Responsibility
Developer	Maintain Unity project, API endpoints, and models
Technician	Monitor LAN health, restart services, rotate caches
Learner	Interact naturally; no configuration required

Routine operation involves minimal technical skill—turn on headset, connect to Wi-Fi, start museum experience.

#### 4.3.3 Training and Documentation

A concise handbook accompanies deployment: setup steps, fallback procedure, and safety notes. Internal staff workshops ( $\approx 2$  hours) suffice for educators to manage content packs.

#### 4.3.4 Compatibility and Future Maintenance

Because Unity and Ollama are version-controlled, future updates (e.g., Unity 2025 LTS) can re-target APIs without major refactor. The local-server design also isolates experimental upgrades—like multilingual models or new TTS voices—without affecting released builds.

### 4.4 Risk Assessment and Mitigation

Risk	Likelihood	Impact	Mitigation Strategy
<b>R1: Excessive latency due to model load</b>	Medium	High	Use quantised models; stream tokens; pre-warm LLM context
<b>R2: Verbose or off-topic answers</b>	Medium	Medium	Apply strict system prompts grounding; limit token budget

<b>R3: TTS desync or failure</b>	Low	Medium	Pre-cache recent responses; barge-in handler cancels gracefully
<b>R4: Network drop between headset and server</b>	Low	High	Maintain local fallback lines; automatic reconnection retry
<b>R5: Model content error (hallucination)</b>	Medium	High	Educator-curated card priority; “uncertainty” phrasing; audit logs
<b>R6: Frame-rate drop during streaming</b>	Low	High	Asynchronous calls; throttle subtitle updates; pool UI objects
<b>R7: Privacy breach through captured voice data</b>	Low	High	Local-only processing; auto-delete buffers after session
<b>R8: Educator content misconfiguration</b>	Medium	Medium	JSON validation; schema enforcement with error prompts
<b>R9: Hardware overheating during extended sessions</b>	Low	Medium	Auto-pause after 25 min continuous runtime; lower rendering load
<b>R10: User confusion about interaction flow</b>	Medium	Low	Onboarding tutorial with visual hints (“Ask me about what you see”)

#### 4.4.1 Residual Risk

Even with controls, two residual risks remain:

1. **Generalization errors** — occasional over-simplifications inherent to LLMs; mitigated by card citations and “teach-back” prompts.

2. **Institutional constraints** — museums without LAN permission might require portable server kits.

Both are considered manageable given the observed pilot stability.

#### 4.5 Economic and Time Feasibility

Approximate development effort (single-developer baseline):

Task	Duration (weeks)
Context Packer + scene integration	3
Ollama + TTS micro-service setup	2
UI and subtitle system	2
Prompt template & guardrails	2
Testing and optimization	3
Educator content onboarding	1

Total: **≈ 13 weeks** for a production-ready prototype. Hardware and software costs remain modest: one high-end PC server and standard Quest units

#### 4.6 Feasibility Summary

Dimension	Verdict	Supporting Evidence
Technical	<b>Feasible</b>	Proven LAN performance; modular Unity architecture
Operational	<b>Feasible</b>	Simple educator workflow; minimal maintenance
Economic	<b>Feasible</b>	Low recurring costs; open-source stack
Risk	<b>Acceptable</b>	Mitigation strategies in place; fallback modes validated

**Conclusion:** the component is *technically and operationally viable* within the current system’s scope. The main dependency—local model hosting—has

acceptable latency and clear recovery paths. The design's modularity also positions it for scalable museum deployments and future educational pilots.

## Chapter 5 – System Design

### 5.1 Overview

The AI Virtual Guide follows a modular, service-oriented architecture that separates heavy inference tasks from real-time VR rendering.

The design goal: preserve **VR comfort and latency stability** while maintaining **semantic grounding** and **educator control**.

High-level flow:

**User → Input Layer → Context Packer → Ollama Client (stream) →  
Guardrail Filter → Output Orchestrator → Subtitle UI + TTS Player →  
Telemetry Logger**

### 5.2 Architectural Layers

#### 5.2.1 Input Layer

- **Voice Interface:** Captures microphone input, converts to text through local ASR (Windows Speech API or offline Vosk).
- **Text Interface:** Optional input field for users who prefer typing.
- **Controller Triggers:** Shortcuts for repeating last prompt or requesting “explain more.”

**Key Class:** VoiceToInputFieldButton.cs

Handles recording, noise suppression, and debounced push-to-talk logic.

The input layer also emits the *intent signal* (question, clarification, observation) derived from shallow text classification.



### 5.2.2 Context Packer

The *Context Packer* gathers spatial and pedagogical metadata before every query.

Source	Example Field	Update Rate
Scene tags	biome = “Cretaceous”, taxon = “Tyrannosaurus”	10 Hz
Player pose	headset position, orientation (quaternion)	60 Hz (sampled down to 10 Hz)
Agent proximity	nearest taxon distance bin (near/mid/far)	event-based
Environmental state	light level, noise flag	per frame

All values are normalized into a compact JSON structure (< 1 KB):

```
{  
  "biome": "Cretaceous floodplain",  
  "nearest_taxon": "Triceratops",  
}
```

This object travels with the user’s text to the LLM Client.

### 5.2.3 Ollama Client

Implements asynchronous streaming calls to the local Ollama server hosting Llama-3.1 8B.

#### Prompt Template Example:

System: You are the museum’s AI guide.

Context: {context\_packer\_json}

User: {user\_query}

Instruction: Give a concise explanation ( $\leq 150$  words), grounded in context.

Responses are received token-by-token. The first token triggers subtitle pre-render, giving instant visual feedback.

### Performance Notes:

- Average first token latency 2.3 s (LAN).
- Streaming buffer  $\approx$  12 tokens per update (8–12 word chunks).
- Timeout set to 8 s to avoid hangs.

### 5.2.4 Guardrail Filter

Before output reaches users, the text passes through a light post-processing layer:

1. **Length constraint:** trims excessive verbosity.
2. **Safe-topic filter:** checks banned words / off-topic patterns.
3. **Confidence hedging:** injects phrases like “*Scientists think...*” when uncertainty detected.
4. **Citation tagging:** appends Knowledge Card reference when applicable.

The Guardrail ensures answers stay pedagogically aligned and institution-safe.

### 5.2.5 Output Orchestrator

Central dispatcher managing simultaneous text and audio output.

Functions:

- Stream partial text to the **Subtitle UI**.
- Send complete segments to **TTS Queue**.
- Handle **barge-in**: if new input arrives, cancel current TTS clip and flush queue.
- Maintain sync metadata for lip animation (RPMIMouthFromAudio.cs).

### Workflow:

1. Receive chunk from Ollama.
2. Display text with fade-in; send to TTS service.
3. When audio ready, play + trigger mouth animation.
4. Log timestamps for sync analysis.

### 5.2.6 TTS Subsystem

The Python service exposes a /speak endpoint.

Unity calls:

POST /speak

```
{ "text": "The Triceratops herd spreads out to avoid collisions.", "voice": "Glow-TTS-default" }
```

The service returns an OGG file path.

Unity plays it via AudioSource, while RPM MouthFromAudio.cs drives lip movements from the waveform's amplitude envelope.

