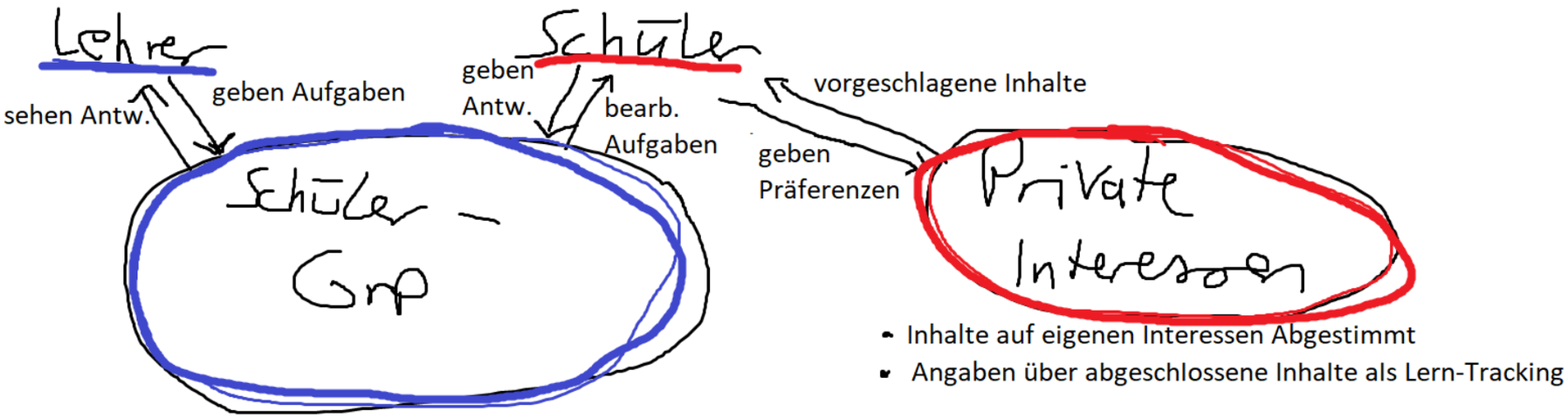
System über Ernährungsbildung - Audit 2

Ernährungsbildung für Jugendliche in der Schule sowie privat.
Gruppe: Sven, Perit, Caro
Github: Repository

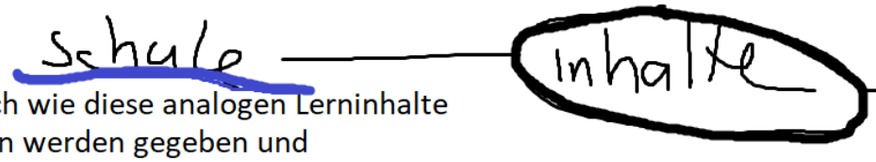
Erstes Konzept

Erstes Konzept



- Chat mit Lehrer (Gruppe und Privat)
- Aufgaben die im Unterricht bearb. werden sollen
- Lehrer kann Fortschritt (privat/Schule?) der Schüler sehen

- Inhalte auf eigenen Interessen Abgestimmt
- Angaben über abgeschlossene Inhalte als Lern-Tracking



- didaktisch wie diese analogen Lerninhalte
- Aufgaben werden gegeben und an alle Schüler "weitergeleitet
- Aufgaben können ggf. digital bearbeitet werden (Texte)

hier könnte man ein analoges Arbeitsblatt nehmen und digital gestalten (jeder Schüler sieht: Das ist das Thema, das sind eure Aufgaben -> Antworten als Text angeben, später diskussion führen. Wenns Passt nur Fragen Stellen und in Klasse live diskutieren)

Kein Fokus auf "Wie sollen die Aufgaben aussehen", das sollten nur Beispiele bleiben, weil wir kein didaktisches wissen haben -> deshalb Vorlage der Analogen Blätter!
-> ggf über mehr Funktionen Nachdenken, je nach Anforderungen der Aufgaben ;)

- plakativere, modernere, aufregende Inhalte
  wie bei Analogen Inhalten ;)
- Gamifizierte Inhalte hier plazieren ggf. Quizzes, kleine minigames, etc.
- neben Inhalte persönlicher Interessen auch "allgemeine Aufgaben anzeigen damit quizzes sinn machen
  -> Technisch gesehen: Themenbereiche ausdenken und Inhalte damit Taggen

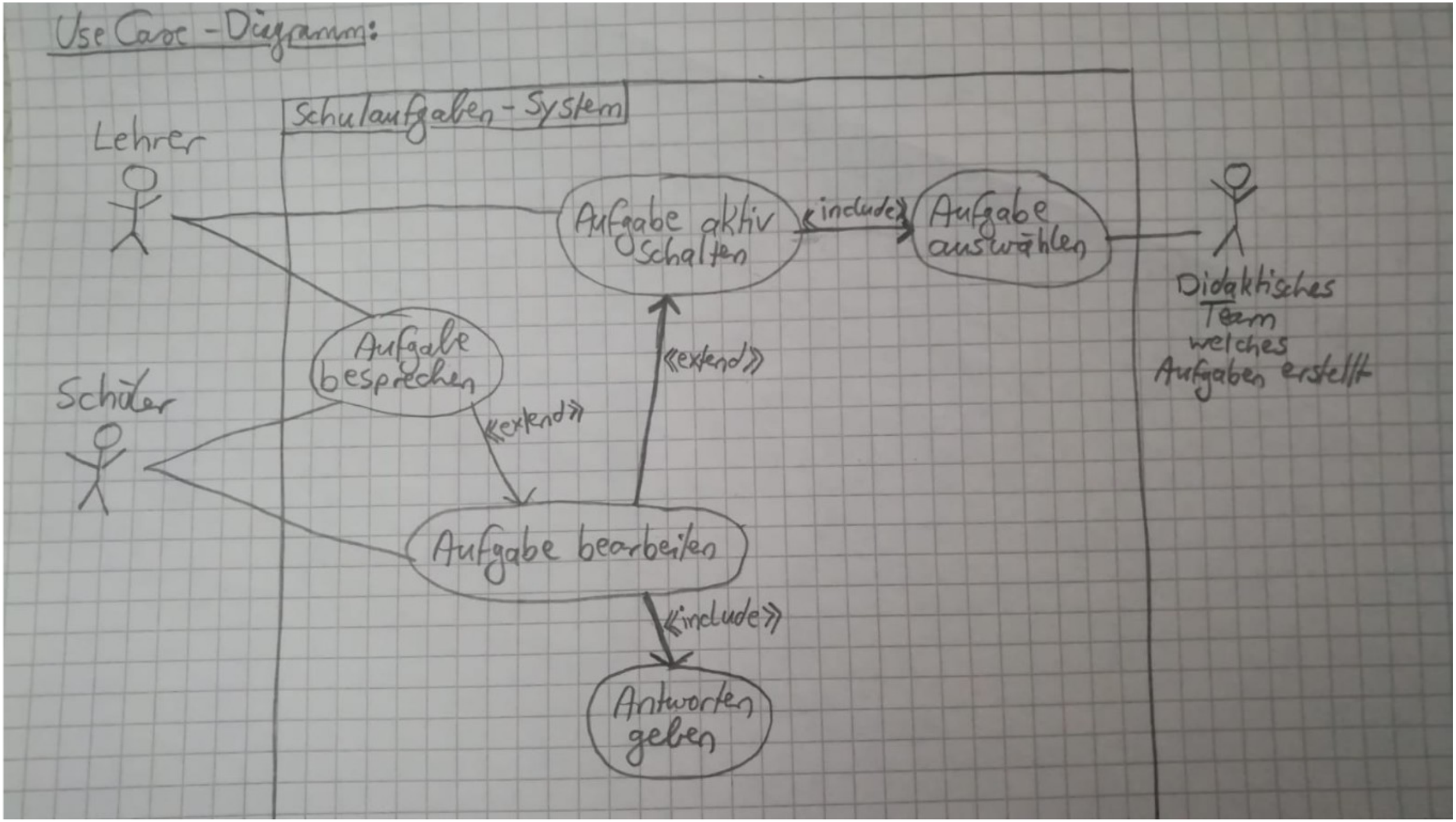
→ Konkrete Gamifizierungen ausarbeiten!
→ Tags ausarbeiten!

