	<p><b>Pflichtenheft Tamagotchi “Ohsom“</b>  <b>Thema: Modul 226</b></p>
	<p><b>David Roth</b>  <b>Natalie Schumacher</b></p>

Projektleiter  
Roger Zaugg

Pratteln, Januar 2014

## Dokumentenmanagement

Version/Status: s. unten

Datum: 26.02.2014

Autoren: Natalie Schumacher (NAS), David Roth (DR)

Dateiname: Pflichtenheft\_RothSchumacher.docx

## Änderungsgeschichte

Vers.	Datum	Autoren	Status	Änderungen
1.0	29.01.2014	NAS, DR	Initial	
1.1	30.01.2014	NAS	Weiterführen	Erstellen des wichtigsten Inhalts, Einfügen der Screens für die Benutzeroberfläche
1.2	05.02.2014	NAS, DR	Vollenden	ERM eingefügt
1.3	05.02.2014	DR	Vollenden	Angleichen der GUI-Prototypen kleinere Formatänderungen
1.4	21.02.2014	NAS	Anpassungen	Anpassen des Pflichtenheftes an Vorschläge des Lehrers
1.5	22.02.2014	DR	Anpassungen	Hardware-Spezifikationen
1.6	24.02.2014	NAS	Anpassungen	Produktfunktionen in Funktionen eingliedern und Wunschkriterien markieren
1.7	24.02.2014	DR	Anpassungen	Produktumgebung Hardware spezifiziert
1.8	25.02.2014	NAS	Anpassungen	Kapitel Zielbestimmung abgeschlossen

## Management Summary

Dies ist das Pflichtenheft für die im Modul 226 entwickelte Applikation. Es soll im Grunde genommen abbilden, was wir gedenken, im Rahmen dieses Unterrichts zu erreichen und was aus unserer Arbeit resultieren soll (=> Applikation). Dies im möglichst professionellen Rahmen mit einem, dem Standard entsprechenden Pflichtenheft.

# Inhaltsverzeichnis

1	Zielbestimmung .....	6
1.1	Musskriterien .....	6
1.2	Wunschkriterien.....	6
1.3	Abgrenzungskriterien .....	7
2	Produkteinsatz .....	8
2.1	Anwendungsbereich .....	8
2.2	Zielgruppe .....	8
3	Produktumgebung .....	9
3.1	Software .....	9
3.2	Hardware.....	9
4	Produktfunktionen .....	10
5	Produktdaten .....	13
5.1	Welche Daten müssen nicht gespeichert werden?.....	13
5.2	Menge der Daten.....	13
5.3	ERM .....	14
6	Produktleistungen .....	15
7	Benutzeroberfläche.....	16
7.1	Die ersten Prototypen.....	16
7.1.1	Komponenten .....	16
7.1.2	Übersichtsinterface .....	16
7.1.3	Anmeldeformular .....	17
7.1.4	Tamagotchi .....	17
7.1.5	Shop .....	19
7.1.6	Konfigurationseinstellungen ändern.....	19
7.1.7	Nachrichtenübersicht .....	20
7.1.8	Nachricht schreiben .....	20
7.1.9	Nachricht lesen .....	20
8	Qualitätsziele .....	21
8.1	Konsistenz.....	21
8.2	Selbstbeschreibungsfähigkeit - Einfachheit.....	21
9	Testfälle .....	22

9.1	Erstellen eines Users .....	22
9.2	Tamagotchi einen Namen geben .....	23
9.3	Einkauf tätigen.....	23
9.4	Tamagotchi Medizin verabreichen.....	24
9.5	Neuen Highscore erreichen.....	24
9.6	Nachricht versenden (Wunschkriterium) .....	25
10	Entwicklungsumgebung.....	26
10.1	Software .....	26
10.1.1	IDE.....	26
10.1.2	Grafiksoftware.....	26
10.1.3	Arbeit mit der Datenbank.....	26
10.2	Hardware.....	27
11	Ergänzungen .....	28
11.1	Installationshinweise .....	28
11.2	Lizensierung.....	28
11.3	Sonstiges .....	28

# 1 Zielbestimmung

Das Produkt ist eine Interpretation zweier Informatik-Lehrlinge des Tamagotchis als Desktop-Anwendung. Aus diesem Grund orientiert es sich bei den Kriterien stark an dem Tamagotchi-Handheld.

## 1.1 Musskriterien

### User:

Die Applikation ist als Mehrusersystem gedacht, in der mehrere User sich anmelden können. Eine Mehrfachanmeldung desselben Users soll bewusst eingeschränkt werden. In Minispielen erreichte Highscores sind spielerübergreifend und von allen einsehbar und ermöglichen ein Wetteifern untereinander.

### Applikation allgemein:

Der User kann seinen Account anlegen, sich bei seinem Tamagotchi einloggen und diesem einen Namen geben. Daraufhin erhält es ein zufälliges Geschlecht und man kann beginnen zu spielen. Das Tamagotchi hat ein anklickbares Menü, wo man die verschiedenen Pflegeoptionen aussuchen kann. Es ist möglich, im Shop Items zu erwerben, die täglich wechseln und in der Datenbank festgelegt sind. Diese kann er seinem Tamagotchi zu essen, zu trinken oder zum Spielen geben.

