QUEST: INTERACTIVE AI-NOVEL

Farham Khademi, Dylan Pereira, Sharon Dsouza
Clive Lewis, Dhruv Joshi, Ajla Hate, Mrunmai Nagtode

Seidenberg School Of Computer Science and Information Systems, Pace University
15 Beekman St, New York, NY 10038

farham.khademi@pace.edu, dylan.pereira@pace.edu, sd37385n@pace.edu,
cl11588n@pace.edu, dj17292n@pace.edu,
ah71782n@pace.edu, mn14098n@pace.edu

***Abstract*— Digital interactive novels have gained popularity ever since they were introduced in the 1980s. Today with the current advancements in artificial intelligence, digital interactive novels have the opportunity to provide dynamic experiences where the user directly shapes the story. This paper introduces Quest, an immersive AI-driven web-based interactive novel. The Platform allows users to dive into a thrilling world of imagination by blending storytelling with player agency. In Quest, readers can play customizable themes, stories and characters, make choices using natural language input, dice roll decisions and interact with NonPlayable Characters to directly impact the story's progression. Quest addresses problems like writer's block, static narratives, game predictability and in-flexibility through dynamic context generation, player agency.**

***Keywords*— Artificial Intelligence, Role-Playing-Games (RPGs), Large Language Model (LLM), Non-Playable Characters (NPC), Interactive Storytelling, Real-time Interaction**

I. INTRODUCTION

Interactive storytelling has emerged as a compelling form of digital engagement, blending novels with the agency-driven dynamics of role-playing games (RPGs). As web technologies and artificial intelligence (AI) continue to evolve, so does the potential for more dynamic and personalized narrative experiences.

Traditional RPGs are often limited by rigid plot structures or single-path narratives. While some offer branching decision trees, they typically lack the adaptability required to respond fluidly to user input.

AI presents a unique opportunity to address these challenges. Recent advances in natural language processing (NLP) and real-time interaction models enable AI to function as a dynamic narrator that is capable of guiding stories based on player decisions, improvising new plot developments, and maintaining narrative coherence. When integrated with choice-driven mechanics like dice rolls, AI can replicate the fluidity and spontaneity of traditional tabletop storytelling within a digital environment.

This paper introduces Quest as an innovative solution to the creative limitations faced by readers, writers, and gamers alike. It explores central research questions such as: How can AI enhance user agency and narrative engagement in interactive storytelling? What design mechanisms ensure coherence while supporting emergent narrative structures? How can multiplayer interactions enrich digital storytelling experiences?. Quest seeks to redefine the way stories are experienced at the intersection of AI and creativity.

II. Literature Review

The increase in interactive applications of stories narration has been parallel to advances in artificial intelligence and web technologies, allowing real -time narrative generation, the dynamic interaction of the user and the management of persistent content. This section reviews similar products in the market and relevant investigations that report the design of an interactive novel web application with AI.

A. Dungeon AI

AI Dungeon by Latitude is a notable example of open interactive fiction promoted by OpenAi GPT models. The system allows players to enter any action or dialogue, and the AI responds generating coherent history continuations. Studies have analyzed narrative coherence and user participation within AI Dungeon, pointing out both their creativity and challenges to maintain the narrative structure on extended interactions [1]. The platform demonstrates the viability of models based on transformers in the narrative of stories but also reveals limitations in context management and content filtering.

B. Ink and branch narrative systems

Inklewriter and ink are tools developed for the elaboration of structured and branched narratives. Although they do not use the generation, their design principles have been studied in the context of interactive narrative theory. Reed et al. [2] emphasizes the importance of author control and predefined narrative arches in the maintenance of narrative coherence, which suggests that hybrid systems that combine paths with scripts with improvisation of AI can offer the best of both worlds.

C. Mobile interactive stories applications

Platforms such as options and episodes use visual narration and structured decision trees to offer an attractive experience with limited but shocking user decisions. Studies on player participation in mobile interactive fiction (for example, Ryan et al. [3]) show that visual elements, consistent characters and emotional decisions contribute significantly to user retention and satisfaction. These findings report UI/UX considerations for the novel web -based platforms.