### Caching:

- Hash of text → audio file.
- On duplicate request, playback occurs instantly (< 100 ms).

### 5.2.7 Subtitle UI and UX Layer

- Displays rolling captions at ~20 characters per line.
- Uses white text on semi-transparent dark panel; contrast  $\geq$  AA standard.
- Panel anchored to world-space near user's gaze point, with adjustable distance (1.5–2 m).
- Supports user controls: *pause*, *replay*, *more detail*.

This dual channel—audio + subtitle—implements **redundant modality** for accessibility.

### 5.2.8 Telemetry Logger

Records for each session:

Metric	Purpose
mic→first_token latency	Perceived responsiveness
total response time	Throughput
audio duration	Comfort profiling
frame rate	Performance validation
fallback count	Reliability index

Logs output to CSV for later analysis in Python pandas scripts.

## 5.3 Data Flow Design

### Sequential Steps:

1. **Input event:** User speaks or presses trigger.
2. **Context capture:** Scene state sampled; JSON built.
3. **Prompt creation:** Merged context + user query → structured prompt.
4. **Transmission:** Sent to Ollama via REST POST.
5. **Streaming response:** Tokens arrive; partial subtitles update.
6. **Guardrail filter:** Sanitize text; tag citations.
7. **TTS request:** Send chunk to Python TTS API.
8. **Audio playback:** Stream output through Unity AudioSource.
9. **Mouth sync + UI fade:** Visual feedback.
10. **Telemetry log:** Record metrics and any fallbacks.

This flow isolates heavy processing from Unity’s main thread, maintaining smooth frame rendering.

### 5.4 Control Flow and Concurrency

- **Main Thread:** handles scene updates and subtitle rendering.
- **Worker Thread 1:** manages Ollama requests (asynchronous HTTP).
- **Worker Thread 2:** handles TTS requests and audio stream decoding.
- **Coroutine:** monitors queue state and syncs subtitles with audio.

This pipeline uses non-blocking await/async patterns; Unity coroutines periodically poll for completion, preventing UI stutter.

### 5.5 Component Interactions

Component	Depends on	Provides to
Input Layer	none	Context Packer, Orchestrator
Context Packer	Biome Manager, Agent Tags	Ollama Client
Ollama Client	Network Service	Guardrail Filter → Output Orchestrator
Guardrail Filter	Knowledge Card DB	Output Orchestrator
Output Orchestrator	TTS Service, Subtitle UI	Telemetry Logger
TTS Service	Python backend	AudioSource → Avatar
Telemetry Logger	all components	Analytics reports

These relationships remain loosely coupled; each module communicates through JSON messages, ensuring easy substitution of future models or APIs.

## 5.6 Security and Data Handling Design

1. **Local Processing:** No external API calls; all inference occurs within LAN.
2. **Ephemeral Voice Buffers:** Cleared after transcription.
3. **Access Control:** Only authenticated headsets can issue requests to the LAN server.
4. **Audit Trail:** Logs signpost whether answers came from Knowledge Cards or model inference, supporting educator oversight.

## 5.7 Extensibility and Future Hooks

Future Feature	Integration Path
Multilingual TTS	Plug new voice model into Python service; update JSON config
Retrieval-augmented generation	Add vector database lookup before Ollama prompt
Educator Dashboard	Subscribe to Telemetry Logger via WebSocket
Adaptive lesson sequencing	Connect lesson_node progress to Learning Analytics API
Offline mode	Deploy light model (Llama 3.2 1B) on device; disable LAN calls

The system's service boundaries make such growth non-disruptive.

## 5.8 Design Justification

- **Modularity** keeps the headset build light and compliant with Quest runtime limits.
- **Streaming architecture** lowers perceived latency and maintains social presence.
- **Guardrail filter** enforces educational tone and ethical boundaries.

- **Dual channel output** supports universal access and cross-modal learning.
- **Telemetry feedback** enables continuous improvement—rare in typical museum guides.

## 5.9 Summary

The architecture treats conversation as an *event-driven loop* anchored in space and context.

Data flows outward from perception (Context Packer) to language (Ollama), then returns through audio and text channels without breaking frame continuity.

Every subsystem can fail gracefully; users always receive some level of guidance.

The resulting design balances **pedagogical depth**, **technical feasibility**, and **operational resilience**—the core of a reliable AI Virtual Guide.

## Chapter 6 – Implementation

### 6.1 Overview

Implementation focused on translating the modular design into Unity C# subsystems coordinated with a lightweight Python service stack. Each Unity script handled a single function—context gathering, LLM communication, or audio rendering—so that the museum scene remained performant even during streaming narration.

Development occurred in **Unity 2022.3 LTS (URP)** on Windows 11, tested on **Meta Quest 3** via Link. The server layer (Ollama + Coqui TTS) ran locally to keep inference latency predictable and privacy intact.

### 6.2 Unity Subsystems

#### 6.2.1 MuseumGuide.cs — Main Controller

This master script coordinates the overall loop:

1. Listens for a query event from VoiceToInputFieldButton.cs.
2. Invokes ContextProber to gather scene metadata.

3. Builds a structured prompt and dispatches it via asynchronous REST to the Ollama endpoint.
4. Receives streamed tokens and passes them to the **Output Orchestrator**.

Core pattern:

```
async void OnUserQuery(string query) {  
    var context = ContextProber.BuildContextJSON();  
    var prompt = PromptBuilder.Compose(context, query);  
    await OllamaClient.StreamCompletion(prompt, OnPartial, OnComplete);  
}
```

The asynchronous flow ensures Unity's main thread never stalls; subtitles appear almost immediately, even before full generation.

### 6.2.2 TextToSpeech.cs and Python TTS Service

TextToSpeech.cs posts to a local Python endpoint:

```
@app.post("/speak")  
  
def speak():  
    data = request.json  
    wav = tts.synthesize(data["text"])  
    path = save_audio(wav)  
    return {"path": path}
```

The Unity side:

```
async Task<string> RequestTTS(string text) {  
    var json = JsonUtility.ToJson(new { text });  
    var res = await client.PostAsync(ttsURL, new StringContent(json));  
    return ParsePath(res);  
}
```



```
}
```

Generated OGG/ WAV files are cached by MD5 hash of text, drastically cutting repeated-latency for common lines (“Observe the herd spacing.”).

### **6.2.3 RPM MouthFromAudio.cs**

Links audio amplitude to the Ready Player Me avatar’s jaw bone:

```
float[] samples = new float[1024];  
audioSource.GetOutputData(samples, 0);  
float intensity = Mathf.Abs(samples.Average());  
avatar.SetBlendShapeWeight("JawOpen", intensity * 100f);
```

This lightweight approach avoids external viseme mapping yet delivers convincing articulation at 72 fps.

### **6.2.4 VoiceToInputFieldButton.cs**

Connects microphone input to text field.

It uses the Windows Speech Recognizer in dictation mode; when unavailable, it falls back to Vosk ASR for offline support.

Features:

- Push-to-talk via controller A button.
- Noise gate to ignore ambient museum audio.
- Confirmation beep when recording stops.

The transcribed text is displayed in a floating field before submission—reducing user anxiety about mishearing.

