Pflichtenheft

<Projektname> Plattform zum Vergleich von Spiele-KIs

<Projektnummer> Gruppe 2

Änderungshistorie				
Version	**Datum**	**Kapitel**	**Änderung**	**Name**
0.1	25.04.2024	Alle	Anlegen und Füllen	Justine Buß
0.2	29.04.2024	3	Formulieren	Justine Buß
0.3	30.04.2024	4, 5, 7, 8.5	Formulieren	Justine Buß
1.0	30.04.2024	Alle	Links, Diagramme, Überarbeiten	Justine Buß

Pflichtenheft

Herausgeber	Technische Hochschule Mittelhessen – FB06 Mathematik, Naturwissenschaften und Informatik	
Dateiname	Pflichtenheft_ver_1.0	
Dokumentenbezeichnung	Pflichtenheft: Plattform zum Vergleich von Spiele-KIs	
Version	1.0	
Stand	Mittwoch, 1. Mai 2024	
Status	In Bearbeitung	
Autor	Justine Buß	
Inhaltlich geprüft von	?	
Freigegeben von	...	
Ansprechpartner	Justine Buß	justine.buss@mni.thm.de
Kurzinfo	„Technische Hochschule Mittelhessen Softwaretechnikprojekt. Pflichtenheft“	

Pflichtenheft

Inhaltsverzeichnis

Abbildungssverzeichnis

Tabellenverzeichnis

Pflichtenheft

1 Einleitung

Das vorliegende Pflichtenheft enthält die an das zu entwickelnde Projekt gestellten funktionalen sowie qualitativen Anforderungen. Mit diesen werden die Rahmenbedingungen für die Entwicklung festgelegt, die im Pflichtenheft detailliert ausgestaltet werden.

2 Allgemeines

2.1 Ziel und Zweck des Dokuments

Dieses Pflichtenheft beschreibt ein umfassendes Dokument, das die Anforderungen und Spezifikationen für die Entwicklung eines bestimmten Produkts, Projekts oder Systems festlegt. Es dient als zentrales Referenzdokument für alle Beteiligten und definiert klar den Umfang, die Funktionen und die Qualitätsanforderungen der zu entwickelnden „Plattform zum Vergleich von Spiele-KIs“. Das Hauptziel dieses Dokuments ist es, sicherzustellen, dass alle Stakeholder (Vgl. 3.3 Zielgruppen) ein einheitliches Verständnis von den Anforderungen haben und dass das Endprodukt die Erwartungen und Bedürfnisse der Benutzer erfüllt. Durch die detaillierte Ausgestaltung der Anforderungen im Pflichtenheft wird ein gemeinsames Verständnis über den Umfang und die Ziele des Projekts geschaffen, was eine reibungslose Umsetzung und erfolgreiche Lieferung ermöglicht.

Das ist nur die Meta-Beschr.

2.2 Ausgangssituation

→ Technischer Fokus

In Anbetracht der wachsenden Bedeutung künstlicher Intelligenz (Vgl. 9 Abkürzungsverzeichnis) in verschiedenen gesellschaftlichen Bereichen ist es zunehmend erforderlich, ein tieferes Verständnis für KI-Konzepte zu entwickeln und diese anzuwenden. Insbesondere im Bildungsumfeld wird es als entscheidend erachtet, Studierenden praxisnahe Erfahrungen zu ermöglichen, um ihre Fähigkeiten in der KI-Entwicklung zu vertiefen. Vor diesem Hintergrund gewinnt das Konzept des Lernens und Lehrens mit spielerischen Methoden zunehmend an Bedeutung. Der Einsatz von interaktiven Plattformen und erlebnisorientierten Ansätzen bietet eine effektive Möglichkeit, das Verständnis und die Anwendung von KI-Konzepten zu fördern. Durch die Integration dynamischer Elemente in den Lernprozess können Interessierte und Studierende auf unterhaltsame Weise praxisnahe Erfahrungen sammeln und ihre Fähigkeiten in der KI-Entwicklung auf ansprechende Art und Weise vertiefen. Diese Herangehensweise erleichtert nicht nur das Verständnis komplexer Konzepte, sondern trägt auch dazu bei, das Interesse und die Motivation der Lernenden zu steigern, da sie aktiv am Prozess beteiligt sind und direktes Feedback erhalten.

Wdh zum Lastenheft

Um diesen Bedarf zu decken, ist es entscheidend, eine benutzerfreundliche Plattform zu schaffen, die das Verhalten von KIs durch interaktive Erfahrungen erkundet, verschiedene KIs vergleicht und das Testen eigener KIs ermöglicht. Während bereits existierende Plattformen einzelne Spiele mit KI-Unterstützung anbieten, fehlt es bisher an einer ganzheitlichen Lösung, die mehrere Spiele integriert und es den Nutzern ermöglicht, sowohl gegen andere Nutzer und verschiedene KIs anzutreten als auch eigene KI-Entwicklungen hochzuladen und zu testen.

2.3 Projektbezug

Das Softwaretechnikprojekt an der Technischen Hochschule Mittelhessen unter der Leitung von Prof. Dr. Frank Kammer strebt mit der Entwicklung einer Plattform, die den Benutzern eine interaktive Möglichkeit zum Sammeln praxisnaher Erfahrungen ermöglicht, die Schließung genau dieser Lücke an. Diese Plattform soll es den Nutzern ermöglichen, nicht nur gegen andere Nutzer anzutreten, sondern auch gegen verschiedene KIs zu spielen. Dabei ist es entscheidend, dass die Plattform benutzerfreundlich gestaltet ist und eine Vielzahl von Spielen unterstützt, um den Lernprozess im Bereich der KI-Entwicklung effektiv und ansprechend zu gestalten.