### Technisch:

- Datenanbindung: MySQL-Datenbank
- Programmiersprache: Java

## 1.2 Wunschkriterien

### User:

Bei der Anmeldung wird die Passwort - Stärke überprüft und dem User angezeigt.

Die einzelnen User können einander Nachrichten zuschicken, diese beantworten oder löschen. Der User kann auch sein Passwort anfordern, falls er dieses vergessen hat.

### Applikation allgemein:

Der User kann ein Nutzerhandbuch aufrufen, das ihn bei der Benutzung der Applikation anleitet. Er kann dieses Nutzerhandbuch über einen Hilfe-Button aufrufen.

Der User kann sein Tamagotchi und die Steuerung personalisieren. Er kann in den Konfigurationseinstellungen die Hotkeys einstellen mit welchen er, statt des anklickbaren Menüs, die Pflegeoptionen aufrufen kann und dem Tamagotchi ein

eigenes Bild zuweisen statt dem Standardbild. Auch das Passwort kann geändert werden.

Das Tamagotchi bewegt sich in regelmässigen Bewegungsintervallen über den Tamagotchi-Hintergrund und durchlebt verschiedenste Entwicklungsstufen (Ei-Zustand, geschlüpft, Teenager, Erwachsener, Senior). Auch gibt es Animationen für das Waschen des Tamagotchis und der Hintergrund des Tamagotchis ändert sich während es schläft und das Tamagotchi ist während dieser Zeit nicht benutzbar.

#### **Technisch:**

Es sollen regelmässig manuelle Datenbank - Cleanups durchgeführt werden.

### **1.3 Abgrenzungskriterien**

#### **User:**

Bewusst ist keine Single Player - Applikation vorgesehen.

Die Applikation ist nicht so gedacht, dass User ihre Tamagotchis miteinander spielen lassen können, wie das beim Tamagotchi-Handhelden der Fall ist.

#### **Funktionalität:**

Allgemein zielt das Tamagotchi nicht darauf ab, dem Tamagotchi-Handhelden exakt gleichzukommen, sondern soll sich auch in der Umsetzung seiner Funktionen von diesem abheben.

Die User können untereinander kommunizieren, für die Tamagotchis ist das, nicht wie beim Original, aber nicht vorgesehen.

## 2 Produkteinsatz

Das Produkt kommt wohl vorwiegend im Heimbetrieb zur Anwendung und wird von den Kindern zur spielerischen Selbstunterhaltung genutzt. Dies vorwiegend im kleineren Rahmen, wobei jedoch eine mögliche kommerzielle Nutzung im grösseren Rahmen nicht ausgeschlossen wird (Kinder – Tagesstätte als Beispiel).

### 2.1 Anwendungsbereich

Unsere Interpretation des traditionellen Tamagotchi zielt, wie auch dieses selbst, darauf ab, sich im Genre „Unterhaltung“ anzusiedeln und dem User, von dem es verwendet wird, eine unterhaltsame Zeit zu bereiten.

### 2.2 Zielgruppe

Realistisch gesehen eignet sich die Applikation eher für eine jüngere, vorwiegend weibliche Zielgruppe. Wobei Kenner und eingefleischte Tamagotchifans es ebenfalls zu würdigen wissen könnten. Generell gehen wir jedoch eher von Mädchen im Alter zwischen 6 und 12 aus. Auch Jungen in diesem Altersbereich könnten Spass daran haben, aufgrund des geplanten Minispiels „Space Invaders“ und dem damit verbundenen „Kräftemessen“.



## **3 Produktumgebung**

### **3.1 Software**

Da wir die Sprache Java zur Entwicklung dieses Produkts verwendet haben, sollte es plattformübergreifend funktionstüchtig sein, das heisst, es kann sowohl auf einem Linux-OS als auch auf einem Windows-OS funktionieren.

Darüber hinaus wird eine Verbindung zwischen unserer Applikation und der Datenbank MySQL bestehen.

### **3.2 Hardware**

Das Programm sollte auch auf älteren Rechnern funktionieren da keine allzu rechenintensiven Aufgaben erledigt werden. Es wird darauf geachtet, dass das Programm mindestens auf einem 7 Jahre alten Laptop funktioniert (siehe 10.2 Hardware). Alles was älter ist, könnte funktionieren es wird jedoch keine Garantie auf die Geschwindigkeit und Nutzbarkeit der Applikation gegeben.

## 4 Produktfunktionen

Wir gehen bei den Produktfunktionen davon aus, dass wir die ganzen Kriterien (sowohl Wunsch – als auch Musskriterien) einfügen konnten, deswegen sind auch Funktionen, die später vielleicht nicht in der Applikation vorkommen, hier aufgelistet).

### 1. Anmeldung

Der User gibt die notwendigen Daten (Nickname, Email, Passwort) über sich ein und kann einen Account und ein dazugehöriges Tamagotchi erstellen lassen und direkt anfangen zu spielen.

### 2. Login

Der User kann sich mit Username und Passwort anmelden

### 3. Highscores einsehen

Auf dem Startbildschirm lassen sich alle Highscores anderer User einsehen.

### 4. Passwort vergessen

Die Emailadresse, die bei der Anmeldung angegeben wurde, wird jetzt dazu verwendet, einem ein neues Passwort zuzusenden, auf das das Passwort des Account zwischenzeitlich zurückgesetzt wurde.

