

Projektarbeit IT-Projekt

Frühling 2017



Kamisado

The duel of the dragon towers

Gruppe Wind Chuckers

Lukas Weber

Cyrill Füglister

Lukas Kunz

Robin Lehner

Thomas Bosshard

Supervision by Lukas Frey

Responsible professor: Brad Richards

Olten, 10. März 2017



Inhaltsverzeichnis

1	Einle	itung	5
	1.1	Ziele	5
	1.2	Umsetzungs- und Projektziele	5
2	Stak	eholder Analyse	6
	2.1	Stakeholder in den Systemgrenzen	6
	2.2	Stakeholder im Detail	6
	2.3	Diagramme	8
	2.3.1	Einteilung nach Wichtigkeit / Einfluss	8
	2.3.2	Einteilung nach Wichtigkeit / Motivation	9
3	Kont	extanalyse	10
	3.1	Systemabgrenzung	10
	3.2	Systemgrenze	10
	3.3	Kontextgrenze	10
	3.4	Ausserhalb der Kontextgrenze	10
4	Anfo	rderungsermittlung	11
5	Use	Case Modellierung	12
	5.1	100 - Login / Logout	13
	5.2	200 - Benutzerverwaltung	16
	5.3	300 - Spielumgebung	21
	5.4	400 - Rangliste	27
	5.5	500 - Technisch	29
	5.6	600 - Zusätzliche optionale Use Cases	32
6	Endŗ	rodukt Use Cases	39
	6.1	Einzel- oder Mehrplayer	39
7	Klass	endiagramm	40
	7.1	Abstract MVC	40
	7.2	Allgemeine Klassen	41
	7.3	Splash-Screen	42
	7.4	Client	43
	7.5	Message	44
	7.6	Login	46
	7.7	Main Menu	47
	7.8	Game Menu	48
	7.8.1		
	7.8.2		
	7.8.3		
	7.9	Al Steuerung	
	7.10	Friends	

7	'.11	Lobi	oy	54
7	'.12	Turc	orial	55
7	7.13	Usei	Menu	56
7	7.14	Auff	orderung zur neuen Runde	57
7	'.15	Steu	erung der Menus	58
7	'.16	Serv	er	59
7	'.17	AI		60
8	Entit	y-Rel	ationship-Modell	62
9	Rela	tiona	les-Modell	63
10	Vert	eilun	gsdiagramm	64
11	Qual	itätsa	anforderungen	65
12	GUI-	Besta	ndteile	66
1	2.1	Logi	n	66
1	2.2	Lobi	oy	66
1	2.3	Maii	n Menu	67
1	2.4	Frie	nds Menu	67
1	2.5	Add	Friends	68
1	2.6	Gam	ne Menu	69
13	Test	konze	pt	70
1	.3.1	Einf	ührung	70
1	.3.2	Zwe	ck	70
1	.3.3	Beze	eichnen der Testfälle	70
1	.3.4	Abn	ahme- und Testkriterien	70
	13.4	.1	Erfassung Testfälle und Anweisungsabdeckung	70
	13.4	.2	Unit-Tests	70
	13.4	.3	Testergebnisse	71
1	.3.5	Vorg	gehen	71
	13.5	.1	Beginn Projekt	71
	13.5	.2	Aufbau Testing	71
	13.5	.3	Unit-Tests	71
	13.5	.4	Implementierung des neuen Codes	71
	13.5	.5	Neuster Stand	71
	13.5	.6	Freigabe Komplettpacket	71
1	3.6	Test	protokoll	72
14	Abbi	ldung	gsverzeichnis	78
15		_	erzeichnis	
16	Anha	ang		80
1	.6.1	AI		80
	16.1	.1	Heuristik	80

n w Fachhochschule Nordwestschweiz

	16.1.	.2	Gewichtung	80
	16.1.	.3	Steuerung	81
1	6.2	Clien	nt	82
1	6.3	User	Menu	83
1	6.4	Logir	1	85
1	6.5	Spiel	l	86
1	6.6	Tuto	rial	88
1	6.7	Haup	ot-Menu	89
1	6.8	Frien	nds-Menu	90
	16.8.	.1	Add Friends	91
1	6.9	Lobb	ру	92
1	6.10	Se	erver	93
17	Old (GUI-M	1ockup's	94
1	7.1	Logir	n	94
1	7.2	Lobb	ру	94
1	7.3	Mair	n Menu	95
1	7.4	Frien	nds Menu	95
1	7.5	Add	Friends	96
1	7 6	Gam	e Menu	96



1 Einleitung

Das Spiel Kamisado wurde im Jahr 2009 vom HUCH & friends-Verlag lanciert. Das abstrakte Brettspiel kann von zwei Personen gespielt werden.