D. Creative tools assisted by AI-AI

Applications such as noveli offer tools of creative writers for stories co -author with AI, using NLP to generate prose in the user's style. Research from Kremininski et al. [4] Investigate the co-creativity of Human-Ai in the narrative generation, highlighting how AI can act as a collaborator and a source of inspiration. This approach is aligned with the objectives of an interactive novel where the user entry guides the narrative flow, while the AI provides dynamic content of the story.

E. Web technologies and real -time interaction

The use of modern web development frameworks such as React and Backend Solutions such as Mongodb for interactive applications has been explored in software engineering studies. For example, Teymourzadeh et al. [5] Explore web applications in real time using NOSQL databases, emphasizing response capacity and scalability: critical characteristics for interactive platforms where the states of history and user routes must be stored and recovered efficiently.

III. METHODOLOGY

The Quest platform is architected as a modular, scalable, and extensible AI-driven interactive storytelling engine. It leverages large language models (LLMs), session-aware user orchestration, and voice-based I/O to deliver a deeply immersive and dynamic narrative experience. The system design is illustrated across six core diagrams that provide insights into architectural layout, data modeling, user interaction sequences, and system boundaries.

A. Fig. 1 Architecture Diagram

Design of Modular Components in an Architecture Diagram

The architecture ensures maintainability and a clear division of responsibilities by using a client-server model with separate front-end, back-end, and external AI service connections.

To enable real-time interactivity, the frontend layer (client application) is created utilizing a contemporary JavaScript framework like React. It includes voice input/output widgets that use the Web Speech API or comparable libraries, a dynamic display container for the tale with formatting options, and a rich text editor for free-form story entry. Additionally, the interface may be user-configurable through customizing tools including accessibility switches, font selections, and themes. For seamless live updates, back-end communication is managed by WebSocket or RESTful APIs.

Python with FastAPI or Node.js with Express are two examples of scalable frameworks used in the development of the Backend Layer (Application Server). The Multiplayer Turn Controller, which controls round-robin logic for multi-user sessions; the Narrative Engine, which connects to the LLM via APIs and enhances prompts with memory context and directives; the Dice Roll Engine, which simulates probabilistic outcomes using pseudo-random number generation; the Session Manager, which tracks stateful user sessions and maintains context; and the Voice Service Layer, which interfaces with third-party text-to-speech and speech-to-text APIs like Google, Azure, or OpenAI Whisper. Session data is persistently retained to provide continuity even though client interactions are stateless.

To manage natural language generation, the AI Integration Layer makes use of models from OpenAI's GPT family or similar substitutes. In order to lessen hallucinations, preserve consistency in character and plot development, and enforce predetermined narrative limitations, such as ethical boundaries, this layer integrates quick engineering and rule-based adjustments.

Lastly, we used NoSQL MongoDB and S3 Buckets because it provides a stable and expandable basis for the program.

B. Fig. 2 ER Diagram – Data Schema and Relational Mapping

There are six entities in Quest’s ER Diagram, which are Themes, Stories, NPCs, GameSession, Characters, and Logs.

Starting form Themes. One theme can have multiple stories, while one story can be assigned to only one theme.

The Stories entity also has a relationship with GameSession and NPCs. One story can have multiple NPCs, while one NPC can only be assigned to a single story. The Story entity can also be assigned to multiple game sessions, but one game session can only have one story assigned to it.

The Game session entity has two other relationships, one with the Characters entity, and other with the Logs entity. A Game session can have multiple characters (Protagonists), while a character can only belong to a specific game session. A game session also has many logs, but a log can only belong to a single Game Session.

C. Fig. 3 Context Diagram – System Boundaries and External Interfaces

The external entities and how they interact with the Quest system are identified in this high-level diagram.
Outside Players:
End Users: Voice or user interface interactions between human players.
- LLM Provider: Asynchronously invoke external API (like OpenAI) with context prompts.
Voice APIs: Text-to-speech (output) and speech-to-text (input) services.