### 5. Hilfe

Der Hilfe – Button, der auf jeder Maske zu finden ist, führt direkt zum Nutzerhandbuch, das einem hilft, sich im Spiel zurechtzufinden.

### 6. Configuration

- Der User hat die Möglichkeit für alle Funktionen und Menüpunkte, die man beim Tamagotchi hat, Hotkeys einzustellen, also als Beispiel möchte der User, wann immer er „S“ drückt, dass der Shop aufgerufen wird. Dies könnte er dann in den Konfigurationseinstellungen erledigen.
- Darüber hinaus ist es möglich, ein eigenes Bild für das Tamagotchi einzustellen, damit man ein benutzerdefiniertes Tamagotchi hat. Dieses sollte in den Formaten PNG oder JPEG vorliegen und 50x100 nicht überschreiten.
- Der User kann auch sein Passwort ändern, falls er es vergessen und eines der Standardpasswörter erhalten hat.

### 7. Tamagotchi

#### • Name geben

Nachdem man das Tamagotchi resetet oder neu erstellt hat, hat man die Möglichkeit, seinem Tamagotchi einen kurzen aber hoffentlich prägnanten Namen zu geben, den er bis zu seinem Tod oder seiner Zurücksetzung trägt.

#### • Reseten

Der User hat die Möglichkeit sein Tamagotchi auf die Standardwerte zurückzusetzen und ganz von vorne zu beginnen.

Der User kann die einzelnen Menüpunkte anklicken oder die Hotkeys verwenden (siehe Configuration) um auf die einzelnen Pflegeoptionen beim Tamagotchi Zugriff zu haben.

- Werte und Inventar einsehen

Der User hat die Möglichkeit die Werte (Hunger, Durst, Sauberkeit) seines Tamagotchis einzusehen und die Items, die er für das Tamagotchi gekauft hat. Bei den Werten kann er nichts weiter tun, als sich einen Überblick zu verschaffen. Beim Inventar kann er die Items dann auch gleich nutzen, sofern es nicht Nahrungsmittel sind, die er nur beim effektiven Füttern oder Tränken benutzen kann.

- Füttern

Der User kann seinem Tamagotchi entweder gekauftes Essen oder eine der Standardmahlzeiten zu Essen geben.

- Zu Trinken geben

Der User kann seinem Tamagotchi entweder ein gekauftes Getränk oder eines der Standardgetränke zu Trinken geben.

- Schlafen legen

Der User kann das Tamagotchi vorzeitig schlafen legen. Das Tamagotchi wird in den Schlafzustand versetzt und der User kann solange nichts mit dem Tamagotchi machen, bis es vollkommen ausgeschlafen ist (nach Ablauf der Zeitspanne n).

- Waschen

Der User kann das Tamagotchi waschen, woraufhin eine Animation oder dergleichen ablaufen wird, in deren Verlauf das Tamagotchi ganz sauber wird.

- Medizin geben

Das Tamagotchi erhält von der gekauften Medizin und der Gesundheitszustand wird auf gesund gesetzt.

- Spielen

Es gibt ein effektives Minispiel: Space Invaders das man mit seinem Tamagotchi spielen kann. In diesem kann man das Tamagotchi so steuern, dass es alle Raumschiffe ausschaltet.

## 8. Nachrichtenübersicht einsehen

In der Nachrichtenübersicht kann der User schnell und einfach seine Nachrichten durchschauen und je nach Menge durch die Nachrichten blättern. Es ist ihm ebenso möglich die Nachricht direkt zu löschen und zu lesen.

## 9. Nachricht schreiben

Der User kann durch Auswahl eines anderen Users per Dropdown (sofern er auf die Nachricht geantwortet hat ist der User bereits ausgewählt), der Angabe eines Titels und eines Nachrichtentextes einem anderen User eine Nachricht schicken.

## 10. Nachricht lesen

Der User kann bei der detaillierten Ansicht einsehen, wann und von wem die Nachricht verschickt wurde, sowie deren Inhalt.

### 11. Nachricht löschen

Der User kann Nachrichten auch löschen indem er direkt bei der detaillierten Ansicht oder auf der Nachrichtengesamtübersicht auf „löschen“ klickt.

### 12. Im Shop einkaufen

Im Shop kann der User seinem Tamagotchi Gegenstände (Nahrungsmittel und Spielzeug => täglich wechselnd) und Medizin kaufen. Entweder er wählt einen bestimmten Gegenstand aus und kauft diesen oder er wählt die Menge der Medizin aus, die er kaufen möchte.

