**Dokumentation**

**‚Zuhtari‘**

**Master Projekt**

Von Sabine Lueck

**Inhalt**

[1. Beschreibung und Ziel der App 4](#_Toc460062373)

[2. App-Übersicht 5](#_Toc460062374)

[3. Appfunktionen 7](#_Toc460062375)

[3.x Funktion: [Name der Funktion] 7](#_Toc460062376)

[Anmelde Bereich: 8](#_Toc460062377)

[3.1 Funktion: Nutzer Registrieren 8](#_Toc460062378)

[3.2 Funktion: Nutzer anmelden 8](#_Toc460062379)

[3.3 Funktion: Nutzer automatisch anmelden 9](#_Toc460062380)

[3.4 Funktion: Nutzer abmelden 9](#_Toc460062381)

[3.5 Funktion: Passwort vergessen 10](#_Toc460062382)

[Menu Bereich: 11](#_Toc460062383)

[3.6 Funktion: Neues Spiel starten 11](#_Toc460062384)

[3.7 Funktion: Spiel weiterspielen 11](#_Toc460062385)

[3.8 Funktion: Spiel Daten Löschen 11](#_Toc460062386)

[Spiel Bereich: 13](#_Toc460062387)

[3.9 Funktion: Monster Züchten 13](#_Toc460062388)

[3.10 Funktion: Zuchtergbnis Vorhersage 13](#_Toc460062389)

[3.11 Funktion: Monster Buch ansehen 14](#_Toc460062390)

[3.12 Funktion: Monster Buch neuer Eintrag 14](#_Toc460062391)

[3.13 Funktion: Monster Buch Update Eintrag 15](#_Toc460062392)

[3.14 Funktion: Monster Overlay Ansehen 15](#_Toc460062393)

[3.15 Funktion: Monster Status Update 15](#_Toc460062394)

[3.16 Funktion: Neues Monster Aufnehmen 16](#_Toc460062395)

[3.17 Funktion: Monster freilassen 16](#_Toc460062396)

[3.18 Funktion: Monster Interaktion 17](#_Toc460062397)

[3.19 Funktion: GPS Aktivierung (Wird nicht Umgesetzt) 17](#_Toc460062398)

[3.20 Funktion: Online Market Eintrag erstellen (verkaufen) 17](#_Toc460062399)

[3.21 Funktion: Online Market Eintrag bearbeiten 18](#_Toc460062400)

[3.22 Funktion: Online Market Eintrag löschen 18](#_Toc460062401)

[3.23 Funktion: Online Market Eintrag erhalten (kaufen) 19](#_Toc460062402)

[3.24 Funktion: Online Market Eintrag einsehen 19](#_Toc460062403)

[3.25 Funktion: Lokale Notification Push “Ei bereit zum schlüpfen” 20](#_Toc460062404)

[3.26 Funktion: Lokale Notification Push “Energie Leiste vill” 20](#_Toc460062405)

[3.27 Funktion: Lokale Notification Push “GPS (xx endeckt)” 20](#_Toc460062406)

[3.28 Funktion: Online Notification Push “Market Item wurde gekauft” 21](#_Toc460062407)

[3.29 Funktion: Neue Monster Bilder erstellen 21](#_Toc460062408)

[3.30 Funktion: Alte lokale Monster Bilder Löschen 21](#_Toc460062409)

[3.31 Funktion: Charackter erstellen 22](#_Toc460062410)

[3.32 Funktion: Charackter bearbeiten 22](#_Toc460062411)

[3.33 Funktion: erstellen lokale DB 22](#_Toc460062412)

[3.34 Funktion: lesen lokale DB 23](#_Toc460062413)

[3.35 Funktion: editieren lokale DB 23](#_Toc460062414)

[3.36 Funktion: erstellen eines Online Eintrages 23](#_Toc460062415)

[3.37 Funktion: lesen online DB 24](#_Toc460062416)

[3.38 Funktion: updaten online DB 24](#_Toc460062417)

[4. “Lesson Learned” und Probleme während der Umsetzung 25](#_Toc460062418)

[5. Ausblick 26](#_Toc460062419)

[6. Timeframe 26](#_Toc460062420)

[7. Unterstützungen 27](#_Toc460062421)

1. Beschreibung und Ziel der App

Das Spiel „Zuhtari“ stellt ein Hybrides Gameplay zwischen Tamagotchi (Virtual Pet Simulator) und einer Zucht-Simulation vor. Die Idee des Spieles, besteht darin dem Spieler neben seinen (virtuellen) täglichen Aufgaben, sich um die Haustiere zu kümmern, dieses auch mit anderen, auf seiner Farm befindlichen Haustieren zu kreuzen, um so neue kombination und unentdeckte Haustiere mit in ihrer Sammlung aufzunehmen.

Die Applikation hat zwei Ziele:

1. Der Spieler soll durch verschiedene kombinatorische Mittel versuchen ein seltenes Monster heranzuzüchten. (Hierbei spielen mehr als nur den Gene der beiden Monster eine Rolle.)
2. Spieler Daten-Sammulung, Online-Markt Entwicklung und Analyse von Aktivitäten.

Hinweis: Die Prototype App wurde in Unity erstellt, um eine schnellere Umsetzung zu sichern (Aufgrund von schon gewonnen Erfahrungen in anderen Projekten). Der eigentliche Plan den Prototypen mit Hilfe von Xamarine zu erstellen wurde auf Grund von unvorhersehbaren Problemen beim Aufsetzen der Entwicklungsumgebung auf einem neuen System und der daraus resultierende hohe Zeitaufwand, verworfen fuers erste.

Server (Tomcat 7 und MySQL) wurde ueber Openshift Online aufgesetzt und laeuft darueber. Fuer den Server wurde IntelliJ IDEA Ultimate verwendet.

