



Projektarbeit im Modul WebE, des Studiengangs

BSC Informatik

Von

Abdelrahman Abdelwahed & Chris Wüthrich

Eingereicht bei:

Lauwiner Philipp
Web Engineering
INF-P-WT002, ZH-Sa-1, HS24/25

Zürich, xx.xx.2025

Zusammenfassung

Abstract

Inhaltsverzeichnis

Abkürzungen.....	v
1 Einleitung	1
1.1 Projektidee & Vision	1
1.2 Vorstellung Projektmitglieder	2
1.2.1 Abdelrahman Abdelwahed	2
1.2.2 Chris Wüthrich	2
1.3 Termine	2
1.4 Rahmenbedingungen (Auszug Moodle).....	2
1.5 Vorgehensmodell.....	4
2 Initialisierung.....	5
2.1 Formale Angaben	5
2.2 Technische Ressourcen	5
2.3 Technologie Stack	5
2.4 Spielziel und Spielregeln	6
2.4.1 Spielziel	6
2.4.2 Spielregeln.....	6
2.5 Meilensteine	7
3 Konzept.....	9
3.1 Anforderungskatalog	9
3.1.1 Funktionale Anforderungen	10
3.1.2 Nicht funktionale Anforderungen / Qualitätsanforderungen	27
3.2 Analyse	30
3.2.1 Systemidee	30
3.2.2 Produktübersicht	30
3.2.3 Grundlegende Use-Cases.....	30
3.2.4 Client / Server Kommunikation.....	33
3.2.5 Frontend Mockups	43
4 Realisierung	45
4.1 Testplan	45
4.2 Testprotokolle.....	45
5 Einführung.....	46
5.1 Inbetriebnahme	46
5.2 Erfüllungsgrad der Anforderungen	46
5.2.1 Funktionale Anforderungen	46

6	Projektmanagement.....	51
6.1	Azure Boards.....	51
6.2	Projekt-Tagebuch	52
6.3	Projektstand	53
6.3.1	PVA 1	53
7	Verzeichnisse.....	54
7.1	Abbildungsverzeichnis	54
7.2	Tabellenverzeichnis.....	55
7.3	Quellenverzeichnis	57
7.4	Hilfsmittelverzeichnis	58
8	Anhang	59
9	Selbständigkeitserklärung	60

Abkürzungen

EFZ.....*Eidgenössisches Fähigkeitszeugnis*
HF.....*Höhere Fachschule*
ICT.....*information and communications technolog*
MMO.....*Massively Multiplayer Online Game*

1 Einleitung

Dieser Teil der Dokumentation soll einen Einstieg in das durchgeführte Projekt bieten. Dabei sollen die Projektidee und die Vision beschrieben und die einzelnen Projektmitglieder kurz vorgestellt werden. In diesem Abschnitt wird zudem erläutert, welche Termine eingehalten werden müssen und was die allgemeinen Rahmenbedingungen der Projektarbeit sind.

1.1 Projektidee & Vision

Die Idee des Projekts ist es, ein browserbasiertes Mehrspieler-Spiel zu entwickeln, das Spass mit Lernen verbindet. Im Mittelpunkt steht ein innovatives Spielkonzept, bei dem die Spieler mathematische Aufgaben lösen müssen, um sich im Spiel vorwärts zu bewegen und Punkte zu sammeln. Diese Aufgaben sind in verschiedene Schwierigkeitsgrade unterteilt und ermöglichen es den Spielern, sowohl ihre Fähigkeiten zu verbessern als auch strategisch zu spielen.

Das Ziel des Projekts ist es, eine unterhaltsame und zugleich lehrreiche Umgebung zu schaffen, in der die Spieler durch das Lösen von Aufgaben ihre Spielfigur voranbringen und gegen andere antreten können. Die Vision des Spiels ist es, sowohl Lerninhalte als auch das Spielerlebnis harmonisch zu vereinen und damit eine Plattform zu bieten, die sowohl Spass macht als auch Wissen fördert.

Besonders wichtig ist dabei die Möglichkeit, das Spiel im Mehrspieler-Modus zu erleben, was die Interaktivität und den Wettbewerbscharakter verstärkt. Durch die Integration von Highscores und Belohnungssystemen wird der Anreiz geschaffen, sich stetig zu verbessern und neue Erfolge zu erzielen. Zusätzlich trägt die Echtzeit-Chat-Funktion dazu bei, dass die Spieler miteinander kommunizieren und sich gegenseitig herausfordern können.

Das Projekt soll nicht nur ein Spiel bieten, sondern eine dynamische Lernplattform, die sowohl für den Bildungsbereich als auch für das private Spielen attraktiv ist.

1.2 Vorstellung Projektmitglieder

1.2.1 Abdelrahman Abdelwahed

1.2.2 Chris Wüthrich

Nach der Ausbildung zum Elektroinstallateur EFZ verbrachte ich weitere 5 Jahre in der Baubranche. Mitte 2016 entschied ich mich für einen Quereinstieg in die Welt der Informationstechnologien und arbeitete neu als ICT System Engineer. Begleitend zu meiner neuen Tätigkeit absolvierte ich eine Höhere Fachschule im Bereich Informatik. Während meines HF-Studiums habe ich Einblicke in die Welt der Softwareentwicklung gewonnen und mich dafür begeistert. Nach meinem Abschluss beschloss ich mich einen Schritt weiter zu gehen und meine Fähigkeiten in der Entwicklung von Applikationen durch ein Bachelor-Studium zu vertiefen. In meiner Freizeit und in schulischen Projekten setze ich mich immer wieder mit der Softwareentwicklung auseinander und verfolge meine Leidenschaft dafür.

1.3 Termine

Im Laufe des Projektes werden folgende Termine vorgegeben, die eingehalten werden müssen.

Datum	Beschreibung
07.09.2024	Kickoff
28.09.2024	1. Meilenstein
26.10.2024	2. Meilenstein
23.11.2024	3. Meilenstein
28.12.2024	4. Meilenstein
04.01.2025	Präsentation
19.01.2025	Abgabe der Projektarbeit

Tabelle 1: Termine

1.4 Rahmenbedingungen (Auszug Moodle)

Im Rahmen dieses Projekts wird eine Softwarelösung entwickelt, die entweder ein Mehrspieler-Spiel oder eine Applikation beinhaltet, bei der mehrere Benutzer gleichzeitig Inhalte bearbeiten können. Das Projekt kann entweder einzeln oder in Zweiergruppen erarbeitet werden. Die Arbeit wird abschliessend bewertet und macht 100% der Modulnote aus. Die Bewertung erfolgt über regelmässige Meilenstein-Abgaben sowie eine finale Abgabe der

Semesterarbeit. Die genauen Bewertungskriterien sind über eine Rubrik definiert und jederzeit einsehbar.

Ein zentrales Element des Projekts ist die strikte Trennung zwischen Client und Server. Der Server ist für die gesamte Anwendungslogik sowie die Berechtigungen verantwortlich und stellt die sogenannte source of truth dar. Der Client hingegen sorgt für eine benutzerfreundliche Darstellung und ermöglicht es, Änderungen in Echtzeit darzustellen. Diese Trennung gewährleistet, dass alle Benutzer dieselben Informationen sehen und die Datenintegrität jederzeit sichergestellt ist.

Um die Anforderungen des Projekts zu erfüllen, müssen folgende Bedingungen realisiert werden:

- **Responsives Frontend:** Die Benutzeroberfläche muss auf verschiedenen Endgeräten einwandfrei funktionieren und anpassbar sein.
- **Aktions-Logik über das Backend:** Alle Aktionen und Spielzüge müssen serverseitig verarbeitet und validiert werden.
- **Persistenz-Layer im Backend:** Daten, wie Benutzeraktionen und Spielzüge, müssen serverseitig persistiert werden, sodass sie auch nach Spielende nachvollziehbar bleiben.
- **Spielzugsnachverfolgung:** Es muss sichergestellt sein, dass alle Spielzüge auch nach dem Abschluss einer Spielsitzung überprüfbar sind.
- **Speicherung der Benutzeraktionen:** Jede Aktion eines Benutzers muss protokolliert und mit dem entsprechenden Benutzeraccount verknüpft werden.
- **Chat-Funktionalität:** Eine Echtzeit-Chat-Funktion ist obligatorisch und muss über WebSockets realisiert werden, um eine reibungslose Kommunikation zwischen den Benutzern zu ermöglichen.
- **Kommunikationsprotokoll:** Ein ausgereiftes Kommunikationsprotokoll muss implementiert werden, das auf die spezifischen Anforderungen des Projekts abgestimmt ist.

Optionale Features wie eine Highscore-Liste oder eine Nutzungsstatistik können zusätzlich implementiert werden, um den Funktionsumfang der Anwendung zu erweitern.

Diese Rahmenbedingungen bilden die Grundlage für die technische und funktionale Umsetzung des Projekts und gewährleisten eine strukturierte und nachhaltige Entwicklung der Software.

1.5 Vorgehensmodell

Am Kick-Off Meeting wurde von den Projektmitgliedern beschlossen, das Projekt im Phasenkonzept durchzuführen. Das bedeutet, dass das Projekt in die Phasen Initialisierung, Konzept, Realisierung und Einführung aufgeteilt wird. Damit das Projekt korrekt zum Schluss kommt, wurde eine zusätzliche Phase Abschluss hinzugefügt. Dadurch soll vermieden werden, dass das Projekt ohne Rückblick und korrekter Teamauflösung beendet wird. Die Abbildung 1 zeigt das Vorgehensmodell grob. Dabei sind zwischen den einzelnen Phasen Meilensteine geplant.

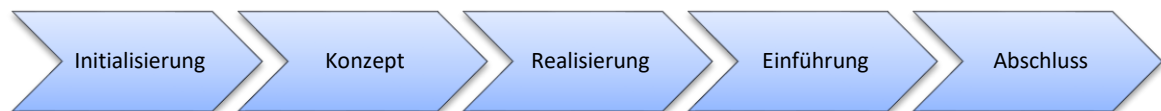


Abbildung 1: Vorgehensmodell

2 Initialisierung

Die Initialisierungsphase dient dazu, alle wichtigen Eckpunkte für das Projekt festzuhalten. Dadurch wird sichergestellt, dass sich alle Projektmitglieder auf dem gleichen Wissensstand befinden und ihnen das zu lösende Problem bewusst ist.

2.1 Formale Angaben

Modul	WebE
Art / Typ	Projektarbeit, deren Umsetzung und Dokumentation
Studiengang	Informatik Bachelor of Science SUPSI
Semester	HS 24/25
Projekt	Webbasierte Spieleentwicklung
Projektname	Adder
Projektmitglieder	Abdelrahman Abdelwahed Chris Wüthrich
Dozent	Lauwiner Philipp
Projektstart	07. September 2024
Projektabgabe	19. Januar 2024

Tabelle 2: Formale Angaben

2.2 Technische Ressourcen

Um das Projekt realisieren zu können sind folgende Ressourcen zwingend nötig:

- Für jedes Projektmitglied ist ein Computer mit Zugriff auf Teams, Word, Moodle von FFHS, GITLab und einer IDE notwendig.
- Einen Server, um die Datenbank und den Webserver zu betreiben. (Azure)
- Eine passende Domäne für flexible Zugriffe. (optional)
- Azure DevOps

2.3 Technologie Stack

Das Frontend des vorliegenden Projektes wird mit ReactJS realisiert. Dazu wird ein Gross-
teil des Programmcodes mit JavaScript Code implementiert.

Für Design und Responsive Design Komponenten wird CSS3 verwendet.

Um die Struktur der Web-Applikation zu erhalten, wird HTML5 mithilfe von React JS erstellt. Die Datenhaltung wird mit einer relationalen PostgreSQL Datenbank realisiert, worin Profil- und Spieldaten persistent gespeichert werden. Der Zugriff auf diese Datenbank, sowie die Spiellogik wird im Backend der Applikation erfolgen. Das Backend wird durch einen Express Server räsentiert, bei welchem es sich um ein Web-Framework von Node.js handelt.

Das Testen des JavaScript Codes, wird voraussichtlich mit dem Framework Jest umgesetzt, welches sich ideal für React Applikationen eignet. Die E2E Tests werden hingegen mit Cypress durchgeführt.

Folgende Liste zeigt eine Übersicht der verwendeten Technologien.