- Evtl. Anzahl x an Aufgaben von ThemenGruppen (wie bei Duolingo), dann Quiz etc. als mini Test?

Github Link - Erstes Konzept

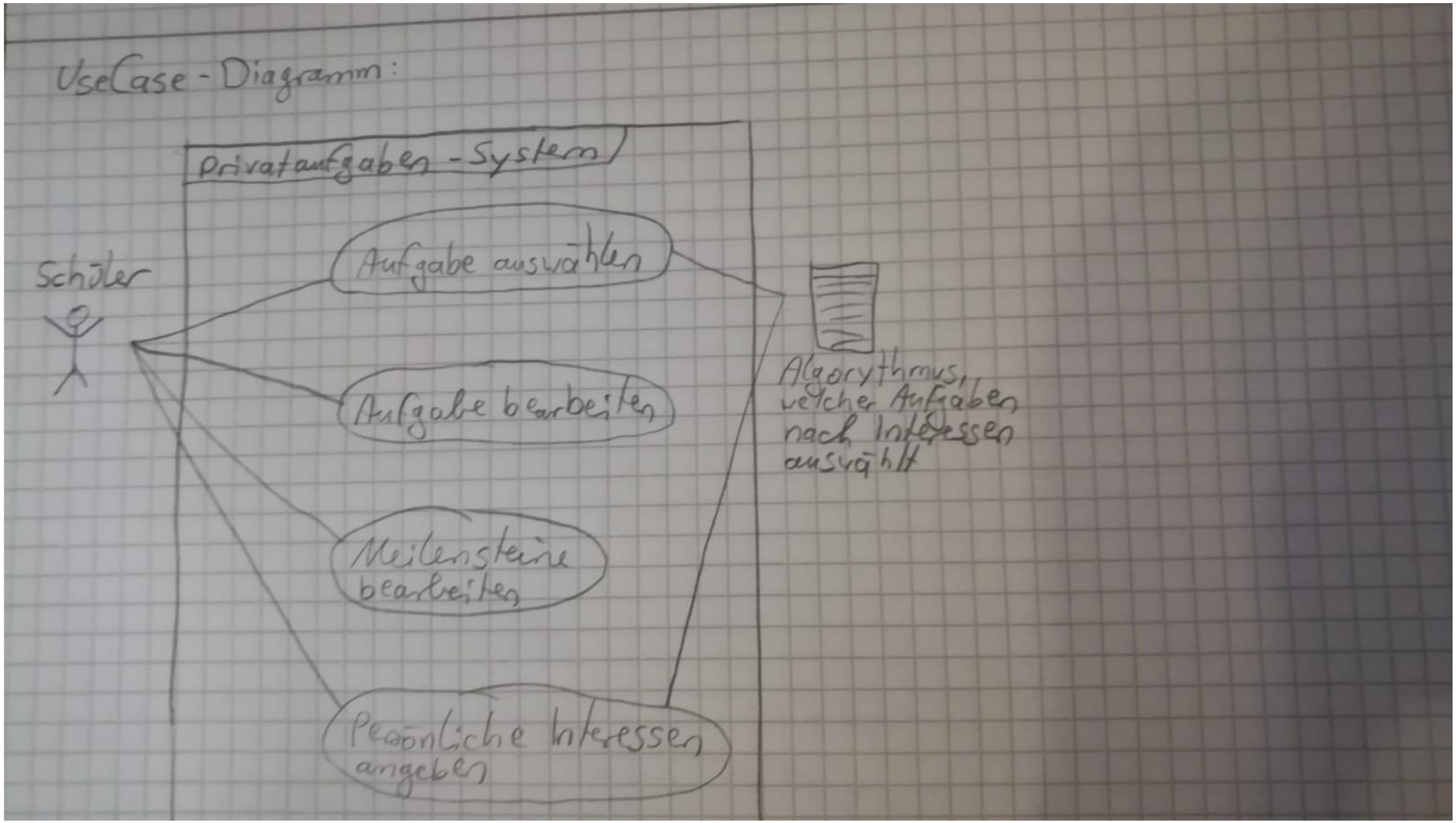
Use Cases

Use Case - Schulaufgaben-System



Github Link - Use Case Diagramm - Schulaufgabensystem

Use Case - Privataufgaben-System



Github Link - Use Case Diagramm - Privataufgabensystem - Iteriert
Use Case - Diagramm - Privataufgabensystem - Alt

Funktionsübersicht

Funktionsübersicht

3. Schulaufgaben-Verwaltungssystem

Funktionsname: Aktivierung und Bearbeitung von Schulaufgaben

Zweck: Ermöglicht Lehrern die Aktivierung von didaktisch vorbereiteten Aufgaben für ihre Schüler. Schüler können die Aufgaben einsehen, mit Lehrern besprechen und Lösungen einreichen.

Grundlegender Ablauf für Lehrer:

a. Der Lehrer meldet sich im System an und navigiert zum Aufgabenbereich.
b. Der Lehrer wählt aus einer Liste von vom didaktischen Team vorbereiteten Aufgaben eine aus.
c. Der Lehrer aktiviert die gewählte Aufgabe für die entsprechende Klasse oder Gruppe.
d. Die Aufgabe wird für die Schüler sichtbar und kann von ihnen bearbeitet werden.