2. App-Übersicht

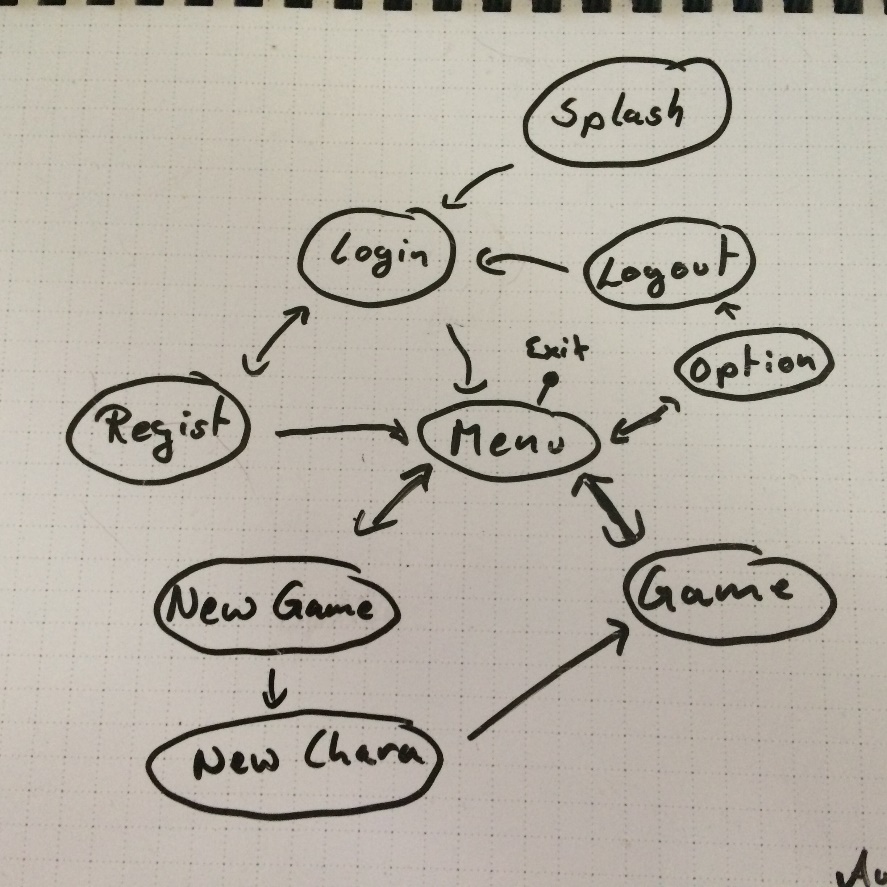
****

Bild: App Screen Flow 1

**Soll-Stand:**

* (Sehr Hoch) Erstellen des Lastenheftes mit den Anforderungen der App sowie Gewichtungen der Funktionen
* (Hoch) Aufsetzen der DB Online (nach einiger Überlegung ist die Online Variante für dieses Endlos Spiel um einiges besser weil der Nutzer/Spieler so nicht an ein Bestimmtes Gerät gebunden ist)
* (Hoch) Online Funktionalitäten: Login/Logoff/Registrieren/Daten speichern-updaten-löschen/abrufen
* (Hoch) Bildgenerierung per Laufzeit (Es werden in der App nur Templates mitgeliefert welche die Funktion später zu neuen Bildern Zusammenfassen soll.)
* (Mittel)Online Markplatz (Besonders wichtig fürs BigData)
* (Mittel)Monster zucht (Kombinieren von verschiedenen Monstern-Genen)
* (Niedrig)Einbau von Pflege der Monster (ala Tamagotchi Gameplay)
* ~~(Niedrig)Auf Monster suche via GPS function~~

**Ist-Stand:**

**Dokumentation:**

Lastenheft und README: erstellt

**Server:**

Server und Online Datenbank (MySQL) wurden aufgesetzt und laufen.

Login/Logout/Registrieren/Versenden von Neuen Passwort Email Funktion wurde Implementiert.

**Client:**

Erster Prototype (Click Dummy) der App ueber Unity wurde aufgesetzt.

Login/Logout/Registrieren/Auto. Anmeldung/Passwort Vergessen wurde Implementiert.

Monster Picture Builder wurde Implementiert.

# 3. Appfunktionen

Hinweis:

Nicht alle Funktionalitäten, welche hier aufgelistet sind, wurden im ersten Entwurf (Prototype) für das Master Projekt umgesetzt. Während einige Funktionen schon fest geplant sind, sind andere noch als CONCEPT im Status bereich vermerkt. Diese Funktionen werden erst noch weiter Recherchiert um zu sehen ob sie in einer späteren Iteration der App mit eingebaut werden.

Der Prioritaeten Status, wurde fuer die erste Prototype Umsetzung nicht als SOLL Stand genommen, sondern als Hinweis auf welche Punkte sich im Verlauf der Entwicklung der App mehr Konzentriert werden soll.

## 3.x Funktion: [Name der Funktion]

|  |  |
| --- | --- |
| Funktionsbeschreibung | Funktionsbeschreibung |
| Priority | Hoch/Mittel/Niedrig |
| Status | Implementiert / Nicht Implementiert |
| Input | Erforderliche Nutzereingaben für die Funktion. |
| Output | Ausgabe bzw. Ereignisse der Funktion |
| Vorbereitungen | Vorrausetzung welche für den Aufruf der Funktion von Nöten sind. |
| Nachbedingungen | Geforderte Nachbedingung welche von der Funktion erfüllt werden sollen. |
| Fehlerfälle | Auffahlende Fehlerfälle welche von der Funktion abgefangen werden sollen. |
| Ausblick | * Moegliche Erweitung nach Implementierung/Folge schritte welche noch gemacht warden muessen. |

## Anmelde Bereich:

Im Anmelde Bereich geht es Um die Funktionalität sich mit einem Nutzer Account an- und abzumelden und diesen zu Registrieren.

