**Hackathon Project Phases Template**

**Project Title:**

Playful AI: Intelligent Board Game Opponents and Advisors

**Team Name:**

Playing warriors

**Team Members:**

* Ch.Nohitha sri
* K.Chandhana sahithi
* T.Sneha
* K.Srivalli

**Phase-1:**

**Objective:**

1. Develop AI-driven opponents that adapt to player strategies for an engaging board game experience.
2. Create intelligent advisors that provide real-time insights and suggestions to improve gameplay.
3. Enhance game balance and challenge by leveraging machine learning and strategic analysis.
4. Foster an interactive and dynamic gaming environment through AI-driven decision-making.

**Key Points:**

**1.Problem Statement:**

Board games have long been a popular form of entertainment, but finding challenging and engaging opponents or receiving strategic advice can be difficult. For casual players, it’s often hard to find a suitable level of challenge or to get valuable insights into game strategies. For board game enthusiasts and professionals, there is a need for advanced tools that can simulate intelligent opponents and offer strategic guidance. An AI-powered solution that enhances gameplay by providing intelligent opponents and strategic advice can address these challenges, enriching the board game experience for everyone involved.

**2.Proposed Solution:**

* AI Opponents – Adaptive machine learning models for dynamic gameplay.
* Smart Advisors – Real-time strategy suggestions based on game state.
* Balanced Gameplay – Reinforcement learning for fair difficulty.
* Scalable System – Modular AI for various board games.

**3.Target Users:**

* **Board Game Enthusiasts** – Casual and competitive players seeking intelligent AI opponents.
* **Game Developers** – Creators looking to integrate smart AI into their board games.
* **Educators & Learners** – Students and teachers using AI for strategic thinking and learning.
* **AI & Game Researchers** – Professionals exploring AI-driven gameplay mechanics

**4.Expected Outcome:**

Playful AI can transform the experience of casual board game players by offering intelligent opponents that adapt to different skill levels. Players select their preferred game and difficulty level, and the AI generates a dynamic opponent capable of providing a balanced challenge. This ensures that players have an engaging and enjoyable experience, regardless of their skill level. By offering opponents that adjust their strategy based on the player’s actions, Playful AI helps maintain interest and provides a satisfying gaming experience.

**Phase-2: Requirement Analysis**

**Objective:**

**Key Points:**

1. **Technical Requirements:**
   * Programming Language: Python
   * Backend: Google Gemini Flash API
   * Frontend: Streamlit Web Framework
2. **Functional Requirements:**

**** AI Opponent Generation

* Provide AI-generated opponents with adjustable difficulty.
* Adapt AI behavior using reinforcement learning for a balanced challenge.

 Real-Time Strategic Advice

* Offer move recommendations based on the current game state.
* Explain reasoning behind AI suggestions for player learning.

 G**ame Selection & Customization**

* Allow users to choose from different board games.
* Enable difficulty level selection for AI opponents.

 **User Interaction & Gameplay Experience**

* Support interactive gameplay via the Streamlit web interface.
* Ensure smooth real-time responses for a seamless gaming experience.

 **Scalability & Integration**

* Enable easy integration of new board games with modular AI architecture.
* Use API-based queries to interact with the Google Gemini Flash API.

 **Performance & Fair Play**

* Ensure AI opponents do not have unfair advantages.
* Optimize response times for real-time interactions.