Anzuwendende Technologien:

- Programmiersprache: Python
- Softwareplattform: Docker
- Maschinelles Lernen Framework: TensorFlow mit Keras
- KI-Entwicklungsframework: AlphaZero

2.4 Beteiligte

Rolle	Name	Fachbereich/ Studienfach	Kontaktinformation
Projektleitung	Thorben Jones	MNI/Ingenieur-Informatik	thorben.jones@mni.thm.de
Stellvertretende Projektleitung	Justine Buß	MNI/Ingenieur-Informatik	justine.buss@mni.thm.de
Teammitglied	Alexander Roos	MNI/Ingenieur-Informatik	alexander.roos@mni.thm.de
Teammitglied	Maximilian Bachmann	MNI/Informatik	maximilian.lars.bachmann@mni.thm.de
Teammitglied	Omar Karkotli	MNI/Informatik	omar.karkotli@mni.thm.de
Teammitglied	Sven Reinhard	MNI/Informatik	sven.roman.reinhard@mni.thm.de

Teammitglied	Pascal Waldschmidt	MNI/Informatik	pascal.waldschmidt@mni.thm.de
Dozent	Prof. Dr. Frank Kammer	MNI	frank.kammer@mni.thm.de

Tabelle 1: Interne Beteiligung

3 Konzept

3.1 Ziele der Entwickelnden

Unser Softwaretechnikprojekt verfolgt das Ziel, eine interaktive Plattform zu entwickeln, die mehrere Zwecke erfüllt. Wir als Entwickelnde möchten dabei nicht nur unsere Kompetenzen erweitern, sondern auch neue Technologien und Konzepte wie künstliche Intelligenz kennenlernen und anwenden. Die Umsetzung dieses Projekts bietet uns die Möglichkeit, praktische Erfahrungen zu sammeln und uns auf zukünftige Herausforderungen in der Berufswelt vorzubereiten. Darüber hinaus sehen wir es als spannende Gelegenheit, gemeinsam mit unseren Kommilitonen ein innovatives Projekt zu realisieren und anderen die Möglichkeit zu geben, von unserer Plattform zu profitieren.

3.2 Ziele der Nutzenden

Für die potenziellen Nutzer der Plattform bietet sie eine Vielzahl von Vorteilen. Sie ermöglicht es, gegen KIs oder andere Spieler anzutreten, eigene KIs gegen vortrainierte Lösungen antreten zu lassen und dabei Spielspaß zu haben. Darüber hinaus soll sie den Nutzer als Werkzeug dienen, um die Funktionsweise von KIs durch interaktive Spieleerfahrung genauer zu verstehen und ihnen die Möglichkeit von Vielfachspielen und Zuganalysen bieten.

Wdh

3.3 Zielgruppen

Die Zielgruppen für unsere Plattform sind vielfältig. Zum einen richten es sich an Studierende der Hochschule, die in verschiedenen Modulen den Einsatz von KI kennenlernen und ihre eigenen KIs auf der Plattform testen können. Dadurch soll eine Lern- und Lehrunterstützung geboten und ein Einblick in vortrainierte KI-Lösungen ermöglicht werden. Zum anderen richtet sich das Projekt an Entwickler, die ihre Kenntnisse und Fähigkeiten im Bereich der KI-Entwicklung erweitern möchten und neue Anwendungen erforschen wollen. Schließlich spricht die Plattform alle KI-Interessenten und Enthusiasten an, die mehr über die Entwicklung und Anwendung von KI erfahren möchten, sowie eigene KIs testen und verbessern wollen.

3 Implementierungsentwurf

3.1 Gesamtsystem

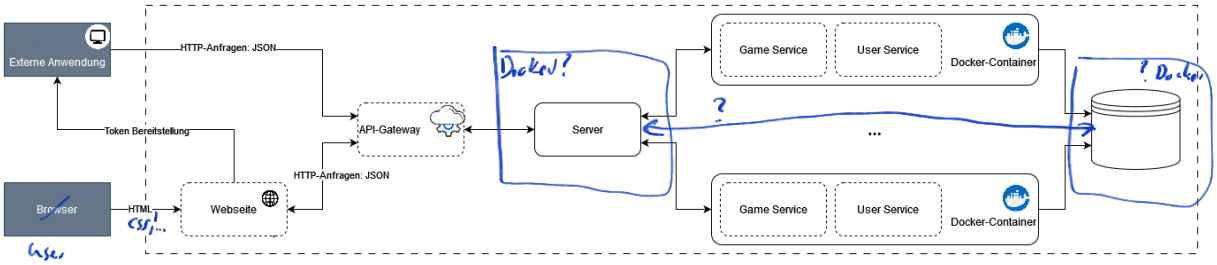


Abbildung 1: Architekturentwurf Gesamtsystem

Im Rahmen des Implementierungsentwurfs wird die Gesamtarchitektur des Systems umfassend beschrieben. Dies umfasst die verschiedenen Systemkomponenten sowie deren Interaktionen, Interdependenzen und Schnittstellen. Das Ziel dieses Abschnitts ist es, einen klaren Überblick über die geplante Softwarelösung zu geben und die grundlegenden Designentscheidungen zu erläutern. Dabei wird insbesondere auf die Integration der Komponenten und die Struktur des Gesamtsystems eingegangen.

Die Architektur des Systems basiert auf dem Client-Server Prinzip ergänzt mit Microservice-Ansätzen. Die Hauptkomponente stellt ein Server dar, der als Vermittler zwischen den Clients (einer Webseite und externen Anwendungen) und den dahinterliegenden Docker-Containern fungiert. Die Clients kommunizieren über eine API (~~Vgl. 9 Abkürzungsverzeichnis~~) mit dem Server, der Anfragen an die entsprechenden Docker-Container weiterleitet. Die Docker-Container stellen die verschiedenen Spielfunktionen bereit und können je nach Anforderung skaliert werden. Der Implementierungsentwurf fokussiert sich darauf, wie der Server die Interaktionen zwischen den Clients und den Docker-Containern verwaltet und wie die Komponenten nahtlos integriert sind, um die gewünschte Softwarelösung effizient zu realisieren.