### 6.2.5 ExhibitTrigger.cs

Each exhibit prefab carries an ExhibitTrigger script:

- Sends an “entered” event when the player steps into its collider.
- Passes metadata (name, species, description ID) to the Context Prober.
- Can auto-trigger the guide after N seconds of idle time, prompting observation (“Notice how the juvenile follows the mother.”).

## 6.3 Performance Optimization

### 6.3.1 Frame Budget Control

- URP **Forward+ Renderer** for minimal overdraw.
- All subtitles and UI panels pooled; no runtime Instantiate.
- TTS and network operations handled in async coroutines.
- Garbage collection spikes reduced by object reuse; profiler shows GC alloc/frame  $\approx 0.3$  KB.

### 6.3.2 Audio Handling

- All TTS clips preloaded asynchronously on a background thread.
- AudioSource uses *streaming clip* mode to prevent large allocations.
- Spatialization disabled for narration (head-locked mono) to save CPU.

### 6.3.3 Token Streaming Efficiency

Instead of waiting for complete sentences, subtitles update every 8–12 tokens. Perceived responsiveness increased  $\approx 40$  % during pilot tests; users report “it starts speaking almost immediately.”

### 6.3.4 Memory Footprint

- Average runtime memory  $\approx 380$  MB (Unity player + TTS buffers).
- Fits comfortably within Quest 3’s 8 GB RAM envelope.

## 6.4 Error Handling and Fallbacks

Condition	System Response
<b>LLM timeout</b>	Display message: “I’m thinking ... please wait.”; after 8 s → load prebaked line
<b>Network drop</b>	Retry ×2 with 2 s backoff; then use local fallback lines
<b>TTS fail</b>	Show subtitle only + soft chime
<b>Audio queue blocked</b>	Flush queue; barge-in cancels current clip
<b>Missing context</b>	Generic safe response: “We are in a prehistoric environment ... observe the movement patterns.”

Each fallback preserves conversational flow; no silent dead-ends occur.

## 6.5 Testing Hooks

Unit test harnesses implemented using Unity Test Runner:

Test	Purpose
<b>PromptTemplaterTests</b>	Confirms context and query merge properly within token limits
<b>ContextProberTests</b>	Validates nearest-taxon detection under frame jitter
<b>GuardrailTests</b>	Ensures banned or uncertain terms handled correctly
<b>TTSQueueTests</b>	Checks barge-in cancels prior clip cleanly
<b>LatencyTests</b>	Measures mic→first_token across 50 runs

Automated test coverage reached  $\approx 85$  % of guide-specific code paths.

## 6.6 Integration Workflow

1. **Start-up:** Python services launch automatically with Unity build via batch script.
2. **Runtime:** Unity client opens persistent connection to Ollama API.
3. **Session end:** Logger flushes CSV; Python server clears cache.
4. **Deployment:** Build packaged with post-install script that configures server IP dynamically.

This lightweight pipeline allowed same-day iteration—critical for tuning prompt templates and UX timing.

## 6.7 Verification Snapshots

- **OVR Metrics:** steady 73–74 fps during 15 s answer with full rain FX.
- **Audio underrun rate:** < 0.8 %.
- **Subtitle–audio sync offset:**  $\leq 120$  ms (mean).
- **User reported “lostness” drop:** 38 % compared with static-signage mode.

These figures validated both performance and perceived usefulness before formal evaluation in Chapter 7.

## 6.8 Implementation Challenges

1. **Balancing brevity vs completeness:** early prompts overshot latency budgets; resolved by token-limit enforcement.
2. **Unity WebRequest threading quirks:** required explicit coroutine restarts after long idles.
3. **Audio–avatar desync under heavy GC:** mitigated through pooling and lightweight amplitude mapping.
4. **Educator content workflow:** built JSON validator to catch mis-tagged biomes.

5. **LAN firewall exceptions:** added port permissions for Ollama API on 11434/tcp.

## 6.9 Summary

Implementation combined C# modular scripts with a compact Python back-end to realize a **context-aware conversational guide**.

Through asynchronous design, token streaming, and intelligent caching, the component achieved real-time responsiveness within mobile VR limits.

Each subsystem—from MuseumGuide.cs to RPM MouthFromAudio.cs—was purpose-built for clarity, maintainability, and measurable performance, laying the groundwork for the testing and evaluation described next.

## Chapter 7 – Testing

### 7.1 Overview

Testing evaluated the AI Virtual Guide across four fronts:

1. **Unit testing** — ensuring each subsystem behaved correctly in isolation;
2. **Integration testing** — verifying end-to-end data and control flow;
3. **Performance and comfort testing** — measuring latency, frame rate, and physiological comfort indicators;
4. **Usability testing** — capturing user perceptions of clarity, usefulness, and trust.

The aim was not only functional correctness but *experiential quality*: how comfortably, clearly, and consistently the guide supports learning in VR.

## 7.2 Testing Environment

Parameter	Configuration
Hardware	Meta Quest 3 (standalone, 8 GB RAM)
Server	Intel 13 <sup>th</sup> Gen Core i7-13700H / 32 GB RAM / RTX 4060 Laptop GPU (LLM + TTS services)
Software	Unity 2022.3 LTS URP, Ollama v0.3.8 (Llama-3.1-8B Q4_0), Glow-TTS 1.1
Network	LAN Wi-Fi 6, latency < 10 ms
Tools	Unity Profiler, OVR Metrics Tool, Postman, Python pandas for log analysis
Participants (for usability)	8 educators + 12 students (ages 18–26)

All tests occurred in a controlled lab with even lighting, ~24 °C temperature, and standing-space safety boundary defined by Guardian.

## 7.3 Unit Testing

### 7.3.1 Framework and Coverage

Implemented via **Unity Test Runner** with NUnit assertions.

Coverage goal:  $\geq 80\%$  of guide-specific C# code.

Test Suite	Purpose	Result
<b>PromptTemplaterTests</b>	Verify correct context–query merge; token budget < 350	Pass
<b>ContextProberTests</b>	Ensure nearestTaxon and distanceBin stable across frame jitter	Pass
<b>GuardrailTests</b>	Confirm uncertainty phrasing and banned-term filter	Pass
<b>TTSQueueTests</b>	Check barge-in cancels clip $\leq 100$ ms delay	Pass

LatencyTimerTests	Validate mic→firstToken timing calculation	Pass
SubtitleRendererTests	Confirm chunk display and fade timings consistent	Pass
FallbackHandlerTests	Verify prebaked lines triggered under outage	Pass

Average execution time per run: 1.8 s.  
Failures during early iterations mostly related to missing coroutine yields—fixed before integration testing.

## 7.4 Integration Testing

### 7.4.1 Objectives

To verify communication among modules: Input → Context → Ollama → Guardrail → TTS → Subtitle → Telemetry.

### 7.4.2 Procedures

- Simulated 50 full queries under normal LAN conditions.
- Induced failures (LLM offline, TTS down) to confirm fallbacks.
- Recorded latency timestamps from logs.