- Content Moderation Service *(optional)*: Removes or filters offensive material from narratives that are generated.
Data Movement:
User input → backend processing → prepared prompt → transmitted to LLM → response → returned and either transformed to audio output or presented on the user interface.
Parallel logging is used for analytics, recovery, and session persistence.

D. Fig. 4.1 Sequence Diagram – Start Game

The bootstrapping stage of a new story session is modeled by this sequence.

When the user accomplishes that using the user interface, a new game session is started. The backend responds by allocating a distinct session_id and starting the appropriate game state. A prompt template that is appropriate for the chosen genre or tale type is loaded. Using this template, the system creates an initial prompt and uploads it to the language model (LLM). The LLM generates the first narrative response, which is then shown to the user and saved for future use. The response is also provided as voice narration if TTS technology is enabled.

For each game session, this procedure provides a consistent and engaging starting state.

E. Fig. 4.2 Sequence Diagram – Player Turn

Natural language comprehension, state management, and rule-based unpredictability are highlighted in the main gameplay loop.

1. An action is submitted by the user (for example, "I try to unlock the door.").

2. The system looks for logical constraints and adds this input to the current context.

3. Action type (difficulty class, skill modifications) determines when the dice roll simulation starts.

4. The completed prompt is created and sent to LLM.

5. Success/failure modifiers are included in the LLM response.

6. The plot advances, and if multiplayer, control shifts to the following player.

This asynchronous event paradigm allows for replicable story routes across user sessions while maintaining high context fidelity.

F. Fig. 4.3 Sequence Diagram – End Game

A terminal state (win/loss/end condition) is reached in the story.
1. Story conclusion is decided by the user or the system.
2. Prompt engineering is used to construct the final summary (e.g., "Summarize what happened in this story.").
3. Post-game statistics are calculated by the system:
Important decisions
The rate of success
- Narrative arcs traveled
4. Data from sessions is stored and preserved.
5. Optional: Options for sharing and user comments are shown.

Social interaction layers, leaderboards, and analytics can all be integrated with a formal end-of-game workflow.

IV. RESULTS

This section presents the functionality, system utility, and technical discoveries of the Quest application, an AI-driven interactive novel.

A. Functionality & System Utility

Quest's functionality successfully accomplished its main objective of allowing users to interact with novels with complete user agency, to create a personalized experience. The AI-driven narrator dynamically continues a context based on the preset data and the unconstrained user input. Quest also uses a logging system for context retention and awareness. This formation ensures that the narrator's story maintains logical continuity and reduces AI hallucinations.

In Quest, users are not bound to specific keywords or commands to progress in the story. Its natural language processing allows for a more dynamic interaction with the contexts generated. Users can shape the story to their understanding, making the experience much more relatable. This also ensures each playthrough differs from prior playthroughs, which significantly increases replay value.

Quest also uses a dice roll mechanism to ensure each playthrough is different from prior playthroughs. This mechanism introduces an additional layer of unpredictability to each playthrough. The dice roll mechanism also overrides the over agreeable nature of the model, which creates a more realistic experience by ensuring that the protagonist is not always successful in accomplishing an interaction.

Quest has multiplayer experience, which allows users to invite friends to join as extra protagonists so they can shape the narrative together. It uses turn-based logic to determine which protagonist can interact with the context next. This reduces potential chaos and ensures an organized representation of data to both the user and the AI-narrator. This structured interaction helps in developing a more contextually consistent experience.

Quest functions effectively without additional model training. Relevant game session data, such as Story description, NPCs' context, and game rules, are prompt-injected into the AI's prompt, which allows the narrator to function effectively without the need to train the model. This makes Quest a significantly flexible platform that can be adapted to provide various chatbot experiences just by modifying the variables injected into its prompt.

Code-based logic is integrated with natural language prompts to cue the narrator to trigger meaningful events, such as concluding the story or when a specified threshold has been met. This mechanism gives developers more control over the narrator by defining what information is available to, or withheld from, the narrator using code-based logic.