### Wunschkriterium

## 5 Produktdaten

Wo? DB, File, ...	MySQL-Datenbank
-------------------	-----------------

### 5.1 Welche Daten müssen nicht gespeichert werden?

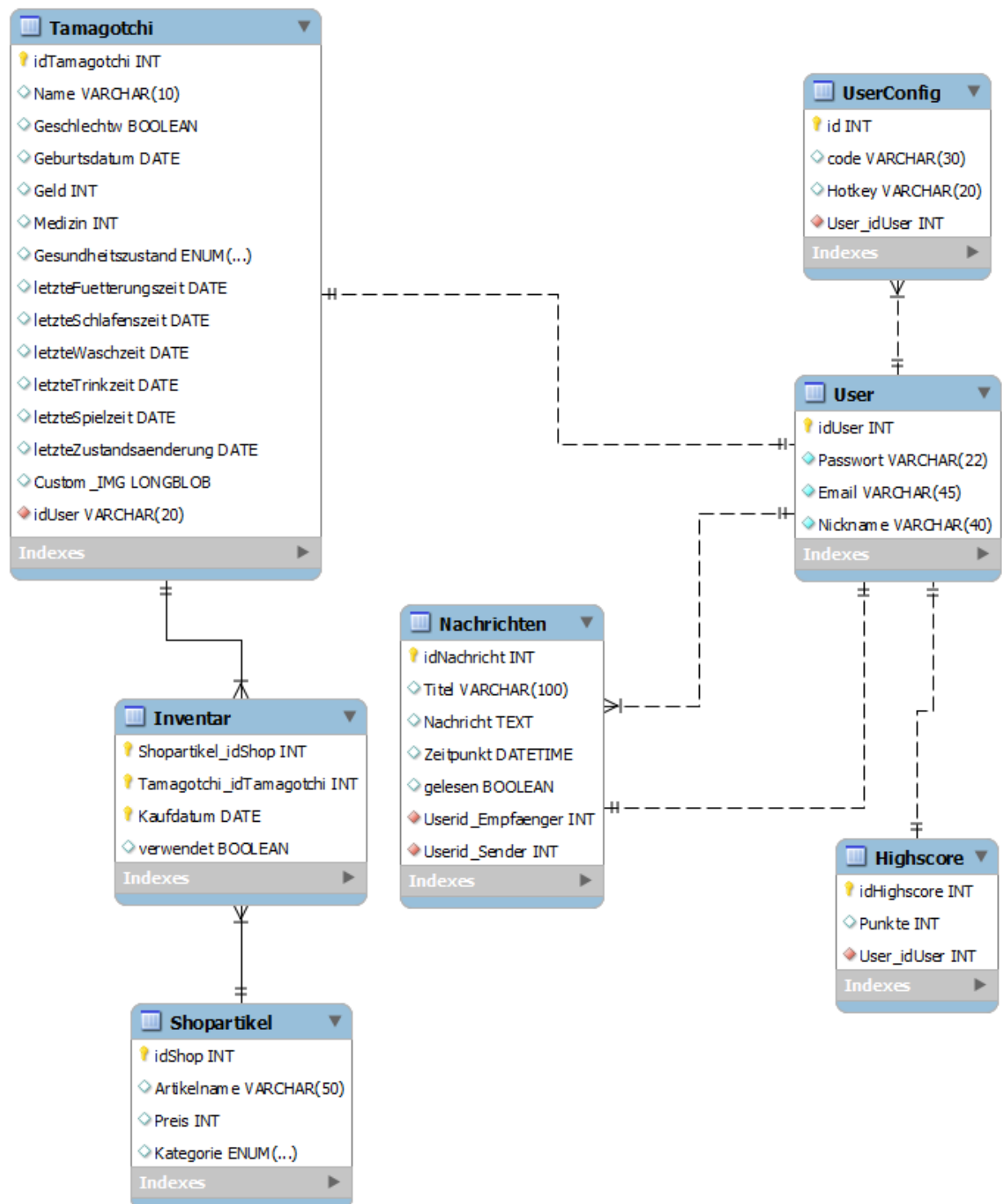
Diejenigen Daten, die aus allgemeineren Daten errechnet werden, müssen nicht längerfristig in der Datenbank gespeichert werden, um Redundanz zu vermeiden und Konsistenz zu gewährleisten, siehe ERM:

1. Evolutionsstand
2. Hungerwert
3. Schmutzgrad
4. Usw.

### 5.2 Menge der Daten

- Max. 50 User
  - ⇒ Je ein Tamagotchi
    - Zur gleichen Zeit je 10 Artikel im Inventar
  - ⇒ Je eine Konfiguration
- Max. 20 Nachrichten pro User werden gespeichert

### 5.3 ERM



## 6 Produktleistungen





<b>Menge der Datensätze</b>	siehe (Kapitel 5.1 „Menge der Daten“)
<b>Plattformkonformität</b>	Soll auf mehreren Plattformen (Linux, Windows, etc) laufen
<b>Zugreifbarkeit</b>	Die Applikation soll abhängig vom Internetzugang für die Verbindung zur Datenbank (keine Verbindung => keine funktionierende Applikation) je nachdem schneller oder langsamer sein und unabhängig davon immer laufen können (Vermeidung von technischen Problemen).

