**Hackathon Project Phases Template** for the Playful Ai project

# Hackathon Project Phases Template

**Project Title:**

**Playful Ai: Intelligent Board Game Opponents and Advisors**

**Team Name:**

**CodeXplosion**

**Team Members:**

Manish Kain

Sujith

Daivaghna

Jyothika

## Phase-1: Brainstorming & Ideation

**Objective:**

**Playful AI: Intelligent Board Game Opponents and Advisors aims to make board**

**games more fun and engaging using AI. It creates smart opponents that**

**challenge players and advisors that suggest the best moves. The AI can learn,**

**adapt to different skill levels, and even teach strategies, making games**

**exciting for everyone.**

**Key Points:**

1. **Problem Statement:**

○ Many board games rely on human opponents, which can make it difficult for players to practice,

improve, or find a challenging match at any time. Existing AI opponents often lack adaptability,

making them either too easy or too difficult. Additionally, players may struggle to learn advanced

strategies without proper guidance

○ Playful AI: Intelligent Board Game Opponents and Advisors aims to solve

this by developing AI that can act as both a competitive opponent and a

helpful advisor. The AI will adapt to different skill levels, provide strategic

recommendations, and enhance the overall gameplay experience, making

board games more accessible, educational, and engaging for all players.

.

1. **Proposed Solution:**

○ Adaptive AI Opponents – AI that adjusts its difficulty based on the player's skill level, providing a balanced and engaging challenge

○ Strategic Advisors – AI-powered guidance that helps players understand game mechanics, suggest optimal moves, and teach strategies

1. **Target Users:**

○ Board Game Enthusiasts.

○  **Beginner**s and Learners

○  **Solo Player**s.

1. **Expected Outcome:**

○ Smarter AI Opponents – AI that adapts to different skill levels, making games more engaging.

○ Increased Player Engagement – More enjoyable gameplay for both solo and multiplayer modes.

## Phase-2: Requirement Analysis

**Objective:**

Define the technical and functional requirements for the Playful ai board Games.

**Key Points:**

1. **Technical Requirements:**

○ Programming Language: **Python and Node.js**

○ Backend: **FastAPI (Python) or Node.js**

○ Frontend: **React.js (or Next.js for Server-Side Rendering)**

○ Database: PostgreSQL (SQL) or MongoDB (NoSQL).

1. **Functional Requirements:**

○ AI Opponent & Gameplay – Intelligent ai opponents to enhance the gameplay.

○ AI Advisor & Hints – Stratergies advisor inbuilt in game for the player to help to tackle.

○ User Interface & Experience – Clean UI for better game navigation.

○ Optimized performance.

1. **Constraints & Challenges:**

○ AI Complexity – Developing AI that plays well across different skill levels is challenging.

○ Real-time Performance – Less game crashes , more smooth gameplay.

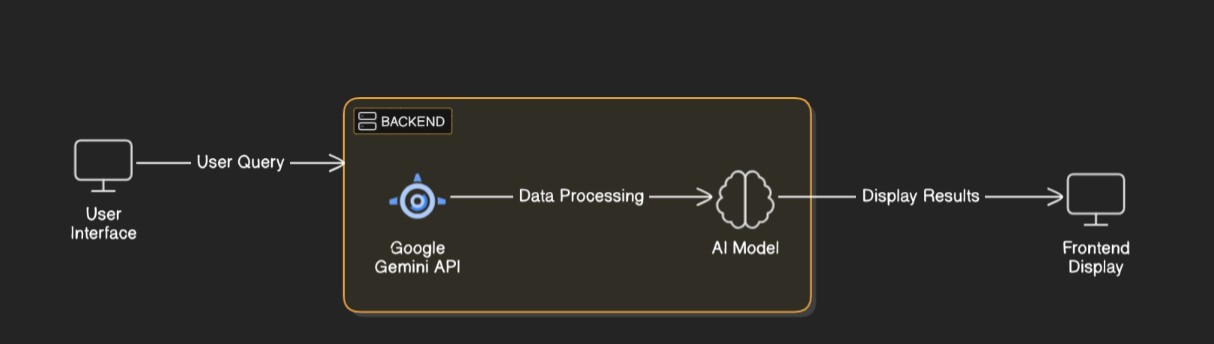
○ Game Variety – Multiple game modes for the player to play any game of his choice.