- **Frontend:** ReactJS, HTML5, CSS3, JavaScript
- **Backend:** Node.js, Express, PostgreSQL, JavaScript
- **DevOps:** Azure DevOps / Gitlab
- **Testing:** Jest, Cypress

2.4 Spielziel und Spielregeln

Adder ist ein Massively Multiplayer Online Game (MMO), das auf dem Spielprinzip von Slither.io basiert, welches im Jahr 2016 von Steve Howse veröffentlicht wurde [1]. In Adder steuern die Spieler eine Spielfigur, die einer Schlange ähnelt, durch eine endlose, offene Spielwelt. Dabei konkurrieren sie mit anderen Spielern um die grösste und mächtigste Schlange auf dem Spielfeld. Im Gegensatz zu Slither.io integriert Adder zusätzlich Mathematikaufgaben in das Gameplay, welche den Spielern die Möglichkeit bieten, durch das Lösen dieser Aufgaben zusätzliche Punkte zu sammeln. Dieses innovative Element fordert nicht nur die Geschicklichkeit der Spieler, sondern auch deren mathematisches Denken heraus und bringt eine strategische Komponente ins Spiel.

2.4.1 Spielziel

Das primäre Ziel in Adder ist es, die längste und mächtigste Schlange auf dem Spielfeld zu werden. Dies wird durch das Aufsammeln von Punkten erreicht, die in der Spielwelt verteilt sind. Die Punkte erscheinen entweder zufällig oder werden durch bestimmte Ereignisse, wie das Zerfallen anderer Schlangen oder das Lösen von Mathematikaufgaben, freigesetzt. Je grösser die Schlange wird, desto höher steigt der Spieler in der Rangliste auf, was den kompetitiven Charakter des Spiels unterstreicht.

2.4.2 Spielregeln

Bewegung und Punkteaufnahme

Die Spieler bewegen ihre Schlange über das Spielfeld und sammeln dabei Punkte ein, die das Wachstum der Schlange in Länge und Dicke fördern. Diese Punkte erscheinen zufällig auf dem Spielfeld oder werden durch das Zerfallen anderer Schlangen freigesetzt, wenn diese durch eine Kollision mit anderen Spielern oder den Spielfeldbegrenzungen „Game

Over“ gehen. Je nach Ursprung haben die Punkte unterschiedliche Werte, die den Fortschritt der Schlange massgeblich beeinflussen.

Mathematikaufgaben und Punktevergabe

Ein besonderes Feature von Adder ist die Integration von Mathematikaufgaben. Während des Spiels erscheinen regelmässig mathematische Aufgaben auf dem Bildschirm. Auf dem Spielfeld tauchen Zahlen auf, von denen eine die richtige Lösung der gestellten Aufgabe ist. Wenn der Spieler die richtige Zahl einsammelt, erhält er einen grossen Bonus in Form von zusätzlichen Punkten. Sollte der Spieler jedoch eine falsche Zahl aufheben, werden ihm Punkte abgezogen. Dieses Element erfordert von den Spielern, dass sie neben der Bewegung ihrer Schlange auch auf die Aufgaben und die richtige Lösung achten, was das Spiel dynamischer und herausfordernder macht.

Punktegewichtung

Die Punkte, die auf dem Spielfeld gesammelt werden können, haben je nach ihrem Ursprung unterschiedliche Werte:

- Zufällig gewachsene Punkte haben einen Wert von **1 bis 5 Punkten**.
- Punkte, die durch das Zerfallen einer anderen Schlange freigegeben werden, haben einen höheren Wert von **9 Punkten**.
- Das Lösen einer Mathematikaufgabe belohnt den Spieler mit einem Bonus von **50 Punkten**.
- Das Einsammeln einer falschen Zahl, die nicht der Lösung der Mathematikaufgabe entspricht, führt zu einem Abzug von **50 Punkten**.

Risiko und Belohnung

Das Punktesystem in Adder basiert auf einer Mischung aus strategischem Risiko und Belohnung. Während das Aufsammeln zufälliger Punkte ein konstantes Wachstum der Schlange ermöglicht, bietet das Lösen von Mathematikaufgaben eine Gelegenheit, schneller und effizienter Punkte zu sammeln. Die falsche Lösung kann jedoch schnell zum Verhängnis werden, da sie nicht nur den Punktestand, sondern auch die Position des Spielers in der Rangliste gefährden kann. Dieses System hält die Spannung aufrecht und fordert die Spieler dazu auf, ihre Entscheidungen sorgfältig abzuwägen.

2.5 Meilensteine

ID	Work Item Type	Title	State	Tags	Start Date	Target Date	Iteration Path
6	Epic	1. Meilenstein	Done	PVA 1	07/09/2024 02:00:00	28/09/2024 02:00:00	Adder\Sprint 1

17	Epic	2. Meilenstein	To Do	PVA 2	05/10/2024 14:00:00	26/10/2024 02:00:00	Adder\Sprint 2
18	Epic	3. Meilenstein	To Do	PVA 3	02/12/2024 13:00:00	23/11/2024 01:00:00	Adder\Sprint 3
19	Epic	4. Meilenstein	To Do	PVA 4	30/11/2024 13:00:00	28/12/2024 01:00:00	Adder\Sprint 4
20	Epic	Präsentation	To Do	PVA 5	30/09/2024 14:00:00	04/01/2025 09:45:00	Adder\Sprint 4
21	Epic	5: Meilenstein: Projekttagbabe	To Do	PVA 5	04/01/2025 13:00:00	18/01/2025 01:00:00	Adder\Sprint 5

3 Konzept

3.1 Anforderungskatalog

Nachfolgend werden die funktionalen und nichtfunktionalen Anforderungen, welche durch die Stakeholder an die Applikation gestellt werden, definiert.

Für die Formulierung der Anforderungen wird die in Abbildung 2 beschriebene Syntax verwendet. Zudem wird eine Unterteilung in Muss- und Soll-Anforderungen gemacht. Dabei gilt nachfolgende Definition für diese Unterteilung.

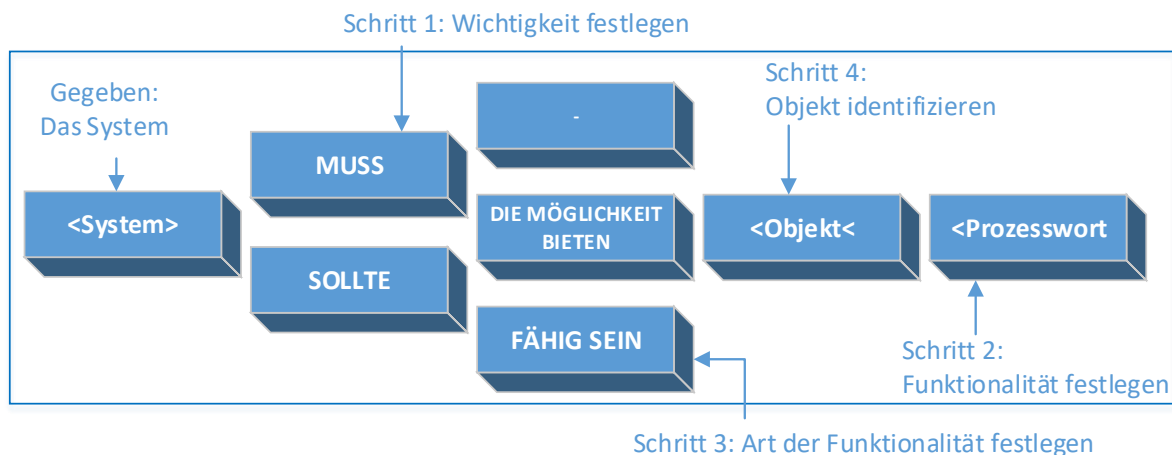


Abbildung 2: Überarbeitete Schablone zur Definition von Anforderungen [2, p. 361].

MUSS

Alle Anforderungen, die mit MUSS formuliert sind, sind verpflichtend in der Umsetzung. Die Abnahme eines Produkts kann verweigert werden, sollte das System einer MUSS-Anforderung nicht entsprechen. Dabei gilt:

0 = nicht erfüllt

1 = erfüllt

SOLLTE

Formulierungen mit SOLLTE stellen einen Wunsch eines Stakeholders dar. Sie sind nicht verpflichtend und müssen nicht erfüllt werden. Allerdings erhöht ihre Umsetzung die Zufriedenheit der Stakeholder und ihre Dokumentation verbessert die Zusammenarbeit und Kommunikation zwischen Stakeholdern und Entwicklern/Auftragnehmern. Dabei werden die einzelnen Anforderungen nach ihrer Gewichtung bewertet. Die Gewichtung sagt aus, wie wichtig diese Soll-Anforderung für das Projektteam ist. Die Gewichtung geht von 1 – 3, wobei Ganzzahl-Schritte gemacht werden. Es gilt:

1 = unwichtig

3 = sehr wichtig

3.1.1 Funktionale Anforderungen

Die funktionalen Anforderungen definieren die spezifischen Fähigkeiten und Verhaltensweisen des Systems, die notwendig sind, um die Kernaufgaben des Spiels zu erfüllen. Sie beschreiben die Interaktionen zwischen dem Nutzer und dem System sowie die Reaktionen des Systems auf bestimmte Eingaben oder Aktionen. In diesem Kapitel werden die verschiedenen funktionalen Anforderungen des Spiels Adder detailliert aufgeführt, um sicherzustellen, dass alle wesentlichen Funktionen umgesetzt werden und den Erwartungen der Nutzer entsprechen. Dazu zählen unter anderem Spielmechanismen, die Benutzerführung sowie die Interaktionen innerhalb der Spielwelt.

3.1.1.1 Muss-Anforderungen

Die Muss-Anforderungen stellen die wesentlichen und unverzichtbaren Funktionen des Systems dar. Sie definieren die Mindestanforderungen, die zwingend erfüllt sein müssen, damit das Spiel wie vorgesehen funktioniert. Diese Anforderungen garantieren das grundlegende Spielerlebnis und die Kernfunktionalitäten, ohne die das Spiel nicht ordnungsgemäss betrieben werden könnte. In diesem Abschnitt werden alle Muss-Anforderungen für das Spiel Adder beschrieben, die sicherstellen, dass die grundlegenden Spielfunktionen, wie das Starten des Spiels, die Bewegung der Spielfigur oder das Punktesystem, reibungslos ablaufen.

Name Spielstart und -beendigung	
ID	MFA-001
Ziel	Das Spiel muss die Möglichkeit bieten, ein Spiel zu starten oder zu beenden.
Ereignis	Der Spieler betätigt den Start- oder Beenden-Button.
Voraussetzung	Der Benutzer muss eingeloggt sein und sich auf dem Startbildschirm befinden.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler betätigt den Start-Button, um ein Spiel zu starten. 2. Der Spieler betätigt den Beenden-Button, um das Spiel zu beenden und den Fortschritt zu speichern.
Alternativablauf	Wenn das Spiel nicht gestartet oder beendet werden kann, wird eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Das Spiel wird gestartet oder beendet. Nach dem Beenden wird eine Zusammenfassung der erreichten Punkte angezeigt.
Nachbedingung Fehler	Eine Fehlermeldung weist den Spieler auf ein Verbindungsproblem oder einen Systemfehler hin.
Klassifizierung	Funktional, MUSS
Aufwand	Gering

Tabelle 3: MFA-001 - Spielstart und -beendigung

KPI-Nr.	Beschreibung
1.0	Das Spiel muss die Möglichkeit bieten, ein Spiel zu starten oder zu beenden.
1.1	Das Spiel muss die Möglichkeit bieten, durch einen Start-Button ein Spiel zu initialisieren.
1.2	Das Spiel muss die Möglichkeit bieten, durch einen Beenden-Button das Spiel ordnungsgemäss zu beenden und den Fortschritt zu speichern.
1.3	Das Spiel muss die Möglichkeit bieten, nach Spielende eine Zusammenfassung der erreichten Punkte anzuzeigen.

Tabelle 4: MFA-001 KPIs

Name Spieler-Registrierung	
ID	MFA-002
Ziel	Das Spiel muss die Möglichkeit bieten, dass sich neue Spieler registrieren können.
Ereignis	Der Spieler betätigt den Registrieren-Button.
Voraussetzung	Der Spieler muss auf der Startseite sein.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler gibt eine valide E-Mail-Adresse ein. 2. Der Spieler gibt einen einzigartigen Benutzernamen ein. 3. Der Spieler gibt ein Passwort ein und bestätigt es durch wiederholte Eingabe.
Alternativablauf	Wenn die eingegebene E-Mail Adresse, der Benutzernamen oder das Passwort ungültig sind, wird eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Der Spieler wird erfolgreich registriert, direkt angemeldet und zum Hauptmenü weitergeleitet.
Nachbedingung Fehler	Eine Fehlermeldung fordert den Spieler auf, die E-Mail Adresse, den Benutzernamen oder das Passwort zu überprüfen und den Vorgang erneut zu versuchen.
Klassifizierung	Funktional, MUSS
Aufwand	Mittel

Tabelle 5: MFA-002 - Spieler-Authentifizierung

KPI-Nr.	Beschreibung
2.0	Das Spiel muss die Möglichkeit bieten, dass sich neue Spieler registrieren können.
2.1	Das Spiel muss die Möglichkeit bieten, dass sich neue Spieler mit einem Benutzernamen, einer E-Mail Adresse und einem Passwort registrieren können.
2.2	Das Spiel muss die Möglichkeit bieten, dass ein Spieler sich mit einer validen und einzigartigen E-Mail-Adresse registriert.
2.3	Der registrierende Spieler muss beim Registrierungsprozess aufgefordert werden, sein Passwort zweimal identisch einzugeben.
2.4	Der registrierende Spieler muss beim Registrierungsprozess aufgefordert werden, einen einzigartigen Benutzernamen einzugeben.