Grundlegender Ablauf für Schüler:

a. Der Schüler meldet sich im System an und sieht die für ihn aktiven Aufgaben.
b. Der Schüler wählt eine Aufgabe aus und hat die Möglichkeit, diese mit dem Lehrer zu besprechen, falls Fragen bestehen.
c. Der Schüler bearbeitet die Aufgabe und reicht seine Antwort über das System ein.
d. Der Lehrer kann die eingereichten Antworten einsehen und Feedback geben.

Github Link - Funktionsbeschreibung

Funktionsübersicht

4. Privataufgaben System

Funktionsname: Individualisierte Aufgabenbearbeitung und Meilensteinerreichung

Zweck: Diese Funktion ermöglicht Schülern die Auswahl und Bearbeitung von Aufgaben basierend auf ihren persönlichen Interessen. Das System verfolgt die Fertigstellung der Aufgaben und kennzeichnet erreichte Meilensteine.

Grundlegender Ablauf:

a. Der Schüler meldet sich im System an und gelangt zum Bereich der individuellen Aufgaben.
b. Auf Basis der vom Schüler angegebenen persönlichen Interessen schlägt das System eine Reihe von Aufgaben vor.
c. Der Schüler wählt eine Aufgabe aus dem vorgeschlagenen Pool aus und beginnt mit der Bearbeitung.
d. Nach Abschluss der Aufgabe aktualisiert das System den Fortschritt des Schülers und markiert die Aufgabe als erledigt.
e. Das Erreichen eines Meilensteins wird vom System registriert und entsprechend visualisiert oder belohnt.

Besondere Berücksichtigung:

a. Der Algorithmus, der Aufgaben vorschlägt, muss adaptiv sein und das Profil und die Vorlieben des Schülers berücksichtigen.
b. Die Aufgaben und der Fortschritt sollten sicher und vertraulich behandelt werden, um die Privatsphäre des Schülers zu schützen.
c. Eine benutzerfreundliche Schnittstelle, die es Schülern ermöglicht, ohne technische Schwierigkeiten Aufgaben zu wählen und zu bearbeiten, ist erforderlich.
d. Das System sollte Feedback und Belohnungen für das Erreichen von Meilensteinen integrieren, um die Motivation der Schüler zu steigern.

Github Link - Funktionsbeschreibung

Datenstruktur

Datenstruktur

Use Case	Variablenname	Datentyp	Sonstige Merkmale	Beschreibung
Kontoerstellung	email	**String**	unique	
	password	**String**	>8 characters, >= 1 number, >= 1 special character	
	name	**Abstract** (first_name (String), last_name (String)		Name des Schülers
	schoolclass	**Abstract** (number (Int), add (Char))		Schulklasse des Schülers (z.B 8b)
	is_email_verified	**Bool**		Flag ob Email verifiziert wurde
Login	email	**String**		
	password	**String**		
Schulgruppen-Management:				
Gruppe erstellen	classgroup	**Abstract** (name(String), course (String), schoolclass (Abstract (number (Int), add (Char)))		Schulklassen-Gruppe, welche im System erstellt werden soll
	admin	**Abstract** (teacher (name(String), is_teacher (Bool))		Name des Admins (Lehrers der classgroup erstellt hat)
	student_list	**Array/List/Dict...** (Abstract (student (name (Abstract (first_name (String), last_name (String)), is_student (Bool)))		Liste aller Teilnehmenden (Schüler)
Gruppe beitreten	classgroup	**Abstract** (name(Abstract (first_name (String), last_name (String)), course (String), schoolclass (Abstract (number (Int), add (Char)))		Name der Schulklassen-Gruppe, welcher der Schüler beitreten möchte
	student	**Abstract** (name (Abstract (first_name (String), last_name (String)), is_student (Bool)		Name des Schülers, welcher beitreten möchte
Schulaufgaben-system:				
Aufgabe Auswählen	task_list	**Array** (maybe Class „Task"?)		Liste aller Aufgaben
Aufgabe aktiv schalten	task	**Abstract** (maybe Class „Task"?)		Aufgabe, welche die Schüler bearbeiten sollen
	is_active	**Bool**		Flag ob Aufgabe aktiv ist (wenn ja können Schüler sie sehen und bearbeiten)
Aufgabe bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		Ggf. Referenz auf Teilaufgaben
	answer_list	**Dict/Array,...** (student (Abstract (name)): answer (String))		Liste, welche Antworten sammelt (ggf. nur vom Lehrer einsehbar)
	correct_answers	**String**	or Array (String)	Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	is_correct	**Bool**		ggf. Flag ob Antwort des Schülers korrekt ist
Antworten geben	student	**Abstract** (name (Abstract (first_name (String), last_name (String)), is_student (Bool)		
	answer	**String, png,**	different datatypes possible	
	task	**Abstract** (maybe Class „Task"?)		Referenz zur Aufgabe
Aufgabe besprechen	chat	**Abstract** (Array (String), name (Abstract), time (datetime))		chat, um Antworten zu besprechen
	is_open	**Bool**		Flag, ob Chat offen ist
Privataufgaben-system:				
Aufgabe Auswählen	task_list	**Array** (maybe Class „Task_Private"?)		
	task_tag	**String**		Tag, mit dem Aufgaben zu Interessen der Schüler vorgeschlagen werden
Aufgabe bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		
	answer_list_private	**List (String)**		
	correct_answers	**String**		Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	is_correct	**Bool**		ggf. Flag ob Antwort korrekt ist
Meilensteine bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		Teilaufgaben
	answer_list_private	**List (String)**		
	correct_answers	**String**		Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	is_correct	**Bool**		ggf. Flag ob Antwort korrekt ist
	is_completed	**Bool**		Flag ob Meilenstein vollständig und korrekt bearbeitet wurde
Persönliche Interessen angeben	interest_tag	**String**		Tag, mit dem Aufgaben zu Interessen der Schüler vorgeschlagen werden (z.B Sport, Kochen, Abnehmen, ...)
	interest_list	**Array** (interest_tag)		Liste aller Vorlieben

Github Link - Datenstruktur

Risikoanalyse

Risikoanalyse

Potentielle Risiken

- Unterschiede in der digitalen Infrastruktur in deutschen Schulen beziehungsweise mangelhafte digitale Infrastruktur in diesen

- Fehlende Bereitschaft im Lehrpersonal / mangelhafte Qualität des Lehrpersonals

- Probleme mit Plattformkompatibilität

- Unterschiedliche technische Fähigkeiten der Schüler

Proof of Concept

Proof of Concept

Ziele

- Funktionsfähigkeit vom Schulaufgaben-System sicherstellen
- Plattformkompatibilität testen => funktioniert Anwendung auf verschiedenen Geräten und Plattformen konsistent und effektiv
- Benutzerfreundlichkeit prüfen
- Prüfung der Tutorials bzw. der Anleitungen

Durchführung

Zur Überprüfung der Anwendung als Proof of Concept wird eine interne Testgruppe von Studierenden eingebunden, die die Anwendung in einer simulierten schulischen Umgebung verwenden wird, insbesondere im Kontext des Schulaufgaben-Systems. Nach der Testphase wird das Feedback mittels eines standardisierten Fragebogens ausgewertet.