3.2 REST API mit FastAPI

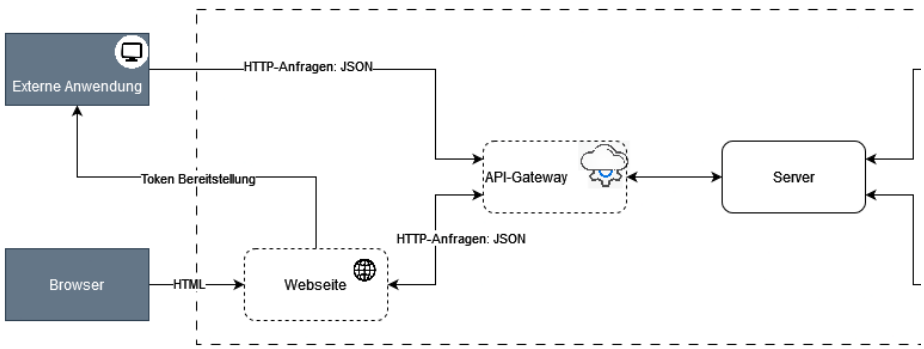


Abbildung 2: Architekturentwurfsausschnitt API Schnittstelle

Die REST API (Vgl. 9 Abkürzungsverzeichnis) wird mithilfe der Python-Bibliothek FastAPI implementiert und fungiert als Schnittstelle für externe Dienste oder Anwendungen, um mit dem Backend zu interagieren. Sie bietet verschiedene Endpunkte, die es ermöglichen, auf bestimmte Aspekte des Spiels oder der Spielinstanzen zuzugreifen. Sie setzt dabei RESTful um und somit den Level 3 Standard des Richardson Maturity Modells. Genauer gesagt HATEOAS (Vgl. 9 Abkürzungsverzeichnis), sodass dem Nutzer der API bei Requests an die API in den Responses mögliche ansteuerbare Endpunkte mitgeliefert werden ausgehend vom aktuellen Zustand in dem sich der Client befindet. Dadurch lässt sich eine Navigation durch die Website und deren Funktionalitäten nahezu ohne Dokumentation handhaben.

So wird es zudem möglich sein, die Interaktionen mit auf der Webseite vorhandenen Interaktionsfeldern, wie Drop-Down-Menüs, Suchleisten, Spielfeldaktionen oder ähnlichem effektiv an das Backend weiterzuleiten und zudem aktualisierte Spielfelder dem Benutzer auf der Weboberfläche mit der Python Bibliothek PyGame ersichtlich zu machen. Zudem bietet die API die Möglichkeit Verbindungen mit externen Anwendungen einzugehen und unterstützt somit eine reine Terminalsteuerung der Software.

//Key beschreibung, .json?

3.3 Socket Server in Python

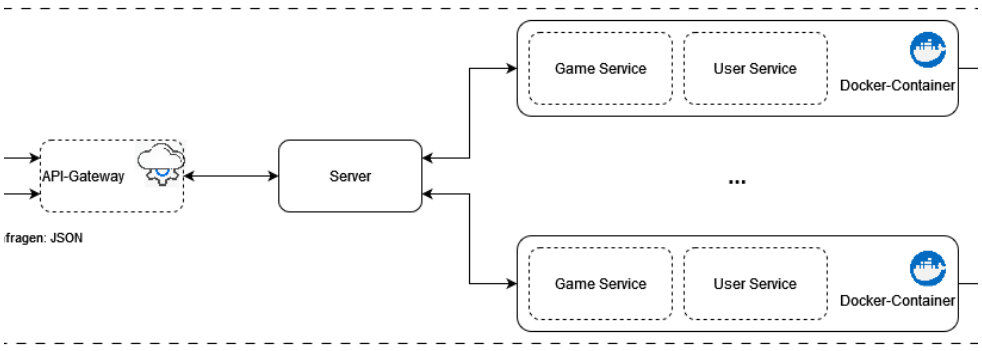


Abbildung 3: Architekturentwurfsausschnitt Server

Die Hauptkomponente des Systems, die die Kommunikation steuert, ist ein Socket Server, der in Python implementiert wird. Sie dient als zentraler Ansprechpartner für alle Anfragen und ihre Hauptaufgabe besteht darin, den Nachrichtenfluss zwischen den verschiedenen Teilen des Systems zu koordinieren. Der Server leitet Anfragen von der API an die entsprechende Spielinstanz weiter und übermittelt Nachrichten von der Spielinstanz zurück an das Frontend oder den externen verbundenen Client.