Tabelle 6: MFA-002 KPIs

Name	Mathematikaufgaben
ID	MFA-003
Ziel	Das Spiel muss mathematische Fragen generieren und sie den Spielern präsentieren.
Ereignis	Mathematikaufgaben werden im Spielverlauf angezeigt.
Voraussetzung	Der Spieler befindet sich in einer Spielsituation, in der eine Mathematikaufgabe gelöst werden muss.
Standardablauf	<ol style="list-style-type: none"> 1. Das Spiel zeigt eine zufällige Frage aus einem Pool an. 2. Der Spieler löst die Aufgabe durch einsammeln der richtigen Antwort mit seiner Spielfigur und erhält Punkte.
Alternativablauf	Der Spieler löst die Mathematikaufgabe falsch, indem er ein falsches Resultat einsammelt.
Nachbedingung Erfolg	Der Spieler erhält bei richtiger Lösung zusätzliche Punkte.
Nachbedingung Fehler	Der Spieler verliert Punkte bei falscher Antwort.
Klassifizierung	Funktional, MUSS
Aufwand	Hoch

Tabelle 7: MFA-003 – Mathematikaufgaben

KPI-Nr.	Beschreibung
3.0	Das Spiel muss verschiedene Arten von mathematischen Aufgaben beinhalten, welche dem Spieler zur Auswahl angeboten werden.
3.1	Das Spiel muss die Möglichkeit bieten, mindestens drei verschiedene Arten von Mathematikfragen auszuwählen.
3.2	Das Spiel muss die Möglichkeit bieten, jede Frage zufällig aus einem Pool von mindestens 20 Fragen pro Art auszuwählen.
3.3	Das Spiel muss die Möglichkeit bieten, die Schwierigkeitsgrade der Fragen dynamisch basierend auf der Leistung des Spielers anzupassen.

Tabelle 8: MFA-003 KPIs

Name	Score-Based Belohnungssystem
ID	MFA-004
Ziel	Das Spiel muss ein Punktesystem enthalten, das die Spieler für das Einsammeln von Bubbles auf dem Spielfeld und das Lösen von Mathematikaufgaben belohnt.
Ereignis	Der Spieler sammelt Punkte durch das Einsammeln von Bubbles oder dem Lösen von Mathematikaufgaben im Spiel.
Voraussetzung	Der Spieler befindet sich im laufenden Spiel.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler sammelt 1-5 Punkte durch das Einsammeln von Bubbles. 2. Der Spieler erhält 50 Punkte durch das Lösen von Mathematikaufgaben.
Alternativablauf	Wenn der Spieler keine Punkte sammelt oder die Aufgabe falsch löst, wird der Punktestand entsprechend nicht geändert.
Nachbedingung Erfolg	Die Punkte werden korrekt auf dem Bildschirm angezeigt und am Ende des Spiels zusammengefasst.
Nachbedingung Fehler	Der alte Punktestand wird unverändert angezeigt.
Klassifizierung	Funktional, MUSS
Aufwand	Mittel

Tabelle 9: MFA-004 - Score-Based Belohnungssystem

KPI-Nr.	Beschreibung
4.0	Das Spiel muss ein Punktesystem enthalten, das die Spieler für das Einsammeln von Bubbles auf dem Spielfeld und das Lösen von Mathematikaufgaben belohnt.
4.1	Das Spiel muss die Möglichkeit bieten, für jeden eingesammelten Bubble Punkte zu vergeben.
4.2	Das Spiel muss die Möglichkeit bieten, für jede korrekt beantwortete Frage Punkte zu vergeben.
4.3	Das Spiel muss die Möglichkeit bieten, dass bei falsch beantworteten Fragen Punkte abgezogen werden.
4.4	Das Spiel muss die Möglichkeit bieten, die Punkte am Ende jedes Levels und Spiels zusammengefasst zu präsentieren.

Tabelle 10: MFA-004 KPIs

Name	Spielerbewegung
ID	MFA-005
Ziel	Das Spiel muss die Möglichkeit bieten, die Spielfigur (Schlange) durch das Spiel zu steuern.
Ereignis	Der Spieler steuert die Spielfigur mit seiner Maus am PC oder einem Touchpad auf einem Smartphone.
Voraussetzung	Der Spieler befindet sich im laufenden Spiel und die Figur ist steuerbar.
1. Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler bewegt die Figur mit den Pfeiltasten oder der Maus. 2. Die Geschwindigkeit der Spielfigur kann manuell erhöht werden, wodurch Punkte verloren gehen.
2. Standardablauf	<ol style="list-style-type: none"> 3. Der Spieler steuert die Spielfigur über seinen eigenen Schwanz.
Alternativablauf	<ol style="list-style-type: none"> 3. Wenn die Figur am Spielfeldrand oder mit einem Gegner kollidiert, wird eine Kollision angezeigt und das Spiel wird für den Spieler beendet.
1. Nachbedingung Erfolg	Der Spieler steuert erfolgreich die Spielfigur, sammelt Punkte und erreicht Ziele.
2. Nachbedingung Erfolg	Der Spieler steuert erfolgreich die Spielfigur über seinen eigenen Schwanz. Kollisionen mit sich selbst gibt es nicht.
Nachbedingung Fehler	Der Spieler ist Game Over und das Spiel wird beendet. Dem Spieler wird sein aktueller Highscore der vergangenen Runde angezeigt.
Klassifizierung	Funktional, MUSS
Aufwand	Hoch

Tabelle 11: MFA-005 – Spielerbewegung

KPI-Nr.	Beschreibung
5.0	Das Spiel muss die Möglichkeit bieten, die Spielfigur (Schlange) durch das Spiel zu steuern.
5.1	Das Spiel muss die Möglichkeit bieten, dass der Spieler die Spielfigur mit den Pfeiltasten oder der Maus steuert.
5.2	Das Spiel muss die Möglichkeit bieten, dass die Geschwindigkeit der Spielfigur manuell erhöht wird, wodurch Punkte verloren gehen.

5.3	Das Spiel muss sicherstellen, dass bei einer Kollision mit einem anderen Spieler oder dem Spielfeldrand das Spiel für den kollidierenden Spieler beendet wird.
------------	--

Tabelle 12: MFA-005 KPIs

Name	Mehrspieler-Funktionalität
ID	MFA-006
Ziel	Das Spiel muss eine Mehrspieler-Funktion unterstützen, die es mehreren Spielern ermöglicht, gleichzeitig zu spielen.
Ereignis	Mehrere Spieler treten gleichzeitig in den gleichen Spielraum ein.
Voraussetzung	Mindestens zwei Spieler müssen eingeloggt sein und sich im gleichen Spielraum befinden.
Standardablauf	<ol style="list-style-type: none"> 1. Die Spieler treten in denselben Spielraum ein. 2. Die Spieler interagieren durch das Sammeln von Punkten und das Bewegen der Schlangen.
Alternativablauf	Der Spieler befindet sich alleine in einem Spielraum.
Nachbedingung Erfolg	Der Mehrspielermodus funktioniert einwandfrei und alle Spieler sind sichtbar.
Nachbedingung Fehler	-
Klassifizierung	Funktional, MUSS
Aufwand	Hoch

Tabelle 13: MFA-006 - Mehrspieler-Funktionalität

KPI-Nr.	Beschreibung
6.0	Das Spiel muss eine Mehrspieler-Funktion unterstützen, die es mehreren Spielern ermöglicht, gleichzeitig zu spielen.
6.1	Das Spiel muss die Möglichkeit bieten, dass mindestens zwei Spieler gleichzeitig im gleichen Spielraum spielen.
6.2	Das Spiel muss die Möglichkeit bieten, dass Spieler den Fortschritt und die Position des anderen Spielers in Echtzeit sehen können.
6.3	Das Spiel muss die Möglichkeit bieten, ein Kommunikationssystem wie einen Chat zur Interaktion zwischen den Spielern zu nutzen.

Tabelle 14: MFA-006 KPIs

Name	Levelauswahl
ID	MFA-007
Ziel	Das Spiel muss eine Funktion zur Auswahl verschiedener Spieltypen bereitstellen, die den Spielern ermöglicht, zwischen unterschiedlichen mathematischen Herausforderungen zu wählen.
Ereignis	Der Spieler wählt ein Level vor Spielbeginn.
Voraussetzung	Der Spieler befindet sich auf dem Levelauswahlbildschirm.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler wählt einen Spieltypen aus. 2. Falls noch keine Instanz des ausgewählten Spieltyps existiert, wird eine neue Instanz generiert. 2. Falls bereits eine Instanz des ausgewählten Spieltyps existiert, wird keine neue Instanz generiert. 3. Der Spieler betritt die Instanz des gewählten Spieltyps.
Alternativablauf	Wenn keine Spieltype ausgewählt wird, wird das Spiel mit dem vorselektionierten Spieltype gespielt.
Nachbedingung Erfolg	Das Spiel startet mit dem vom Spieler gewählten Spieltype.
Nachbedingung Fehler	Eine Fehlermeldung informiert den Spieler, wenn das Level nicht geändert werden kann.
Klassifizierung	Funktional, MUSS
Aufwand	Gering

Tabelle 15: MFA-007 – Levelauswahl

KPI-Nr.	Beschreibung
7.0	Das Spiel muss eine Funktion zur Auswahl verschiedener Spieltypen bereitstellen, die den Spielern ermöglicht, zwischen unterschiedlichen mathematischen Herausforderungen zu wählen.
7.1	Das Spiel muss die Möglichkeit bieten, dass der Spieler aus mindestens drei verschiedenen Spieltypen (z.B. Addition, Subtraktion und Multiplikation) auswählen kann.
7.2	Das Spiel muss die Möglichkeit bieten, dass der Spieler das Level vor Spielbeginn ändern kann.

Tabelle 16: MFA-007 KPIs

Name	Rückmeldungssystem
ID	MFA-008
Ziel	Das Spiel muss visuelle und auditive Rückmeldungen geben, wenn ein Spieler Punkte gewinnt oder verliert.
Ereignis	Der Spieler erhält Rückmeldungen nach jedem Erhalt oder Verlust von Punkten.
Voraussetzung	Der Spieler befindet sich im laufenden Spiel.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler gewinnt Punkte. 2. Das Spiel zeigt eine visuelle Rückmeldung (z.B. grünes Blinken) an. 3. Eine akustische Rückmeldung wird abgespielt.
Alternativablauf	<ol style="list-style-type: none"> 1. Der Spieler verliert Punkte. 2. Das Spiel zeigt eine visuelle Rückmeldung (z.B. rotes Blinken) an. 3. Eine akustische Rückmeldung wird abgespielt.
Nachbedingung Erfolg	Der Spieler erhält eine klare Rückmeldung und der Punktestand wird angepasst.
Nachbedingung Fehler	Eine Fehlermeldung weist den Spieler auf ein Anzeigeproblem bei der Rückmeldung hin.
Klassifizierung	Funktional, MUSS
Aufwand	Mittel

Tabelle 17: MFA-008 – Rückmeldungssystem

KPI-Nr.	Beschreibung
8.0	Das Spiel muss visuelle und auditive Rückmeldungen geben, wenn ein Spieler Punkte gewinnt oder verliert.
8.1	Das Spiel muss die Möglichkeit bieten, eine visuelle Rückmeldung zu geben, wenn eine Aufgabe richtig beantwortet wurde (z.B. Schlange blinkt grün und wird grösser).
8.2	Das Spiel muss die Möglichkeit bieten, eine visuelle Rückmeldung zu geben, wenn eine Aufgabe falsch beantwortet wurde (z.B. Schlange blinkt rot und wird kleiner).
8.3	Das Spiel muss die Möglichkeit bieten, bei jeder Antwort eine kurze akustische Rückmeldung zu geben, die optional deaktivierbar ist.