Das Spielfeld ist ähnlich wie ein Schachbrett aufgebaut. Es gibt 8 Felder in der Breite und 8 Felder in der Tiefe. Insgesamt sind es 64 Felder, welche nach einem bestimmten Muster eingefärbt sind. Dieselben Farben werden auch für die Spielfiguren verwendet. Jeder Spieler ist im Besitz von 8 Türmen.

In einem Zug muss jeder Spieler eine seiner Spielfiguren entweder vorwärts oder diagonal ziehen, ohne dabei andere Figuren zu überspringen. Die Farbe, auf dem der Spielzug endet, bestimmt die Farbe der gegnerischen Spielfigur, mit welcher dieser seinen nächsten Zug machen muss. Ziel ist es die Grundlinie des Gegners zu erreichen.

Es ist auch möglich das Spiel über mehrere Runden zu führen. Dazu wird die Spielfigur, welche die Grundlinie des Gegners erreicht hat mit einem Drachenzahn versehen. Spielfiguren, welche einen Drachenzahn besitzen, sogenannte "Sumos", verfügen über spezielle Fähigkeiten. Zum Beispiel können Sie den Gegner mit einem "Sumo-Stoss" zurückdrängen. Sumo-Figuren bringen Punkte, wobei die Anzahl der Punkte von der Anzahl der Drachenzähne abhängt.

Das Spiel wird beendet, wenn ein Spieler die festgelegte Punktezahl erreicht.

1.1 Ziele

Es ist vorgesehen das Spiel virtuell spielbar zu machen mit einer Server-Client-Variante. Es soll dabei die Option geben das Spiel als Multiplayer oder Singleplayer zu starten.

1.2 Umsetzungs- und Projektziele

Umsetzungsziele	Projektziele		
Multiplayer-Spiel möglich	 Auf dem Server können sich zwei Parteien einloggen via Client Der Server wechselt die Spieler automatisch ab 		
Singleplayer-Spiel möglich	 Der Server hat eine Logik, welche selbst Spielzüge vornehmen kann Der Server wechselt die Spieler automatisch ab 		
Etc.	-		

Tabelle 1: Umsetzungs- und Projektziele



2 Stakeholder Analyse

2.1 Stakeholder in den Systemgrenzen

Die ermittelten Stakeholder stehen wie folgt in Beziehung mit dem System:

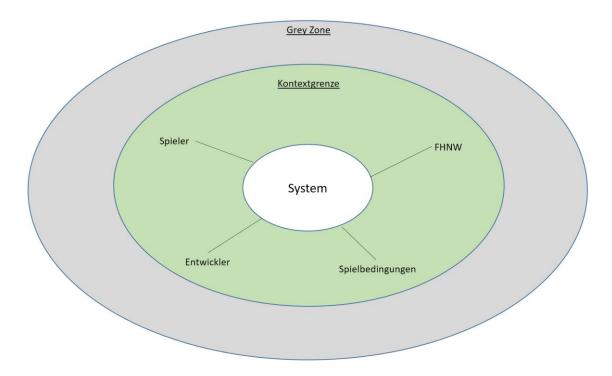


Abbildung 1: Stakeholder in der Systemgrenze

2.2 Stakeholder im Detail

Stakeholder	Spieler	Spieler		
Position	Alle Spieler			
Rolle	Benutzer des S	piels über den Client		
Ziele	Spiel als Single	Spiel als Single-, oder als Multiplayer zu starten und zu spielen		
Wissen	Spielregeln			
		Es ist der Endbenutzer. Der Spieler hat eine Vorstellung wie das System laufen sollte.		
Einfluss	Niedrig	Hat auf die Entwicklung einen geringen Einfluss		
Motivation	Hoch	Wird am Ende mit dem Spiel spielen		

Tabelle 2: Stakeholdergruppe: Spieler



Stakeholder	Entwickl	er		
Position	Entwickelt d	Entwickelt das Spiel		
Rolle	Erstellt das s	pielbare Spiel		
Ziele		Möchte eine möglichst gute Applikation erstellen, welche im Minimum die Minimalanforderungen erfüllt, jedoch wäre es besser weitere Funktionen einzubauen		
Wissen	Weiss wie da	as Spiel entwickelt werden muss		
Wichtigkeit	Hoch	Kennt das Spiel und weiss wie es entwickelt werden muss		
Einfluss	Hoch	Bestimmt welche Anforderungen umgesetzt werden		
Motivation	Will am Schluss eine gute Note erzielen und ein g funktionierendes System abgeben			