○ Storage & Processing Power – Stores the player’s name and his achievements status in the game server.

## Phase-3: Project Design

**Objective:**

Develop the architecture and user flow of the application.



**Key Points:**

1. **System Architecture:**

○ Frontend (User Interface).

○ Backend (API & Game Logic.

○ AI Game Engine.

○ Database.

1. **User Flow:**

○ Step 1: User enters a value in the no.of players window, neither with players nor with

the ai bot.

○ Step 2: The backend **then boot the game and user can select game of his choice**.

○ Step 3: The app processes the data and thus the user can play a board game of his choice neither with his opponents nor with the bot.

1. **UI/UX Considerations:**

* 1. **Minimalist, user-friendly interface** for seamless navigation.

○ **Filters for number of players, and game modes**.

○ **Dark & light mode** for better user experience.

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

**Objective:**

Break down development tasks for efficient completion.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Task** | **Priority** | **Duration** | **Deadline** | **Assigned To** | **Dependencies** | **Expected**  **Outcome** |
| Sprint 1 | Environment Setup  & API Integration | 🔴 High | 6 hours  (Day 1) | End of Day  1 | Member 1 | Google API Key,  Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡  Medium | 2 hours  (Day 1) | End of Day  1 | Member 2 | API response format finalized | Basic UI with input fields |
| Sprint 2 | Vehicle Search &  Comparison | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Member 1& 2 | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling &  Debugging | 🔴 High | 1.5 hours  (Day 2) | Mid-Day 2 | Member 1&4 | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI  Enhancements | 🟡  Medium | 1.5 hours  (Day 2) | Mid-Day 2 | Member 2& 3 | API response, UI layout completed | Responsive UI, better user experience |
| Sprint 3 | Final Presentation  & Deployment | 🟢 Low | 1 hour  (Day 2) | End of Day  2 | Entire Team | Working prototype | Demo-ready project |

**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:**

* 1. **Frontend:** **React.js**,.

○ **Backend:** FastAPI (Python) or

Express.js (Node.js).

○ libraries- Pillow(9.0.0) , tk (tkinter).

○ **Programming Language:** Python

1. **Development Process:**

○ Implement scripts and libraries, (Node.js for running game scripts , importing lib).

○ Develop ui and game **logic, implement game script.**

○ Rendering graphics of the board games.

1. **Challenges & Fixes:**

* 1. **Challenge:** Delayed Game response times..

**Fix:** Implement better game code and server connectivity (Updating game code).

○ **Challenge:** Ai advisor in hard gameplays.

**Fix:** Optimize the ai advisor to instruct in hard gameplays , providing correct hints if necessary.

**Fix the bugs :** Less bugs in the game.

## Phase-6: Functional & Performance Testing

**Objective:**

Ensure that the AutoSage App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional  Testing | Ludo game(Multiplayer 4) | Relevant 4 players names are visible and scores with them. | ✅ Passed | Tester 1 |
| TC-002 | Functional  Testing | Ludo game( Singleplayer) | Solo player with bot | ✅ Passed | Tester 2 |
| TC-003 | Performance  Testing | API response time under  500ms | API should return results quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Game Glitches should be improved. | ✅ Fixed | Develop er |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | Passed - UI work on pc | Tester 2 |
| TC-006 | Deployment  Testing | Host the app using  Nod.js Sharing | App should be accessible online , offline. | 🚀 Deployed | DevOps |

## Final Submission

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