Furthermore, Quest is implemented with accessibility in mind to ensure adoption by a wider audience. Firstly, Quest's frontend is designed to be minimalist and straightforward so users can intuitively use the web app without confusion. Each page has a specified purpose and avoids unnecessary information. Secondly, Quest is prompt-injected to use simplified narration so that while it is descriptive and immersive, a wide range of English speakers, from beginners to fluent, can understand the story. Thirdly, Quest is not just for readers because users can hear the story using its text-to-speech feature and interact with it using its voice-to-text feature. Lastly, we have implemented a customizable UI to allow users to personalize the format of their text for better readability.

Moreover, Quest uses RESTful backend endpoints to increase accessibility for other developers. The backend just needs to receive data through HTTP requests, making it easy to connect with different frontends, such as React, Vue, Unity, and potentially voice assistant systems like Alexa.

B. Discoveries

There were several discoveries during the development of Quest, such as the limitation of OpenAI's API, the impact of integrating code-based logic with prompt injection, and the challenges of prompt engineering. These discoveries helped us execute more effective methods for context generation and AI-guided interactions, while reducing hallucinations.

The first discovery we had during development was OpenAI's API limitation. One limitation discovered about OpenAI's API was that by default, each API call is stateless, therefore, it does not store context in memory for logical continuity. To maintain logical continuity, all of the past contexts must be prompt-injected into each new prompt. This was resolved by implementing a log system that tracks each game session's generated context and user interactions, and prompt-injects them into each new prompt. The log system ultimately acts as the narrator's (Chatbot Agent's) memory. Another limitation of OpenAI's API is its over agreeableness, which led us to our second discovery.

Prompt engineering can be more effective when combined with code-based logic because it allows developers to have more control over the agent's behavior, measure specific variables, and reduce tokens used in each prompt. Since the AI's reasoning is a black box, integrating code-based logic, such as a success or failure threshold, adds controllable boundaries for the developer, giving the developer more dynamic control over the narrator.

The impacts of these boundaries are much easier to pinpoint and measure because code-based logic is more deterministic than probabilistic. This means that if inputs are held constant, the output will not change, making it easier to measure the impact in a specific context based on different responses. To simplify this thought process using an analogy, imagine these code-based logics as guardrails that developers can hold on to so they don't get lost in the AI's brain.

Moreover, code-based logic can be integrated with prompts to significantly reduce token usage costs. At times, conveying a rule or mechanism, such as the dice roll threshold, to the narrator requires wordy prompts. This will increase the number of tokens used, but it might not even work accurately due to its probabilistic nature. Whereas, Code-based logic integrated with the prompts reduces the need for wordy prompts because it relies on an external deterministic logic.

Lastly, while prompt engineering, we discovered methods to eliminate unwanted prompt injections. The most notable approach implemented ensures that user inputs are always treated as a dialogue within the story. This is done by binding user interactions with a specific prefix inside each prompt.

V. CONCLUSION

Summary and recommendations for future work. (What we can be done to improve this application):
Quest revolutionizes interactive storytelling with LLMs. Players shape narratives freely, eliminating rigid rules or the need for a game master. As far as the future of Quest goes, we have several avenues to explore which enhance both gameplay and depth.

Different stories will have different narrator tones. Playing a horror game, your narrator will sound like they're telling a spooky story over a campfire. A game that needs to give you more of a laugh? Why not have Buggy the Clown narrate this story for you? Introducing UI elements to manage health, abilities, and effects will deepen engagement and better track player status throughout the story. A difficult story that big brother and you cannot finish? We can tone that down for you because sometimes you just have to know the ending. Online co-op and cross-device play break barriers of distance and time. A creator mode will enable community-driven content—make your own story or pick from other community-created stories. Everyone loves stats (at least we do). "Play of the Game," MVP, and other analytics encourage social engagement. With the boom of technology like DALL-E and SORA, Quest will be able to further immerse users through AI-generated imagery.