Ausblick Audit 3

Ausblick Audit 3

Geplante Artefakte

- Wireframes
- Klassendiagramm
- Pseudocode
- Prototyp
- Getesteter Prototyp (PoC)

Iteration

Überarbeitete Stakeholder, Erfordernisse und Anforderungen
Angepasstes Domänenmodell
Erweiterte Marktanalyse

Iterationen

Neue Stakeholder:

- Sportinteressierte Schüler:Innen
- Übergewichtige Schüler:Innen
- Untergewichtige Schüler:Innen
- Freundeskreis

Verworfene Stakeholder:

- Männlich, Weiblich
  - Unterschiede waren lediglich leicht bei der Art der Ernährung
  - Keine zwanghafte Drängung in Stereotype (z.B bei der Auswahl der privaten Aufgaben)

Marktanalyse – Neue Erkenntnisse

- Kein digitales System zur Integration im Unterricht für Sekundarschüler gefunden
- Mehrere analoge Medien explizit für Sekundarschüler
  - Manche mit digitalen Bearbeitungsmöglichkeiten
- Wenige weitere digitale Lernkonzepte; existierende sind unzureichend

Artefakte siehe Git:

Stakeholder und Erfordernisse - Iteriert

Domänenmodell Iteriert 2
(Direkter link zum jpg)

Marktanalyse - Iteriert

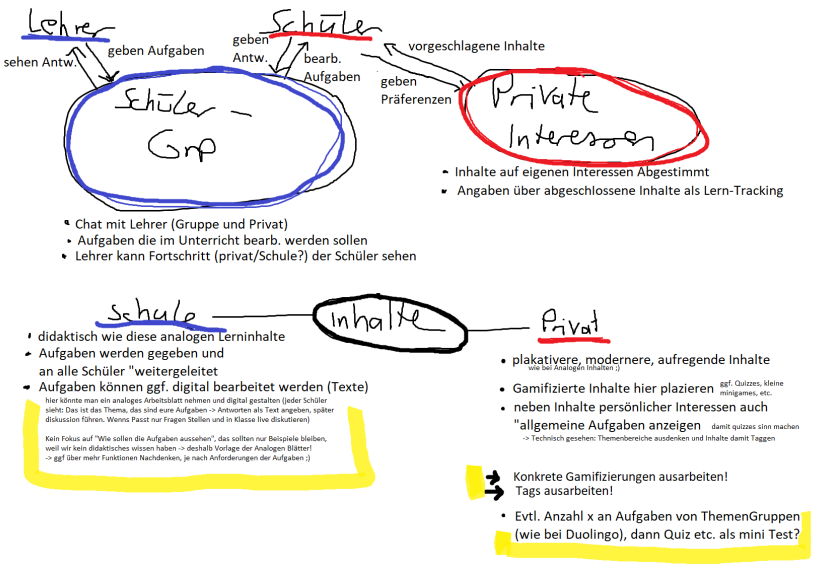
Alte Artefakte finden sie hier

System über Ernährungsbildung - Audit 2

Ernährungsbildung für Jugendliche in der Schule sowie privat.
Gruppe: Sven, Perit, Caro
Github: Repository

Erstes Konzept

Erstes Konzept



Github Link - Erstes Konzept

Erstes Konzept

Das Konzept fing als einfache Mindmap an. Wir hatten bereits die Idee, zwischen schulischer Integration und privatem Gebrauch zu Unterscheiden. Dazu haben wir einige, mögliche Interaktionen der Schüler und Lehrer aufgestellt, welche unsere primären Stakeholder sind. Zusätzlich haben wir erste, mögliche Funktionen (z.B „Chat mit Lehrer“, „Inhalte auf eigene Interessen Abgestimmt“) aufgelistet und diese diskutiert.

Danach haben wir die Inhalte/Aufgaben modelliert, welche ebenfalls zwischen schulischer Integration und privatem Gebrauch unterschieden werden. Die Art der Benutzung, sowie das Umfeld unterscheidet sich nämlich drastisch. Neben dem „gezwungenen“, schulischen Gebrauch, welcher eher fachliches Wissen vermitteln wird (in Anlehnung an diverse analoge Inhalte die bei der Recherche gefunden wurden), sollen die privaten Aufgaben eher realitätsnah und interessant für Jugendliche sein.

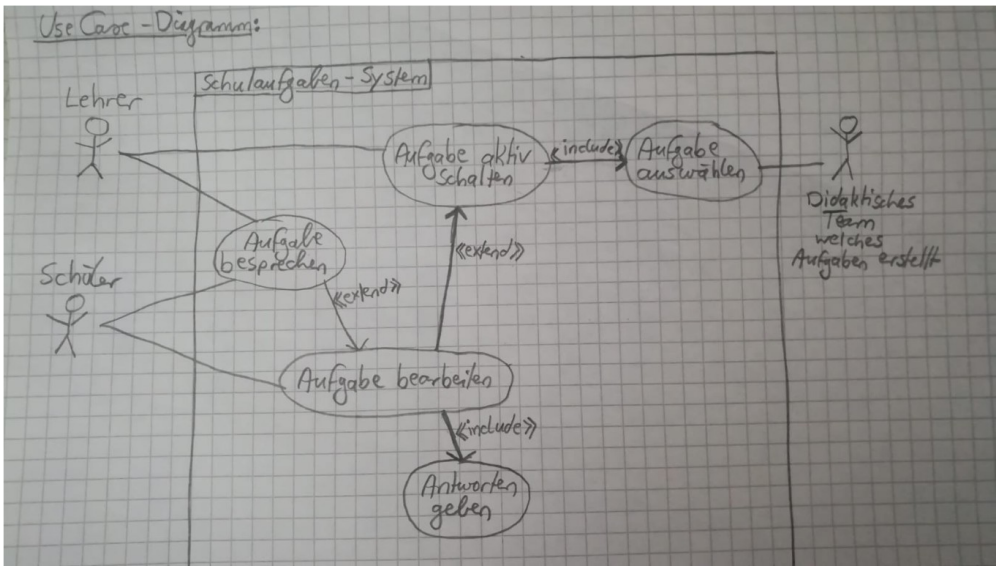
Durch den Feedback-Termin wurden wir angeregt, die Schüler in der Schule für das Thema zu begeistern, damit sie dann privat motiviert sind, sich näher mit Ernährung zu befassen. Dieser Idee stimmen wir zu. Es wird ohne den Kontakt in der Schule mit dem Thema Ernährung schwer, Jugendliche privat dafür zu interessieren und den ersten „Kontakt“ zu diesem Thema aufzubauen.