Tabelle 3: Stakeholdergruppe: Entwickler

Stakeholder	FHNW				
Position	Gibt den Auftrag	Gibt den Auftrag für das IT-Projekt			
Rolle	Gibt Auftrag und	Gibt Auftrag und bewertet Endergebnis			
Ziele		Will mindestens ein funktionierendes System, besser jedoch wenn noch mehr Funktionen eingebaut wurden			
Wissen	Weiss wie das Sp	Weiss wie das Spiel aussehen soll und welche die Minimalforderungen sind			
Wichtigkeit	Hoch	Kennt das Spiel und erstellt die Benotung			
Einfluss	Mittel	Gibt Leitplanken vor, jedoch liegt die Entwicklung komplett beim Entwickler			
Motivation	Hoch	Wird am Schluss das Spiel spielen und bewerten			

Tabelle 4: Stakeholdergruppe: FHNW

Stakeholder	Spielbedir	Spielbedingungen			
Position	Spiels				
Rolle	Gibt die einzuhaltenden Vorschriften bezüglich der Spielregeln vor				
Ziele	Spielregeln eir	Spielregeln einhalten			
Wissen Spielregeln					
Wichtigkeit	Mittel	Gibt die Spielregeln vor, auf den Rest der Umsetzung hat es keinen Einfluss			
Einfluss Hoch Die Spielregeln sind dringend einzuhalter		Die Spielregeln sind dringend einzuhalten			
Motivation	Niedrig	Die Spielregeln sollen eingehalten werden			

Tabelle 5: Stakeholdergruppe: Spielbedingungen

2.3 Diagramme

2.3.1 Einteilung nach Wichtigkeit / Einfluss

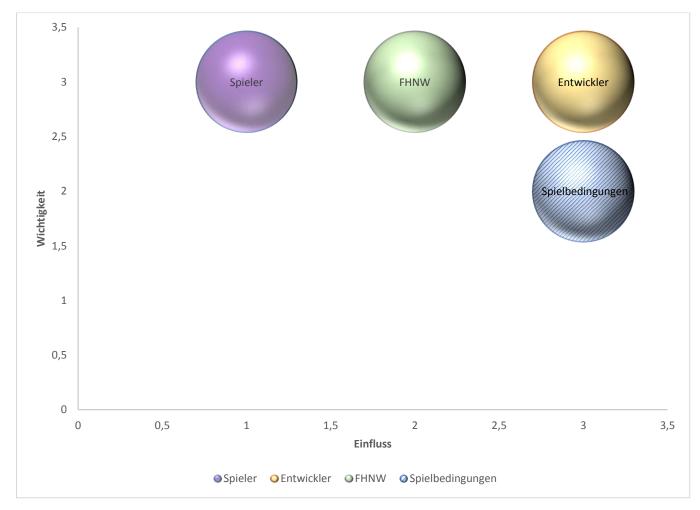


Abbildung 2: Stakeholder: Wichtigkeit und Einfluss

Die Y- Achse zeigt die Wichtigkeit eines Stakeholders. Die Wichtigkeit indexiert, wieviel der Stakeholder beitragen kann, um die Anforderungen an das System konkret zu spezifizieren. Die X- Achse zeigt den Einfluss des Stakeholders. Der Einfluss-Wert sagt aus, über wie viel Macht und Kompetenzen eine Person verfügt.