## 3.1 Funktion: Nutzer Registrieren

|  |  |
| --- | --- |
| Funktionsbeschreibung | Ein Nutzer kann sich, sofern noch nicht geschehen, auf der Online Datenbank registrieren |
| Priorität | * hoch |
| Status | * Implementiert |
| Input | * Name (Nickname), E-Mail (Ein Account per E-Mail.) , Email bestaetigung, neues Passwort |
| Output | * Rückmeldung über Erfolg und Misserfolg der Registrierung (409 Email / 408 Name schon registeriert) |
| Vorbereitungen | * Nutzer klickt auf Registrien im Anmelde Bild (Weiterleitung auf Reg. Seite) |
| Nachbedingungen | * Nutzer ist registriert * Auto. Weiterleitung auf Login |
| Fehlerfälle | * Falsche Eingaben * Keine Internetverbindung * Zugriff auf DB gescheitert * User Daten schon vorhanden |
| Ausblick | * (Client) Extra Feld fuer Passwort bestätigen * (Server) User bekommt eine Email mit einmaligen Code zum aktivieren des Accounts * (Client) Fontend Design |

## 

## 3.2 Funktion: Nutzer anmelden

|  |  |
| --- | --- |
| Funktionsbeschreibung | Ein Nutzer kann sich beim System anmelden |
| Input | * Name + Passwort |
| Priorität | * hoch |
| Status | * Implementiert |
| Output | * Rückmeldung Erfolg (Auto. Weiterleitung auf Menue) * Misserfolg der Anmeldung (Link fuer Paswort vergessen) |
| Vorbereitungen | * Nutzer muss registriert sein * App starten (es ist kein Token für letzte Anmeldung vorhanden) |
| Nachbedingungen | * Nutzer ist angemeldet * Auto. Weiterleitung auf Login * Anlegen von Login token |
| Fehlerfälle | * Nutzer ist nicht registriert * Zugriff auf DB gescheitert * Falsche/Fehlerhafte Eingabe Daten |
| Ausblick | * (Client) Fontend Design |

## 

## 3.3 Funktion: Nutzer automatisch anmelden

|  |  |
| --- | --- |
| Funktionsbeschreibung | Ein Nutzer kann durch das System automatisch angemeldet werden (Anmelde Daten sind vom erfolgreichen Anmelden Gespeichert.) |
| Priorität | * Mittel |
| Status | * Implementiert |
| Input |  |
| Output | * Rückmeldung das localStorage die Login Daten token gefunden hat |
| Vorbereitungen | * Nutzer muss beim letzten starten der App erfolgreich eingeloggt worden sein (token vorhanden) * App starten |
| Nachbedingungen | * Nutzer ist angemeldet * Automatische Weiterleitung zum Hauptmenü |
| Fehlerfälle | * Token nicht mehr vorhanden * Internet Verbindung muss vorhanden sein |
| Ausblick | * Im Moment erfolgt die Auto. Anmeldung im Login-Screen, dies waere wahrscheinlich besser schon im Splashscreen zu erledigen. |

## 

## 3.4 Funktion: Nutzer abmelden

|  |  |
| --- | --- |
| Funktionsbeschreibung | Abmeldung kann über Menü Punkt Option erfolgen.  Ein Nutzer kann sich vom System abmelden |
| Priorität | * hoch |
| Status | * Implenentiert |
| Input | * Abmelde Button klicken |
| Output | * Rückmeldung Erfolg / Misserfolg der Abmeldung |
| Vorbereitungen | * Nutzer ist angemeldet |
| Nachbedingungen | * Nutzer ist abgemeldet * Entfernen des Nutzerinformationen tokens in der localStorage * Auto. Weiterleitung auf Login Screen |
| Fehlerfälle | * Token nicht gefunden |
| Ausblick | * Information bei Misserfolg der Abmeldung * (Client) Fontend Design (Button) |

## 3.5 Funktion: Passwort vergessen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Nuzter kann nach fehlgeschlagener Anmeldung, eine Anfrage fuer ein neues Passwort schicken. |
| Priorität | * Mittel |
| Status | * Implementiert |
| Input | * Input: Email * Button klick |
| Output | * Rückmeldung Erfolg / Misserfolg der Abmeldung |
| Vorbereitungen | * Input Felder wurden alle korrekt befuehlt * Email ist registriert |
| Nachbedingungen | * (Server) Account Daten wurden mit neuen Passwort geupdated * (Server) Email Versendung mit neuen Passwort * (Client)Anzeige ueber Versendung des neuen Passwortes |
| Fehlerfälle | * Email nicht gefunden im System * Keine Internet Verbindung * DB Daten Fehler |
| Ausblick | * (Server) Anpassung des Email Designs * (Client) Fontend Design |

## Menu Bereich:

Im Menü Bereich kann der Nutzer das Spiel Neu bzw. Weiterspielen, oder die App einfach Beenden.