3.4 Dockerbasierte Spielinstanzen

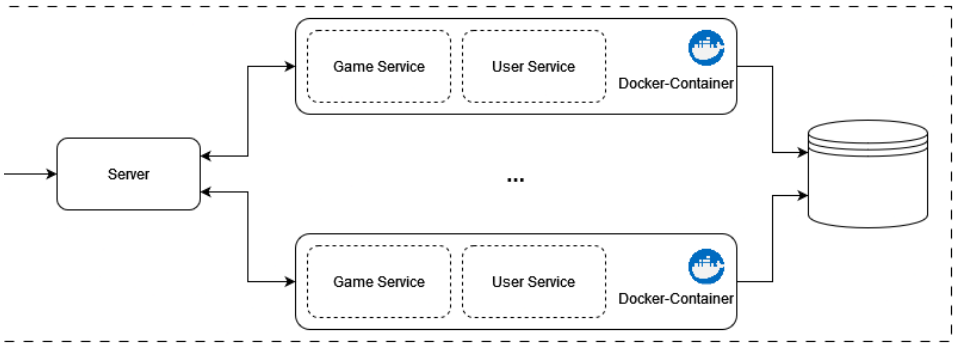


Abbildung 4: Architekturentwurfsausschnitt Docker Container

Zur Bereitstellung eines Mehrspielerbetriebs und der Unterstützung zukünftiger Skalierbarkeit werden die Spielinstanzen in separaten Docker Containern ausgeführt. Jede Spielinstanz kommuniziert dabei über einen internen Socket mit dem zentralen Server. Bei dessen Start meldet sich jede Spielinstanz als Game Client am Server an. Wird eine neue Instanz erstellt, generiert der Server einen eindeutigen Schlüssel, der diesem Docker Container zugeordnet wird. Bei zukünftiger Kommunikation verwaltet der Server diese Schlüssel und leitet Anfragen entsprechend an die Docker Container weiter oder von den Containern zu den entsprechenden Clients.

Innerhalb der Docker Container werden die Spielinstanzen mit Python und den entsprechenden Bibliotheken, die für die Ausführung des jeweiligen Spiels erforderlich sind, implementiert. Beim Erzeugen der Container greift Docker auf die entsprechend benötigten vortrainierten KI-Implementierungen in einer Datenbank zu //format. Das Trainieren der KIs erfolgt auf einem Unsuperviced Learning Paradigma, dem Reinforcement Learning. Der Ansatz orientiert sich an der AlphaZero Lösung von Google DeepMind und spezifischer auf einem angepassten alphazero framework. Die Python Bibliotheken Keras und TensorFlow (sowie davon abhängige Bibliotheken) und PyTorch unterstützen das Trainieren der KIs in neuralen Netzen.

Wie Webseite Node JS-Server?

Verwaltung der Docker-Cont?
Monitoring des Systems?

4 Funktionale (Detail-) Anforderungen

4.1 Must Haves

4.1.1 Webseite

Über gängige Webbrowser erreichbare Webseite, die eine benutzerfreundliche Navigation bietet. Sie sollte reaktionsschnell sein und Inhalte schnell laden, um eine angenehme Erfahrung für Besucher sicherzustellen. Alle Funktionen und Inhalte der Webseite müssen klar strukturiert und leicht zugänglich sein, damit Benutzer problemlos durch die Seiten navigieren können. Wichtige Informationen sollten deutlich sichtbar platziert sein, um eine intuitive Nutzung zu ermöglichen.

Die Webseite muss ein gut sichtbares Impressum und klare Datenschutzbestimmungen enthalten, die den rechtlichen Anforderungen entsprechen. Diese Informationen sollten leicht auffindbar und verständlich formuliert sein, um Transparenz und Vertrauen bei den Nutzern zu fördern. Darüber hinaus müssen die Datenschutzbestimmungen die Erhebung, Verarbeitung und Nutzung von personenbezogenen Daten klar darlegen und die Rechte der Benutzer in Bezug auf ihre Daten deutlich beschreiben.

Mockups ? ?

HTML, CSS, ..? Vue, ..?

4.1.2 Externe Anwendung

Zusätzlich zur herkömmlichen Browser-Navigation soll es möglich sein, über eine Terminaleingabe mit der API und dem dahinterliegenden Server zu interagieren. Diese Funktionalität ermöglicht es Benutzern, bestimmte Aktionen oder Abfragen direkt über eine programmatische Schnittstelle auszuführen, ohne die Webseite besuchen zu müssen. Die API-Kommunikation sollte klar dokumentiert und einfach zu nutzen sein, um eine reibungslose Interaktion mit dem System zu gewährleisten. Gleichzeitig wird durch das Umsetzen des RESTful Paradigmas ein leichteres Navigieren über das Terminal ermöglicht, da ansteuerbare Endpunkte ausgehend vom aktuellen Zustand bei response-Nachrichten mitgeliefert werden.

Auf der Webseite wird zu diesem Zwecke die Verbindungsanleitung hinterlegt und bei Konfiguration über die Seite zu entsprechender Zeit präsentiert.

4.1.3 Umgesetzte Spiele

Es werden verschieden komplexe Spiele angeboten, wobei eine Erweiterung des Spielepools durch Möglichkeiten der Skalierbarkeit im Backend bereitgestellt werden.

Auf welchen Projekten soll hier aufgebaut werden?

- ***Othello (Reversi)***: Strategisches Brettspiel für zwei Spieler. Ziel des Spiels ist es, die meisten Steine auf dem Spielbrett zu besitzen, indem man gegnerische Steine einkreist und sie in die eigene Farbe umdreht.
- ***Tic-Tac-Toe***: Ist ein klassisches Zwei-Spieler-Spiel, bei dem die Spieler abwechselnd ihre Symbole (üblicherweise X und O) in einem Raster platzieren. Das Ziel ist es, drei Symbole in einer Reihe, Spalte oder Diagonale zu platzieren, bevor der Gegner dies schafft. Das Spiel endet entweder in einem Sieg, einer Niederlage oder einem Unentschieden.
- ***Vier gewinnt***: Ist ein strategisches Brettspiel für zwei Spieler. Ziel des Spiels ist es, als Erster vier seiner eigenen Spielsteine horizontal, vertikal oder diagonal in einer Linie zu platzieren. Die Spieler wechseln sich ab, einen Spielstein in eine Spalte fallen zu lassen.
- ***Nim***: Ist ein mathematisches Strategiespiel, bei dem zwei Spieler abwechselnd eine Anzahl von Objekten aus verschiedenen Haufen nehmen. Das Ziel ist es, den Gegner dazu zu bringen, den letzten Gegenstand zu nehmen.
- ***Dame***: Ist ein klassisches Brettspiel für zwei Spieler. Die Spieler bewegen abwechselnd ihre Spielsteine diagonal über das Spielbrett. Ziel ist es, die gegnerischen Spielsteine zu schlagen, indem man über sie hüpft. Das Spiel endet, wenn ein Spieler alle gegnerischen Steine schlägt oder der Gegner nicht mehr ziehen kann.