## 7 Benutzeroberfläche

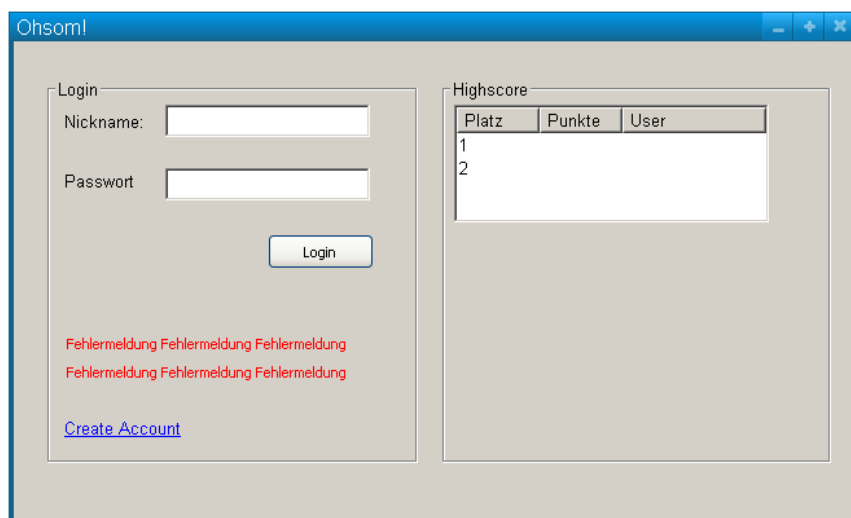
### 7.1 Die ersten Prototypen

Bei den nachfolgenden Gui – Prototypen handelt es sich lediglich um die Darstellungsversuche von Vorstellungen, die wir uns über das Endprodukt gemacht haben, nicht um festgelegte Ziele, die wir so und nicht anders durchzuführen gedenken.

#### 7.1.1 Komponenten

Allgemeine Komponenten	
	Buttons wie „Hilfe“ oder „Einstellung“ müssen klar erkennbar und einfach zu finden sein. Beide sollten sich eher am oberen Rand eines jeden Fensters befinden (Hilfe = Fragezeichen, Zahnrad = Einstellung)
	Passender Nachrichtenbutton
Spezifischere Komponenten	
	Beim geplanten Minispiel „Space Invaders“ gilt es, Raumschiffe abzuschiesen. Wenn ein Raumschiff getroffen wird, sieht man eine andere Ansicht, wie wenn es noch rumfliegt.
	Tamagotchi-Figur, die im Tamagotchi-Rahmen herumläuft und ihren Zustand ändern kann.

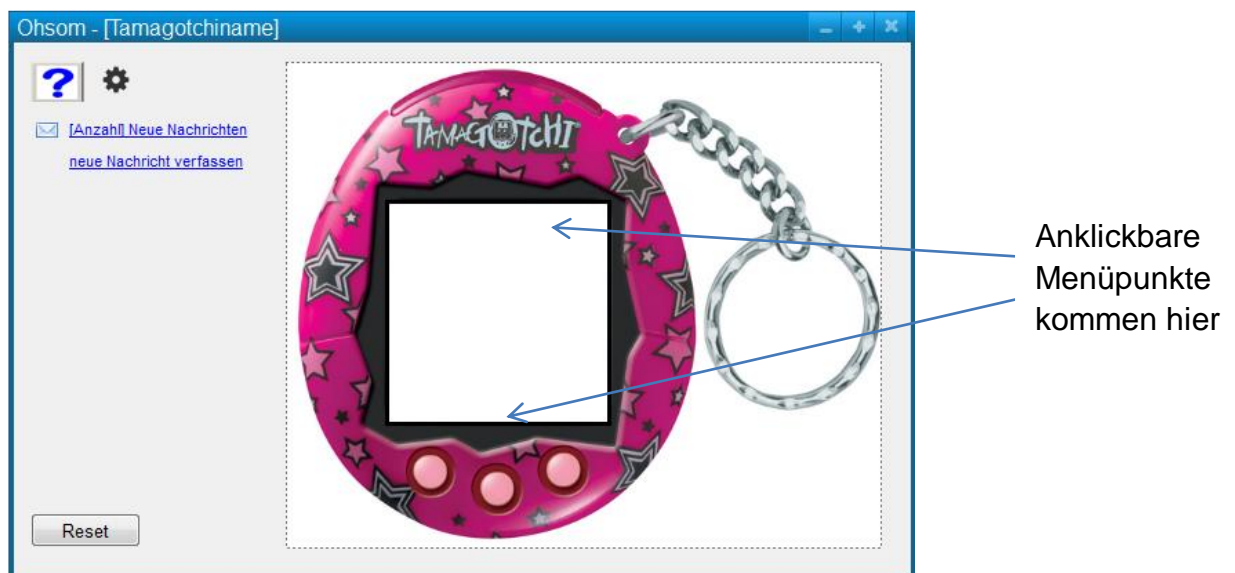
#### 7.1.2 Übersichtsinterface





### 7.1.3 Anmeldeformular

### 7.1.4 Tamagotchi



Menüpunkte (innerhalb des Tamagotchihandheld-rahmens):

- Werte (des Tamagotchis => Hunger, Durst, Hygiene, usw.)
- Füttern



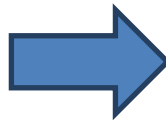
Man hat ein paar Standardmöglichkeiten und kann zwischen diesen mittels Pfeiltasten wählen (eventuell konfigurierbar => Customize). Wenn man letzten Endes eines der Nahrungsmittel möchte, muss man es nur anklicken.

- Trinken geben

Gleich wie Füttern

- Waschen
- Schlafen legen

Das Tamagotchi wechselt seinen Zustand und entweder der Hintergrund wird verändert (dunkel) oder das Tamagotchi => anderes Aussehen.



- Spielen

Minispiel: Space Invaders (das Tamagotchi kämpft gegen plötzlich auftauchende Raumschiffe)

- Inventar
- Shopping

### 7.1.5 Shop

Shop

Medizin Essen Getränke Sonstiges

Medizin: 10    Geld: 100

Medizin kaufen

Wie viel Medizin möchtest du kaufen?