## 3.6 Funktion: Neues Spiel starten

|  |  |
| --- | --- |
| Funktionsbeschreibung | Ein Nutzer kann ein neues Spiel vom Hauptmenü aus starten |
| Priorität | * hoch |
| Status | * Klick Dummy Screen Vorhanden |
| Input | * Klick auf Neues Spiel |
| Output | * Hinweis das (wenn vorhanden) Alte Spieldaten gelöscht werden [Aufruf 4.7 Funktion Spieldaten Löschen] |
| Vorbereitungen | * Nutzer ist angemeldet |
| Nachbedingungen | * Auto. Weiterleitung auf Character-Erstellungs Seite, sobald Hinweis bestätigt wurde. |
| Fehlerfälle | * Fehler beim erstellen der lokalen DB * Fehler bei der Weiterleitung |
| Ausblick | * (Client) Funktionalitaet einbauen * (Client) Hinweis falls Avatar schon vorhanden ist * (Client) Fontend Design |

## 3.7 Funktion: Spiel weiterspielen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Ein Nutzer kann ein schon gestartetes Spiel vom Hauptmenü aus fortserten |
| Priorität | * Hoch |
| Status | * Klick Dummy Screen Vorhanden |
| Input | * Klick auf Spiel fortzenen |
| Output | * Lade Bildschirm |
| Vorbereitungen | * Zugriff auf lokale DB auf dem mobilen gerät muss vorhanden sein * Spielstand muss vorhanden sein |
| Nachbedingungen | * Auto. Weiterleitung auf Spiel Seite, sobald alle Daten geladen wurden * Laden der Spiel Daten |
| Fehlerfälle | * DB wurde nicht gefunden * Weiterleitung schlägt fehl |
| Ausblick | * Funktionalitaet einbauen * (Client) Fontend Design |

## 

## 3.8 Funktion: Spiel Daten Löschen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Spieldaten löschen. Hierbei diwrd ein der Avatar in der Datenbank inaktiv gesetzt. |
| Priorität | * Mittel |
| Status | * Token in der DB vorhanden |
| Input | * Bestätigung der Abfrage ob Nutzer sicher ist das dieser dieser seinen alten Spielstand überschreiben will |
| Output |  |
| Vorbereitungen | * Nutzer will ein Neues Spiel Starten. |
| Nachbedingungen | * Rückmeldung an das System das Löschung erfolgreich war |
| Fehlerfälle | * Fehler während der Löschung der Daten. |
| Ausblick | * (Client)Funktionalitaet einbauen * (Server) Avatar wird nach x- aus dem System geloescht * (Server) Avatar Daten token auf Inaktiv setzen |

## Spiel Bereich:

## 3.9 Funktion: Monster Züchten

|  |  |
| --- | --- |
| Funktionsbeschreibung | Nutzer wählt zwei Monster und Kombiniert diese. |
| Priorität | * Hoch |
| Status | * Klick Dummy Screen Vorhanden |
| Input | * Auswahl von zwei Monsters * Bestätigung der Kombination |
| Output | * Anzeige von gelegten Eiern [Zufall 1-3 (3 Rare)] * Anzeige der Dauer des Schlüpfens |
| Vorbereitungen | * Mind. zwei Monster muessen vorhanden sein. * Aufruf der [Funktion: Zucht vorhersage 4.9] |
| Nachbedingungen | * Update des Zucht Screens * Neuer Eintrag in das Farm Buch |
| Fehlerfälle | * Keine Monster zur Kombination vorhanden * Fehler beim Speichern der neuen Monster Daten |
| Ausblick | * (Server) Neu und Updaten von Daten * (Client) Funktionalitaet des Zuchtprozesses * (Client) Fontend Design |

## 

## 3.10 Funktion: Zuchtergbnis Vorhersage

|  |  |
| --- | --- |
| Funktionsbeschreibung | Anzeige von Vier Möglichen Monstern, welche die Kombination von zwei Eltern Monstern ergeben würden   * Vorhersage beschränkt sich auf 5 Versuche! |
| Priorität | * Mittel |
| Status | * Klick Dummy Screen Vorhanden |
| Input | * Klick auf Vorhersage (Standard Max. 5 Versuche, neuer Versuch alle 10min.) |
| Output | * Anzeige der Möglichen Ergebnisse (Adult Version) * Anzahl der noch möglichen Vorhersagen |
| Vorbereitungen | * Eltern Monster wurden schon ausgewählt |
| Nachbedingungen | * Adult Version Bilder werden mit in das Monster Buch aufgenommen (als Gesehen markiert) * Senden der neuen/update Monster Daten an Server * Senden des genutzten Tokens an Server |
| Fehlerfälle | * Fehler während des Erstellens der Bilder * Fehler beim Monster Buch eintrag |
| Ausblick | * (Server) Updaten/Erstellen der Daten * (Client) Updaten der temp. lokalen Daten * (Server) Scheduler fuer neue Token der Vorhersage * (Client) Fontend Design |

## 

## 3.11 Funktion: Monster Buch ansehen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Daten des Monster Buches anzeigen. |
| Priorität | * Hoch |
| Status | * Klick Dummy Overlay des Monster Buches Vorhanden |
| Input | * Klick Auf Monster Buch im Player Overlay auf dem Farm Screen |
| Output | * Anzeige Monster Buch Informations Layers |
| Vorbereitungen | * Player Overlay muss geoeffnet sein. |
| Nachbedingungen |  |
| Fehlerfälle | * Fehler beim lessen der temp. lokalen Daten |
| Ausblick | * (Client) Funktionaliaet der Anzeige der Monster Daten * (Client) Fontend Design |

## 

## 3.12 Funktion: Monster Buch neuer Eintrag

|  |  |
| --- | --- |
| Funktionsbeschreibung | Sobald ein neues Monster welches noch nicht Dokumentiert wurde, durch Zucht Vorhersage angezeigt wird, bzw. Nach schluepfen eines Eies, wird es in die Online & temp. lokale Daten im Monster Buch Abschnitt aufgenommen. |
| Priorität | Hoch |
| Status | Nicht implementiert |
| Input |  |
| Output | * Notiz das Monster Buch ein neuen Eintrag erhalten hat |
| Vorbereitungen | * Aufruf durch [Funktion 4.9] bzw. [Funkton 4.18] * Kontrolle ob Monster Eintrag schon existiert |
| Nachbedingungen | * Monster Buch hat neuen Eintrag in der DB |
| Fehlerfälle | * Eintrag in der DB fehlgeschlagen |
| Ausblick | * Implementieren |