- ***Go***: Ist ein strategisches Brettspiel für zwei Spieler. Die Spieler platzieren abwechselnd ihre Steine auf einem Gitterbrett, um Territorium zu beanspruchen und gegnerische Steine einzuschließen. Ziel ist es nach einem voll besetzten Brett oder einem Aufgeben, das meiste Territorium zu besitzen.
- ***Waldmeister***: Ist ein taktisches Brettspiel, bei dem die Spieler um die größten Gruppen von gleichhohen oder gleichfarbigen Bäumen konkurrieren.

4.1.4 Spielekonfiguration

Sowohl die Webseite als auch die Terminalunterstützung stellen eine Möglichkeit bereit neben dem zu spielenden Spiel weitere Einstellungen vorzunehmen.

So lassen sich zudem folgende Konfigurationen vornehmen:

- ***Spielmodus***: Zur Unterstützung vieler Anwendungsbereiche werden verschiedene Spielmodi angeboten. Dazu gehören, Spieler gegen Spieler, Spieler gegen von uns vortrainierte KIs, eine KI-Implementierung des Spielers gegen eine vorab trainierte KI, sowie zwei Spieler-KIs.
- ***Spielfeldoptionen***: Bei einfacheren Spielen wie Tic-Tac-Toe, Othello und Vier gewinnt werden unterschiedlich große Spielfelder angeboten, sodass nicht nur die Standardgrößen, wie 3x3 oder 4x4 angeboten werden.
- ***Spielkonfigurationen***: Es wird eine Möglichkeit geben einige Einstellungen in Bezug auf das Spiel vorzunehmen. Dazu gehört vor allem das Auswählen der Spielsteine und damit der Wahl, welcher Spieler das Spiel beginnt.

4.1.5 Spielebeitritt

Nach der Konfiguration eines Spiels, wird ein //key generiert, der es einem Mitspielenden ermöglicht dem Spieleraum beizutreten (im Spieler-Spieler Modus) oder seine KI-Implementierung über die API mit diesem Raum zu verbinden (im Spieler-KI Modus). Dies ist auf zwei Spielende begrenzt.

4.1.6 Spielerepräsentation und -navigation

Da die Spiele sowohl über das Terminal als auch über die Webseite ausgeführt werden können, unterstützen beide Instanzen die Repräsentation des Spielfeldes.

In der Terminalnavigation werden Spielzüge mit entsprechenden Befehlen ausgeführt oder Verbindungsaktionen ausgeführt. Das Spielfeld wird dabei im Terminal dargestellt, sodass eine vollkommen von der Webseite unabhängige Spielerfahrung möglich ist.

Die Webseite unterstützt eine graphische Darstellung des Spiels. Dies wird durch PyGame, eine mächtige Python Bibliothek, gewährleistet. Zudem ermöglicht die Webseite eine mausgesteuerte Bedienung und Interaktion mit den Spielsteinen auf dem Spielfeld, sowie den zusätzlichen Aktionen der Seiten (bspw. Buttons, Suchleiste).

4.1.7 Unterstütze Spielefunktionen

Da einer Spieleinstanz spezifische Räume mit Spielen zugeordnet sind, ist es möglich, Spiele abzubrechen oder aufzugeben und dann mit einem neuen Spielanfang fortzufahren. Spieler haben dabei jederzeit die Option, einen Raum zu schließen und zur Hauptseite zurückzukehren, um entweder das aktuelle Spiel zu verlassen oder ein neues Spiel zu beginnen.

Zudem dienen zwei Optionen vor allem der genaueren Analyse von KI-Verhalten. So wird eine Möglichkeit der Mehrfachspiele für den KI-KI Modus geboten, bei der keine graphische Oberfläche unterstützt wird, jedoch eine statistische Auswertung der Siege und Niederlagen folgt. Zusätzlich lassen sich nach Vollendung eines Spiels die vergangenen Züge durch ein Zeitleistenverhalten betrachten.

4.2 Nice-To-Haves

← Konfigmöglichkeiten des Admins??

4.2.1 Weitere Spiele

Es werden weitere verschieden komplexe Spiele zu dem Spielepool aus 4.1.3 Umgesetzte Spiele angeboten.

- ***Schach*:** Ist ein klassisches strategisches Brettspiel für zwei Spieler. Ziel ist es, den gegnerischen König matt zu setzen, was bedeutet, dass der König bedroht ist und keine Fluchtmöglichkeit hat. Die Spieler bewegen abwechselnd ihre Schachfiguren über das Spielbrett, wobei jede Figur spezielle Bewegungsmöglichkeiten hat.
- ***Mühle*:** Ist ein traditionelles Brettspiel für zwei Spieler. Das Ziel ist es, drei eigene Spielsteine in einer Reihe (Mühle) zu platzieren, wodurch man einen gegnerischen Stein entfernen kann. Hat ein Spieler keine Zugmöglichkeiten mehr, oder besitzt weniger als zwei Steine, hat er verloren.
- ***Halma*:** Ist ein strategisches Brettspiel für zwei bis sechs Spieler (in unserer Implementierung für zwei). Das Ziel ist es, alle eigenen Spielsteine in das gegenüberliegende Feld zu bringen, indem man über benachbarte Spielsteine springt.
- ***Abalone*:** Ist ein taktisches Brettspiel für zwei Spieler. Ziel ist es, sechs gegnerische Kugeln vom Spielbrett zu drängen, indem man mit eigenen Kugeln eine Linie bildet und die Kugeln des Gegners schiebt.
- ***Hex*:** Ist ein strategisches Zwei-Spieler-Brettspiel, bei dem das Ziel darin besteht, eine durchgehende Verbindung von einer Seite des Spielbretts zur gegenüberliegenden Seite zu schaffen, bevor der Gegner dies tut. Die Spieler platzieren abwechselnd ihre Steine auf einem sechseckigen Raster und versuchen, eine ununterbrochene Linie ihrer Farbe zu bilden.