While Quest is designed to be a simple and straightforward MVP, its implications can alter how current interactive stories and visual novels are both played and created. AI elevates storytelling from static to adaptive and multidimensional. Anyone can now dive into rich plots without prior creative writing skills—do what you know and let AI handle the gaps. It enables historical roleplay, ethical simulations, and comprehension training. Quest can be used to explore plot possibilities, what-ifs, character developments, and more. It can also serve as a medium for reflective storytelling or emotional exploration. Voice/text integration ensures accessibility for motor or visual impairments.

Quest expands beyond gaming into education, entertainment, and content creation. Some of the projections we see happening include but are not limited to the following: Language learners gain fluency through interactive conversations. Streamers can host audience-driven adventures. Designers can test branching plots. Its foundation enables ever-evolving narrative worlds. As far as the future of Quest goes, we have several avenues to explore which enhance both gameplay and depth. As technology evolves, so will the gameplay it enables, opening the door to boundless worlds waiting to be discovered and shaped, whether by a solo player or a group of friends.

VI. ACKNOWLEDGMENT

This capstone project would not have been possible without the guidance, support, and encouragement of instructor Henry Wong and Pace University.

VII. REFERENCES:

[1] N. C. Kim, J. S. Park, and Y. S. Lee, "A Study on Interactive Storytelling Using GPT-based Natural Language Generation," Journal of AI & Society, vol. 35, no. 2, pp. 185–197, 2021.

[2] A. Reed, J. Garbe, and M. Wardrip-Fruin, "Narrative Coherence in Interactive Story Systems," in Proc. Int. Conf. on Interactive Digital Storytelling (ICIDS), 2016, pp. 139–151.

[3] M.-L. Ryan, D. L. Koenitz, and A. Mateas, "Interactive Narrative as a Hybrid Art Form: Researching Mobile Storytelling," Digital Humanities Quarterly, vol. 13, no. 2, 2019.

[4] M. Kreminski, A. Dickinson, and M. Wardrip-Fruin, "Felt Presence in Interactive Narrative: AI as Co-Author," in Proc. of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE), 2020, pp. 45–52.

[5] R. Teymourzadeh, T. A. Bak, and M. Y. I. Idris, "Implementation of Real-Time Web Applications Using Node.js and MongoDB," in Proc. IEEE Int. Conf. on Computer and Information Sciences (ICCOINS), Kuala Lumpur, Malaysia, 2014, pp. 1–6.

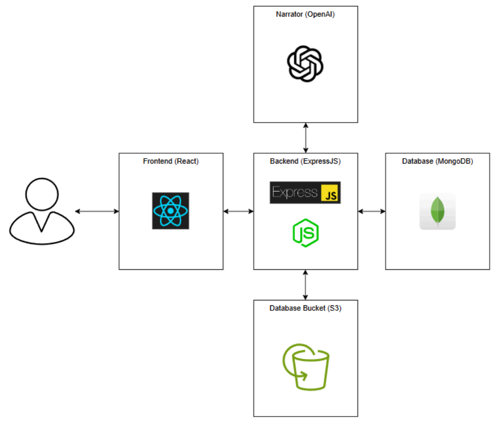


Fig. 1 Architecture Diagram

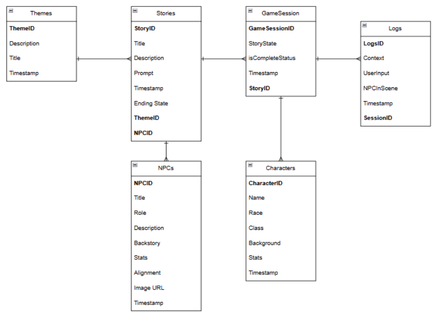


Fig. 2 ER Diagram – Data Schema and Relational Mapping

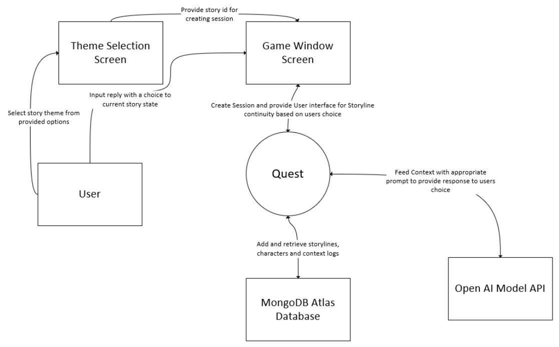

Provide story id for creating session
Theme Selection Screen
Game Window Screen
Input reply with a choice to current story state
Create Session and provide User interface for Storyline continuity based on users choice
Select story theme from provided options
Quest
Feed Context with appropriate prompt to provide response to users choice
User
Add and retrieve storylines, characters and context logs
MongoDB Atlas Database
Open AI Model API