Nach diesem Brainstorming haben wir den privaten Gebrauch, sowie den schulischen Gebrauch als die zwei wichtigsten Use-Cases bzw. Benutzungsfelder festgelegt. Wobei der schulische Gebrauch, wie bereits erwähnt, zu priorisieren ist.

Github: Erstes Konzept

Use Cases

Use Case - Schulaufgaben-System



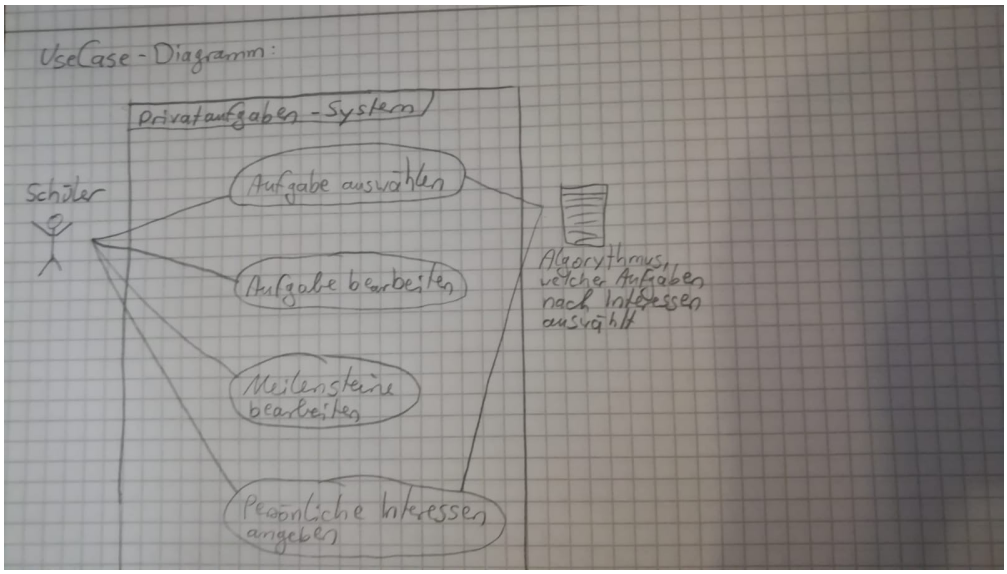
Github Link - Use Case Diagramm - Schulaufgabensystem

Use Case – Schulaufgabensystem:

Der Lehrer schaltet einen Aufgabe im System aktiv. Das beinhaltet, dass diese zuvor ausgewählt werden musste. Die Aufgaben werden theoretisch vom didaktischen Team unserer „Firma“ erstellt. Wie genau dieses Team aussehen soll, wurde noch nicht ausgearbeitet.
Schüler können nun die Aufgabe bearbeiten, sowie Antworten geben. Später kann die Aufgabe besprochen werden. Dies kann direkt im Klassenzimmer geschehen, doch für uns schien es zeitgemäß, eventuell auch eine Möglichkeit für Schüler anzubieten, die aus Gründen nicht in der Klasse sein können. Das wurde auch durch Feedback bestärkt.

(Use Case Diagramm - Schulaufgabensystem)

Use Case - Privataufgaben-System



Github Link - Use Case Diagramm - Privataufgabensystem - Iteriert
Use Case - Diagramm - Privataufgabensystem - Alt

Use Case – Privataufgaben-System:
Hier werden Aufgaben eventuell von einem Algorithmus für den jeweiligen Schüler angezeigt (anhand von angegebenen Interessen). Diese kann der Schüler dann auswählen, um sie zu bearbeiten. Schüler können auch Meilensteine bearbeiten, um ihr Wissen zu testen. Damit der Algorithmus Aufgaben besser anzeigen kann, könne Schüler auch Interessen angeben.
(Use Case - Privataufgabensystem - iteriert)
(Use Case – Privataufgaben-System - Alt)

Es wurden noch weitere Use Case Diagramme erstellt, diese finden sie hier (Weitere Use Case Diagramme)

Funktionsübersicht

Funktionsübersicht

3. Schulaufgaben-Verwaltungssystem

Funktionsname: Aktivierung und Bearbeitung von Schulaufgaben

Zweck: Ermöglicht Lehrern die Aktivierung von didaktisch vorbereiteten Aufgaben für ihre Schüler. Schüler können die Aufgaben einsehen, mit Lehrern besprechen und Lösungen einreichen.

Grundlegender Ablauf für Lehrer:

a. Der Lehrer meldet sich im System an und navigiert zum Aufgabenbereich.
b. Der Lehrer wählt aus einer Liste von vom didaktischen Team vorbereiteten Aufgaben eine aus.
c. Der Lehrer aktiviert die gewählte Aufgabe für die entsprechende Klasse oder Gruppe.
d. Die Aufgabe wird für die Schüler sichtbar und kann von ihnen bearbeitet werden.

Grundlegender Ablauf für Schüler:

a. Der Schüler meldet sich im System an und sieht die für ihn aktiven Aufgaben.
b. Der Schüler wählt eine Aufgabe aus und hat die Möglichkeit, diese mit dem Lehrer zu besprechen, falls Fragen bestehen.
c. Der Schüler bearbeitet die Aufgabe und reicht seine Antwort über das System ein.
d. Der Lehrer kann die eingereichten Antworten einsehen und Feedback geben.

Github Link - Funktionsbeschreibung

Schulaufgaben-Verwaltungssystem:

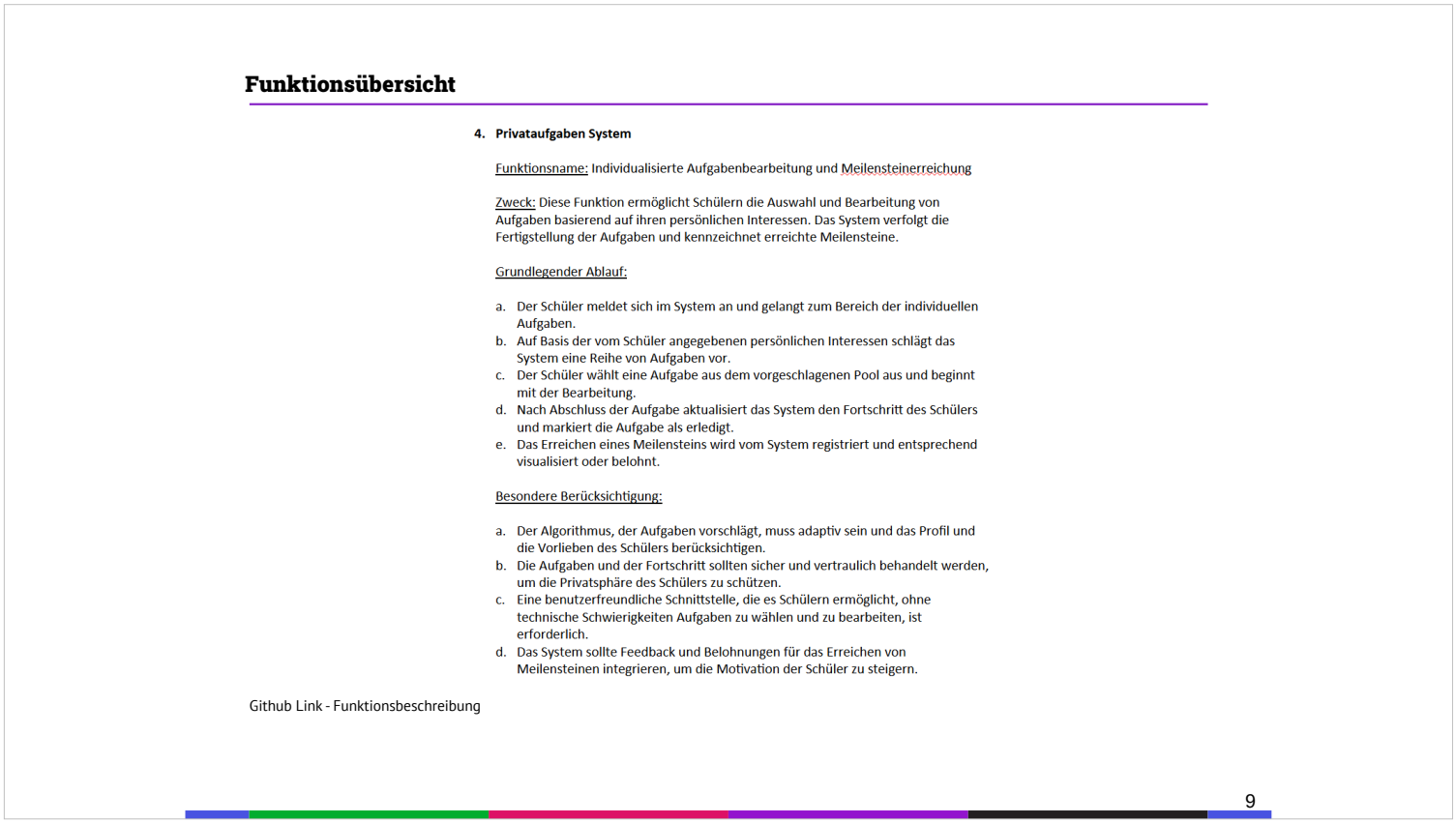
Die Funktionsübersicht wurde mit Blick auf eine intuitive Benutzerführung gestaltet, um Lehrern und Schülern eine einfache Navigation und Interaktion zu ermöglichen.

Die Auswahlprozesse für Lehrer sind darauf ausgerichtet, Zeit zu sparen und die Effizienz zu erhöhen, indem schnell auf vorbereitete Aufgaben zugegriffen werden kann.

Die Sichtbarkeit und Bearbeitbarkeit der Aufgaben für Schüler spiegelt die pädagogische Intention wider, ein selbstgesteuertes Lernen zu fördern.

Die Einbindung von Feedback-Mechanismen trägt zur ständigen Verbesserung der Lehr- und Lernprozesse bei.

Git: Funktionsbeschreibung



Funktionsübersicht

4. **Privataufgaben System**

Funktionsname: Individualisierte Aufgabenbearbeitung und Meilensteinerreichung

Zweck: Diese Funktion ermöglicht Schülern die Auswahl und Bearbeitung von Aufgaben basierend auf ihren persönlichen Interessen. Das System verfolgt die Fertigstellung der Aufgaben und kennzeichnet erreichte Meilensteine.

Grundlegender Ablauf:

a. Der Schüler meldet sich im System an und gelangt zum Bereich der individuellen Aufgaben.
b. Auf Basis der vom Schüler angegebenen persönlichen Interessen schlägt das System eine Reihe von Aufgaben vor.
c. Der Schüler wählt eine Aufgabe aus dem vorgeschlagenen Pool aus und beginnt mit der Bearbeitung.
d. Nach Abschluss der Aufgabe aktualisiert das System den Fortschritt des Schülers und markiert die Aufgabe als erledigt.
e. Das Erreichen eines Meilensteins wird vom System registriert und entsprechend visualisiert oder belohnt.

Besondere Berücksichtigung:

a. Der Algorithmus, der Aufgaben vorschlägt, muss adaptiv sein und das Profil und die Vorlieben des Schülers berücksichtigen.
b. Die Aufgaben und der Fortschritt sollten sicher und vertraulich behandelt werden, um die Privatsphäre des Schülers zu schützen.
c. Eine benutzerfreundliche Schnittstelle, die es Schülern ermöglicht, ohne technische Schwierigkeiten Aufgaben zu wählen und zu bearbeiten, ist erforderlich.
d. Das System sollte Feedback und Belohnungen für das Erreichen von Meilensteinen integrieren, um die Motivation der Schüler zu steigern.

Github Link - Funktionsbeschreibung

Privataufgaben System:

Die Darstellung individueller Aufgaben berücksichtigt die persönlichen Interessen und Bedürfnisse der Schüler, um die Motivation zu steigern und ein personalisiertes Lernerlebnis zu bieten.

Der Ablauf ist so konzipiert, dass er die Selbstständigkeit der Schüler unterstützt und ihnen ermöglicht, eigenverantwortlich Meilensteine zu setzen und zu erreichen.

Die spezielle Berücksichtigung adaptiver Algorithmen stellt sicher, dass die Aufgaben dynamisch an das Lernprofil der Schüler angepasst werden.

Datenschutz und Benutzerfreundlichkeit sind zentrale Elemente des Designs, um eine sichere und barrierefreie Nutzung zu gewährleisten.