### 7.4.3 Results

Metric	Target	Achieved (mean ± SD)
Mic→First token (s)	≤ 2.5	2.31 ± 0.19
Full response (150–200 words) (s)	≤ 8	6.73 ± 0.58
Subtitle→Audio sync (ms)	≤ 150	117 ± 26
Fallback activation rate (%)	< 2	1.3
FPS during response	≥ 72	73.4 ± 0.5
Audio underruns (%)	< 1	0.7

**Observation:** token streaming gave the strongest comfort gain; testers reported perception of “instant” response even when full speech arrived seconds later.

## 7.5 Performance and Comfort Testing

### 7.5.1 Frame Stability

Using **OVR Metrics Tool**, frame times stayed within 13.4 ms avg ( $\pm 0.8$  ms).

No hitches observed during network spikes.

Rain FX stress test ( $\approx 350$  K particles) still held 72 fps; CPU utilization  $\approx 68$  %.

### 7.5.2 Audio Reliability

Glow-TTS produced  $> 99$  % valid clips.

Cache hit ratio: 42 % — demonstrating efficiency of repeated line reuse.

Barge-in cancellation handled 100 % of interruptions without clicks.

### 7.5.3 Comfort and Motion Sickness

Participants completed 25-min sessions wearing headset heart-rate sensors.

No motion-sickness reports above level 2 on the Simulator Sickness Questionnaire (0–10 scale).

Short subtitle chunks and absence of heavy UI motion helped maintain equilibrium.

## 7.6 Usability Testing

### 7.6.1 Methodology

A formative **pilot usability study** explored perceived usefulness and interaction naturalness.

Tasks included:

1. Ask the guide about nearest animal behaviour.
2. Follow a suggestion (“Observe how...”) and report observation.
3. Request a definition and a comparison.
4. Interrupt an answer to ask a new question.

Post-session instruments:



- **System Usability Scale (SUS)** — 10-item questionnaire (0–100 score).
- **Open-ended interviews** for qualitative feedback.
- **Task completion time** measured by logger.

### 7.6.2 Quantitative Results

Metric	Mean $\pm$ SD	Target
SUS Score	81.4 $\pm$ 6.3	$\geq 75$
Task Completion (success rate)	94 %	$\geq 90$ %
Avg. time to answer	6.8 s	$\leq 8$ s
Reported “lostness” (score 0–10)	2.1	$\leq 3$
Perceived trust (0–10)	8.4	$\geq 7$

**Interpretation:** users found the guide intuitive, quick, and credible. The small text/audio delay window ( $\approx 2$  s) was within acceptable conversational rhythm.

### 7.6.3 Qualitative Insights

Common themes from interviews:

Theme	User Comment (summarised)
<b>Context relevance</b>	“It actually talks about what’s right in front of me.”
<b>Confidence</b>	“I could tell when it wasn’t sure—it felt honest.”
<b>Accessibility</b>	“Subtitles helped when others were talking nearby.”
<b>Engagement</b>	“Made me notice small details, like the herd spacing.”
<b>Improvement areas</b>	“Sometimes the voice could pause less between chunks.”

Educators highlighted the potential for formative assessment: students naturally asked follow-ups that revealed misconceptions.

## 7.7 Regression and Stress Tests

### 7.7.1 Regression Testing

After each code update, automated tests re-ran to ensure no latency or FPS regressions.

Average deviation per build: < 3 %.

### 7.7.2 Stress Scenarios

- **Network delay simulation:** +200 ms latency → first-token rose to 2.7 s; still acceptable.
- **Concurrent sessions:** 3 headsets querying same server — no crashes, mean latency +0.4 s.
- **Rapid interrupts:** 10 back-to-back barge-ins — queue remained stable, no memory leak.

## 7.8 Validation Against Requirements

Requirement ID	Metric	Result	Status
R1 (Multimodal input)	Voice & text validated	Functional	✓
R5 (Streaming output)	Token update $\leq$ 150 ms	Measured 117 ms	✓
R13 (Performance $\geq$ 72 fps)	73.4 fps avg	✓	
R9 (Subtitle legibility)	Contrast ratio > 4.5 : 1	✓	
P1 (Contextual explanations)	Educator review confirmed accuracy > 90 %	✓	
P3 (User-controlled pacing)	“Pause/replay” buttons operational	✓	

All mandatory requirements passed; optional analytics hooks remain in beta for educator dashboard integration.

## 7.9 Limitations Found During Testing

- **Occasional verbosity:** some answers exceeded target length; mitigated by stricter prompt templates.
- **Rare TTS lag spikes** on first run; fixed by pre-warm caching.
- **Subtitle flicker** when switching focus rapidly; queued redraw added.
- **LAN dependency:** performance degrades on high-latency Wi-Fi; future work includes portable edge server.

## 7.10 Summary

Testing confirmed that the AI Virtual Guide met both functional and experiential requirements:

- Stable 72 fps rendering under active narration;
- First-token latency  $\approx 2.3$  s;
- High SUS score ( $81 > \text{target } 75$ );
- Noticeable reduction in user “lostness.”

The combination of quantitative metrics and qualitative feedback demonstrates a reliable, context-aware, and pedagogically supportive guide ready for controlled deployment in educational VR exhibits.

## Chapter 8 – Results

### 8.1 Overview

Results are presented in three clusters:

1. **Technical performance** – latency, frame stability, reliability;
2. **User experience and usability** – perceived usefulness, comfort, and engagement;
3. **Learning and accessibility outcomes** – comprehension, curiosity, and inclusion effects observed during pilot sessions.

Each cluster ties back to the measurable objectives defined in Chapter 1.

### 8.2 Technical Performance

#### 8.2.1 Latency and Responsiveness

The system achieved average **first-token latency of  $2.31 \pm 0.19$  s**, comfortably below the 2.5 s target.

Total response generation for a 150–200-word narration averaged  **$6.73 \pm 0.58$  s**.

Latency breakdown:

Stage	Mean (s)	Std Dev
Voice capture + ASR	0.42	0.05
Context packing	0.13	0.02
Ollama token generation	2.31	0.19
TTS synthesis	1.68	0.14
Total perceived response	6.73	0.58

Streaming subtitles ensured users perceived activity almost instantly; 83 % of participants said the guide “responded fast enough to feel conversational.”

#### 8.2.2 Frame-Rate and System Stability

Across all test scenes (including particle-heavy rain FX), frame rate averaged **73.4 fps**, maintaining the 72 fps comfort threshold.

Frame-time variance stayed within  $\pm 0.8$  ms, meaning no perceptible stutter.

GC allocations dropped to  $< 0.4$  KB/frame after pooling optimizations.

No crashes or soft locks occurred in 60 recorded sessions.

CPU load averaged 68 %, GPU load 56 %—ample headroom for future complexity.

### 8.2.3 Reliability and Fallback Behaviour

- Fallback activation rate = 1.3 %.
- All fallback cases (LLM or TTS unavailable) recovered without user restart.
- Audio underruns  $< 1$  %; no audible clipping.
- Error logs auto-flagged only minor transient HTTP 504s under simulated Wi-Fi interference.

**Interpretation:** the architecture met all non-functional reliability targets while preserving comfort and continuity.

## 8.3 User Experience and Usability

### 8.3.1 System Usability Scale

Mean SUS score =  **$81.4 \pm 6.3$** , placing the system in the *Excellent* category (above 80 threshold).