**3.Constraints & Challenges:**

 **Computational Efficiency** – Optimize AI models and use caching to reduce response time.

 **API Reliability** – Implement fallback mechanisms in case of API failures.

 **Scalability** – Design a modular AI framework for easy integration of new games.

 **Adaptive AI** – Use reinforcement learning with feedback loops to balance difficulty.

**Phase-3: Project Design**

**Objective:**

Develop the architecture and user flow of the application**.**

**+--------------------------+**

| User Input |

| (Move: e2e4, g8f6, etc.)|

+------------+-------------+

|

v

+--------------------------+

| Validate Move |

| - Check legality |

| - Store in history |

+------------+-------------+

|

v

+--------------------------+

| AI Move |

| - Send FEN to Gemini AI |

| - Receive UCI move |

| - Validate AI move |

+------------+-------------+

|

v

+--------------------------+

| Update Board |

| - Push move to board |

| - Render chessboard |

| - Show move history |

+------------+-------------+

|

v

+--------------------------+

| Check Game Status |

| - Ongoing / Checkmate |

| - Stalemate / Draw |

+------------+-------------+

|

v

+--------------------------+

| Restart or End Game |

| - Reset state |

| - Show final result |

+--------------------------+

**1. Architecture Overview**

**A modular, API-driven architecture ensures scalability and efficiency.**

**Components:**

1. **Frontend (User Interface) –** Streamlit
   * User dashboard for game selection & AI settings.
   * Interactive board game UI for real-time gameplay.
2. **Backend –** Google Gemini Flash API
   * Processes game state and generates AI moves.
   * Provides strategic advice based on AI calculations.
3. **AI Engine**
   * Uses reinforcement learning to adapt difficulty.
   * Implements decision trees and heuristics for different board games.
4. **API Layer**
   * Handles communication between frontend and AI engine.
   * Ensures secure and efficient data exchanges.

**2. User Flow**

**Step 1:** User Access & Game Selection

* User opens the Streamlit web app.
* Selects a board game (e.g., Chess, Go, Checkers).
* Chooses AI difficulty (Beginner, Intermediate, Advanced).

**Step 2:** Game Setup & AI Opponent Initialization

* AI opponent is generated using Google Gemini Flash API.
* The game board is displayed with an interactive interface.

**Step 3:** Gameplay Interaction

* User makes a move.
* AI processes the move and provides a response.
* Strategic hints are displayed based on user actions.

**Step 4:** Adaptive Learning & Fair Play

* AI adjusts strategy dynamically based on user performance.
* Reinforcement learning ensures fair challenge balance.

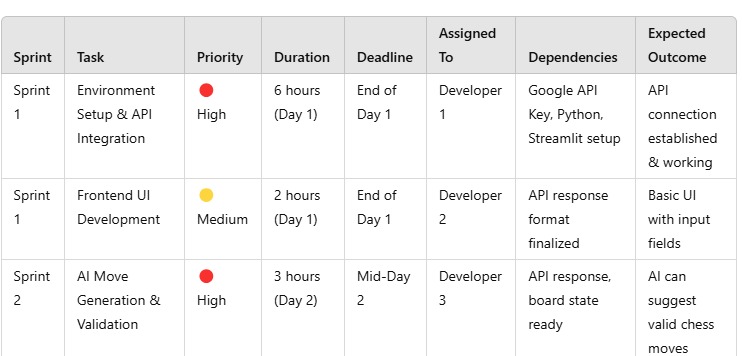
**Step 5:** Game Completion & Insights

* Game ends when a win/loss/draw condition is met.
* AI provides a summary of key moves and strategy insights.

**Phase-4: Project Planning (Agile Methodologies)**

**Objective:**

Break down development tasks for efficient completion.





**Sprint Planning with Priorities**

**Sprint 1 – Setup & Integration (Day 1)**

**(🔴 High Priority)** Set up the **environment** & install dependencies.  
 **(🔴 High Priority)** Integrate **Google Gemini API**.  
 **(🟡 Medium Priority)** Build a **basic UI with input fields**.

**Sprint 2 – Core Features & Debugging (Day 2)**

**(🔴 High Priority)** Implement **search & comparison functionalities**.  
 **(🔴 High Priority)** Debug API issues & handle **errors in queries**.

**Sprint 3 – Testing, Enhancements & Submission (Day 2)**

**(🟡 Medium Priority)** Test API responses, refine UI, & fix UI bugs.  
 **(🟢 Low Priority)** Final **demo preparation & deployment**.

**Phase-5: Project Development**

**Objective:**

Implement core features of the Playful AI.

**Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** Streamlit
   * **Backend:** Google Gemini Flash API
   * **Programming Language:** Python
2. **Development Process:**
   * Implement **API key authentication** and **Gemini API integration**.
   * Develop **to get suggestion from ai**
   * Optimize **search queries for performance and relevance**.
3. **Challenges & Fixes:**
   * **Challenge:** Delayed API response times.  
      **Fix:** Implement **caching** to store frequently queried results.
   * **Challenge:** Limited API calls per minute.  
      **Fix:** Optimize queries to fetch **only necessary data**.

**Final Submission**

1. Project Report Based on the templates
2. Demo Video (3-5 Minutes)
3. GitHub/Code Repository Link
4. Presentation