Git: Funktionsbeschreibung

Datenstruktur

Datenstruktur

Use Case	Variablenname	Datentyp	Sonstige Merkmale	Beschreibung
Kontoerstellung	email	**String**	unique	
	password	**String**	>8 characters, >= 1 number, >= 1 special character	
	name	**Abstract** (first_name (String), last_name (String)		Name des Schülers
	schoolclass	**Abstract** (number (Int), add (Char))		Schulklasse des Schülers (z.B 8b)
	is_email_verified	**Bool**		Flag ob Email verifiziert wurde
Login	email	**String**		
	password	**String**		
Schulgruppen-Management:				
Gruppe erstellen	classgroup	**Abstract** (name(String), course (String), schoolclass (Abstract (number (Int), add (Char)))		Schulklassen-Gruppe, welche im System erstellt werden soll
	admin	**Abstract** (teacher (name(String), is_teacher (Bool))		Name des Admins (Lehrers der classgroup erstellt hat)
	student_list	**Array/Liste/Dict...** (Abstract (student (name (Abstract (first_name (String)), last_name (String)), is_student (Bool)))		Liste aller Teilnehmenden (Schüler)
Gruppe beitreten	classgroup	**Abstract** (name(Abstract (first_name (String), last_name (String)), course (String), schoolclass (Abstract (number (Int), add (Char)))		Name der Schulklassen-Gruppe, welcher der Schüler beitreten möchte
	student	**Abstract** (name (Abstract (first_name (String), last_name (String)), is_student (Bool)		Name des Schülers, welcher beitreten möchte
Schulaufgaben-system:				
Aufgabe Auswählen	task_list	**Array** (maybe Class „Task"?)		Liste aller Aufgaben
Aufgabe aktiv schalten	task	**Abstract** (maybe Class „Task"?)		Aufgabe, welche die Schüler bearbeiten sollen
	is_active	**Bool**		Flag ob Aufgabe aktiv ist (wenn ja können Schüler sie sehen und bearbeiten)
Aufgabe bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		Ggf. Referenz auf Teilaufgaben
	answer_list	**Dict/Array,...** (student (Abstract (name)): answer (String))		Liste, welche Antworten sammelt (ggf. nur vom Lehrer einsehbar)
	correct_answers	**String**	or Array (String)	Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	is_correct	**Bool**		ggf. Flag ob Antwort des Schülers korrekt ist
Antworten geben	student	**Abstract** (name (Abstract (first_name (String), last_name (String)), is_student (Bool)		
	answer	**String, png,**	different datatypes possible	
	task	**Abstract** (maybe Class „Task"?)		Referenz zur Aufgabe
Aufgabe besprechen	chat	**Abstract** (Array (String), name (Abstract), time (datetime))		chat, um Antworten zu besprechen
	is_open	**Bool**		Flag, ob Chat offen ist
Privataufgaben-system:				
Aufgabe Auswählen	task_list	**Array** (maybe Class „Task_Private"?)		
	task_tag	**String**		Tag, mit dem Aufgaben zu Interessen der Schüler vorgeschlagen werden
Aufgabe bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		
	answer_list_private	**List (String)**		Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	correct_answers	**String**		
	is_correct	**Bool**		ggf. Flag ob Antwort korrekt ist
Meilensteine bearbeiten	sub_task	**Abstract** (maybe Class „Task_Type"?)		Teilaufgaben
	answer_list_private	**List (String)**		Liste mit ggf. richtigen Antworten, welche vom System vorgegeben sind
	correct_answers	**String**		ggf. Flag ob Antwort korrekt ist
	is_correct	**Bool**		
	is_completed	**Bool**		Flag ob Meilenstein vollständig und korrekt bearbeitet wurde
Persönliche Interessen angeben	interest_tag	**String**		Tag, mit dem Aufgaben zu Interessen der Schüler vorgeschlagen werden (z.B Sport, Kochen, Abnehmen, ...)
	interest_list	**Array** (interest_tag)		Liste aller Vorlieben

Github Link - Datenstruktur

Datenstruktur:

Hier haben wir uns erste Gedanken zu einer möglichen Datenstruktur gemacht. Gleichzeitig haben wir uns Gedanken über die potentielle Programmstruktur gemacht. Auffällig ist, dass wir wahrscheinlich eine eigene Klasse für Aufgaben brauchen, sowie bei Benutzern zwischen Lehrern und Schülern unterscheiden müssen. Orientiert haben wir uns an den Use Cases, und was für Daten dort wahrscheinlich gebraucht werden. Eine Erklärung für den Kontext ist ebenfalls enthalten.

Git: Datenstruktur

Risikoanalyse

Risikoanalyse

Potentielle Risiken

- Unterschiede in der digitalen Infrastruktur in deutschen Schulen beziehungsweise mangelhafte digitale Infrastruktur in diesen
- Fehlende Bereitschaft im Lehrpersonal / mangelhafte Qualität des Lehrpersonals
- Probleme mit Plattformkompatibilität
- Unterschiedliche technische Fähigkeiten der Schüler

An dieser Stelle sind potentielle Risiken in der Entwicklung stichpunktartig gennant. Eine ausführlichere Erklärung der Risiken mit Ideen für Lösungsstrategien sind im git unter: Risikoanalyse

Proof of Concept

Proof of Concept

Ziele

- Funktionsfähigkeit vom Schulaufgaben-System sicherstellen
- Plattformkompatibilität testen => funktioniert Anwendung auf verschiedenen Geräten und Plattformen konsistent und effektiv
- Benutzerfreundlichkeit prüfen
- Prüfung der Tutorials bzw. der Anleitungen

Durchführung

Zur Überprüfung der Anwendung als Proof of Concept wird eine interne Testgruppe von Studierenden eingebunden, die die Anwendung in einer simulierten schulischen Umgebung verwenden wird, insbesondere im Kontext des Schulaufgaben-Systems. Nach der Testphase wird das Feedback mittels eines standardisierten Fragebogens ausgewertet.

Hier ist Idee noch einmal genauer formuliert: Zur Überprüfung der Anwendung als Proof of Concept wird eine interne Testgruppe von Studierenden eingebunden, die die Anwendung in einer simulierten schulischen Umgebung verwenden wird, insbesondere im Kontext des Schulaufgaben-Systems. Dabei sollen Teammitglieder oder Kommiliton:innen abwechselnd die Rollen von Lehrer:innen und Schüler:innen übernehmen, um vielfältige Perspektiven und Erfahrungen einzubeziehen. Ein besonderer Fokus liegt darauf, die Plattformkompatibilität durch die Verwendung verschiedener Geräte zu testen. Nach der Testphase wird das Feedback mittels eines standardisierten Fragebogens ausgewertet, um Erkenntnisse über die Benutzerfreundlichkeit, Funktionalität und mögliche Verbesserungsbereiche zu gewinnen.