15    kaufen

Kosten: 150

Fehlermeldungen (Bspw: zu wenig Geld!)

Das Medizins - Interface unterscheidet sich von den anderen, weil es keine verschiedenen Medikamente zur Auswahl gibt, Nahrungsmittel, als Beispiel, hingegen schon.

### 7.1.6 Konfigurationseinstellungen ändern

Konfigurationseinstellungen

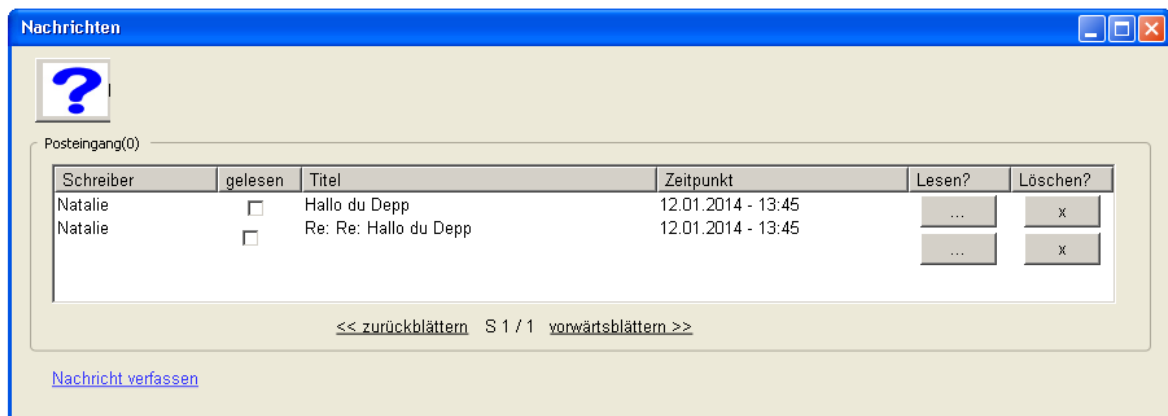
Hotkeys Tamagotchi

Optionen

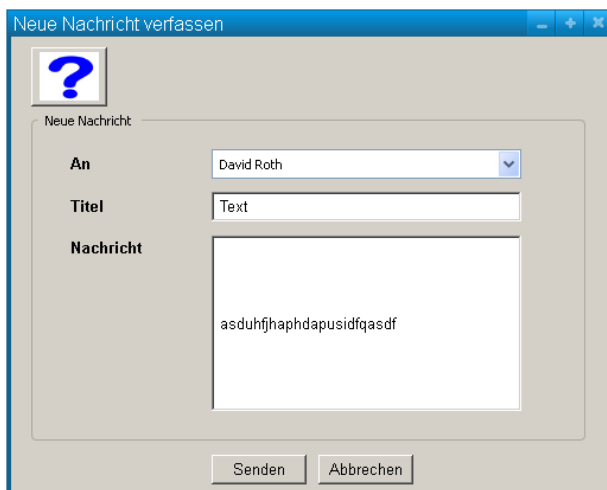
Werte anzeigen	Text
füttern	Text
schlafenlegen	Text
zu Trinken geben	Text
spielen	Text
shopping	Text
Waschen	Text
Inventar	Text

Speichern    Cancel

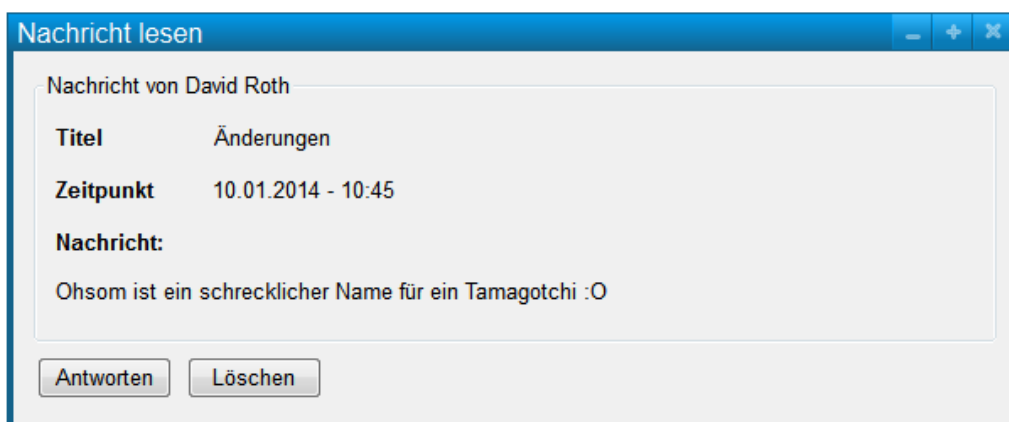
### 7.1.7 Nachrichtenübersicht



### 7.1.8 Nachricht schreiben



### 7.1.9 Nachricht lesen



## 8 Qualitätsziele

Anforderung	Sehr wichtig	Ziemlich wichtig	neutral	Nicht relevant
Aufgabenangemessenheit	x			
Selbstbeschreibungsfähigkeit	x			
Steuerbarkeit	x			
Erwartungskonformität	x			
Fehlertoleranz		x		
Individualisierbarkeit				x
Lernförderlichkeit			x	

### 8.1 Konsistenz

Eine gewisse Konsistenz soll eingehalten werden, der deutlich erkennbare Hilfe – Button, als treffendes Beispiel, soll immer an derselben Stelle anzutreffen sein. Auch soll es gewisse Gliederungen wie beispielsweise die Unterteilung des Fensters in Menü – und Content – Bereich geben. Das ermöglicht dem User, eine gewisse Gewohnheit einzustellen und nur in einem gewissen Bereich des Fensters nach bestimmten Controls zu suchen, die ihm bereits bekannt sind.