Tabelle 18: MFA-008 KPIs

Name	Leaderboard
ID	MFA-009
Ziel	Das Spiel muss eine Rangliste enthalten, welche die besten Spieler basierend auf ihrem Highscore anzeigt.
Ereignis	Das Leaderboard wird nach Abschluss eines Spiels angezeigt.
Voraussetzung	Der Spieler hat das Spiel beendet und die Punktzahl wurde berechnet.
Standardablauf	<ol style="list-style-type: none"> 1. Das Spiel zeigt die Top 10 Spieler basierend auf den erzielten Punkten an. 2. Der eigene Spieler wird mit Rangnummer im Leaderboard unter den Top 10 oder direkt unterhalb angezeigt, abhängig vom Rang des Spielers.
Alternativablauf	Wenn keine Punkte oder Spieler vorhanden sind, wird das Leaderboard leer angezeigt.
Nachbedingung Erfolg	Das Leaderboard zeigt korrekt die besten Spieler basierend auf der Punktzahl an.
Nachbedingung Fehler	Eine Fehlermeldung informiert den Spieler über einen Fehler beim Laden der Rangliste.
Klassifizierung	Funktional, MUSS
Aufwand	Gering

Tabelle 19: MFA-009 – Leaderboard

KPI-Nr.	Beschreibung
9.0	Das Spiel muss eine Rangliste enthalten, welche die besten Spieler basierend auf ihrem Highscore anzeigt.
9.1	Das Spiel muss die Möglichkeit bieten, die Top 10 Spieler basierend auf ihrem Highscore im Leaderboard anzuzeigen.
9.2	Das Spiel muss die Möglichkeit bieten, das Leaderboard nach wöchentlichen, monatlichen und Gesamt-Ergebnissen zu filtern.
9.3	Das Spiel muss die Möglichkeit bieten, die Spieler nach Benutzernamen und Punktzahl zu sortieren.
9.4	Das Spiel muss die Möglichkeit bieten, im Leaderboard anhand einer Volltextsuche einen Benutzernamen zu suchen.

Tabelle 20: MFA-009 KPIs

Name	Spielerfortschritt
ID	MFA-010
Ziel	Das Spiel muss den Fortschritt und die Errungenschaften des Spielers speichern und beim erneuten Einloggen wiederherstellen können.
Ereignis	Der Spieler kehrt nach dem Einloggen ins Spiel zurück.
Voraussetzung	Der Spieler ist eingeloggt und hat bereits Fortschritte im Spiel erzielt.
Standardablauf	<ol style="list-style-type: none"> 1. Der Fortschritt des Spielers wird automatisch nach jedem abgeschlossenen Spiel gespeichert. 2. Der Spieler kann seinen Fortschritt (Errungenschaften, Highscores) im Benutzerprofil einsehen.
Alternativablauf	Wenn der Fortschritt nicht gespeichert wurde, wird der Spieler darüber informiert und aufgefordert, erneut zu spielen.
Nachbedingung Erfolg	Der Spieler kehrt erfolgreich zu seinem vorherigen Spielstand zurück.
Nachbedingung Fehler	Eine Fehlermeldung informiert den Spieler über Speicherprobleme oder Verbindungsfehler beim Speichern des Fortschritts.
Klassifizierung	Funktional, MUSS
Aufwand	Mittel

Tabelle 21: MFA-010 – Spielerfortschritt

KPI-Nr.	Beschreibung
10.0	Das Spiel muss den Fortschritt des Spielers speichern und beim erneuten Einloggen wiederherstellen können.
10.1	Das Spiel muss die Möglichkeit bieten, dass der Fortschritt des Spielers automatisch nach jedem abgeschlossenen Spiel gespeichert wird.
10.2	Das Spiel muss die Möglichkeit bieten, dass Spieler ihren Fortschritt über eine Profilseite einsehen können, welche Statistiken wie gespielte Level, Gesamtpunkte und erreichte Erfolge/Badges anzeigt.

Tabelle 22: MFA-010 KPIs

Name Spieler-Anmeldung	
ID	MFA-011
Ziel	Das Spiel muss die Möglichkeit bieten, dass sich registrierte Spieler anmelden können.
Ereignis	Der Spieler möchte sich im Spiel einloggen.
Voraussetzung	Der Spieler muss sich im Anmeldebereich befinden.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler gibt seine korrekte und valide E-Mail-Adresse ein. 2. Der Spieler gibt sein korrektes Passwort ein. 3. Der Spieler drückt auf den Anmelden-Button
Alternativablauf	Wenn die eingegebene E-Mail Adresse oder das Passwort ungültig sind, wird eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Der Spieler wird erfolgreich angemeldet und zum Hauptmenü weitergeleitet.
Nachbedingung Fehler	Eine Fehlermeldung fordert den Spieler auf, die E-Mail Adresse, oder das Passwort zu überprüfen und den Vorgang erneut zu versuchen.
Klassifizierung	Funktional, MUSS
Aufwand	Mittel

Tabelle 23: MFA-002 - Spieler-Authentifizierung

KPI-Nr.	Beschreibung
2.0	Das Spiel muss die Möglichkeit bieten, dass sich registrierte Benutzer mit einer E-Mail Adresse und einem Passwort anmelden können.

Tabelle 24: MFA-010 KPIs

3.1.1.2 Soll-Anforderungen

Name	Passwort Reset
ID	SFA-001
Ziel	Das Spiel sollte die Möglichkeit bieten, dass ein Spieler sein Passwort über eine Passwort-vergessen-Funktion zurücksetzen kann.
Ereignis	Der Spieler betätigt den Passwort-vergessen-Button.
Voraussetzung	Der Spieler befindet sich auf der Login-Seite.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler gibt seine E-Mail-Adresse ein, um einen Reset-Link zu erhalten. 2. Der Spieler klickt auf den Link in der E-Mail und setzt sein Passwort zurück.
Alternativablauf	Wenn die E-Mail-Adresse nicht validiert werden kann, wird eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Der Spieler kann erfolgreich sein Passwort zurücksetzen und sich mit dem neuen Passwort einloggen.
Nachbedingung Fehler	Eine Fehlermeldung weist den Spieler auf Verbindungsprobleme oder einen Fehler bei der E-Mail-Adresse hin.
Gewichtung	2
Klassifizierung	Funktional, SOLL
Aufwand	Mittel

Tabelle 25: SFA-001 - Passwort Reset

Name	Benutzerverwaltung
ID	SFA-002
Ziel	Das Spiel sollte die Möglichkeit bieten, dass ein Spieler seine Benutzerinformationen (Passwort, Benutzername und E-Mail Adresse) ändern kann.
Ereignis	Der Spieler navigiert zur Profilseite und aktualisiert seine Informationen.
Voraussetzung	Der Spieler ist eingeloggt und befindet sich auf seiner Profilseite.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler gibt die gewünschten Änderungen ein. 2. Das Spiel speichert die geänderten Benutzerinformationen. 3. Eine Erfolgsmeldung wird angezeigt.
Alternativablauf	Wenn die eingegebenen Daten nicht valide sind, wird eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Die Benutzerinformationen werden erfolgreich aktualisiert und der Spieler erhält eine Bestätigung.
Nachbedingung Fehler	Eine Fehlermeldung informiert den Spieler bei einem Fehler in der Dateneingabe oder bei Verbindungsproblemen.
Gewichtung	3
Klassifizierung	Funktional, SOLL
Aufwand	Hoch

Tabelle 26: SFA-002 – Benutzerverwaltung

Name	Achievement-Based Belohnungssystem
ID	SFA-003
Ziel	Das Spiel sollte eine Funktionalität beinhalten, welche es Spielern erlaubt spezielle Ziele abzuschliessen und dadurch Badges zu erhalten.
Ereignis	Der Spieler schliesst eine besondere Aufgabe ab.
Voraussetzung	Der Spieler befindet sich im laufenden Spiel und erfüllt die Bedingung für ein Achievement.
Standardablauf	<ol style="list-style-type: none"> 1. Das Spiel überprüft, ob der Spieler die Bedingung für ein Achievement erfüllt hat. 2. Der Spieler erhält das Badge und eine Benachrichtigung.
Alternativablauf	Wenn der Fortschritt für das Achievement nicht korrekt aufgezeichnet wird, erhält der Spieler keine Benachrichtigung.
Nachbedingung Erfolg	Der Spieler erhält das Badge erfolgreich und es wird im Profil angezeigt.
Nachbedingung Fehler	-
Gewichtung	3
Klassifizierung	Funktional, SOLL
Aufwand	Hoch

Tabelle 27 SFA-003 - Achievement-Based Belohnungssystem:

Name	Werbung
ID	SFA-004
Ziel	Das Spiel sollte einen Bereich enthalten, an welchem Werbung eingespielt werden kann.
Ereignis	Die Werbung wird während des Spiels oder auf dem Startbildschirm angezeigt.
Voraussetzung	Der Spieler befindet sich auf der Startseite oder in einer Spielpause.
Standardablauf	<ol style="list-style-type: none"> 1. Die Werbung wird auf dem Bildschirm angezeigt. 2. Der Spieler kann das Spiel fortsetzen, nachdem die Werbung angezeigt wurde.
Alternativablauf	Wenn die Werbung nicht geladen werden kann, wird dem Spieler eine Fehlermeldung angezeigt.
Nachbedingung Erfolg	Die Werbung wird erfolgreich geladen und dem Spieler angezeigt.
Nachbedingung Fehler	Eine Fehlermeldung weist den Spieler auf ein Problem beim Laden der Werbung hin.
Gewichtung	1
Klassifizierung	Funktional, SOLL
Aufwand	Mittel

Tabelle 28: SFA-004 – Werbung

Name	Werbung deaktivieren (Abonnement)
ID	SFA-004.1
Ziel	Das Spiel sollte die Möglichkeit bieten, dass Benutzer ein Abonnement abschliessen können, um die Werbung nicht angezeigt zu bekommen.
Ereignis	Der Spieler schliesst ein Abonnement ab, um Werbung zu deaktivieren.
Voraussetzung	Der Spieler befindet sich auf der Abonnement-Seite.
Standardablauf	<ol style="list-style-type: none"> 1. Der Spieler schliesst das Abonnement erfolgreich ab. 2. Das Spiel stellt sicher, dass der Spieler keine Werbung mehr angezeigt bekommt.
Alternativablauf	Wenn das Abonnement nicht abgeschlossen werden kann, erhält der Spieler eine Fehlermeldung.
Nachbedingung Erfolg	Der Spieler schliesst das Abonnement erfolgreich ab und die Werbung wird entfernt.
Nachbedingung Fehler	Eine Fehlermeldung weist den Spieler auf Probleme bei der Abonnement-Abwicklung hin.
Gewichtung	1
Klassifizierung	Funktional, SOLL
Aufwand	Mittel

Tabelle 29: SFA-004.1 - Werbung deaktivieren (Abonnement)

3.1.2 Nicht funktionale Anforderungen / Qualitätsanforderungen

Die nicht funktionalen Anforderungen definieren Wünsche der Stakeholder, sowie Rahmenbedingungen welche unabhängig der Funktionsabdeckung erfüllt werden müssen oder sollen. Dies umfasst architektonische Grundprinzipien sowie Bedingungen an Zeit und Kosten. Auch diese Anforderungen werden wieder anhand der vordefinierten Syntaxe in Muss und Soll unterteilt.

3.1.2.1 Muss-Anforderungen

Folgende nicht funktionalen Anforderungen müssen für einen erfolgreichen Projektabschluss zwingend erfüllt werden.

Kategorie	Nr.	Unter-Nr.	Beschreibung	Erfüllt
Verfügbarkeit	MNFA-001	1.0	Das Spiel muss lediglich spielbar sein, wenn eine aktive Netzwerkverbindung zum Server besteht.	0/1
Benutzerfreundlichkeit	MNFA-002	2.0	Die Funktionen MFA 9.0 und 10.0 müssen von der Hauptansicht mit maximal 1 Klick erreicht werden.	0/1
		2.1	Die Funktionen MFA 1.2 muss während dem Spiel mit maximal 1 Klick erreicht werden.	0/1
Performance	MNFA-003	3.0	Die Suchfunktion im Leaderboard muss innerhalb 1 Sekunde, die ersten Ergebnisse darstellen, wenn eine Internetverbindung von mindestens 40 Mbit/s besteht.	0/1
		3.1	Das Leaderboard muss mindestens 100 Einträge enthalten können.	0/1

		3.2	Das Spiel muss bei einer Netzwerkverbindung von 20Mbit/s einwandfrei funktionieren.	
Systemumgebung	MNFA-004	4.0	Das Spiel muss im Browser «Google Chrome Version 129.0.6668.70» ausgeführt werden können.	0/1
		4.1	Das Spiel muss funktionieren, ohne dass der Benutzer zusätzliche Software oder Extensions installieren muss.	0/1
		4.2	Die Aktions-Logik des Spiels muss im Backend implementiert sein.	0/1
		4.3	Das Backend des Spiels muss einen Persistenz-Layer aufweisen.	0/1
		4.4	Das Spiel muss ein ausgereiftes, auf das Projekt abgestimmtes Kommunikationsprotokoll verwenden.	
Sprache	MNFA-005	5.0	Die Benutzeroberfläche muss auf Deutsch verfügbar sein.	0/1
Datenunabhängigkeit	MNFA-006	6.0	Das Anwendungsprogramm und die Datenhaltung müssen unabhängig voneinander betrieben werden können.	0/1
Speicherzugriff	MNFA-007	7.0	Die Zeit des Datenbank-Zugriffs, vom Backend zum Persistenz-Layer aus gesehen, muss weniger als 1s betragen, wenn eine Internetverbindung von mindestens 40 Mbit/s besteht.	0/1

		7.1	Mehrere Benutzer müssen gleichzeitig auf die Datenbank zugreifen können.	0/1
		7.2	Der Aufbau der Datenbank muss vor dem Frontend verborgen bleiben. (Information Hiding)	0/1

Tabelle 30: Nicht funktionale Muss-Anforderungen

3.1.2.2 Soll-Anforderungen

Des Weiteren soll das Spiel, wenn möglich, folgende nicht funktionalen Anforderungen erfüllen.