4.2.2 Spielekonfiguration

- ***Schwierigkeitsgrad***: Zur Erweiterung der Spielemöglichkeiten werden verschieden gut trainierte KIs angeboten. Damit erweitert sich die 4.1.4 Spielekonfiguration um einen Schwierigkeitsgrad.

4.2.3 Webseitenerweiterung

Zur Unterstützung des Spielspaß und Belohnung guter Spiele und KI-Implementierungen wird eine Anzeigetafel mit Punktedarstellungen der in einem Spiel erspielten Punkte von KI oder Spieler hinzugefügt. Um dies zu realisieren werden Benutzeridentifikationen mit der Rückmeldung eines Benutzernamens eingeführt.

Es wird eine Möglichkeit bereitgestellt, Spiele für spätere Analysen und Fortsetzungen des Spiels abzuspeichern. Dazu gehört zudem eine Lademöglichkeit der abgespeicherten Spiele. Dazu wird keine Benutzeridentifikation benötigt.

Die Webseite steht zudem in einem weiteren Farbmodus zur Verfügung, sodass nicht nur ein Light-Mode, sondern auch ein Dark-Mode angeboten werden. Die Einstellung lassen sich auf den verschiedenen Seiten der Webseite intuitiv anwählen.

4.3 If-Time-Allows

4.3.1 Zufallsspiele

Es werden weitere Spiele zu dem Spielpool aus 4.2.1 Weitere Spiele hinzugefügt. Diese beinhaltet jedoch den alphazero Formalien nach nicht regelkonformen Bedingungen. Sie werden bestimmt durch Zufallskomponenten im Spiel (bspw. Würfel).

4.3.2 Webseitenerweiterung

Zur Erweiterung der Webseite, wird zunehmend die Barrierefreiheit der Webseite ausgebaut. Sie unterstützt dabei nicht nur Screen Reader, sondern auch vollständig mausfreie Bedienung. Dazu zählt auch die Bereitstellung der Webseite für mobile Geräte, die die Spieleoberfläche und Navigation in einer Toucheingabe erweitern, und eine Mehrsprachenunterstützung der Webseitentexte und Anleitungen in mindestens Englisch und Deutsch.

5 Qualitative Anforderungen

5.1 Allgemeine Anforderungen

Für die Benutzeroberfläche wird eine hohe Benutzerfreundlichkeit angestrebt, wodurch eine intuitive Navigation und Interaktion gewährleistet werden sollen. Dies umfasst klare und verständliche Menüstrukturen, Schaltflächen und Layouts, die den Nutzern eine einfache und angenehme Bedienung ermöglichen. Zudem sollen Fehlermeldungen präzise und informativ sein,

um den Benutzern bei auftretenden Problemen eine effiziente Fehlerbehebung zu ermöglichen. Das System soll eine hohe Zuverlässigkeit aufweisen und eine hohe Verfügbarkeit sicherstellen, um Unterbrechungen im Betrieb auf ein Minimum zu reduzieren.

5.2 Gesetzliche Anforderungen

Das System muss sämtlichen relevanten gesetzlichen Vorgaben und Bestimmungen entsprechen, insbesondere in Bezug auf Datenschutz und Sicherheit. Dies beinhaltet die Einhaltung der DSGVO (Datenschutz-Grundverordnung) sowie anderer lokaler Datenschutzgesetze und -vorschriften.

↘ konkreter?

5.3 Technische Anforderungen

Das System soll plattformübergreifend funktionieren und auf einer Vielzahl von Endgeräten nahtlos und fehlerfrei laufen. Es muss mit gängigen Webbrowsern wie Chrome und Firefox einwandfrei funktionieren.

genaue: Web, Version
Mobile Version?

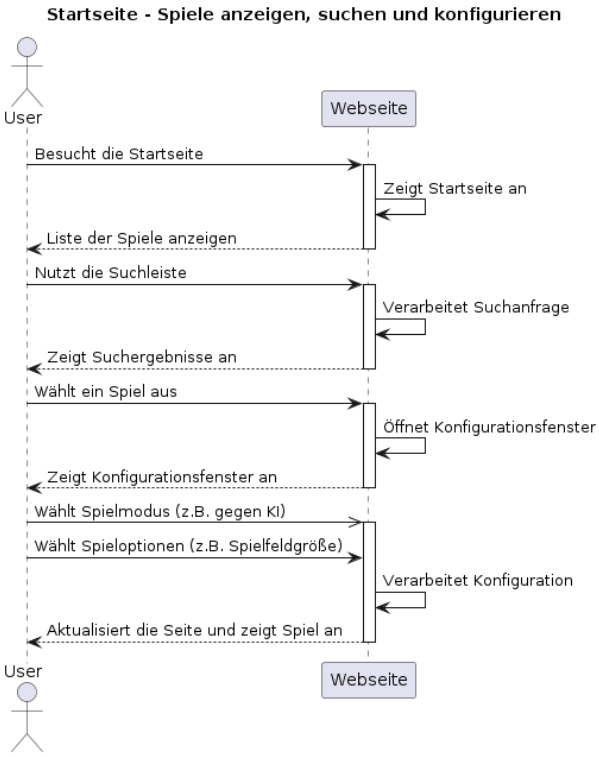
5.4 Weitere Anforderungen

Das System soll skalierbar sein, um zukünftiges Wachstum und eine steigende Anzahl von Benutzern zu unterstützen, ohne die Leistung zu beeinträchtigen. Zusätzlich soll die Client-Server Architektur mit kombinierten Microservices die Flexibilität für zukünftige Erweiterungen und Anpassungen bereitstellen.