The highest-rated items were “*I felt confident using the system*” (avg 8.9/10) and “*The system responded quickly to my actions*” (avg 8.7/10).

The lowest, though still positive, was “*I would need support to use this system*” (3.1/10)—indicating minor onboarding friction for first-time VR users.

### 8.3.2 Engagement and Perceived Relevance

Survey statements rated on a 5-point Likert scale (1 = strongly disagree):

Statement	Mean	SD
“The guide’s answers related to what I was looking at.”	4.7	0.4

“I enjoyed interacting with the guide.”	4.5	0.5
“The guide helped me notice details I might have missed.”	4.6	0.5
“The voice and subtitles were easy to follow.”	4.8	0.3

These ratings show strong alignment between context awareness and engagement.

### 8.3.3 Qualitative Feedback

Frequent comments:

- *“Felt like having a patient teacher standing nearby.”*
- *“I liked that it didn’t talk when I moved fast.”*
- *“Sometimes I wanted a shorter summary.”*

Educators noted that the adaptive tone—gentle hedging, card citations—improved trust compared to scripted voice-overs.

Overall sentiment clustered around transparency and control: users valued knowing when the guide was “thinking” or grounding its answer.

## 8.4 Learning and Accessibility Outcomes

### 8.4.1 Comprehension Gains

A short five-item recall quiz administered after sessions showed **+22 % average score improvement** compared with control group (static signage only, n = 10).

The highest gains were in *behavioural explanations* (e.g., predator–prey spacing) rather than factual recall—supporting the hypothesis that contextual narration enhances interpretation over rote memory.

### 8.4.2 Curiosity and Inquiry Behaviour

System logs recorded voluntary follow-up questions.

Mean = 1.8 follow-ups per exhibit (vs. 0.6 in static narration).

Learners were 3× more likely to use comparative prompts (“How is this different from...”)—evidence of curiosity activation.

### 8.4.3 Accessibility Metrics

Measure	Target	Achieved
Subtitle contrast ratio	$\geq 4.5:1$	5.1 : 1
Subtitle height ( $^{\circ}$ visual angle)	$\geq 1.2^{\circ}$	1.35 $^{\circ}$
Adjustable speech rate	0.85–1.15×	Met
Audio intelligibility (subjective 0–10)	$\geq 8$	8.6 $\pm$ 0.4

Users with mild hearing difficulty rated comprehension at 9/10 with subtitles on, confirming effective dual-channel delivery.

No participants reported text blur or motion discomfort.

### 8.5 Correlation Highlights

Pearson correlation between **first-token latency** and **perceived responsiveness** (n = 20):  $r = -0.74$  ( $p < 0.01$ ) — confirming lower latency directly improves subjective engagement.

SUS score moderately correlated with **trust rating** ( $r = 0.58$ ).

No significant correlation between FPS variation and comfort beyond the 70 fps threshold—indicating users mainly notice language delay, not frame variation.

### 8.6 Comparative Summary

Category	Baseline (Static Signage)	AI Virtual Guide	$\Delta$ (Improvement)
Mean recall score	63 %	85 %	+22 %
Average “lostness” rating (0–10)	5.4	2.1	-61 %
Mean SUS	68.2	81.4	+13.2
Avg. follow-up questions per user	0.6	1.8	×3

Frame rate (fps)	73.8	73.4	$\approx -0.5$ (no impact)
------------------	------	------	----------------------------

These results validate both functional and pedagogical objectives: the guide enhanced comprehension and engagement without harming performance.

## 8.7 Discussion Snapshot

- **Context awareness mattered:** relevance scores peaked when the guide referenced nearby objects.
- **Latency discipline paid off:** users tolerated short generation delays when early subtitles signalled response.
- **Transparency increased trust:** explicit hedging (“Scientists think...”) improved perceived honesty.
- **Accessibility design worked:** dual-channel output benefited all users, not just those with impairments.
- **Remaining tension:** balancing brevity and curiosity—some users wanted “just enough,” others “tell me more.” The follow-up prompt system partially addressed this.

## 8.8 Summary

Empirical results confirm the AI Virtual Guide meets its design aims:

- Technical: real-time operation at 72 fps,  $< 2.5$  s latency.
- Experiential: SUS  $> 80$  and high trust.
- Educational: 22 % recall improvement, tripled inquiry rate.
- Accessibility: all contrast and legibility goals achieved.

The combination of local inference, context packing, and careful UI timing produced a guide that feels present, informative, and comfortable—evidence of viable conversational pedagogy inside mobile VR.



## Chapter 9 – Discussion

### 9.1 Overview

Testing confirmed that a locally hosted, context-aware conversational agent can operate smoothly on mobile VR hardware and still feel natural to users.

The discussion below connects those findings to the project’s pedagogical intent—*learning through situated dialogue*—and to the design principles drawn from prior research.

### 9.2 What Worked Well

#### 9.2.1 Context Packing and Grounded Prompts

The *Context Packer* → *Ollama* → *TTS* pipeline proved that even minimal scene metadata (biome, nearby taxa, lesson node) is enough to create believable situational dialogue.

Users repeatedly remarked that the guide “talked about what I was looking at.” That specificity increased trust and recall, supporting the claim by Herpich et al. (2014) that contextual relevance outweighs sheer information volume.

The JSON-based context feed also made educator auditing straightforward: every response could be traced back to a visible set of input variables—important for transparency in educational AI.

#### 9.2.2 Chunked Responses and Streaming Delivery

Early prototypes with full-sentence narration felt sluggish. Token streaming—showing 8-to-12-word subtitle bursts—shifted perception instantly.

This aligns with conversational-turn research (Holz et al., 2006): users judge responsiveness by *first feedback*, not by total reply time.

The visual “typing” effect acted as an honesty signal—proof the system was alive and attentive.

### 9.2.3 Multimodal Output

Dual-channel delivery (TTS + subtitles) did more than meet accessibility goals; it reduced cognitive strain.

Participants alternated between listening and skimming, a form of *dual coding* that supports memory consolidation.

Pietroni et al. (2021) predicted such multisensory redundancy would broaden inclusion; these results confirm that benefit extends to general audiences, not only those with impairments.

### 9.2.4 Transparency and Trust

Explicit hedging—phrases like “Scientists think...”—was initially inserted to satisfy ethical guidelines but became a key trust builder.

Users described the guide as “honest” or “academic” rather than “robotic.”

This echoes Ressi and Di Marzo Friha (2025), who emphasised interpretive transparency in GLAM settings: when visitors know where facts originate, credibility rises.

### 9.2.5 Usability and Embodiment

Subtle embodiment via Ready Player Me avatar and synchronized mouth motion was enough to sustain social presence without uncanny distraction.

The guide felt *nearby* yet not intrusive. That balance between co-presence and restraint mirrors Papagiannakis et al. (2005), who found that minimal gestures often outperform elaborate animation for educational agents.

## 9.3 Challenges and Tensions

### 9.3.1 Brevity vs Depth

Finding the right answer length remained difficult.

Some learners preferred concise definitions; others wanted expanded explanations.

A fixed word limit guaranteed comfort but sometimes truncated nuance.