Kategorie	Nr.	Unter-Nr.	Beschreibung	Gewichtung
Verfügbarkeit	SNFA-001	1.0	Das Spiel sollte ohne aktiver Netzwerkverbindung zum Server genutzt werden können (Offline-Modus).	2
Benutzerfreundlichkeit	SNFA-002	2.0	Das Spiel sollte die Möglichkeit bieten, dass Spieler die visuelle Benutzeroberfläche anpassen können (Light- & Dark-Mode).	1
		2.1	Die Buttons für die Interaktion mit dem Spiel, sollten so gewählt sein, dass keine schriftliche Erklärung ihrer Funktion benötigt wird.	2
		2.2	Das Spiel sollte bei der Nutzung auf Smartphones die Möglichkeit bieten, das automatische drehen des Bildschirms zu blockieren.	1
Performance	SNFA-003	3.0	Das Spiel sollte bei einer Netzwerkverbindung von 5Mbit/s einwandfrei funktionieren.	3

Systemumgebung	SNFA-004	4.0	Das Spiel sollte im Browser «Safari Version 18.1» ausgeführt werden können.	3
		4.1	Das Spiel sollte im Browser «Edge Version 128.0.2739.42» ausgeführt werden können.	3
		4.2	Die Benutzeroberfläche sollte auf Englisch verfügbar sein.	2
Sicherheit	SNFA-005	5.0	Die gespeicherten Daten sollten verschlüsselt abgelegt werden.	3
Speicherzugriff	SNFA-006	6.0	Die Zeit des Datenbank-Zugriffs, vom Backend zum Persistenz-Layer aus gesehen, sollte weniger als 10ms betragen, wenn eine Internetverbindung von mindestens 40 Mbit/s besteht.	1

Tabelle 31: Nicht funktionale Soll-Anforderungen

3.2 Analyse

3.2.1 Systemidee

3.2.2 Produktübersicht

3.2.3 Grundlegende Use-Cases

Dem Benutzer sollen einige Grundlegenden Funktionen zur Verfügung stehen. Diese Funktionen wurden als Use-Case Diagramme visualisiert und anschliessend beschrieben. Es wird in erster Linie zwischen registrierten und nicht registrierten Benutzern, da die Nutzung des Spiels ohne gültiges Benutzerkonto nicht möglich sein soll

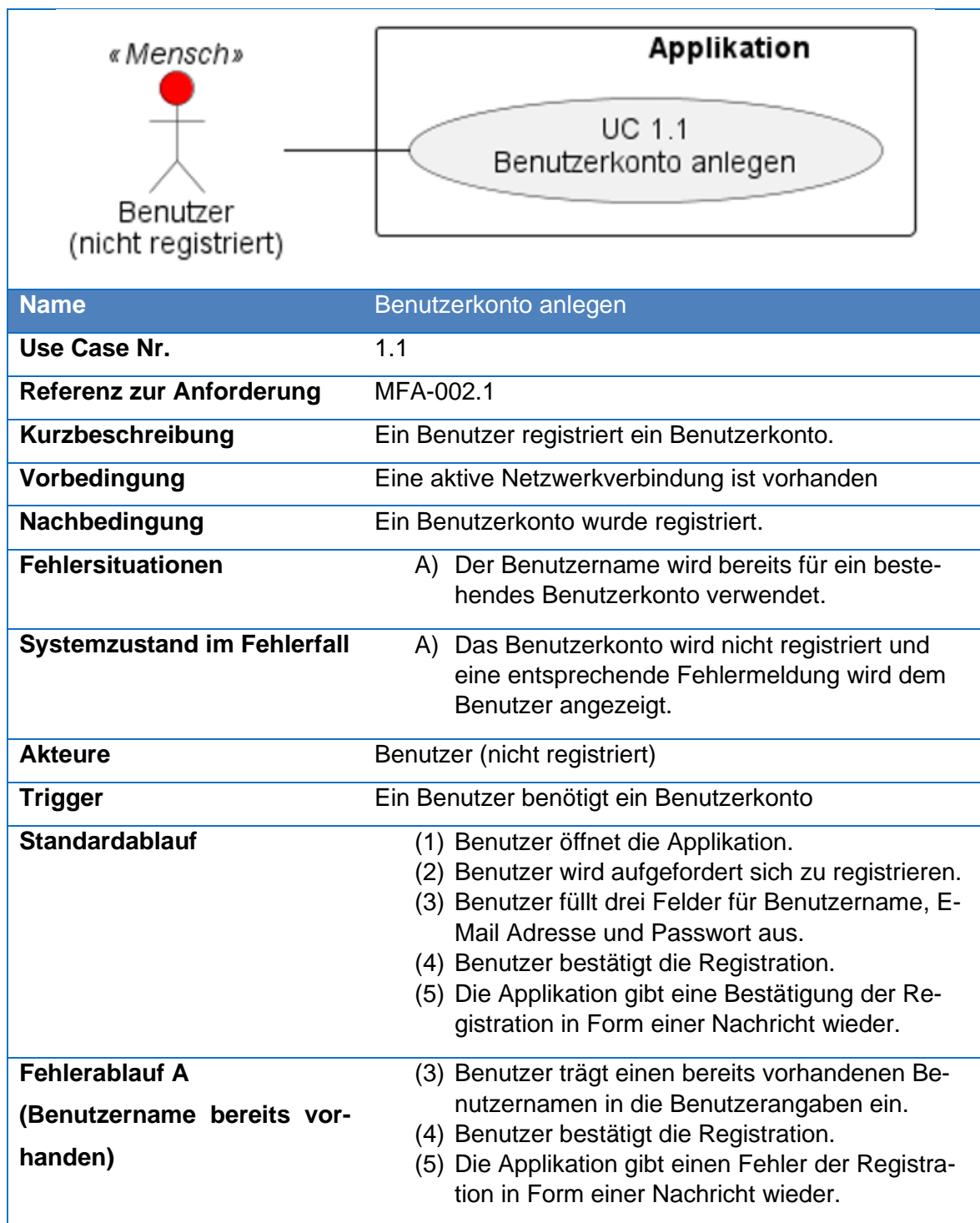


Tabelle 32: Use-Case 1.1 - Benutzerkonto anlegen

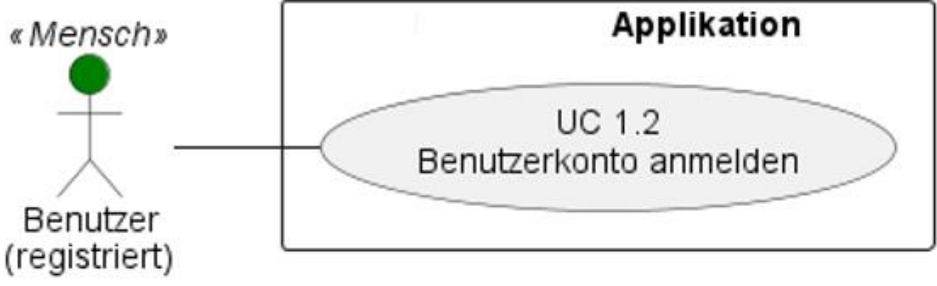
	
Name	Benutzerkonto anmelden
Use Case Nr.	1.2
Referenz zur Anforderung	MFA-002.5
Kurzbeschreibung	Ein Benutzer meldet sich an.
Vorbedingung	1. Eine aktive Netzwerkverbindung ist vorhanden 2. Der Benutzer verfügt über ein Benutzerkonto.
Nachbedingung	Der Benutzer ist angemeldet
Fehlersituationen	A) Der Benutzername existiert nicht. B) Das Passwort des Benutzers ist falsch.
Systemzustand im Fehlerfall	A) Der Benutzer wird nicht angemeldet und eine entsprechende Fehlermeldung wird auf der Applikation angezeigt. B) Der Benutzer wird nicht angemeldet und eine entsprechende Fehlermeldung wird auf der Applikation angezeigt.
Akteure	Benutzer (registriert)
Trigger	Ein Benutzer möchte sich anmelden.
Standardablauf	(1) Benutzer öffnet die Applikation. (2) Benutzer gibt seinen Benutzernamen und sein Passwort ein. (3) Benutzer bestätigt Anmeldung. (4) Benutzer wird erfolgreich angemeldet und das Dashboard der Applikation erscheint.
Fehlerablauf A (Benutzername existiert nicht)	(4) Die Applikation zeigt eine entsprechende Fehlermeldung an. (5) Benutzer wurde nicht angemeldet.
Fehlerablauf B (Falsches Passwort wurde eingegeben)	(4) Die Applikation zeigt eine entsprechende Fehlermeldung an. (5) Benutzer wurde nicht angemeldet.

Tabelle 33: Use-Case 1.2 – Benutzerkonto Anmelden

3.2.4 Client / Server Kommunikation

Die Kommunikation zwischen Client und Server findet über WebSockets statt. Es werden die Spieler- und Spieldaten damit übertragen wie zum Beispiel Spieleraktionen und Bewegungen, Spielerzustandsaktualisierungen, das Beitreten und Verlassen von Spieler und die Kollisionen und Ereignisse. Die Spieleraktionen ermöglichen es dem Server, die Position des Spielers zu aktualisieren und die Updates zu übertragen, um einen synchronisierten Spielzustand zu gewährleisten. Spielerzustandsaktualisierungen stellen sicher, dass jeder Client den neuesten Spielzustand hat, sodass alle Spieler dieselben Positionen und Interaktionen sehen. Wenn neue Spieler das Spiel verlassen oder beitreten ermöglicht es dem Server, alle Clients zu benachrichtigen, wenn sich die Spielumgebung aufgrund von Spieleraktivitäten ändert. Durch die Kollisionen und Ereignisse wird die Spiellogik damit aktualisiert, z. B., wenn ein Spieler stirbt oder nach dem Verzehr von Nahrung oder anderen Spielern grösser wird. Schliesslich werden auch die Spielerpunktzahl hält die Clients über die Rangliste oder den Fortschritt (Leaderboard) der Spieler auf dem Laufenden.

3.2.4.1 Status Antworten HTTP

http Status	Beschreibung
200 OK	Für alle erfolgreichen Operationen.
201 CREATED	Wird bei POST-Methoden verwendet, wenn die Operation erfolgreich war.
202 ACCEPTED	Wird verwendet, um Anforderungen an den Service zu bestätigen.
400 BAD REQUEST	Ungültige Anfrage. Dies kann verwendet werden, wenn die Überprüfung der clientseitigen Eingabe fehlschlägt.
401 UNAUTHORIZED	Wenn der User nicht authentifiziert ist und daher den Service nicht abfragen kann.
403 FORBIDDEN	Wenn der User zwar authentifiziert, jedoch nicht autorisiert ist den Service abzufragen.
404 NOT FOUND	Wenn die angefragte Ressource nicht vorhanden ist.
409 CONFLICT	Konflikt (z.B. Benutzername oder E-Mail bereits vorhanden)
503 SERVICE UNAVAILABLE	Wenn der angefragte Service vorübergehend nicht zur Verfügung steht.