## 

## 3.13 Funktion: Monster Buch Update Eintrag

|  |  |
| --- | --- |
| Funktionsbeschreibung | Update der Monster Daten im Monster Buch. |
| Priorität | * Hoch |
| Status | * nicht implementiert |
| Input |  |
| Output | * Notiz das Daten geupdated wurden |
| Vorbereitungen | * Kontrolle ob Monster Daten schon im Monster Buch existieren |
| Nachbedingungen | * (Client) Notification/Hinweis Anzeige * (Server) Update DB Eintrag * (Client) Update temp. local Data |
| Fehlerfälle | * Monster Daten konnten nicht gefunden werden * Fehler beim Zugriff der DB |
| Ausblick | * Implementieren |

## 

## 3.14 Funktion: Monster Overlay Ansehen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Informationen des Einzelnen Monsters auf der Farm einsehen |
| Priorität | * Hoch |
| Status | * Klick Dummy Overlay Vorhanden |
| Input | * Klick auf Monster |
| Output | * Anzeige Monster Info Layer |
| Vorbereitungen | * Laden der Monster Daten aus lokalen Daten |
| Nachbedingungen |  |
| Fehlerfälle | * Fehler beim Zugriff auf DB |
| Ausblick | * (Client) Funktionalitaet * (Client) Frontend Design |

## 

## 3.15 Funktion: Monster Status Update

|  |  |
| --- | --- |
| Funktionsbeschreibung | Update der Monster Daten |
| Priorität | * Mittel |
| Status | * Nicht Implentiert |
| Input |  |
| Output | * Notiz das Daten geupdated wurden |
| Vorbereitungen | * DB Monster Datenaktualiseren |
| Nachbedingungen | * Bei Stage Erhöhung wird Monster Bild auf der Spiel Seite ausgetauscht. * Updaten der Daten anzeige im Frontend * Senden der Daten an lokalen und online DB |
| Fehlerfälle | * Fehler beim Zugriff auf DB |
| Ausblick | * Implementieren |

## 

## 3.16 Funktion: Neues Monster Aufnehmen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Neues Monster wird der Kollektion auf der Farm hinzugefügt.  Meist erfolgt diese Funktion nachdem ein Ei geschlüpft ist.   * Idee neue Monster koennen auch bei Mini Spiele erworben warden (Concept) |
| Priorität | * hoch |
| Status | * nicht Implementiert |
| Input | * Bestätigung des schlüfen des Eies * Bestaetigen der Aufnahme des neuen Monsters (Concept) |
| Output | * Hinzufügen des neuen Monsters auf der Farm und dieses Anzeigen |
| Vorbereitungen |  |
| Nachbedingungen | * Eintrag des neuen Monsters ins Farmbuch * Neuer Eintrag in die DB online und local. |
| Fehlerfälle | * Fehler beim Zugriff auf DB |
| Ausblick | * Implementieren |

## 

## 3.17 Funktion: Monster freilassen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Der Nutzer kann sich dafür entscheiden ein Monster freizulassen. Das heisst das es von der Farm entfernt wird und aus dem Farmbuch als solches vermerkt wird. |
| Priorität | * mittel |
| Status | * nicht Implementiert |
| Input | * Bestätigen des freilassens des Monsters |
| Output | * Notiz das Monster freigelassen wurde und Daten geupdated wurden. * Monster von der Spiele Seite entfernen |
| Vorbereitungen | * Klick auf Freilassen Button |
| Nachbedingungen | * Update der Monster Daten * Update der Spiele Seite |
| Fehlerfälle |  |
| Ausblick | * Implementieren |

## 

## 3.18 Funktion: Monster Interaktion

|  |  |
| --- | --- |
| Funktionsbeschreibung | Mit den einzelnen Monsters agieren via Gesture. (Streicheln, Füttern) |
| Priorität | * niedrig |
| Status | * nicht implementiert |
| Input | * Gesture |
| Output | * Affection Punkte Sammeln |
| Vorbereitungen | * Monster bekommt eine Notiz welches den Spieler signalisiert das es aufmerksamkeit möchte * Anklick des Monsters |
| Nachbedingungen | * Notiz verschwindet * Monster Daten werden geupdated |
| Fehlerfälle |  |
| Ausblick | * Implementieren |

## 

## 3.19 Funktion: GPS Aktivierung (Wird nicht Umgesetzt)

|  |  |
| --- | --- |
| Funktionsbeschreibung | Aktivieren der GPS Funktion des Gerätes. (Schritte zählen des Spielers und ab einer bestimmten Anzahl wird der Zufall ein Monster bzw. Item gefunden) |
| Priorität | * ~~Niedrig~~ |
| Status | * Concept |
| Input | * Monster auf die “Wiese” schicken. |
| Output | * Monster ist auf der Spiel Seite abwesend |
| Vorbereitungen | * GPS aktivieren |
| Nachbedingungen | * Zeitanzeige für wieder kehren des Monsters vom Spaziergang * Monster ist für die Dauer des Spazierganges nicht verfügbar |
| Fehlerfälle |  |

## 

## 3.20 Funktion: Online Market Eintrag erstellen (verkaufen)

|  |  |
| --- | --- |
| Funktionsbeschreibung | Der Nutzer hat die möglichkeit eines seiner gezüchteten Monsters auf den Markt zum verkauf anzubieten. Dies gilt auch für Items welcher gefunden werden. |
| Priorität | * mittel |
| Status | * nicht imeplentiert * ClickDummy Marketplace Overlay Implimentiert |
| Input | * Klick auf Verkaufen Button im Online Market Layer Verkaufen. * Eintrag alle nötigen Daten zum Verkauf   + Dauer des Verkaufes   + Preis |
| Output | * Notiz das der Butzer etwas auf den Online Market Platz angelegt hat |
| Vorbereitungen | * Prüfen der Online Verbindung |
| Nachbedingungen | * Daten an den Server senden * Item/Monster vom Nutzer entfernen, Daten Updaten   + Bei ablauf der Verkauf Zeit wird Monster/Item dem Nutzer einfach wieder hinzugefügt * Bestätigung des Servers abwarten das alles geklappt hat |
| Fehlerfälle | * Fehler bei Verbindungsaufbau * Fehler beim Update der Daten auf dem Client * DB Fehler |
| Ausblick | * Implementieren der Funktion * (Client) Frontend Design |