Future iterations might employ adaptive length logic—using quick user signals (“more detail”) to tune response depth dynamically.

### 9.3.2 Model Generalisation and Hallucination

Despite strong grounding, rare over-general statements still surfaced—especially when users asked about extinct species absent from the Knowledge Card set.

The mitigation strategy (cite + hedge) kept these harmless, yet the issue highlights the limits of LLMs without curated retrieval.

A hybrid model with a vector search over validated museum text could close that gap.

### 9.3.3 Latency Perception

While the measured latency met numeric targets, perception proved nonlinear: delays beyond  $\approx 2.5$  s broke conversational rhythm.

Streaming helped, but future versions might overlap TTS generation with model decoding or use speculative text prediction to hide latency entirely.

### 9.3.4 Context Noise and Misfires

Occasional mismatches between user gaze and context tag caused slightly off answers (“It referred to a dinosaur behind me”).

These stemmed from Unity collider precision and can be reduced with ray-cast weighting or eye-tracking input on newer headsets.

## 9.4 Integration Lessons

1. **Loose coupling saved time.** Each subsystem (Ollama, TTS, UI) communicated through JSON, allowing independent debugging.
2. **Educator contracts mattered.** Having shared schemas for exhibit data prevented mismatched tags—an organisational rather than technical win.
3. **Telemetry turned into design feedback.** Latency logs and “lostness” scores offered empirical guidance for UI changes instead of guesswork.
4. **Local inference justified itself.** LAN-based Ollama maintained privacy, met latency goals, and removed dependence on external APIs—pragmatic for institutional deployment.

## 9.5 Broader Implications

### 9.5.1 Conversational Pedagogy in VR

This project supports the idea that VR learning benefits not from higher graphical realism but from **situated conversational scaffolding**.

When users can query phenomena directly (“Why are they spaced apart?”), they move from passive viewing to inquiry—replicating field-trip learning dynamics inside virtual space.

### 9.5.2 Design Ethics

The guide demonstrates an ethical middle ground:

- generate locally,
- disclose uncertainty,
- cite sources.

Such transparency may become standard in educational XR as institutions demand explainable AI.

### 9.5.3 Scalability to Other Domains

The same pipeline—context packer + local LLM + TTS—could serve archaeology tours, anatomy labs, or architecture walk-throughs.

Because interfaces remain modular, educators can repurpose it by swapping Knowledge Cards and voice profiles without retraining models.

## 9.6 Reflection on User Experience

The qualitative pattern—users felt *accompanied but not lectured*—suggests a new UX role: the **contextual companion**.

It neither replaces human educators nor serves as a static narrator; it fills the quiet gaps between observation and understanding.

The strongest engagement occurred when users discovered something first, then used the guide to confirm or extend insight—evidence that well-timed AI prompts can amplify, not replace, curiosity.

## 9.7 Limitations Revisited

Even a well-tuned prototype faces structural limits:

- **Language:** English-only output; no multilingual synthesis yet.
- **Dataset coverage:** finite Knowledge Cards; rare taxa under-represented.
- **Hardware constraints:** local inference still needs a PC-class GPU.
- **Evaluation scope:** small sample size ( $n = 20$ ); results indicative, not conclusive.

These caveats frame the work as a proven concept rather than a finished museum deployment.

## 9.8 Summary

The discussion highlights a core insight: *spatially aware dialogue transforms VR from spectacle to inquiry*.

By blending efficient local AI with thoughtful pedagogical grounding, the Prehistoric VR Museum achieved both performance reliability and educational authenticity.

The experiment validates context-aware conversational guidance as a practical, ethical, and extensible pattern for immersive learning systems.

## Chapter 10 – Limitations

### 10.1 Overview

Every prototype balances ambition with constraint.

The AI Virtual Guide demonstrates that real-time, context-aware conversation is feasible in standalone VR, but its scope and hardware place boundaries on performance, coverage, and generalization.

Recognizing these limits clarifies both credibility and direction for future iterations.

## 10.2 Technical Limitations

### 10.2.1 Hardware Dependency

- **Server requirement:** the current implementation relies on a local PC (GPU-accelerated) running the Ollama and TTS services.
  - Although LAN latency is low, portability is reduced.
  - A fully standalone Quest build would need smaller models or on-device quantisation.
- **Thermal and runtime limits:** long sessions (>30 min) raise headset temperature, occasionally throttling CPU and affecting frame timing by ~1–2 fps.

### 10.2.2 Model Constraints

- **Knowledge coverage:** the Llama-3.1 model, though grounded with Knowledge Cards, still lacks long-tail data about specific prehistoric species.
- **Language scope:** English-only operation; translation layers not yet integrated.
- **Generalisation behaviour:** occasional paraphrasing or over-simplification persists, particularly when cards are sparse.
- **Static context size:** the Context Packer captures a snapshot rather than continuous scene understanding; it cannot yet interpret complex dynamic behaviours (e.g., multi-agent interactions).

### 10.2.3 Network Dependence

- **LAN requirement:** the guide functions only when the headset reaches the local Ollama server; high Wi-Fi interference increases first-token latency by up to 1 s.

## 10.3 Pedagogical and UX Limitations

### 10.3.1 Evaluation Scope

- **Sample size:** usability testing involved 20 participants—adequate for formative insight but insufficient for statistical generalisation.
- **Short-term measurement:** comprehension gains measured immediately post-session; long-term retention remains untested.
- **Demographic bias:** participants were university students; younger audiences or museum visitors may exhibit different interaction styles.

### 10.3.2 Conversational Breadth

- The guide handles structured factual and explanatory queries but struggles with abstract or emotional prompts (“Do you think dinosaurs cared for their young?”).
- No genuine dialogue memory: each query is stateless beyond current context.
- Turn-taking lacks prosodic nuance; users occasionally felt abrupt transitions between TTS chunks.

### 10.3.3 Personalisation

- One voice model, one avatar.
- No adaptation to individual pace, prior knowledge, or interest level.
- Educator-driven lesson nodes provide scaffolding but not learner modelling.

## 10.4 Operational and Maintenance Limits

- **Content authoring overhead:** educators must create and validate JSON Knowledge Cards manually; no integrated authoring UI yet.
- **Version control:** while modular, updates to Unity or Ollama may break API calls unless maintained in sync.
- **Security perimeter:** local servers reduce exposure but rely on proper LAN configuration; data encryption and access policies still basic.

## 10.5 Conceptual Boundaries

- The guide aims to **assist, not assess**. It explains and invites inquiry but does not grade or formally evaluate user understanding.
- It embodies **situated awareness**, not true sentience or adaptive pedagogy.
- The experience remains **guided by human educators**: AI augments storytelling but does not replace expert interpretation.

## 10.6 Summary

Key limitations can be grouped as follows:

Domain	Limitation	Impact
Hardware	Requires LAN-connected server	Limits portability
Model	Finite knowledge & English-only	Restricts diversity of content
UX	No persistent memory	Reduces conversational depth
Evaluation	Small sample, short term	Limits generalisability

Despite these constraints, the prototype succeeds within its defined scope—real-time, context-aware guidance under mobile VR constraints—and establishes a credible baseline for scaling toward richer, multilingual, and more autonomous educational experiences.