2.3.2 Einteilung nach Wichtigkeit / Motivation

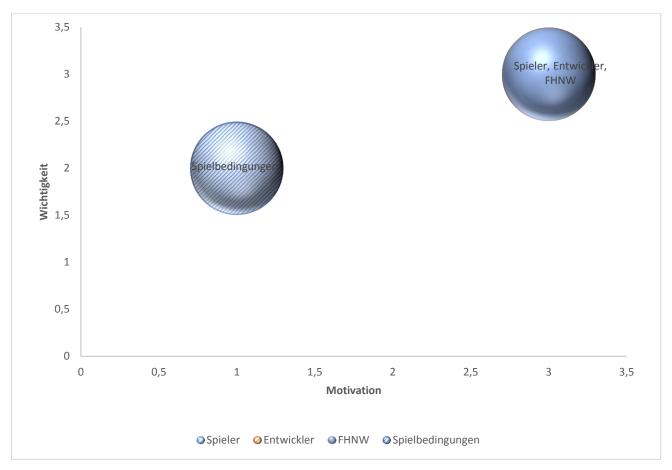


Abbildung 3: Stakeholder: Wichtigkeit und Motivation

Die Y- Achse zeigt die Wichtigkeit eines Stakeholders. Die Wichtigkeit indexiert, wieviel der Stakeholder beitragen kann, um die Anforderungen an das System konkret zu spezifizieren. Die X- Achse zeigt die Motivation des Stakeholders. Die Motivation unterstreicht das Interesse der unterschiedlichen Stakeholder für das Projekt.



3 Kontextanalyse

3.1 Systemabgrenzung

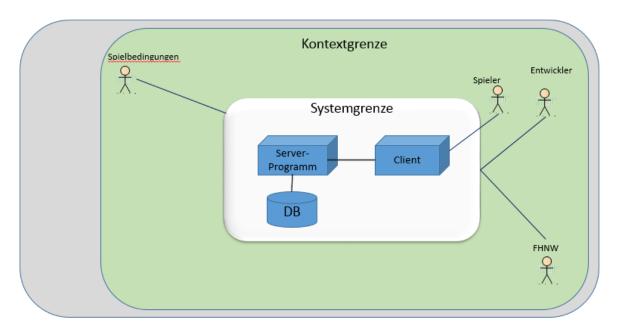


Abbildung 4: Systemabgrenzung

3.2 Systemgrenze

Das System besteht aus einem Server- und einem Client-Programm welches zur Benutzung zur Verfügung steht.

3.3 Kontextgrenze

Innerhalb der Kontextgrenze stehen die Stakeholder mit folgenden Tätigkeiten:

- Der Spieler beginnt das Spiel im Single- oder Multi-Player-Modus auf dem Client
- Der Entwickler entwickelt das System nach erhobenen Use Cases
- Die FHNW erhält das Spiel am Ende des Projektes und benotet dieses
- Die Spielbedingungen geben die Spielregeln vor, welche im Spiel implementiert sein müssen

3.4 Ausserhalb der Kontextgrenze

Ausserhalb der Kontextgrenze befindet sich kein Stakeholder.



4 Anforderungsermittlung

Um die Anforderungen korrekt ermitteln zu können, wurde hauptsächlich die Dokumentation der FHNW benutzt. Des Weiteren wurden zusätzliche Informationen aus dem Internet und von der offiziellen Spiel-Seite herbeigezogen um mehr Details zu erfahren. Grundsätzlich soll beim Projekt die "Lifecycle-Methode" angewendet werden, da in der Grösse dieses Projektes eine iterative Methode wenig Sinn ergibt. Damit das Projekt am Schluss den Systemtest wie auch den Akzeptanztest problemlos besteht, soll das System von Zeit zu Zeit einer unbeteiligten Person gezeigt werden. Dadurch soll verifiziert werden, dass das Spiel leicht verständlich ist.