Tabelle 34: Zeigt die unterschiedlichen http-Status Antworten

3.2.4.2 Netzwerkprotokoll

register	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "register", "username": "player123", "email": "player123@example.com", "password": "hashed_password_here" }</pre>
register_success	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "register_success", "userId": "607f1f77bcf86cd799439011", "username": "player123" }</pre>
register_fail	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "register_fail", "error": "Email already registered" }</pre>
Status Codes	
200	Registration Successful
400	Bad Request (z.B. fehlende Parameter, ungültiges Format)
409	Conflict (z.B. Benutzernamen oder E-Mail bereits vorhanden)
500	Internal Server Error

Tabelle 35: Netzwerkprotokoll des Registrationsprozesses

Login	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "login", "username": "player123", "password": "hashed_password_here" }</pre>
login_success	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "login_success", "userId": "607f1f77bcf86cd799439011", "username": "player123", "token": "abc123securetoken" }</pre>
login_fail	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "login_fail", "error": "Invalid username or password" }</pre>
Status Codes	
200	Login Successful
400	Bad Request (fehlende Parameter oder ungültiges Format)
401	Unauthorized (falsches Passwort oder ungültiges Token)
500	Internal Server Error (bei unerwarteten Problemen)

Tabelle 36: Netzwerkprotokoll des Login Prozesses

Player Move (Broadcast)	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "move", "playerId": "607f1f77bcf86cd799439011", "direction": "up" // or "down", "left", "right" "position": { "x": 123.45, "y": 678.90 } }</pre>
move_broadcast	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "move_broadcast", "playerId": "607f1f77bcf86cd799439011", "position": { "x": 100, "y": 200 }, "direction": "up" }</pre>
Status Codes	
200	Move processed successfully
400	Invalid move parameters (Ausserhalb des Spielbereich, fehlende Daten)
500	Internal Server Error
Player Death (Broadcast)	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "death_broadcast",</pre>

	<pre>" playerId ": "607f1f77bcf86cd799439011", "username": "player123" }</pre>
Status Codes	
200	Death broadcasted successfully
500	Internal Server Error

Abbildung 3: Netzwerkprotokoll Player Death

Spawn Item	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "spawn_item", "itemId": "item123", "position": { "x": 100.0, "y": 200.0 } }</pre>
Status Codes	
200	Item spawned successfully
500	Internal Server Error
Item Collected	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "collect_item", "userId": "607f1f77bcf86cd799439011", "itemId": "item123" }</pre>
item_collected_broadcast	
Channel	WebSocket (JSON)
From → To	Server → All Clients

Parameters	<pre>{ "type": "item_collected_broadcast", "userId": "607f1f77bcf86cd799439011", "username": "player123", "itemId": "item123" }</pre>
Status Codes	
200	Item collected successfully
404	Item not found or already collected
500	Internal Server Error

Leaderboard	
Channel	WebSocket (JSON)
From → To	Server → Client
Parameters	<pre>{ "type": "leaderboard", "players": [{ "userId": "607f1f77bcf86cd799439011", "username": "player123", "score": 1500 }, { "userId": "607f1f77bcf86cd799439012", "username": "player456", "score": 1400 }] }</pre>
Status Sodes	
200	Leaderboard retrieved successfully
500	Internal Server Error
New Player Joined	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "player_joined", "userId": "607f1f77bcf86cd799439011", "username": "player123" }</pre>
Status Codes	
200	Player joined successfully
500	Internal Server Error

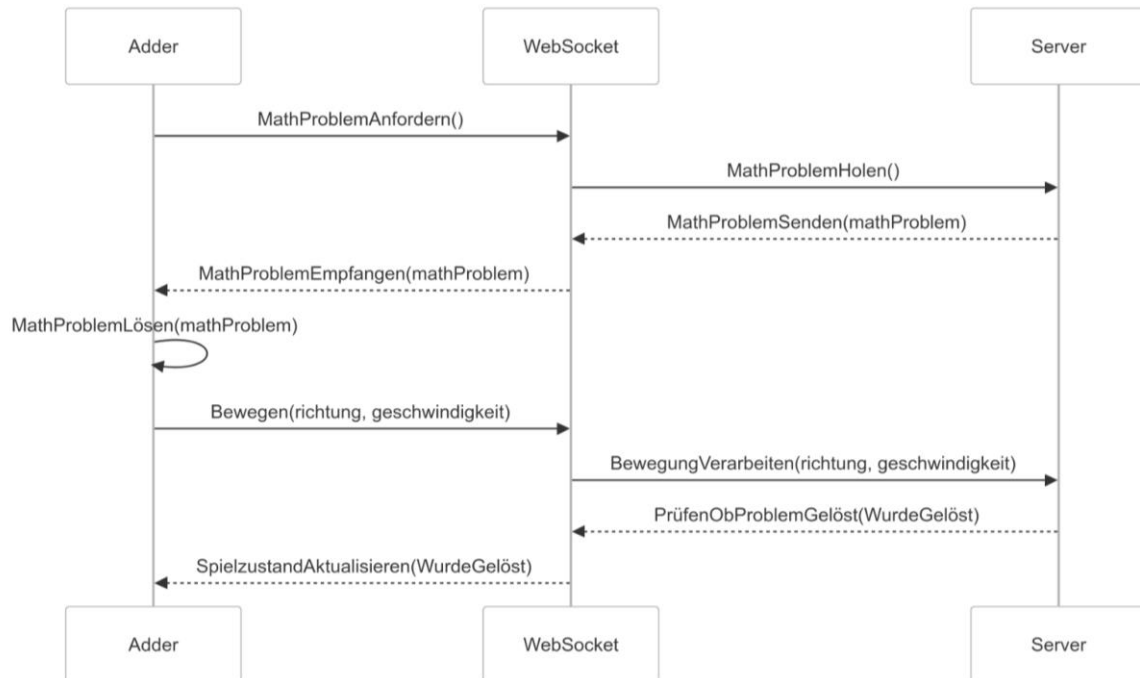
New Game Round Start	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "round_start", "roundId": "round123", "startTime": "2024-10-23T12:00:00Z" }</pre>
Status Codes	
200	Round started successfully
500	Internal Server Error
Player Disconnected	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "player_disconnected", "userId": "607f1f77bcf86cd799439011", "username": "player123" }</pre>
Status Codes	
200	Round disconnected successfully
500	Internal Server Error

Player Score Update	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "score_update", "userId": "607f1f77bcf86cd799439011", "score": 1500 }</pre>
Status Codes	
200	Score updated successfully
500	Internal Server Error
Chat Message	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "chat_message", "userId": "607f1f77bcf86cd799439011", "message": "Hello, all!" }</pre>
Status Codes	
200	Message sent successfully
400	Invalid message content
500	Internal Server Error

chat_broadcast	
Channel	WebSocket (JSON)
From → To	Client → Server
Parameters	<pre>{ "type": "chat_broadcast", "userId": "607f1f77bcf86cd799439011", "username": "player123", "message": "Hello, all!" }</pre>
Game Over	
Channel	WebSocket (JSON)
From → To	Server → All Clients
Parameters	<pre>{ "type": "game_over", "winnerId": "607f1f77bcf86cd799439011", "username": "player123", "score": 2000 }</pre>
Status Codes	
200	Game over broadcasted successfully
500	Internal Server Error

Tabelle 37: Netzwerkprotokoll der Spielerbewegung

3.2.4.3 Sequenzdiagramm

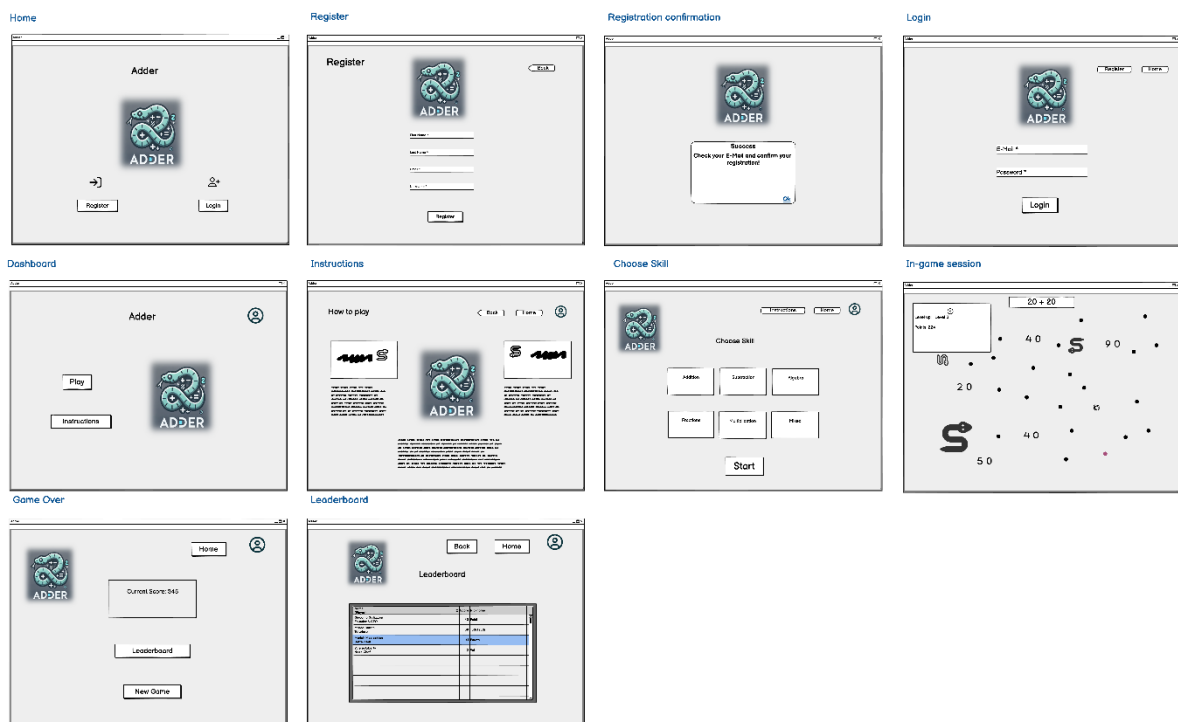


3.2.5 Frontend Mockups

Als Mockup und Wireframe Tool wurde eine Testversion von «Balsamiq» verwendet.

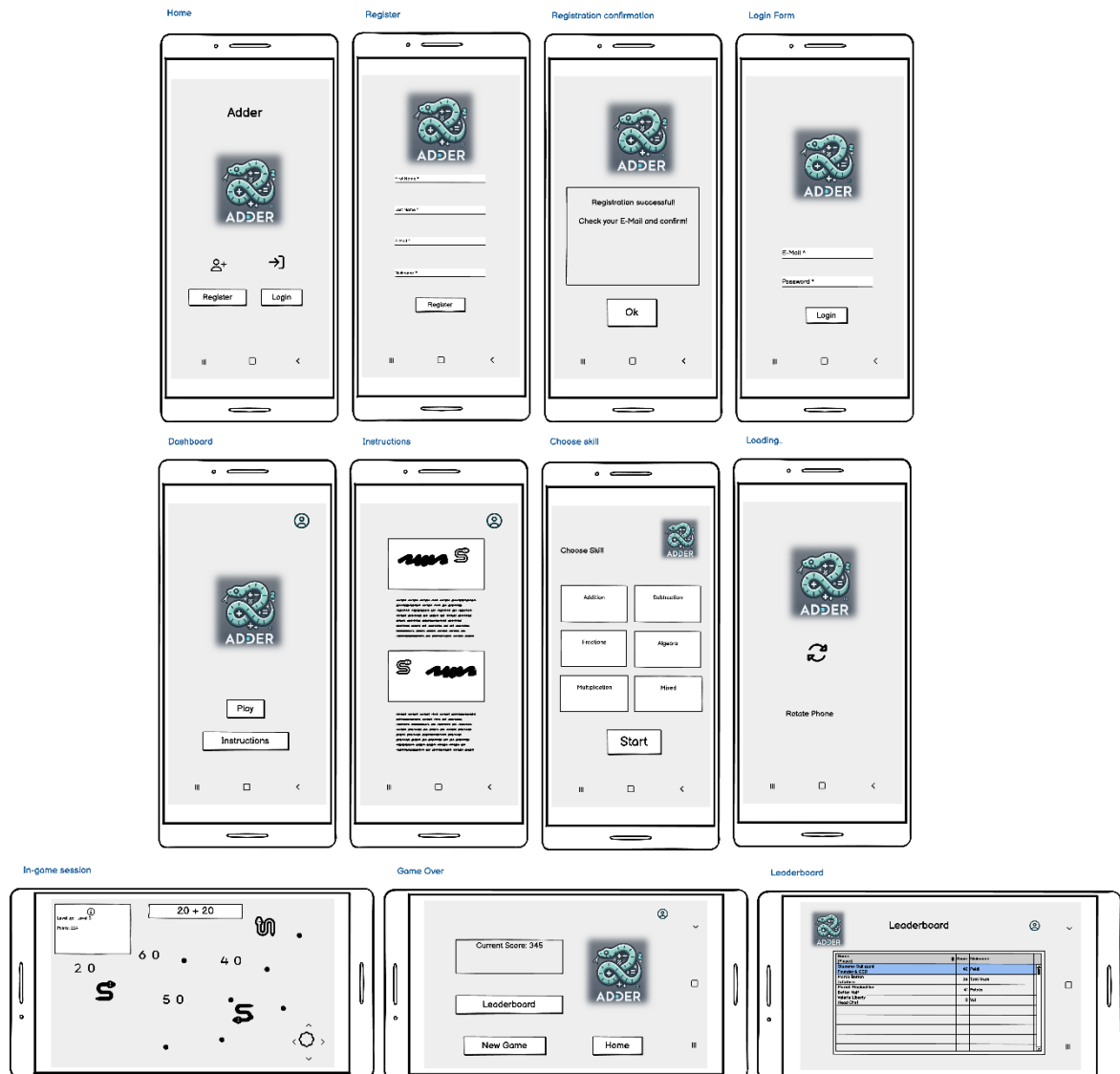
3.2.5.1 Desktop

Desktop



3.2.5.2 Mobile

Mobile



4 Realisierung

4.1 Testplan

4.2 Testprotokolle

5 Einführung

5.1 Inbetriebnahme

5.2 Erfüllungsgrad der Anforderungen

5.2.1 Funktionale Anforderungen

5.2.1.1 Muss-Anforderungen

Nr.	Unter-Nr.	Beschreibung	Erfüllt
MFA 1	1.0	Spielstart und -beendigung Das Spiel muss die Möglichkeit bieten, ein Spiel zu starten oder zu beenden.	0/1
	1.1	Das Spiel muss die Möglichkeit bieten, durch einen Start-Button ein Spiel zu initialisieren.	
	1.2	Das Spiel muss die Möglichkeit bieten, durch einen Beenden-Button das Spiel ordnungsgemäss zu beenden und den Fortschritt zu speichern.	
	1.3	Das Spiel muss die Möglichkeit bieten, nach Spielende eine Zusammenfassung der erreichten Punkte anzuzeigen.	
MFA 2	2.0	Spieler-Authentifizierung Das Spiel muss die Möglichkeit bieten, dass sich neue Spieler registrieren und anmelden können.	0/1
	2.1	Das Spiel muss die Möglichkeit bieten, dass sich neue Spieler mit einem Benutzernamen, einer E-Mail Adresse und einem Passwort registrieren können.	
	2.2	Das Spiel muss die Möglichkeit bieten, dass ein Spieler sich mit einer validen und einzigartigen E-Mail-Adresse registriert.	