## 

## 3.21 Funktion: Online Market Eintrag bearbeiten

|  |  |
| --- | --- |
| Funktionsbeschreibung | Der Nutzer hat die möglichkeit die Einträge im Market, welche selber angelegt wurden auch zu bearbeiten. |
| Priorität | * niedrig |
| Status |  |
| Input | * Klick auf eigen erstellte Market Eintrag zum Bearbeiten * Eingabe der Änderung in den vorgesehenen Feldern |
| Output | * Notiz das Änderungen gespeichert wurden |
| Vorbereitungen | * Eintrag muss vorhanden sein * Daten werden aus dem Server geladen |
| Nachbedingungen | * Daten an den Server schicken. * Bestätigung des Servers abwarten das alles geklappt hat |
| Fehlerfälle | * Fehler bei Verbindungsaufbau * Fehler beim Update der Daten auf dem Client * DB Fehler |
| Ausblick | * Implementieren * (Client) Frontend Design |

## 

## 3.22 Funktion: Online Market Eintrag löschen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Der Nutzer kann seinen Market Eintrag Löschen bzw. Den Verkauf seiner Ware abbrechen und somit für andere nicht mehr ersichtlich machen. |
| Priorität | * niedrig |
| Status | * MarketOverlay Click Dummy Implementiert * Nicht Implementieren |
| Input | * Klick auf Abbrechen Button im Online Market Verkauf Item Layer |
| Output | * Notiz das der Verkauf abgebrochen wurde |
| Vorbereitungen | * Laden der Daten vom Server |
| Nachbedingungen | * Update der Server Daten * Ware wird dem Nutzer wieder freigschaltet.   + Zu beachten ist das die Daten eines Verkauften Produktes in der lokalen DB noch vorhanden sind, jedoch einen besonderen ticker haben wodurch sie zum normalen interagieren nicht zur verfügung stehen |
| Fehlerfälle | * Fehler bei Verbindungsaufbau * Fehler beim Update der Daten auf dem Client * DB Fehler |
| Ausblick | * Implementieren * (Client) Frontend Design |

## 3.23 Funktion: Online Market Eintrag erhalten (kaufen)

|  |  |
| --- | --- |
| Funktionsbeschreibung | Der Nutzer hat eine Ware aus dem Market gekauft. |
| Priorität | * niedrig |
| Status | * nicht Implementieren |
| Input | * Klick auf Kaufen der Ware im Market Layer |
| Output | * Notiz das die Ware der Farm hinzugefügt wurde   + Item im Inventory   + Monster auf Farm Seite |
| Vorbereitungen | * Laden der Server Daten |
| Nachbedingungen | * Neue Einträge in die Lokale DB von Käufer, Ware zum Inventar7Farm hinzufügen * Senden des Server an Client welcher Ware zum Verkauf angeboten hat. |
| Fehlerfälle | * Fehler bei Verbindungsaufbau * Fehler beim Update der Daten auf dem Client * DB Fehler |
| Ausblick | * Implementieren |

## 

## 3.24 Funktion: Online Market Eintrag einsehen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Nutzer kann sich die Informationen der Ware welche zum Verkauf angeboten wird in Detail ansehen. |
| Priorität | * niedrig |
| Status | * nicht Implementiert |
| Input | * Klick auf Ware in der Liste |
| Output | * Layer mit informationen der Ware |
| Vorbereitungen | * Laden der Daten vom Server für die Detailierte Anzeige |
| Nachbedingungen |  |
| Fehlerfälle | * Fehler bei Verbindungsaufbau * Fehler beim Update der Daten auf dem Client * DB Fehler |
| Ausblick | * Implementieren |

## 3.25 Funktion: Lokale Notification Push “Ei bereit zum schlüpfen”

|  |  |
| --- | --- |
| Funktionsbeschreibung | Senden einer lokalen Notification push sobald ein Ei bereit zum schlüpfen. |
| Priorität | * niedrig |
| Status | * nicht Implementiert |
| Input | * Klick auf Notification |
| Output | * Notification Push mit der Info “Monster schlüpft” |
| Vorbereitungen | * Brüt phase des Eies ist abgelaufen * App läuft im hintergrund |
| Nachbedingungen | * Öffnen der App auf die Zucht Seite |
| Fehlerfälle | * Timer probleme zum triggern der Notification |
| Ausblick | * Implementieren |

## 3.26 Funktion: Lokale Notification Push “Energie Leiste vill”

|  |  |
| --- | --- |
| Funktionsbeschreibung | Senden einer lokalen Notification push sobald der Energy Balken des Charackters wieder voll ist. |
| Priorität | * niedrig |
| Status | * nicht Implementiert |
| Input | * Klick auf Notification |
| Output | * Notification Push mit der Info “Energy gefüllt” |
| Vorbereitungen | * Balken ist wieder gefüllt * App läuft im hintergrund |
| Nachbedingungen | * Öffnen der App auf die Farm Seite |
| Fehlerfälle | * Timer probleme zum triggern der Notification |
| Ausblick | * Implementieren |