Fig. 3 Context Diagram

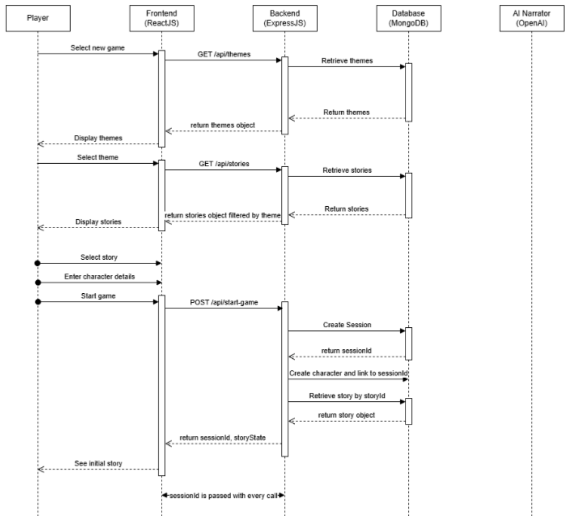


Fig. 4.1 Sequence Diagram - Start Game

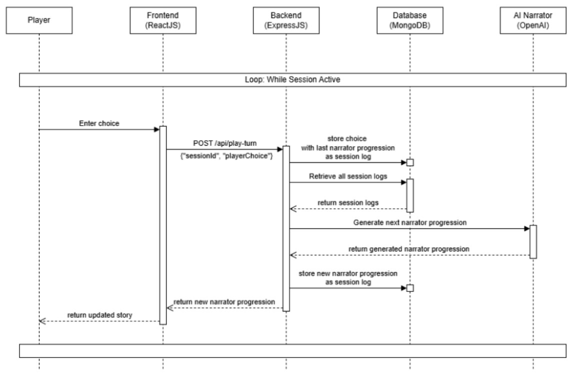

Player
Frontend (ReactJS)
Backend (ExpressJS)
Database (MongoDB)
AI Narrator (OpenAI)
Loop: While Session Active
Enter choice
POST /api/play-turn {"sessionId", "playerChoice"}
store choice with last narrator progression as session log
Retrieve all session logs
return session logs
Generate next narrator progression
return generated narrator progression
store new narrator progression as session log
return new narrator progression
return updated story

Fig. 4.2 Sequence Diagram - Player Turn

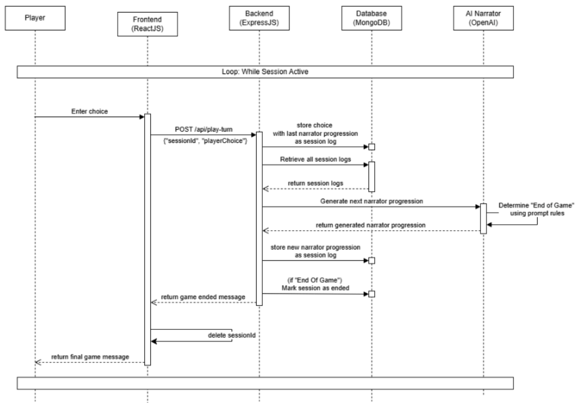

Player
Frontend (ReactJS)
Backend (ExpressJS)
Database (MongoDB)
AI Narrator (OpenAI)
Loop: While Session Active
Enter choice
POST /api/play-turn {"sessionId", "playerChoice"}
store choice with last narrator progression as session log
Retrieve all session logs
return session logs
Generate next narrator progression
Determine "End of Game" using prompt rules
return generated narrator progression
store new narrator progression as session log
(if "End Of Game") Mark session as ended
return game ended message
delete sessionId
return final game message