Git: Proof of Concept

Ausblick Audit 3

Ausblick Audit 3

Geplante Artefakte

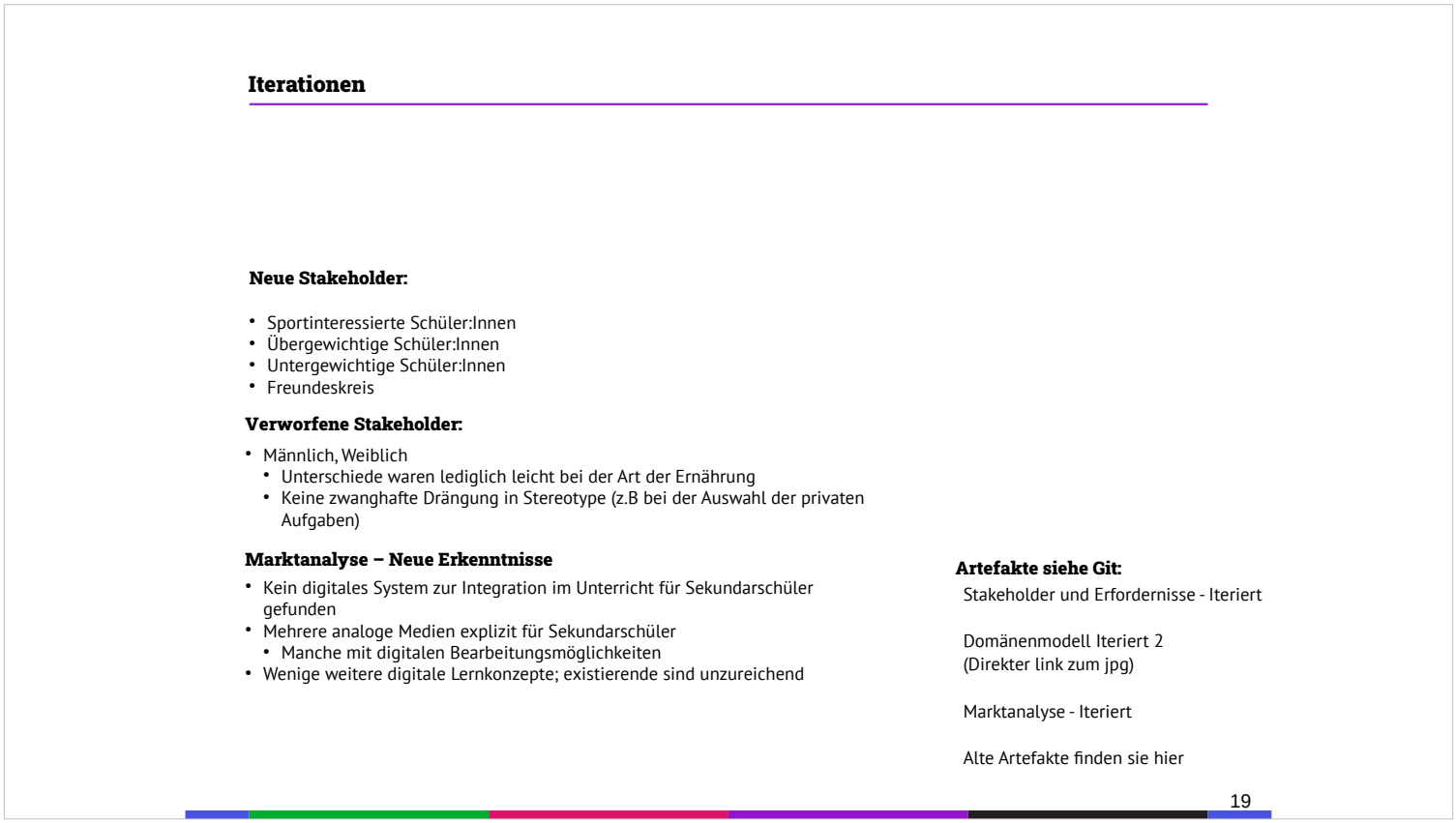
- Wireframes
- Klassendiagramm
- Pseudocode
- Prototyp
- Getesteter Prototyp (PoC)

Ausblick Audit 3:
Geplante Artefakte:
- Wireframes
- Klassendiagramm
- Pseudocode
- Prototyp
- Getesteter Prototyp (PoC)

Git (ganz am Ende): README

Iteration

Überarbeitete Stakeholder, Erfordernisse und Anforderungen
Angepasstes Domänenmodell
Erweiterte Marktanalyse



Stakeholder:

Nach der Kritik aus Audit 1 haben wir die Stakeholder entsprechend überarbeitet. Hinzugefügt wurden die Folgenden:

- Sportinteressierte Schüler:Innen:
Da Schüler eine große Gruppe sind, hat es Sinn gemacht, diese in kleinere Interessensgruppen zu unterscheiden, welche sich auf Ernährung beziehen. Viele Jugendliche fangen an, sich für Sport zu interessieren, jedoch nicht alle. Um die Bedürfnisse dieser Gruppe mit einzubinden, haben wir Sportinteressierte Schüler:innen hinzugefügt.

- Übergewichtige Schüler:Innen:
Laut bereits genannten Quellen und Daten nimmt Übergewicht bei Kindern und Jugendlichen nachweislich zu. Daher erschien es uns passend, die Bedürfnisse und Blickwinkel dieser wichtigen Gruppe hinzuzufügen.

- Untergewichtige Schüler:Innen
Es gibt weniger Untergewichtige als Übergewichtige Jugendliche, doch auch die Ansichten dieser Gruppe sollten beim Thema Ernährung nicht ignoriert werden. Untergewicht kann schnell lebensgefährlich werden. Neben allgemeinen Inhalten hauptsächlich Inhalte zu Übergewicht und Abnehmen, sowie Sport darzustellen, erschien uns für potentiell untergewichtige Jugendliche zu riskant.

- Freundeskreis
Jugendliche lassen sich sehr durch ihren Freundeskreis beeinflussen. Daher sollte dieser durch seinen potentiell großen Einfluss als Stakeholder nicht fehlen.

Wir haben auch zwei potentielle Stakeholder verworfen. Im, für Audit 1, iterierten Domänenmodell (Domänenmodell Iteriert) haben wir männliche und weibliche Schüler:Innen hinzugefügt. Wir haben uns gegen diese Unterscheidung entschieden, da uns einige Unterschiede in der Ernährung nicht aussagekräftig genug erschienen das weitläufige Thema Ernährung so stark im Geschlecht zu unterscheiden. Während uns bei der Bearbeitung immer klarer wurde, wie unser System aussehen sollte, sahen wir keinen Grund, explizit zwischen Geschlechtern zu unterscheiden. Neben festen, wissensbasierten Schulaufgaben soll das System Interessen-bezogene Inhalte vorschlagen. Wir sahen die Gefahr, Schüler zu sehr in Geschlechterrollen zu drängen.

In Zuge dessen haben wir ebenfalls das Domänenmodell iteriert (Domänenmodell Iteriert 2)

Die Ergebnisse der ebenfalls iterierten Marktanalyse finden sie hier (Marktanalyse – Iteriert)
Die alte Marktanalyse finden sie hier (Marktanalyse - Alt)