**Projektauftrag: "Archon Reimagined" als KI-gesteuerte Web-App**

**Ziel:** Entwickle eine browserbasierte Neuinterpretation des Strategiespiels "Archon" mit einem starken Fokus auf den Einzelspieler-Modus (Mensch gegen KI). Die Anwendung soll als moderne Web-App mit einem intuitiven User Interface realisiert werden und für eine spätere Erweiterung (z.B. Zwei-Spieler-Modus) modular aufgebaut sein.

**1. Kern-Spezifikationen:**

* **Plattform:** Web-Browser (Chrome, Firefox, Safari, Edge), responsiv für Desktop und Tablet.
* **Technologie-Stack:**
  + **Frontend/UI:** React oder Vue.js.
  + **Grafik/Spielelogik:** Phaser.js oder eine ähnliche 2D-Web-Game-Engine.
  + **Sprache:** TypeScript.
* **Primärer Spielmodus:** **Einzelspieler (Mensch gegen KI)**. Das Spiel muss von Anfang an mit einer funktionierenden KI für den Computergegner entwickelt werden.
* **Sekundärer Spielmodus (Ausbaubar):** Die Architektur soll es ermöglichen, später einfach einen lokalen Zwei-Spieler-Modus ("Hotseat") hinzuzufügen.

**2. KI-Anforderungen (Zentraler Bestandteil):**

* **Strategie-KI (Rundenbasiert):**
  + **Ziel:** Die KI muss intelligente Züge auf dem Strategiebrett ausführen können.
  + **Verhaltenslogik:**
    1. **Bewertung von Zügen:** Die KI sollte potenzielle Züge bewerten, basierend auf Faktoren wie: Angriff auf eine wertvolle gegnerische Figur, Besetzung eines Kraftfeldes, Flucht vor einer überlegenen Bedrohung.
    2. **Situationsanalyse:** Die KI muss den Lichtzyklus in ihre Entscheidungen einbeziehen und Kämpfe auf für sie vorteilhaften Feldern bevorzugen.
    3. **Ressourcenmanagement:** Die KI sollte ihre Zaubersprüche strategisch und nicht verschwenderisch einsetzen.
  + **Schwierigkeitsgrad:** Beginne mit einer "normalen" KI. Die Architektur sollte es erlauben, später einfachere oder schwierigere KI-Profile zu erstellen (z.B. durch Anpassung der Entscheidungslogik oder durch Hinzufügen einer gewissen Zufälligkeit für eine "einfache" KI).
* **Kampf-KI (Echtzeit):**
  + **Ziel:** Die KI muss ihre Figur im Kampf autonom und kompetent steuern.
  + **Verhaltenslogik:**
    1. **Positionierung:** Die KI soll aktiv ausweichen und eine für ihre Figur vorteilhafte Distanz zum Gegner halten (z.B. Abstand bei Fernkämpfern, Nähe bei Nahkämpfern).
    2. **Angriffs-Timing:** Die KI sollte ihre Angriffe gezielt einsetzen und nicht nur spammen.

**3. Design und User Interface (UI/UX):**

* **Design-Philosophie:** Modern, sauber, minimalistisch, mit klaren Vektorgrafiken oder stilisierten 2D-Illustrationen.
* **UI-Anpassungen für Einzelspieler:**
  + Klare visuelle Unterscheidung zwischen Spieler- und KI-Zügen (z.B. durch Hervorhebungen oder kurze Text-Einblendungen wie "Computer am Zug").
  + Möglichkeit, die Geschwindigkeit der KI-Züge anzupassen (z.B. schnell für erfahrene Spieler, langsam und animiert für Anfänger).
* **Interaktionen:** Flüssige Animationen, Hover-Effekte und intuitive Bedienung per Maus/Touch.

**4. Entwicklungsplan (KI-Fokus):**

1. **Phase 1 (Grundgerüst & Prototyping):** Richte das Web-Projekt ein. Erstelle das Spielbrett und die grundlegende UI. Implementiere die manuelle Bewegung von Figuren für Testzwecke.
2. **Phase 2 (Kampf-Modus & Kampf-KI):** Entwickle die Kampfarena. Implementiere die Spielersteuerung (Tastatur) und eine **erste Version der Kampf-KI**, die eine Figur steuern kann (z.B. auf den Spieler zulaufen und angreifen). Dies ist entscheidend, um die Kern-Gameplay-Schleife früh zu testen.
3. **Phase 3 (Strategie-KI):** Entwickle die Logik für die KI auf dem Strategiebrett. Beginne mit einer einfachen Logik: Die KI wählt einen zufälligen, aber gültigen Angriffszug.
4. **Phase 4 (Integration & Gameplay-Schleife):** Verbinde alle Teile. Der Spieler macht einen Zug, dann macht die KI einen Zug. Wenn es zu einem Kampf kommt, wird die Kampf-Szene mit der entsprechenden KI-Figur geladen. Das Ergebnis wird auf das Strategiebrett übertragen.
5. **Phase 5 (KI-Verfeinerung & Features):** Verbessere die Strategie- und Kampf-KI schrittweise gemäß den oben definierten Verhaltenslogiken. Füge den Lichtzyklus, Siegbedingungen und Zaubersprüche hinzu und bringe der KI bei, diese zu nutzen.
6. **Phase 6 (Ausbau & Feinschliff):** Nach Fertigstellung des stabilen Einzelspieler-Modus: Implementiere den optionalen Zwei-Spieler-Modus. Verbessere die UI, füge Sounds hinzu und bereite die App für das Deployment vor.

sk-proj-x9lY629XKcGZ6YQpRPuXlR4b4K8Fl7oaLIvPCntlyqsMqRyZO6sZm74fmH4rb2Jib5ZMEo1kOIT3BlbkFJ2QQuierpW81uQ77fZxFGl\_2DrTpaVZRbvkQX-vutvZlZ1XhpcCdup3rH1wg64fI3ZIBRjHIGYA

[Archon Reimagined](https://archon-dqsnx2.manus.space/)

<https://archon-dqsnx2.manus.space/>

<https://manus.im/app/iXbxQpdgsBr7jmbavzxFte>

**Supabase**

leouxY1FIOG2lFBW

Project URL

<https://ftgnrwxzuecauxwxtkwp.supabase.co>

API Key

anonpublic

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6ImZ0Z25yd3h6dWVjYXV4d3h0a3dwIiwicm9sZSI6ImFub24iLCJpYXQiOjE3NTc4MDcyNjQsImV4cCI6MjA3MzM4MzI2NH0.sylSikqfQnEFazKmJdmwNTEznwkeYbU7VzpS8\_w2-\_w

import { createClient } from '@supabase/supabase-js'

const supabaseUrl = 'https://ftgnrwxzuecauxwxtkwp.supabase.co'

const supabaseKey = process.env.SUPABASE\_KEY

const supabase = createClient(supabaseUrl, supabaseKey)