↓ wie?

6 Webseite

6.1 Startbildschirm

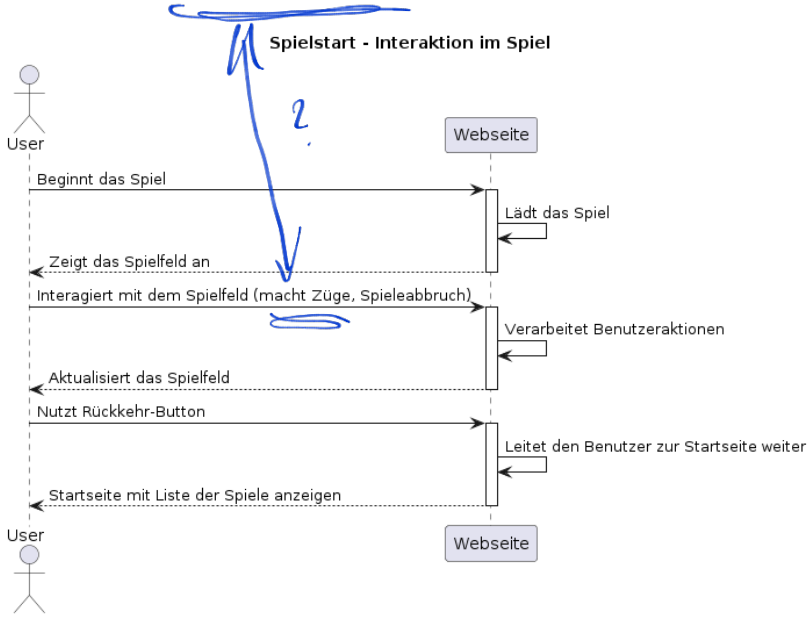


Token für KI?

Ggf. Mockup einfacher

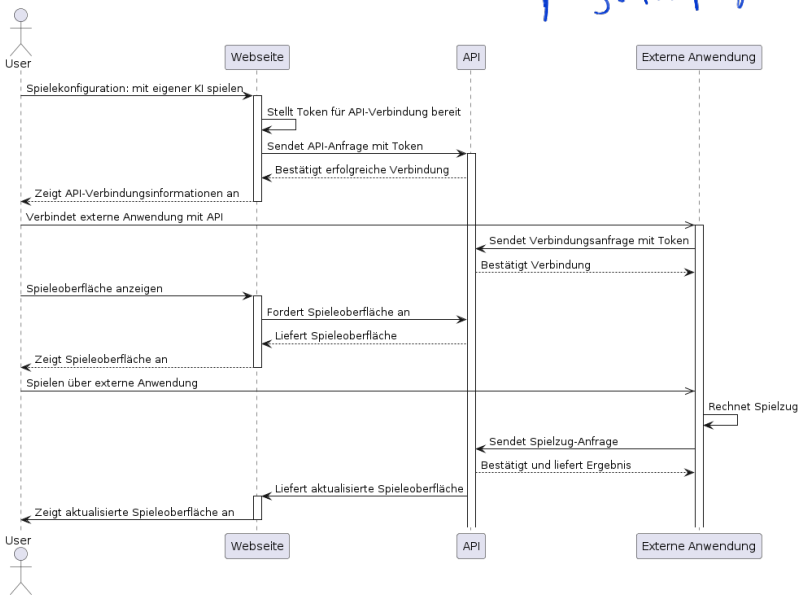
Sequenzdiag über Mockups?

6.2 Spielekonfiguration



6.3 Spieleoberfläche

6.4 ...



Schriftgr ?

1. API-Anfrage mit Unique-Key
2. Webseite zeigt Liste API-Anfragen an, Nutzer kann auswählen und Spiel zuordnen. ggf auch über API
3. Man sieht Spielzüge im Webbrowser

Wie kann man automatisiert 10, 100, ... Spiele mit der API spielen lassen

7 Umfang der Anforderungen

Komponente	Zeitabschätzung	Entwickler
Must-Haves		
- Webseite	Mehrere Issues	
- Externe Anwendung		
- Umgesetzte Spiele		
- Spielekonfiguration		
- Spielebeitritt		
- Spielerepräsentation und -navigation		
- Unterstütze Spielefunktionen		
Nice-To-Haves		
- Weitere Spiele		
- Spielekonfiguration		
- Webseitenerweiterung		
If-Time-Allows		
- Zufallsspiele		
- Webseitenerweiterung		

Tabelle 2: Umfang der Anforderung

8 Rahmenbedingungen

8.1 Zeitplan → Gantt Diagramm

8.2 Technische Anforderungen

8.3 Problemanalyse

8.4 Qualitätssicherung

8.5 Dokumentation

Die umfassende Dokumentation am Ende des Projekts ist von entscheidender Bedeutung, um den reibungslosen Betrieb der Plattform sicherzustellen und zukünftige Entwicklungen zu unterstützen. Sie dient somit als wichtige Referenz für Benutzer, Entwickler und Testteams und unterstützt eine effiziente Nutzung, Wartung und Weiterentwicklung der Plattform im gesamten Lebenszyklus. Sie umfasst verschiedene Komponenten:

- ***Benutzerhandbuch***: Das Benutzerhandbuch enthält detaillierte Anleitungen zur Nutzung der Plattform und ihrer Funktionen. Es bietet den Benutzern klare und verständliche Informationen darüber, wie sie sich anmelden, Spiele spielen, Einstellungen ändern und andere interaktive Funktionen der Plattform nutzen können. Das Ziel ist es, den Benutzern eine einfache und effiziente Erfahrung zu ermöglichen.
- ***Entwicklerdokumentation***: Die Entwicklerdokumentation enthält wichtige Informationen zur Plattformarchitektur, API-Spezifikationen und Implementierungsdetails. Sie richtet sich an Entwickler und technisches Personal, die mit der Plattform arbeiten oder sie weiterentwickeln möchten. Diese Dokumentation erleichtert die Wartung, Erweiterung und Integration neuer Funktionen in die Plattform.
- ***Weiterentwicklungsdokumentation***: Die Weiterentwicklungsdokumentation bietet Anleitungen und Richtlinien zur Erweiterung und Weiterentwicklung der Plattform. Dies umfasst beispielsweise die Integration neuer Spiele, die Verbesserung der Webseitenzugänglichkeit oder die Implementierung zusätzlicher Funktionen. Die Dokumentation soll Entwicklern klare Schritte und Best Practices bieten, um die Plattform erfolgreich zu erweitern und anzupassen.
- ***Testdokumentation***: Die Testdokumentation enthält Berichte über durchgeführte Tests und deren Ergebnisse. Sie dokumentiert den gesamten Testprozess, einschließlich der Teststrategie, Testfälle, durchgeführten Testszenarien und der daraus resultierenden Testergebnisse. Diese Dokumentation ist entscheidend, um die Qualität und Zuverlässigkeit der Plattform zu gewährleisten und etwaige Probleme oder Schwachstellen frühzeitig zu erkennen und zu beheben.

9 Abkürzungsverzeichnis

Abkürzung	Beschreibung
API	Application Programming Interface ist eine Schnittstelle, die definiert, wie verschiedene Softwarekomponenten miteinander interagieren können.
HATEOAS	Hypermedia As The Engine Of Application State ist ein Designmodell für REST-Schnittstellen
KI/AI	Künstliche Intelligenz, auch artifizielle Intelligenz, behandelt die Automatisierung
REST-API	Representational State Transfer beschreibt ein Paradigma für die Softwarearchitektur von Webservices und anderen verteilten Systemen

Tabelle 3: Abkürzungsverzeichnis

10 Anhang

1. Technische Hochschule Mittelhessen (THM), Fachbereich MNI (Medieninformatik und Medientechnik). Abgerufen von https://www.thm.de/mni/ [Zuletzt aufgerufen: 30. April 2024].
2. Kammer, Frank. Prof. Dr. Frank Kammer – Technische Hochschule Mittelhessen. Abgerufen von https://www.thm.de/mni/frank-kammer [Zuletzt aufgerufen: 30. April 2024].
3. Python Software Foundation. (2024). Python. Abgerufen von https://www.python.org/ [Zuletzt aufgerufen: 30. April 2024].
4. Docker Inc. (2024). Docker. Abgerufen von https://www.docker.com/ [Zuletzt aufgerufen: 30. April 2024].
5. Martín Abadi, et al. (2022). Abgerufen von https://www.tensorflow.org/ [Zuletzt aufgerufen: 30. April 2024].
6. Chollet, F., et al. Keras. Abgerufen von https://keras.io/ [Zuletzt aufgerufen: 30. April 2024].
7. David Silver, et al. (2018). AlphaZero: Shedding new light on chess, shogi, and Go. Abgerufen von https://deepmind.google/discover/blog/alphazero-shedding-new-light-on-chess-shogi-and-go/ [Zuletzt aufgerufen: 30. April 2024].
8. Tiangolo. FastAPI. Abgerufen von https://fastapi.tiangolo.com/ am 30. April 2024.
9. Pygame. News – Pygame v2.1.2 documentation. Abgerufen von https://www.pygame.org/news [Zuletzt aufgerufen: 30. April 2024].
10. Socket.IO. (2024). Abgerufen von https://socket.io/ [Zuletzt aufgerufen: 30. April 2024].
11. Silver, D., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Abgerufen von https://discovery.ucl.ac.uk/id/eprint/10069050/1/alphazero_preprint.pdf [Zuletzt aufgerufen: 30. April 2024].
12. Nair, S, et al. (2024). suragnair/alpha-zero-general. Abgerufen von https://github.com/suragnair/alpha-zero-general [Zuletzt aufgerufen: 30. April 2024].
13. Gerhards Spiel und Design. (n.d.). Waldmeister. Abgerufen von https://www.spielewerkstatt.eu/de/strategie-taktik/204-waldmeister.html [Zuletzt aufgerufen: 30. April 2024].
14. DSGVO-Gesetz. (2018). DSVGO Datenschutz-Grundverordnung. Abgerufen von https://dsgvo-gesetz.de/ [Zuletzt aufgerufen: 30. April 2024].
15. Graf, S. (2017). Höchster Reifegrad für REST mit HATEOAS. Heise Developer. Abgerufen von https://www.heise.de/hintergrund/Hoechster-Reifegrad-fuer-REST-mit-HATEOAS-3550392.html?seite=2 [Zuletzt aufgerufen: 30. April 2024].
16. Naeem, M., & Carbone, M. (2020). Design and implementation of an adaptive knowledge-based tutoring system using python. Abgerufen von https://search.informit.org/doi/abs/10.3316/INFORMIT.200742815970454 am 30. April 2024 [Zuletzt aufgerufen: 30. April 2024].
17. PyTorch. (n.d.). Abgerufen von https://pytorch.org/ [Zuletzt aufgerufen: 30. April 2024].