	2.3	Der registrierende Spieler muss beim Registrierungsprozess aufgefordert werden, sein Passwort zweimal identisch einzugeben.	
	2.4	Der registrierende Spieler muss beim Registrierungsprozess aufgefordert werden, einen einzigartigen Benutzernamen einzugeben.	
	2.5	Das Spiel muss die Möglichkeit bieten, dass sich registrierte Benutzer mit einer E-Mail Adresse und einem Passwort anmelden können.	
MFA 3	3.0	Mathematikfragen Das Spiel muss verschiedene Arten von mathematischen Aufgaben beinhalten, welche dem Spieler zur Auswahl angeboten werden.	0/1
	3.1	Das Spiel muss die Möglichkeit bieten, mindestens drei verschiedene Arten von Mathematikfragen auszuwählen.	
	3.2	Das Spiel muss die Möglichkeit bieten, jede Frage zufällig aus einem Pool von mindestens 20 Fragen pro Art auszuwählen.	
	3.3	Das Spiel muss die Möglichkeit bieten, die Schwierigkeitsgrade der Fragen dynamisch basierend auf der Leistung des Spielers anzupassen.	
MFA 4	4.0	Score-Based Belohnungssystem Das Spiel muss ein Punktesystem enthalten, das die Spieler für das Einsammeln von Bubbles auf dem Spielfeld und das Lösen von Mathematikaufgaben belohnt.	0/1
	4.1	Das Spiel muss die Möglichkeit bieten, für jeden eingesammelten Bubble Punkte zu vergeben.	
	4.2	Das Spiel muss die Möglichkeit bieten, für jede korrekt beantwortete Frage Punkte zu vergeben.	
	4.3	Das Spiel muss die Möglichkeit bieten, dass bei falsch beantworteten Fragen Punkte abgezogen werden.	
	4.4	Das Spiel muss die Möglichkeit bieten, die Punkte am Ende jedes Levels und Spiels zusammengefasst zu präsentieren.	

MFA 5	5.0	Spielerbewegung Das Spiel muss die Möglichkeit bieten, die Spielfigur (Schlange) durch das Spiel zu steuern.	0/1
	5.1	Das Spiel muss die Möglichkeit bieten, dass der Spieler die Spielfigur mit den Pfeiltasten oder der Maus steuert.	
	5.2	Das Spiel muss die Möglichkeit bieten, dass die Geschwindigkeit der Spielfigur manuell erhöht wird, wodurch Punkte verloren gehen.	
	5.3	Das Spiel muss sicherstellen, dass bei einer Kollision mit einem anderen Spieler oder dem Spielfeldrand das Spiel für den kollidierenden Spieler beendet wird.	
MFA 6	6.0	Mehrspieler-Funktionalität Das Spiel muss eine Mehrspieler-Funktion unterstützen, die es mehreren Spielern ermöglicht, gleichzeitig zu spielen.	0/1
	6.1	Das Spiel muss die Möglichkeit bieten, dass mindestens zwei Spieler gleichzeitig im gleichen Spielraum spielen.	
	6.2	Das Spiel muss die Möglichkeit bieten, dass Spieler den Fortschritt und die Position des anderen Spielers in Echtzeit sehen können.	
	6.3	Das Spiel muss die Möglichkeit bieten, ein Kommunikationssystem wie einen Chat zur Interaktion zwischen den Spielern zu nutzen.	
MFA 7	7.0	Levelauswahl Das Spiel muss eine Funktion zur Auswahl verschiedener Spieltypen bereitstellen, die den Spielern ermöglicht, zwischen unterschiedlichen mathematischen Herausforderungen zu wählen.	0/1
	7.1	Das Spiel muss die Möglichkeit bieten, dass der Spieler aus mindestens drei verschiedenen Spieltypen (z.B. Addition, Subtraktion und Multiplikation) auswählen kann.	
	7.2	Das Spiel muss die Möglichkeit bieten, dass der Spieler das Level vor Spielbeginn ändern kann.	
MFA 8	8.0	Rückmeldungssystem	0/1

		Das Spiel muss visuelle und auditive Rückmeldungen geben, wenn ein Spieler Punkte gewinnt oder verliert.	
	8.1	Das Spiel muss die Möglichkeit bieten, eine visuelle Rückmeldung zu geben, wenn eine Aufgabe richtig beantwortet wurde (z.B. Schlange blinkt grün und wird grösser).	
	8.2	Das Spiel muss die Möglichkeit bieten, eine visuelle Rückmeldung zu geben, wenn eine Aufgabe falsch beantwortet wurde (z.B. Schlange blinkt rot und wird kleiner).	
	8.3	Das Spiel muss die Möglichkeit bieten, bei jeder Antwort eine kurze akustische Rückmeldung zu geben, die optional deaktivierbar ist.	
MFA 9	9.0	Leaderboard Das Spiel muss eine Rangliste enthalten, welche die besten Spieler basierend auf ihrem Highscore anzeigt.	0/1
	9.1	Das Spiel muss die Möglichkeit bieten, die Top 10 Spieler basierend auf ihrem Highscore im Leaderboard anzuzeigen.	
	9.2	Das Spiel muss die Möglichkeit bieten, das Leaderboard nach wöchentlichen, monatlichen und Gesamt-Ergebnissen zu filtern.	
	9.3	Das Spiel muss die Möglichkeit bieten, die Spieler nach Benutzernamen und Punktzahl zu sortieren.	
	9.4	Das Spiel muss die Möglichkeit bieten, im Leaderboard anhand einer Volltextsuche einen Benutzernamen zu suchen.	
MFA 10	10.0	Spielerfortschritt Das Spiel muss den Fortschritt des Spielers speichern und beim erneuten Einloggen wiederherstellen können.	0/1
	10.1	Das Spiel muss die Möglichkeit bieten, dass der Fortschritt des Spielers automatisch nach jedem abgeschlossenen Spiel gespeichert wird.	
	10.2	Das Spiel muss die Möglichkeit bieten, dass Spieler ihren Fortschritt über eine Profilseite einsehen können, welche	

		Statistiken wie gespielte Level, Gesamtpunkte und erreichte Erfolge/Badges anzeigt.	
--	--	---	--

Tabelle 38: Erfüllungsgrad der funktionale Muss-Anforderungen

5.2.1.2 Soll-Anforderungen

Nr.	Unter-Nr.	Beschreibung	Gewichtung
SFA 1	1.0	Passwort Reset Das Spiel sollte die Möglichkeit bieten, dass ein Spieler sein Passwort über eine Passwort-vergessen-Funktion zurücksetzen kann.	2
SFA 2	2.0	Benutzerverwaltung Das Spiel sollte die Möglichkeit bieten, dass ein Spieler seine Benutzerinformationen (Passwort, Benutzername und E-Mail Adresse) ändern kann.	3
SFA 3	3.0	Achievement-Based Belohnungssystemen Das Spiel sollte eine Funktionalität beinhalten, welche es Spielern erlaubt spezielle Ziele abzuschliessen und dadurch Badges zu erhalten.	3
SFA 4	4.0	Werbung Das Spiel sollte einen Bereich enthalten, an welchem Werbung eingespielt werden kann.	1
	4.1	Das Spiel sollte die Möglichkeit bieten, dass Benutzer ein Abonnement abschliessen können, um die Werbung nicht angezeigt zu bekommen.	1

Tabelle 39: Erfüllungsgrad der funktionalen Soll-Anforderungen

6 Projektmanagement

Als Ergänzung zu unseren Vorgehensmodell in Kapitel 1.5 werden wir den agilen Ansatz nach SCRUM für die Entwicklung des Projekts anwenden. Für eine Reibungslose Zusammenarbeit werden alle Tasks (Work Items) in einem Azure Board erstellt und bearbeitet. Hier entspricht jeden Sprint oder Iteration einen Meilenstein. Es gibt pro PVA einen Meilenstein wo bereits von der FFHS gemäss Moodle Eintrag «Projektarbeit»

<https://moodle.ffhs.ch/mod/page/view.php?id=4752578> vorgegeben ist. Jeden Sprint dauert in gemäss dieser Planung drei Wochen.

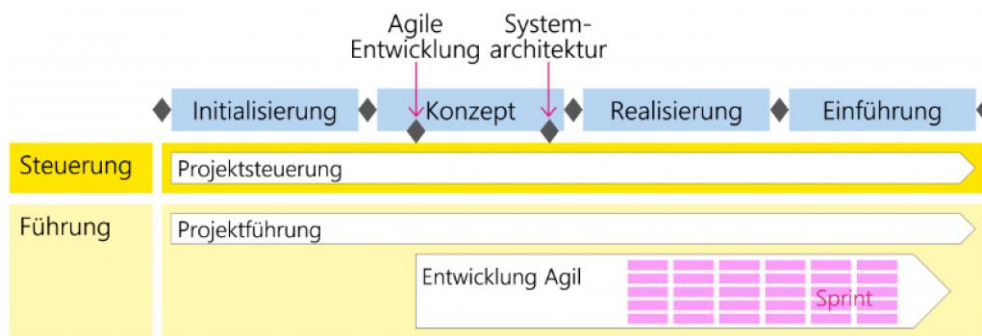


Abbildung 4: Quelle hermes.admin.ch

6.1 Azure Boards

Azure Boards gehört zur Azure DevOps Suite und bietet eine Reihe von Agile-Tools zur Unterstützung der Planung und Nachverfolgung von Arbeit, Codefehlern und Problemen mithilfe von Kanban- und Scrum-Methoden. Pro Meilenstein wird ein Epic erstellt. Alle dazugehörigen Aufwände und Arbeiten werden mit «Issues» angelegt. Je nach Bedarf werden auch «Tasks» zu den «Issues» angelegt.