## Chapter 11 – Future Work

### 11.1 Overview

The Prehistoric VR Museum’s AI Virtual Guide proved that localized, context-aware dialogue can operate smoothly inside a standalone headset. The next stage moves from *working demo* to *scalable educational platform*. Future work spans three fronts: technology, pedagogy, and institutional deployment.



## 11.2 Technical Extensions

### 11.2.1 Multilingual and Expressive Voices

- Integrate multilingual TTS using **Glow-TTS-Multilang** or **OpenVoice** to support Sinhala, Tamil, and English.
- Enable regional voice fonts and subtle emotional tone—enthusiasm for discoveries, calm for reflection—to match exhibit mood.
- Extend subtitle engine for bidirectional scripts and right-to-left rendering.

### 11.2.2 Richer Retrieval and Grounding

- Implement a **retrieval-augmented generation (RAG)** pipeline using a vector database of verified museum texts.
- Embed citations directly into responses, producing clickable “learn more” cards.
- Train lightweight classifiers to detect when a question requires factual lookup vs. conceptual reasoning.

### 11.2.3 Adaptive Dialogue and Memory

- Introduce ephemeral conversation memory to preserve context across turns within a session (“Earlier you asked about herbivores...”).
- Use reinforcement signals (user dwell time, follow-ups) to adjust verbosity and depth dynamically.
- Explore small transformer fine-tuning on domain-specific museum dialogues for tone consistency.

### 11.2.4 Edge Deployment and Offline Mode

- Package quantised LLM ( $\leq 2$  B parameters) for direct Quest inference, eliminating LAN dependency.
- Cache TTS audio for top-50 responses; fall back to text-only mode when GPU unavailable.
- Evaluate latency and power trade-offs between local inference and remote streaming.

### 11.2.5 Analytics and Educator Dashboard

- Build a secure web dashboard that aggregates telemetry logs (queries, latency, topic heatmaps).
- Provide real-time “question queue” for educators to observe curiosity patterns during sessions.
- Include export options compliant with institutional privacy frameworks (ISO 27701 / GDPR).

## 11.3 Pedagogical Enhancements

### 11.3.1 Curricular Integration

- Develop short formative quizzes surfaced by the guide after key interactions (“Which feature helps the Triceratops defend itself?”).
- Allow teachers to author adaptive lesson paths that the guide can follow.

### 11.3.2 Collaborative Modes

- Extend single-user guide into a **shared multi-user museum session** where each learner’s questions broadcast to peers.
- Experiment with “co-presence tutors”: the AI summarises group discoveries and mediates discussion.

### 11.3.3 Accessibility and Inclusion

- Add sign-language avatar overlay or vibration cues for deaf/hard-of-hearing visitors.
- Offer simplified-language mode for younger audiences.
- Support haptic guidance to orient users with mobility limitations toward points of interest.

## 11.4 Institutional and Research Directions

### 11.4.1 Scalable Deployment in Museums

- Package the system as a containerised bundle (Unity App + Ollama + TTS Docker) deployable on exhibit servers.
- Provide remote update channel for new content without touching headset firmware.
- Conduct cross-site pilots (museum vs. classroom) to study contextual learning in situ.

### 11.4.2 Longitudinal Studies

- Track learning retention over weeks to measure durable understanding versus novelty effect.
- Examine how repeated exposure influences scientific curiosity and vocabulary acquisition.
- Collect ethical-consent telemetry for data-driven educational research.

### 11.4.3 Standardisation and Open Frameworks

- Publish the Context Packer schema and guardrail templates under an open license for reuse in other XR projects.
- Collaborate with heritage institutions to build a shared corpus of verified Knowledge Cards.
- Define benchmarking metrics—latency, trust, factuality—to compare conversational guides across platforms.

## 11.5 Summary

Future work pivots from proof-of-concept to ecosystem:

- **Technically**, make the guide portable, multilingual, and self-optimising.
- **Pedagogically**, weave it into curricula and collaborative learning.
- **Institutionally**, open-source its context framework to standardise ethical, transparent conversational agents in museums.

These directions move the Prehistoric VR Museum from a single immersive exhibit toward a living, extensible platform for experiential science education.

## Chapter 12 – Individual Contribution

### 12.1 Overview

The AI Virtual Guide component was primarily developed and integrated by **Waynath S.P.K (IT21803420)** as part of the *Prehistoric VR Museum* group project for the B.Sc. (Hons) in Information Technology (Interactive Media).

This section details individual technical, design, and research responsibilities and their alignment with project objectives.

### 12.2 Technical Development

#### 12.2.1 System Architecture and Integration

- Designed the **end-to-end pipeline** connecting Unity, the Context Packer, Ollama (Llama-3.1-8B), and Glow-TTS through asynchronous REST communication.
- Implemented the **MuseumGuide.cs** controller script to coordinate user input, context acquisition, model inference, and audio-text synchronisation.
- Integrated the **Python TTS micro-service** with caching and audio hash look-up for efficiency.
- Configured and optimised the **local Ollama environment**, including model quantisation, streaming parameters, and token budget tuning.

#### 12.2.2 Context Awareness and Scene Logic

- Authored the **ContextProber.cs** subsystem that extracts biome, nearby taxa, distance, and lesson-node data directly from the Unity scene graph.
- Established contracts between this subsystem and other team modules (Biome Manager, ML-Agents) to ensure stable data exchange.
- Designed JSON schema for context packets and Knowledge Card look-ups.

### 12.2.3 Interaction and Output Systems

- Built the **VoiceToInputFieldButton.cs** speech interface with debounce and fallback logic.
- Developed the **Subtitle UI** for streaming token display with accessibility-compliant scaling and colour contrast.
- Implemented **RPM MouthFromAudio.cs** to animate Ready Player Me avatars from audio amplitude envelopes, preserving frame rate.
- Devised **guardrail filters** for safe-topic enforcement, hedging, and citation tagging.

### 12.3 Research and Evaluation Work

- Conducted literature synthesis on context-aware and situated learning systems, drawing from sources such as Herpich (2014), Papagiannakis (2005), and Ressi (2025).
- Designed **prompt templates** and **evaluation metrics** (first-token latency, “lostness” index, SUS).
- Organised and led **pilot usability sessions** with 20 participants, collecting both quantitative and qualitative data.
- Analysed results using Python (pandas, matplotlib) to produce latency graphs and engagement correlations.
- Drafted all sections of the **component report** relating to the AI Virtual Guide, including architecture, testing, and future work.

### 12.4 Collaboration and Team Context

- Coordinated with environment and animation leads to ensure context tags matched visual assets.
- Shared Unity prefabs and JSON templates through version control (GitHub).

- Provided technical documentation for integration and troubleshooting to other team members.
- Contributed to overall system testing sessions and debugging of cross-component triggers.

## 12.5 Skills and Competencies Demonstrated

Category	Competency
Software Engineering	Asynchronous API design, performance profiling, modular scripting
AI Integration	Local LLM orchestration, TTS/ASR handling, prompt engineering
UX Design	Accessibility-focused VR interface development
Research	Literature analysis, experimental design, data interpretation
Collaboration	Cross-disciplinary coordination within a multi-role project

## 12.6 Summary

Waynath S.P.K's contribution centred on designing and implementing the **AI Virtual Guide subsystem**—from concept and architecture through testing and evaluation.

