Situation Puzzle Game Project Proposal  
Bo Zhu bozhu2 Jun Wen junwen2 Shen Rong shenr3 Yi Wang yw80

Pitch  
A situation puzzle (sometimes called a lateral thinking puzzle) typically presents a strange or intriguing scenario and challenges players to uncover the hidden backstory or missing context. The user can only ask yes/no or short-answer questions to unravel the mystery. Our software leverages the ChatGPT API to provide real-time responses to these player inquiries, guiding them through the puzzle in a fun and intuitive way. This project aims to offer an interactive experience that blends classic puzzle-solving with modern AI-driven insights.

Functionality

1. **Puzzle Library**
   * The application includes a collection of short puzzle scenarios, each requiring players to figure out “why” or “how” a certain event happened.
2. **AI-Powered Question Handling**
   * Players ask yes/no or short-answer questions; ChatGPT responds with context-appropriate answers, offering hints and clarifications.
3. **Adaptive Difficulty**
   * The system adjusts the level of detail in ChatGPT responses based on players’ progress, making it easier or harder to get hints.
4. **Interactive Story Progression**
   * As players unlock clues, they see a short narrative timeline or bullet-point version of the story, adding more depth after each correct discovery.
5. **Leaderboard and Social Features**
   * Timed or scored puzzle modes allow players to see how quickly they can solve a scenario.
   * Users can share completed puzzles and invite friends to compare results.
6. **Puzzle Authoring Tool**
   * An optional backend feature that allows admins or puzzle creators to add new puzzle scenarios through a simple interface.
7. **Hint System and Clue Unlocks**
   * If players get stuck, they can spend “hint points” to get extra clarifications or partial solutions, guided by ChatGPT’s advanced logic.

Components

* **Backend**– A minimal server (e.g., Node.js with Express or Python Flask) that handles user authentication, puzzle data storage, and API requests to ChatGPT.  
  – A database (like SQLite or PostgreSQL) to store user progress, puzzle definitions, and solution reveals.
* **ChatGPT Integration**– Calls to the ChatGPT API are used to handle player questions and generate context-aware yes/no or short-answer replies.  
  – Basic content filtering or moderation layer ensures responses remain appropriate.
* **Frontend**– A web-based interface (React or Vue) that displays puzzle descriptions, captures player questions, and renders ChatGPT replies in real time.  
  – Progress bars or other UI elements show how close the player is to solving the puzzle, based on the revealed clues.

Weekly Planning

| **Week # (Dates)** | **Tasks** |
| --- | --- |
| **Week 1** | 1. Set up the project’s repository and basic backend structure.  2. Implement user registration and login. |
| **Week 2** | 1. Connect the backend to ChatGPT’s API.  2. Create the database schema for puzzle definitions and user progress. |
| **Week 3** | 1. Develop a minimal frontend to display one puzzle scenario and handle user questions.  2. Ensure ChatGPT responses are formatted for yes/no or short answers. |
| **Week 4** | 1. Add advanced puzzle modes (leaderboard, hint system).  2. Test puzzle logic with multiple scenarios. |
| **Week 5** | 1. Expand puzzle library (multiple puzzle sets).  2. Integrate adaptive difficulty for ChatGPT responses. |
| **Week 6** | 1. Implement puzzle authoring tool for admins.  2. Add social or sharing features. |
| **Week 7** | 1. Refine the user experience, adding theming or seasonal events.  2. Bug fixes and final performance optimizations. |

Potential Risks

1. **API Rate Limits and Costs**
   * The ChatGPT API may have usage caps or fees. We can mitigate this through caching frequently asked questions and using fallback responses for repeated queries.
2. **Overly Complex AI Responses**
   * ChatGPT might provide more context or clues than intended. Implementing guardrails or strict prompt engineering ensures consistent, puzzle-appropriate answers.
3. **Database Overhead**
   * Storing user progress and numerous puzzles could grow in complexity. Proper indexing and data modeling will keep lookups efficient.
4. **Security and Cheating**
   * Players might try to manipulate the puzzle or skip steps. We can keep the solution hidden server-side and only reveal partial clues via verified steps.

Teamwork

* **Backend/AI Team**: Sets up database schema, integrates ChatGPT, and implements puzzle logic.
* **Frontend/UI Team**: Builds responsive puzzle interfaces, ensures smooth real-time question-and-answer flow, and manages user engagement features.
* Daily or weekly stand-ups keep tasks aligned. A shared Kanban board (e.g., Trello or GitHub Projects) helps track progress.

By combining the immersive nature of a classic situation puzzle with the dynamic capabilities of ChatGPT, this project aims to deliver an entertaining and mentally stimulating experience. This is different from the other puzzle game because the player can ask whatever question they want, and the AI will give the correct feedback. Different from choosing response from limited options, the player can write all sorts of interesting questions they like and all of them can get a response.