5 Use Case Modellierung

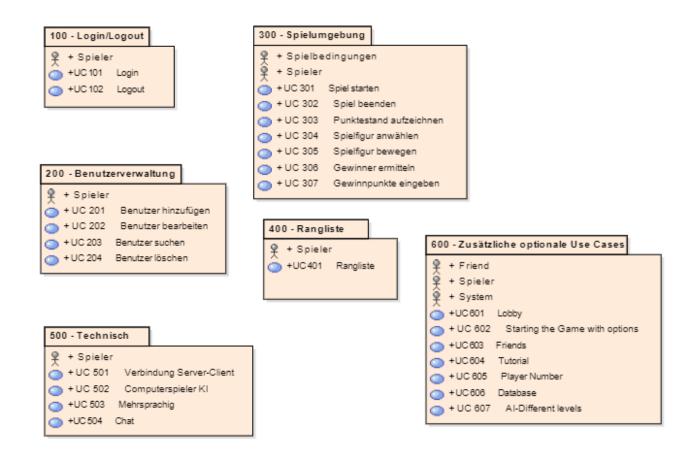


Abbildung 5: Use Case Modellierung Übersicht



5.1 100 - Login / Logout

UC Nr. 101	Login		optional	
Beschreibung	Das System muss dem Benutzer ermöglichen, seinen Benutzernamen und sein Passwort über die Tastatur auf der Benutzeroberfläche einzugeben und sich damit einzuloggen.			
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Der Benutzer loggt sich ein.			
Vorbedingungen	UC 201 -> Benutzer muss im S UC 501 -> Verbindung muss h	•		
Eingaben	Benutzername, Passwort			
	Standard	Alternativ		
	1. Aufruf des Loginfensters		"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite	
	2. Benutzername eingeben	2.a.1 Hinweis: Benutzername falsch eingegeben		
Ablauf (Szenario)	3. Passwort eingeben	3.a.1 Hinweis: Passwort falsch eingegeben		
	4. "Anmelden" klicken			
	5. System überprüft eingaben			
	6. Eingaben korrekt	6.a.1 Eingaben nicht korrekt		
		6.a.2 Benutzer wird zur erneuten Eingabe aufgefordert		
	7. Benutzer angemeldet			
		1	1	
Ausgaben	Logdaten für das Einloggen werden geschrieben			
Nachbedingung	5 50 5			

Tabelle 6: UC NR 101: Login



UC Nr. 102	Logout	optional			
Beschreibung	Das System muss dem Benutzer ermöglichen sich abzumelden				
Verantwortlicher	Spieler				
Beteiligte	-				
Auslöser	Der Benutzer loggt sich aus.				
	UC 101 -> Benutzer muss im System	angemeldet sein			
Vorbedingungen	UC 201 -> Benutzer muss im System erfasst sein				
	UC 501 -> Verbindung muss hergeste	llt sein			
Eingaben	-				
	Standard	Alternativ			
Ablauf (Szenario)	1. Im Main Menu über "Datei" –				
	"Beenden" sich ausloggen				
Ausgaben	Ausgaben -				
Nachbedingung	Nachbedingung Der Benutzer ist ausgeloggt				

Tabelle 7: UC Nr. 102: Logout



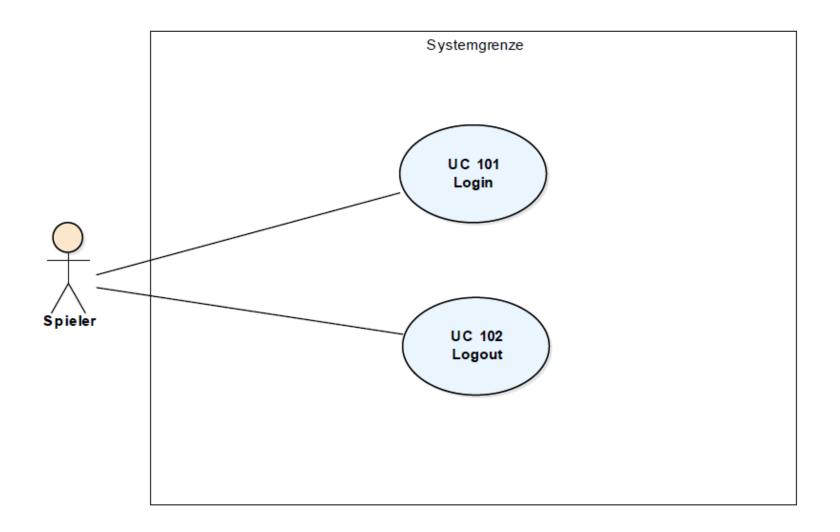


Abbildung 6: UC Diagramm 101 - 102



5.2 200 - Benutzerverwaltung

UC Nr. 201	Benutzer hinzufügen		optional	
Beschreibung	Das System muss das Erstel ermöglichen	len eines Be	enutzers über d	ie Tastatur
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Neuer Benutzer wird erstellt. Im System wird ein neuer Be			er Benutzer erfasst.
Vorbedingungen	UC 501 -> Verbindung muss			
Eingaben	IdVornameNachname	J	• Poin	oTower
	Standard	Alternati	v	
	1. "Benutzer hinzufügen" anklicken			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
	2. Alle Benutzerdaten eingeben			
	3. "Benutzer erstellen" anklicken			
Ablauf (Szenario)	4. Eingaben korrekt und vollständig	Felder sin Pflichtfeld "Eingaber	n ndig" (alle id der) / n nicht (Vorname ist	
		4.a.2 Eing vervollstä korrigiere 4.		
	5. Benutzer erstellt	5.a.1 Mel "Benutze erfasst"	_	
		5.a.1 Zuri Startseite		
Ausgaben	-			
Nachbedingung	Ein neuer Benutzer ist erste	IIt		