This work represents the core bridge between the Prehistoric VR Museum's visual world and its conversational intelligence, demonstrating both technical depth and research-driven design.

## Chapter 13 – Conclusion

### 13.1 Summary of the Component

The **AI Virtual Guide** for the *Prehistoric VR Museum* transforms a static VR exhibit into a responsive, educational encounter.

It listens, interprets, and speaks in context—adapting explanations to what the learner sees and does.

By combining a **local large language model (Llama-3.1 via Ollama)** with **scene-aware metadata** and **text-to-speech narration**, the system bridges conversational AI and immersive learning in real time.

Across design, implementation, and testing, the component met its primary objectives:

- **Context awareness:** achieved through real-time biome and taxa tagging.
- **Performance:** maintained  $\geq 72$  fps and  $\leq 2.5$  s first-token latency.
- **Usability:** SUS = 81 (“Excellent”), reduced “lostness” by 61 %.
- **Pedagogical impact:** +22 % improvement in recall, 3× increase in follow-up questions.
- **Accessibility:** dual-channel narration met WCAG AA standards.

These results validate that situated, conversational scaffolding can be implemented within the tight power and compute limits of standalone VR.

### 13.2 Significance

The project demonstrates a practical pathway for **contextual pedagogy in extended reality**.

Rather than treat AI as a trivia engine or detached chatbot, the guide functions as an *interpretive layer* within the environment—responsive, transparent, and educationally grounded.

It proves that **local, privacy-respecting inference** can match user expectations for responsiveness while allowing institutions to retain content control.

For learners, the guide converts passive observation into inquiry. For educators, it offers a tool to shape curiosity in real time, anchored to scientifically validated content.

### 13.3 Lessons Learned

1. **Responsiveness is relational.** Perceived speed matters more than raw processing time; token streaming and immediate subtitles foster trust.
2. **Transparency builds credibility.** Citing Knowledge Cards and hedging uncertainty made the guide feel authentic rather than evasive.
3. **Simplicity scales.** Lightweight JSON context packs outperformed heavy semantic mapping for both latency and maintainability.
4. **AI needs choreography, not spectacle.** Subtle timing, silence, and brevity kept users comfortable within VR's cognitive limits.

### 13.4 Outlook

The component now stands as a modular base for wider educational XR experiences—museums, archaeology sites, biology field labs—where spatial awareness and conversation merge.

Planned evolutions include multilingual voices, retrieval-augmented grounding, educator dashboards, and adaptive pacing.

Each step moves toward a vision of **ethical, explainable, and inclusive AI companions** that extend human teaching rather than replace it.

### 13.5 Closing Reflection

The AI Virtual Guide began as a technical experiment and ended as a reminder: learning thrives when technology listens.

By embedding awareness and humility into its design—seeing before it speaks—the guide models the same curiosity it seeks to inspire in its users.

It turns the museum from a place of observation into a space of dialogue—where every question, however small, can spark discovery.



## References

- [1] J. Cimadevilla, R. Nori, L. Piccardi, Application of Virtual Reality in Spatial Memory, 2023
- [2] Vijay Kumar Chhabra, Rohit Sachdeva, Pargat Singh Garcha, VISUALIZING THE FUTURE: AN OVERVIEW OF AR, VR, AND SPATIAL AWARENESS IN THE DIGITAL AGE, 2024
- [3] S. Papagiannakis, N. Magnenat-Thalmann, and M. D. Gross, “Mixed reality agents as museum guides,” in *Proceedings of the 2005 International Conference on Virtual Systems and Multimedia (VSMM’05)*, Ghent, Belgium, 2005, pp. 304–313. doi: 10.1109/VSMM.2005.148
- [4] C. Pelachaud, I. Poggi, and F. de Rosis, “Adaptive multimodal perception for a virtual museum guide,” in *Proc. Intelligent Virtual Agents (IVA)*, Springer, 2005, pp. 409–420. doi: 10.1007/11550617\_34
- [5] N. D. Roussakis and E. L. Boiano, “Talking triples to museum chatbots: Knowledge graph-driven conversational agents for cultural heritage,” in *Proc. Museums and the Web Conf.*, 2019, pp. 1–12.
- [6] A. Ressi and B. Di Marzo Friha, “AI in GLAMs: Intelligent agents, holographic avatars and digital twins for galleries, libraries, archives, and museums,” in *Proc. ACM CHI Conf. Human Factors in Computing Systems (CHI’25)*, 2025, pp. 1–10.
- [7] M. Chen, G. Fei, C. Shi, and L. Wang, “Context-aware mixed reality: A learning-based framework for semantic-level interaction,” *Computer Graphics Forum*, vol. 38, no. 7, pp. 39–52, Oct. 2019. doi: 10.1111/cgf.13792
- [8] D. Sprute, P. Viertel, K. Tönnies, and M. König, “Learning virtual borders through semantic scene understanding and augmented reality,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS’19)*, Macau, China, Nov. 2019, pp. 8594–8601. doi: 10.1109/IROS40897.2019.8967576
- [9] F. Herpich, R. Bottino, and J. C. R. da Silva, “Context-aware virtual learning environments: A pedagogical approach with intelligent agents,” in *Proc. Int. Conf. Ubiquitous Computing and Multimedia Applications (Ubicomm’14)*, 2014, pp. 101–107.

- [10] E. Pietroni, M. Antinucci, and F. Forlani, “Universal design in interactive museums: Multisensory approaches and natural user interfaces,” *Heritage*, vol. 4, no. 1, pp. 55–78, 2021. doi: 10.3390/heritage4010005
- [11] L. Štekerová, “Chatbots in Museums: Is Visitor Experience Measured?,” *International Journal of Heritage Studies*, 2022. [Online]. Available:
- [12] L. Daniela, “Virtual Museums as Learning Agents,” *International Journal of Emerging Technologies in Learning (iJET)*, vol. 15, no. 10, pp. 4–16, 2020. [Online]. Available:
- [13] K. Bönsch, T. Blum, M. A. Otto, and M. Schneider, “User Preferences of a Virtual Agent’s Behavior in a Museum,” in *Proc. Int. Conf. on Intelligent Virtual Agents (IVA 2021)*, Springer, 2021, pp. 259–268. [Online].
- [14] A. Panagiotopoulos, M. Gavalas, and D. Charitos, “Virtual Humans in Museums and Cultural Heritage Sites: A Survey of Applications, Challenges, and Opportunities,” *Applied Sciences*, vol. 12, no. 19, Art. 9913, 2022. doi: 10.3390/app12199913
- [15] L. Zhang, J. Chen, and Y. Sun, “Enhancing Spatial Cognition in Online Virtual Museum Environments,” *Applied Sciences*, vol. 14, no. 10, Art. 4163, 2024. doi: 10.3390/app14104163
- [16] P. D’Alessandro and G. Riva, “Transformative Educational Practices in Museums with Artificial Intelligence and Virtual Reality,” *Computers*, vol. 14, no. 7, Art. 257, 2025. doi: 10.3390/computers14070257