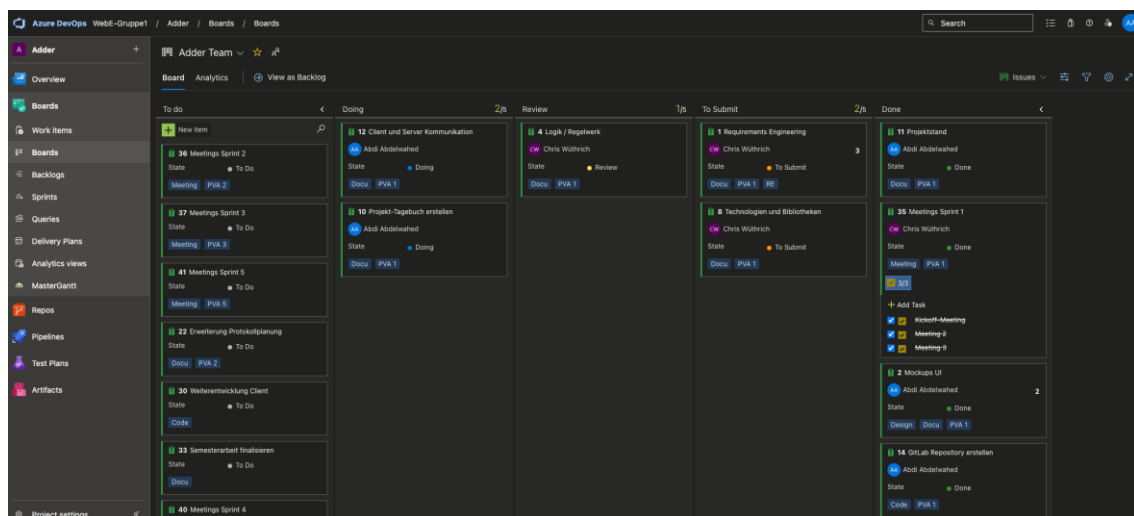


Abbildung 5: Projektteam-Board

Unter «Work Items» wird die Meilenstein Ansicht angezeigt:

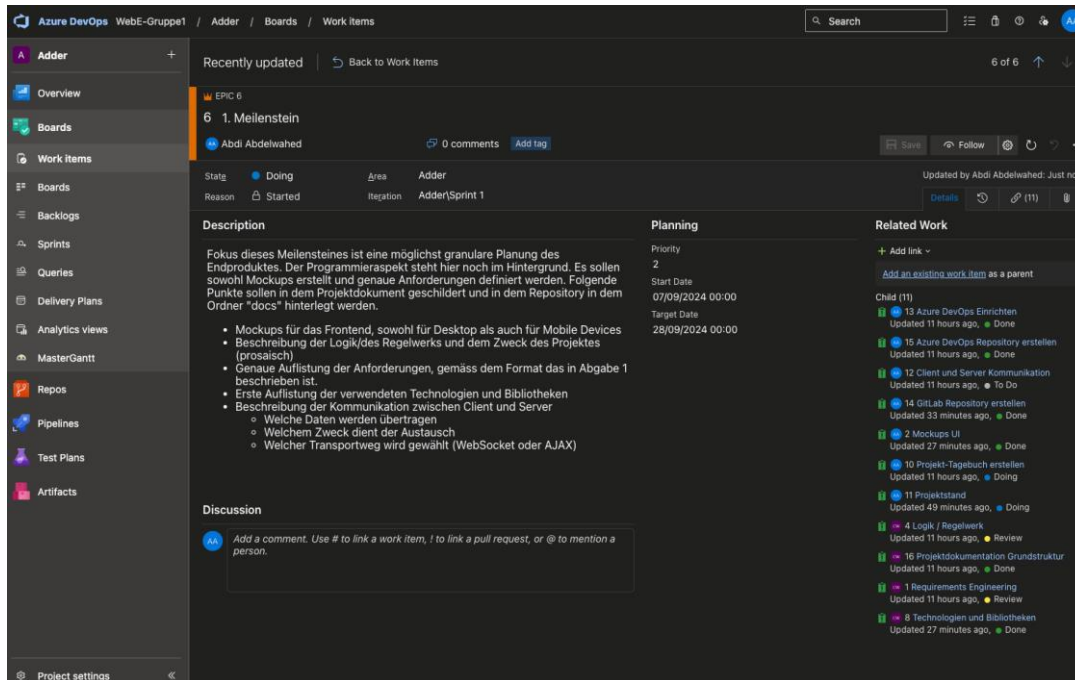


Abbildung 6: Meilenstein Ansicht auf Azure Boards

6.2 Projekt-Tagebuch

Das Projekt-Tagebuch beinhaltet sowohl die geplanten und gehaltenen Meetings im Verlauf des Semesters als auch alle Projektfortschritte. Zu den Meetings, wird in jedem Sprint einen Work Item vom Typ «Issue» angelegt und darunter wird je Meeting einen Work Item vom Typ «Task» angelegt. In jedem Task bzw. Meeting werden die Ziele bereits im Voraus definiert und am Anschluss des Meetings die Ergebnisse nach geführt. Auf der folgenden Abbildung sind die Projektfortschritte und Meetings ersichtlich.

ID	Title	Assigned To	State	Area Path	Tags	Comments	Activity Date
6	1. Meilenstein	Abdi Abdelwahed	Doing	Adder	PVA 1		28/09/2024 14:13:13
42	Meeting 3	Abdi Abdelwahed	Done	Adder	Meeting PVA 1		28/09/2024 15:54:39
11	Projektstand	Abdi Abdelwahed	Done	Adder	Docu PVA 1		28/09/2024 15:54:06
35	Meetings Sprint 1	Chris Wüthrich	Done	Adder	Meeting PVA 1		28/09/2024 15:47:20
10	Projekt-Tagebuch erstellen	Abdi Abdelwahed	Doing	Adder	Docu PVA 1		28/09/2024 15:41:31
8	Technologien und Bibliotheken	Chris Wüthrich	To Submit	Adder	Docu PVA 1	1	28/09/2024 15:39:37
39	Meeting 2	Chris Wüthrich	Done	Adder	Meeting PVA 1		28/09/2024 15:30:49
12	Client und Server Kommunikation	Abdi Abdelwahed	Doing	Adder	Docu PVA 1		28/09/2024 14:28:46
16	Projektdokumentation Grundstruktur	Chris Wüthrich	Done	Adder	Docu PVA 1		28/09/2024 14:25:56
15	Azure DevOps Repository erstellen	Abdi Abdelwahed	Done	Adder	Code PVA 1		28/09/2024 14:25:51
2	Mockups UI	Abdi Abdelwahed	Done	Adder	Design Docu PVA 1		28/09/2024 14:22:33
14	GitLab Repository erstellen	Abdi Abdelwahed	Done	Adder	Code PVA 1	1	28/09/2024 14:22:22
4	Logik / Regelwerk	Chris Wüthrich	Review	Adder	Docu PVA 1	1	28/09/2024 14:22:09
13	Azure DevOps Einrichten	Abdi Abdelwahed	Done	Adder	Code PVA 1		28/09/2024 14:21:55
1	Requirements Engineering	Chris Wüthrich	To Submit	Adder	Docu PVA 1 RE	1	28/09/2024 14:19:11
38	Kickoff-Meeting	Abdi Abdelwahed	Done	Adder	Meeting PVA 1		28/09/2024 14:16:30

Abbildung 7: Projekt-Tagebuch 1. Meilenstein

6.3 Projektstand

6.3.1 PVA 1

ID	Work Item Type	Title	Assigned To	State	Tags	Start Date	Target Date
6	Epic	1. Meilenstein	Abdi Abdelwahed < aabdelwa1@gmail.com >	Doing	PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
13	Issue	Azure DevOps Einrichten	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Code; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
15	Issue	Azure DevOps Repository erstellen	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Code; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
12	Issue	Client und Server Kommunikation	Abdi Abdelwahed < aabdelwa1@gmail.com >	Doing	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
14	Issue	GitLab Repository erstellen	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Code; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
38	Task	Kickoff-Meeting	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Meeting; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
4	Issue	Logik / Regelwerk	Chris Wüthrich < chris.wue-thrich@hotmail.com >	Review	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
39	Task	Meeting 2	Chris Wüthrich < chris.wue-thrich@hotmail.com >	Done	Meeting; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
42	Task	Meeting 3	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Meeting; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
35	Issue	Meetings Sprint 1	Chris Wüthrich < chris.wue-thrich@hotmail.com >	Done	Meeting; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
2	Issue	Mockups UI	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Design; Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
16	Issue	Projektdokumentation Grundstruktur	Chris Wüthrich < chris.wue-thrich@hotmail.com >	Done	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
11	Issue	Projektstand	Abdi Abdelwahed < aabdelwa1@gmail.com >	Done	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
10	Issue	Projekt-Tagebuch erstellen	Abdi Abdelwahed < aabdelwa1@gmail.com >	Doing	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00
1	Issue	Requirements Engineering	Chris Wüthrich < chris.wue-thrich@hotmail.com >	To Submit	Docu; PVA 1; RE	07/09/2024 12:00:00	28/09/2024 00:00:00
8	Issue	Technologien und Bibliotheken	Chris Wüthrich < chris.wue-thrich@hotmail.com >	To Submit	Docu; PVA 1	07/09/2024 12:00:00	28/09/2024 00:00:00

7 Verzeichnisse

7.1 Abbildungsverzeichnis

Abbildung 1: Vorgehensmodell	4
Abbildung 2: Überarbeitete Schablone zur Definition von Anforderungen [2, p. 361].	9
Abbildung 3: Netzwerkprotokoll Player Death	37
Abbildung 4: Quelle hermes.admin.ch	51
Abbildung 5: Projektteam-Board	51
Abbildung 6: Meilenstein Ansicht auf Azure Boards	52
Abbildung 7: Projekt-Tagebuch 1. Meilenstein	52

7.2 Tabellenverzeichnis

Tabelle 1: Termine	2
Tabelle 2: Formale Angaben	5
Tabelle 3: MFA-001 - Spielstart und -beendigung	11
Tabelle 4: MFA-001 KPIs	11
Tabelle 5: MFA-002 - Spieler-Authentifizierung	12
Tabelle 6: MFA-002 KPIs	12
Tabelle 7: MFA-003 – Mathematikaufgaben	13
Tabelle 8: MFA-003 KPIs	13
Tabelle 9: MFA-004 - Score-Based Belohnungssystem	14
Tabelle 10: MFA-004 KPIs	14
Tabelle 11: MFA-005 – Spielerbewegung	15
Tabelle 12: MFA-005 KPIs	16
Tabelle 13: MFA-006 - Mehrspieler-Funktionalität	16
Tabelle 14: MFA-006 KPIs	16
Tabelle 15: MFA-007 – Levelauswahl	17
Tabelle 16: MFA-007 KPIs	17
Tabelle 17: MFA-008 – Rückmeldungssystem	18
Tabelle 18: MFA-008 KPIs	18
Tabelle 19: MFA-009 – Leaderboard	19
Tabelle 20: MFA-009 KPIs	19
Tabelle 21: MFA-010 – Spielerfortschritt	20
Tabelle 22: MFA-010 KPIs	20
Tabelle 23: MFA-002 - Spieler-Authentifizierung	21
Tabelle 24: MFA-010 KPIs	21
Tabelle 25: SFA-001 - Passwort Reset	22
Tabelle 26: SFA-002 – Benutzerverwaltung	23
Tabelle 27 SFA-003 - Achievement-Based Belohnungssystem:	24
Tabelle 28: SFA-004 – Werbung	25
Tabelle 29: SFA-004.1 - Werbung deaktivieren (Abonnement)	26
Tabelle 30: Nicht funktionale Muss-Anforderungen	29
Tabelle 31: Nicht funktionale Soll-Anforderungen	30
Tabelle 32: Use-Case 1.1 - Benutzerkonto anlegen	31
Tabelle 33: Use-Case 1.2 – Benutzerkonto Anmelden	32
Tabelle 34: Zeigt die unterschiedlichen http-Status Antworten	33

Tabelle 35: Netzwerkprotokoll des Registrationsprozesses.....	34
Tabelle 36: Netzwerkprotokoll des Login Prozesses	35
Tabelle 37: Netzwerkprotokoll der Spielerbewegung.....	42
Tabelle 38: Erfüllungsgrad der funktionale Muss-Anforderungen	50
Tabelle 39: Erfüllungsgrad der funktionalen Soll-Anforderungen.....	50

7.3 Quellenverzeichnis

- [1] „WIKIPEDIA,“ 26 09 2024. [Online]. Available: <https://de.wikipedia.org/wiki/Slither.io>.
- [2] C. Rupp und d. SOPHISTen, Requirements-Engineering und -Management - Das Handbuch für Anforderungen in jeder Situation, München: Carl Hanser Verlag, 2021.

7.4 Hilfsmittelverzeichnis

Welches Hilfsmittel wurde eingesetzt?	Wozu wurde das Hilfsmittel eingesetzt?	Betroffene Stellen

8 Anhang

9 Selbständigkeitserklärung

Hiermit erkläre ich,

- dass ich die vorliegende Arbeit selbstständig verfasst habe,
- dass alle sinngemäss und wörtlich übernommenen Textstellen aus fremden Quellen kenntlich gemacht wurden,
- dass alle mit Hilfsmitteln erbrachten Teile der Arbeit präzise deklariert wurden,
- dass keine anderen als die im Hilfsmittelverzeichnis aufgeführten Hilfsmittel verwendet wurden,
- dass das Thema, die Arbeit oder Teile davon nicht bereits Gegenstand eines Leistungsnachweises eines anderen Moduls waren, sofern dies nicht ausdrücklich mit der Referentin oder dem Referenten im Voraus vereinbart wurde,
- dass ich mir bewusst bin, dass meine Arbeit elektronisch auf Plagiate und auf Dittautorschaft menschlichen oder technischen Ursprungs überprüft werden kann und ich hiermit der FFHS das Nutzungsrecht so weit einräume, wie es für diese Verwaltungshandlungen notwendig ist.

Ort, Datum, Unterschrift

Ort, Datum, Unterschrift