## 

## 3.27 Funktion: Lokale Notification Push “GPS (xx endeckt)”

|  |  |
| --- | --- |
| Funktionsbeschreibung | Senden einer lokalen Notification push sobald etwas beim spaziergang des Monsters endeckt wurde. |
| Priorität | * niedrig |
| Status | * nicht Implementiert |
| Input | * Klick auf notification |
| Output | * Notification Push mit der Info “Monster hat etwas gefnden!” |
| Vorbereitungen | * App läuft im hintergrund |
| Nachbedingungen | * Öffnen der App auf die Farm Seite |
| Fehlerfälle | * Timer probleme zum triggern der Notification |
| Ausblick | * Implementieren |

## 

## 3.28 Funktion: Online Notification Push “Market Item wurde gekauft”

|  |  |
| --- | --- |
| Funktionsbeschreibung | Senden einer online Notification push vom Server an client sobald eine im Market gestellte Ware vom client Verkauft wurde. |
| Priorität | * niedrig |
| Status | * nicht Implentiert |
| Input | * Klick auf Notification |
| Output | * Notification Push mit der Info “Ihre Ware wurde gekauft.” |
| Vorbereitungen | * Verbindung zum Internet ist vorhanden. * App läuft |
| Nachbedingungen | * Öffnen der App auf die Zucht Seite |
| Fehlerfälle | * Timer probleme zum triggern der Notification |
| Ausblick | * Implementieren |

## 

Bild erstellen Via Server und erstelltes Bild an User zurueckschicken

Bilder werden auf den Persistenten Daten Speicher von Openshift App erstellt.

Test Link fuer Bild anzeigen:

<https://zuhtariserver-fhemp.rhcloud.com/api/db/userImageShowTest> <-- Testbild welches erstellt wurde (als erstes aufrufen um zu sehen dass das erstellte bild noch nicht vorhanden ist)

Test Link fuer Bild erstellen

<https://zuhtariserver-fhemp.rhcloud.com/api/db/userImageTest> 🡨Link ueber browser aufrufen (vor wird erstellt: combinedTest.png )

## 3.29 Funktion: Neue Monster Bilder erstellen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Die eigentlichen Bilder der Monster werden per Laufzeit aus verschiedenen Bildern zusammen gefasst. |
| Priorität | * hoch |
| Status | * test Function im Server vorhanden |
| Input |  |
| Output | * Anzeige des Erstellten Bildes |
| Vorbereitungen | * Übergabe von Kombination und Farbwerte für das zu erstellende Bild des Monsters |
| Nachbedingungen | * Speichern des Bildes im lokalen Speicher |
| Fehlerfälle | * Fehlende Daten zum erstellen des Bildes. * Speicher voll * Speichern fehlgeschlagen |
| Ausblick | * (Client) Ersetzen der Placeholder Bilder * Ausbauen der Server function um argumente von client zu erhalten und nach diesen die bilder zu erstellen |

## 

## 3.30 Funktion: Alte lokale Monster Bilder Löschen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Löschen aller lokal erstellten Bilder der App. |
| Priorität | * niedrig |
| Status | * Haengt vom Bild erstellen Concept ab |
| Input | * Zumstimmung des Löschens |
| Output | * Notiz das Bilder gelöscht wurden |
| Vorbereitungen | * Aufruf der Alte daten Löschen Funktion |
| Nachbedingungen |  |
| Fehlerfälle | * Lokaler Speicher kann nicht gelesen werden * Fehler während des Löschens |
| Ausblick | * Implementieren |

## 

## 3.31 Funktion: Charackter erstellen

|  |  |
| --- | --- |
| Funktionsbeschreibung | Nutzer legt einen neuen Charackter für das Spiel an, welcher als Avatar für den Nutzer im Game agiert. |
| Priorität | * mittel |
| Status | * Click Dummy Implemtiert |
| Input | * Angabe von Daten: Name, Gender, GB etc. * Weiter via Button klick |
| Output | * Anzeige des Avatars je nach wahl der vorgegebenen Optionen |
| Vorbereitungen | * Nutzer startet neues Spiel |
| Nachbedingungen | * Auto. Weiterleitung auf Farm Seite nach erstellen * Character wird in die lokale DB aufgenommen |
| Fehlerfälle | * Option ist Fehlerhaft * Name ist Blacklisted |

## 

## 3.32 Funktion: Charackter bearbeiten

|  |  |
| --- | --- |
| Funktionsbeschreibung | Während des Spielens ist es möglich seinen Charackter neu Auszustatten. |
| Priorität | * niedrig |
| Status | * concept |
| Input | * Wahl des neuen Ausstatungs-Gegenstandes |
| Output | * Anzeige des geänderten Avatars |
| Vorbereitungen | * Öffnen des Avatar Info Bilders |
| Nachbedingungen | * Aktualisieren des Avatar DB eintrages |
| Fehlerfälle | * DB Fehler |
| Ausblick |  |

## 

## 3.33 Funktion: erstellen lokale DB

|  |  |
| --- | --- |
| Funktionsbeschreibung | Erstellen der temp. lokalen DB im mobilen Speicher (json). |
| Priorität | * hoch |
| Status |  |
| Input |  |
| Output |  |
| Vorbereitungen | * App wird das erste mal gestartet |
| Nachbedingungen | * Erfolgreiches erstellen der lokalen DB |
| Fehlerfälle | * DB ist schon vorhanden * Fehler während des erstellen der DB |

## 3.34 Funktion: lesen lokale DB

|  |  |
| --- | --- |
| Funktionsbeschreibung | Die App soll die Daten der lokalen DB auslesen und wiedergeben können. |
| Prioritäten | * hoch |
| Status |  |
| Input |  |
| Output | * Anzeige der geforderten Daten auf der View. |
| Vorbereitungen | * Vorhanden sein der lokalen DB * Abfrage der Daten |
| Nachbedingungen | * Wiedergabe der ausgelesenen Daten |
| Fehlerfälle | * Daten sind Fehlerhaft * DB ist noch nicht vorhanden |