### 8.2 Selbstbeschreibungsfähigkeit - Einfachheit

Alles soll selbsterklärend und simpel sein. Es ist klar, welcher Button wofür da ist, oder zumindest ausschliessbar, wofür er nicht ist (der Hilfe Button ist nicht dazu da, Einstellungen am Account zu machen als Beispiel). Das ist deswegen nötig, weil das Tamagotchi eine relativ junge Zielgruppe ansprechen soll im Endeffekt und diese vermutlich nicht, mässig oder noch nicht besonders gut lesen und / oder begreifen kann.

Das wiederum bedeutet, dass die Buttons klare Symbole oder Beschriftungen haben müssen.

## 9 Testfälle

### 9.1 Erstellen eines Users

<i>ID: Testfall 1 Erstellen eines Users</i>	
<b>Ziel</b>	Ein neuer User konnte sich anmelden
<b>Akteur</b>	Nicht-User
<b>Vorbedingung</b>	Nicht-User will sich anmelden
<b>Ablauf</b>	<p>Der Nicht-User wählt auf dem Hauptbildschirm den Button „Create Account“, woraufhin sich ein neues Fenster öffnet, das ein Anmeldeformular zeigt. Dort kann er einen Nicknamen eingeben, ein Passwort, das er wiederholen muss (die Passwortstärke wird ebenfalls angezeigt) und seine Email (falls er mal sein Passwort vergessen hat oder falls eine Newsletterfunktion eingeführt werden würde).</p> <p>Wenn er das alles eingegeben hat und auf den Create – Button klickt, wird ein Account erstellt.</p>
<b>Nachbedingung</b>	Ein User- und ein zugehöriger Tamagotchidatensatz werden erstellt.
<b>Sonderfall</b>	<p>⇒ <b>Account wurde nicht erstellt</b></p> <p><b>Fall 1)</b> Es wurde kein Passwort eingegeben oder aus einem anderen Grund stimmen die beiden Passwörter nicht miteinander überein</p> <p><b>Fall 2)</b> Der Nickname ist bereits anderweitig vergeben</p> <p><b>Fall 3)</b> Die Mail wird bereits verwendet (soll Mehrfachanmeldungen zumindest einschränken).</p>

## 9.2 Tamagotchi einen Namen geben

<i>ID: Tamagotchi einen Namen geben</i>	
<b>Ziel</b>	Das Tamagotchi soll einen Namen erhalten
<b>Akteur</b>	User
<b>Vorbedingung</b>	Das Tamagotchi hat noch keinen Namen
<b>Ablauf</b>	Der User tippt den gewünschten Namen in das im Tamagotchirahmen erscheinende Textfeld. Der Name wird gespeichert, das Textfeld verschwindet, man kann mit dem Tamagotchi spielen.
<b>Nachbedingung</b>	Das Tamagotchi hat einen Namen erhalten. Dieser wurde in der Datenbank gespeichert.
<b>Sonderfall</b>	<p>⇒ Der Name konnte nicht gespeichert werden, das Textfeld verschwindet nicht, eine Fehlermeldung erscheint.</p> <p>Der im Textfeld eingegebene Name ist länger als die vorgegebene Zeichenanzahl von 10 Zeichen.</p>

## 9.3 Einkauf tätigen

<i>ID: Einkauf im Shop tätigen</i>	
<b>Ziel</b>	Man konnte seinem Tamagotchi etwas kaufen
<b>Akteur</b>	User
<b>Vorbedingung</b>	Das Tamagotchi hat genug Geld für Medizin (für jede weitere Medizin als Beispiel 10 [Währungsname]). Das Tamagotchi hat nicht bereits 100 (oberster Wert) Medizin.
<b>Ablauf</b>	Der User ruft mit dem Hotkey oder über das anklickbare Menü den Shop auf. Dort geht er auf die Kategorie Medizin und wählt eine gewisse Menge aus. Sobald er auf kaufen klickt, wird seine Anfrage verwertet.
<b>Nachbedingung</b>	Das Tamagotchi hat Geld verloren und mehr Medizin in seinem Besitz.
<b>Sonderfall</b>	<p>⇒ Der User konnte die Medizin nicht kaufen.</p> <p>Der User hat entweder zu wenig Geld oder zu viel Medizin (also aktuelle Medizin + neue Medizin). In diesem Fall müsste die Bestellung neu aufgegeben werden. Eine Fehlermeldung weist einen darauf hin, was genau das Problem ist.</p>