Fig. 4.3 Sequence Diagram - End Game

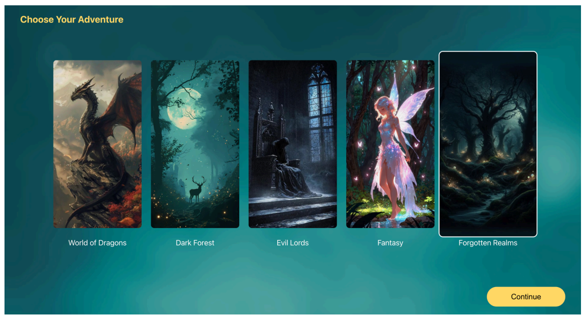
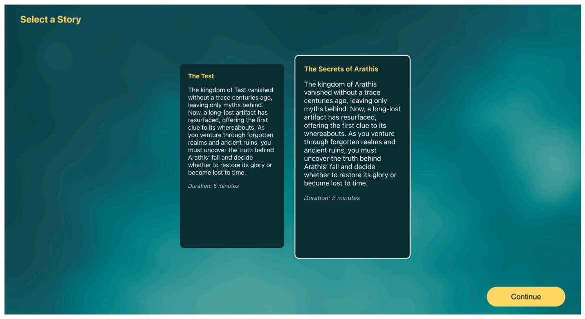


Fig. 5 Theme Selection Page


Select a Story

The Test

The kingdom of Test vanished without a trace centuries ago, leaving only myths behind. Now, a long-lost artifact has resurfaced, offering the first clue to its whereabouts. As you venture through forgotten realms and ancient ruins, you must uncover the truth behind Arathis' fall and decide whether to restore its glory or become lost to time.

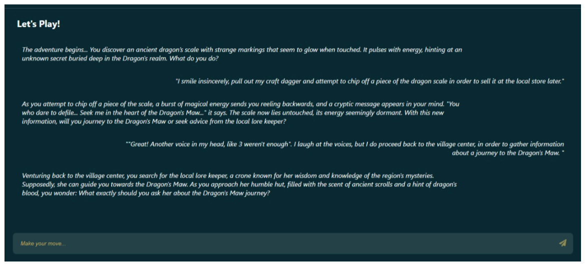
Duration: 5 minutes

The Secrets of Arathis

The kingdom of Arathis vanished without a trace centuries ago, leaving only myths behind. Now, a long-lost artifact has resurfaced, offering the first clue to its whereabouts. As you venture through forgotten realms and ancient ruins, you must uncover the truth behind Arathis' fall and decide whether to restore its glory or become lost to time.

Duration: 5 minutes

Fig. 6 Story Selection Page


Let's Play!

The adventure begins... You discover an ancient dragon's scale with strange markings that seem to glow when touched. It pulses with energy, hinting at an unknown secret buried deep in the Dragon's realm. What do you do?

"I smile insincerely, pull out my craft dagger and attempt to chip off a piece of the dragon scale in order to sell it at the local store later."

As you attempt to chip off a piece of the scale, a burst of magical energy sends you reeling backwards, and a cryptic message appears in your mind. "You who dare to defile... Seek me in the heart of the Dragon's Maw..." it says. The scale now lies untouched, its energy seemingly dormant. With this new information, will you journey to the Dragon's Maw or seek advice from the local lore keeper?

""Great! Another voice in my head, like 3 weren't enough". I laugh at the voices, but I do proceed back to the village center, in order to gather information about a journey to the Dragon's Maw. "

Venturing back to the village center, you search for the local lore keeper, a crone known for her wisdom and knowledge of the region's mysteries. Supposedly, she can guide you towards the Dragon's Maw. As you approach her humble hut, filled with the scent of ancient scrolls and a hint of dragon's blood, you wonder: What exactly should you ask her about the Dragon's Maw journey?

Fig. 7 Game Play Page