## 3.35 Funktion: editieren lokale DB

|  |  |
| --- | --- |
| Funktionsbeschreibung | Die App soll die Daten der lokalen DB bearbeiten (Updaten) |
| Priorität | * hoch |
| Status |  |
| Input | * Zu Speichernde Informationen |
| Output | * Schreiben der Daten in in die DB |
| Vorbereitungen | * Vorhanden sein der lokalen DB * Überprüfen der Daten |
| Nachbedingungen | * Erfolgreiche Schreiben in die DB ohne Fehlermeldung |
| Fehlerfälle | * Daten sind Fehlerhaft * DB ist noch nicht vorhanden |

## 3.36 Funktion: erstellen eines Online Eintrages

|  |  |
| --- | --- |
| Funktionsbeschreibung | Erstellen eines neuen Online Eintrages in die Online DB auf dem Server. |
| Priorität | * hoch |
| Status | * implementiert (Registrieren des Users) |
| Input |  |
| Output |  |
| Vorbereitungen | * Nutzer hat:   + den Online Market aufgerufen   + Neues Monster ausgebrütet   + Charackter erstellt   + Neuen Account erstellt |
| Nachbedingungen | * Daten wurden erfolgreich an den Server geschickt |
| Fehlerfälle | * Fehler mit den Daten * Keine Internet Verbindung * Übertragung Fehlerhaft |
| Ausblick | * (Server) Avatar/Monster/Market etc. |

## 3.37 Funktion: lesen online DB

|  |  |
| --- | --- |
| Funktionsbeschreibung | Die App soll die Daten der online DB abfragen und wiedergeben |
| Priorität | * hoch |
| Status | * Implementiert (User) |
| Input |  |
| Output | * Anzeige der Abgerufenen Daten im View |
| Vorbereitungen | * Anfrage an Server |
| Nachbedingungen | * Wiedergabe der ausgelesenen Daten |
| Fehlerfälle | * Daten sind Fehlerhaft * Keine Internet Verbindung |
| Ausblick | * (Server) Avatar/Monster/Market etc. |

## 3.38 Funktion: updaten online DB

|  |  |
| --- | --- |
| Funktionsbeschreibung | Die App soll die Daten der Online DB updaten |
| Priorität | * hoch |
| Status | * Implementiert (USER) |
| Input |  |
| Output |  |
| Vorbereitungen | * Überprüfen ob Daten welche geupdated werden sollen ok sind |
| Nachbedingungen | * Erfolgreiche Updaten der Daten |
| Fehlerfälle | * Daten sind Fehlerhaft * DB ist noch nicht vorhanden |
| Ausblick | * (Server) Avatar/Monster/Market etc. |

# 4. “Lesson Learned” und Probleme während der Umsetzung

**Big Data Datenbank:**

Es benötigte einiges an Einlesen und Recherche, um sich ein besseres Bild zu machen wenn es um das Thema Big Data ging und welche Optionen für Datenbanken vorhanden sind.

Beste Option waere die MongoDB, welche die JSON dateien der User Sammelt, und der Server diese Spaeter auswerten wird.

*Ueberlegung*: Auswertung der Daten Entweder Ueber eine JSP Webpage fuer den Admin zu haben Oder eine App (Entweder im Spiel Zuhtari oder extra neue App) welche die Statistiken des Spieles wieder gibt.

**Server und Client Entwicklung:**

Erste eigene Umsetzung einer Client – Server App, im MC2 Projekt wurde dieser Part von anderen uebernommen, wodurch auch dieser Part mehr Zeit gekostet hat.

Das MC2 Projekt (Server), diente hier als Recherche, zum Nachschaun wie dort der Server Code umgesetzt wurde.

Waehrend schon Erfahrung vorhanden war mit der Unity Entwicklungsumgebung, so war die Kommunikation mit einem Server von einer mit Unity erstellten App, ein noch unbekannter Faktor.

**Mangel an Erfahrung:**

Wie in jedem neuen Projekt mit geringer bzw. Keiner Erfahrung braucht es einige Zeit vom Entwickler sich der herausforderung zu stellen bzw. für die Recherche und das Testen einzubringen.

**Probleme beim aufsetzen der Entwicklungsumgebung:**

Unerwartete Probleme bei dem Aufsetzen der Xamarine Umgebung auf einem neuen System hat dazu gefuehrt das man sich fuer ein anderes Entwickler Tool (Unity 3D Personal Edition) umentscheiden musste.

# 5. Ausblick

Anfang des SS 17, sollte das Gameplay laufen und die Big Data Datenbank aufgesetzt, um das Projekt weiter fuer die angestrebte Master Arbeit zu verwenden.

1. Implementieren und Umsetzen der Ausblick Sektionen der gelisteten Funktionen
2. Sammeln von Testdaten durch Testusers
3. Erstellen und Ersetzen der Grafiken, Frontend Design
4. Erstellen der BigData DB
5. Dokumentation UpToDate halten
6. Code Refectoring
7. Ueberabreiten des Conceptes (Erstellen einer UML)
8. Erweiterung der Online DB

# 6. Timeframe

Der Timeframe fuer das Master Projekt ist nicht genau und eher als geschaetzter Zeitrahmen da.

(Workday 1day = 8hours)

1. Umsetzung Clickdummy: ~ca. 12h
2. Concept und Research: ~ca. 64h
3. Aufsetzen von Server und Entwicklungsumgebung: ~ca. 40h
4. Umsetzung von Funktionen (Server und Client): ~ca. 66h+
5. Dokumentation und Testen: ~ca. 32h+

# 7. Unterstützungen

Unterstuetzung bei der Erstellung von Icons und Bildern:

Emeline Huo Yue Chang

Unterstuetzung bei Findung und Beseitung von Server Bug beim Aufesetzen:

Bernd Schmidt