## 9.4 Tamagotchi Medizin verabreichen

<i>ID: Tamagotchi Medizin verabreichen</i>	
<b>Ziel</b>	Man konnte sein Tamagotchi von irgendwelchen Beschwerden heilen
<b>Akteur</b>	User
<b>Vorbedingung</b>	Das Tamagotchi ist krank und hat Medizin $\geq 1$ im Besitz
<b>Ablauf</b>	Der User benutzt den Hotkey um die Behandlung durch Medizin aufzurufen oder klickt direkt auf den Menübutton. Das Tamagotchi wird von den Beschwerden geheilt und ändert seinen Gesundheitszustand zu gesund.
<b>Nachbedingung</b>	Das Tamagotchi hat im Besitz $-1$ Medizin und der Gesundheitszustand ist = gesund.
<b>Sonderfall</b>	<p>⇒ Der User konnte dem Tamagotchi keine Medizin verabreichen.</p> <p>Das Tamagotchi ist bereits gesund oder der User hat keine Medizin um das Tamagotchi zu heulen.</p>

## 9.5 Neuen Highscore erreichen

<i>ID: Neuen Highscore erreichen (in der Rangliste aufsteigen)</i>	
<b>Ziel</b>	Man hat einen neuen Highscore erreicht, der auf der Hauptmaske in der Tabelle nach oben rutscht auf der Platzliste
<b>Akteur</b>	User
<b>Vorbedingung</b>	Das Tamagotchi ist nicht krank und schläft nicht und hat Lust zu spielen.
<b>Ablauf</b>	Man wählt ein Minispiel aus (über den Hotkey, sofern die Funktion implementiert wurde, oder über das anklickbare Menü). Man spielt und versucht einen neuen Highscore zu erzielen.
<b>Nachbedingung</b>	Man hat seinen letzten Highscore übertroffen und steigt in der Punkteliste auf.
<b>Sonderfall</b>	<p>⇒ Der User konnte seinen Highscore nicht toppen und bleibt auf dem gleichen Punktestand.</p> <p>Als Folge davon verändert sich natürlich auch seine Platzierung nicht.</p>



## 9.6 Nachricht versenden (Wunschkriterium)

<i>ID: Nachricht verfassen</i>	
<b>Ziel</b>	Man konnte einem gewünschten User eine Nachricht schicken
<b>Akteur</b>	User
<b>Vorbedingung</b>	Keine (man kann immer Nachrichten verschicken)
<b>Ablauf</b>	Man geht im Nachrichtenfenster oder im Tamagotchi - Fenster auf „neue Nachricht verfassen“. Das Nachrichtenfenster öffnet sich und man kann eine neue Nachricht eintippen und abschicken.
<b>Nachbedingung</b>	Die Nachricht wurde, mit standardmässig auf ungelesen gesetzten Zustand verschickt und sollte nun vom anderen Benutzer in seinem Postfach einsehbar sein.
<b>Sonderfall</b>	<p>⇒ Die Nachricht konnte nicht verschickt werden.</p> <p>Alle Felder müssen ausgefüllt sein, damit die Nachricht abgeschickt (bzw. eingetragen) werden kann. Ansonsten kann die Nachricht nicht abgeschickt und nicht von der anderen Person gelesen werden.</p>

## 10 Entwicklungsumgebung

### 10.1 Software

#### 10.1.1 IDE



Als Entwicklungsumgebung wird uns Eclipse in der Version 4.3 dienen, da wir mit dieser in den letzten 1.5 Jahren intensiv gearbeitet haben und wir daher ihre Vorzüge, bzw. die Kniffe und Tricks kennen.

#### 10.1.2 Grafiksoftware



Sollten jegliche gestalterischen Arbeiten anfallen, so bedienen wir uns des kostenlosen Grafikbearbeitungsprogramms GIMP in der Version 2.8.10.

#### 10.1.3 Arbeit mit der Datenbank



Zur Erstellung des ERMs welches unserer Datenbank einen Startaufbau gegeben hat, verwendeten wir die MySQL Workbench v5.2.



Zur späteren Administration der Applikations-Datenbank verwenden wir zusätzlich das RDBMS - Tool phpmyadmin v4.0.9.

## 10.2 Hardware

Das Programm wird auf folgender Hardware entwickelt und getestet:

OS: Arch Linux 64bit/Windows 7 Home Premium 64bit

CPU: i7-3770 @ 3.4GHz

GPU: Nvidia Geforce 640 GT

OS: Ubuntu 13.10 64bit/ Windows Vista 32bit

CPU: Centrino 2 @ 2.5GHz

GPU: Intel 4 Mobile Series

OS: Windows 8 64bit

CPU: i5-3230M @ 2.6 Ghz

GPU: Nvidia Geforce GT 710M

## 11 Ergänzungen

### 11.1 Installationshinweise

#### Bei lokaler Ausführung:

- Dazugehörige SQL – Datei (Name wird nach Abschliessung des Projekts mitgeteilt) muss in der MySQL – Datenbank importiert werden.

### 11.2 Lizenzierung

- WTFPL : Do **W**hatever **T**he **F**uck You Want To **P**ublic **L**icense

### 11.3 Sonstiges

Das Produkt kann letzten Endes leichte Abweichungen im Datenbankaufbau, in der Gui-Gestaltung und in der Fülle der Funktionen besitzen. Das hat unter anderem damit zu tun, dass wir mitten im Programmieren merken können, dass wir etwas nicht so umsetzen wollen, aufgrund der Usability oder weil es überflüssig ist. Wenn bspw. zu viele Daten gespeichert werden, werden wir das einschränken, wenn zu wenig gespeichert werden, werden wir das noch zusätzlich ergänzen.