Tabelle 8: UC Nr. 201: Benutzer hinzufügen



UC Nr. 202	Benutzer bearbeite	Benutzer bearbeiten optional				
Beschreibung	Das System muss dem Ben	Das System muss dem Benutzer die Bearbeitung von Vornamen, Nachnamen				
Describering	über die Tastatur ermöglich	über die Tastatur ermöglichen				
Verantwortlicher	Spieler	Spieler				
Beteiligte	-	-				
Auslöser	Benutzer will die Benutzerd	Benutzer will die Benutzerdaten korrigieren / ändern.				
Vorbedingungen	UC 201 -> Der Benutzer, we UC 101 -> Der Benutzer mu	ıss im System	n angemeldet s			
	UC 501 -> Verbindung mus.	s nergestellt	sein			
Eingaben	Vorname					
	Nachname					
	Standard	Alternativ				
	1. Benutzer suchen	Alternativ		"Datei" und dann		
	<includes case<br="" use="">203>></includes>			"Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite		
	2. "Change User / create User" anklicken					
	3. Personendaten ändern und altes Benutzerpasswort eingeben					
Ablant (Cronoria)	4. "Ändern" drücken					
Ablauf (Szenario)	5. Eingaben korrekt und vollständig	nicht ausg	ndig" (alle d ler) / n nicht Vorname ist gefüllt)			
	C. Doguttary hooving that	5.a.2 Einga vervollstäi korrigierei 4.				
	6. Benutzer bearbeitet					
Ausgaben	-					
Nachbedingung		adaton words	on im Systam h	oim aucgowählten		
ivaciineuiiiguiig	Die aktualisierten Personendaten werden im System beim ausgewählten Benutzer gespeichert					

Tabelle 9: UC Nr. 202: Benutzer bearbeiten



UC Nr. 203	Benutzer su	ıchen			optional		
Beschreibung	Das System muss dem Benutzer die Suche von aktiven und inaktiven Spielern ermöglichen						
Verantwortlicher	Spieler	Spieler					
Beteiligte	-	-					
Auslöser	Ein Benutzer soll g	gesucht werden.					
	UC 201 -> Der Ber	UC 201 -> Der Benutzer muss im System erfasst sein					
Vorbedingungen	UC 101 -> Der Ber	utzer muss im Syste	m angeme	eldet sein			
	UC 501 -> Verbind	lung muss hergestell	t sein				
	• Id						
Eingaben	Vornam	е					
	 Nachnai 	me					
	Standard		Alternativ	,			
	1. User Menu				"Datei" und		
	aufrufen				dann "Ende"		
					anklicken.		
					Ausstieg aus		
					dem Use Case		
					und zurück zur		
					Startseite		
	2. Eingabe der				Startseite		
	Personendaten						
	zur Suche						
	3. "Suchen"	3.a.1 Benutzer					
	anklicken	nicht gefunden					
		3.a.2 Eingaben	3.b.1				
		korrigieren /	Benutze	er			
		vervollständigen	nicht				
		weiter mit 4.	gefunde	en			
Alderes (Comments)			3.b.2 U0	3.c.1 Auftritt eines			
Ablauf (Szenario)			201 ->	Verbindungsfehlers			
			"Benutz				
			hinzufü	-			
			" anklick				
			3.b.3	3.c.2 Das System			
			Weiter				
			UC 201	"Verbindung konnte			
				nicht aufgebaut			
				werden, bitte überprüfen Sie die			
				Einstellungen und			
				kontaktieren Sie den			
				Systemadministrator."			
	4. Benutzer			Systemaummstrator.			
	gefunden						
	5. Benutzer und						
	dessen Punkte						
	werden						
	angezeigt						
				'			
Ausgaben	-						
Nachbedingung	Der Benutzer wur	de gefunden und dei	r Account	aufgerufen			

Tabelle 10: UC Nr. 203: Benutzer suchen



UC Nr. 204	Benutzer löschen		optional		
Beschreibung	Das System muss dem Benu Spielern ermöglichen	Das System muss dem Benutzer das Löschen von einem eindeutig definierten Spielern ermöglichen			
Verantwortlicher	Spieler				
Beteiligte	-				
Auslöser	Der Spieler spielt das Spiel r	nicht mehr	und der Account soll gelöscht werden		
Vorbedingungen	welcher gelöscht werden so UC 101 -> Der Benutzer mu	UC 201 -> Der Benutzer, welcher die Löschung vornimmt und der Benutzer, welcher gelöscht werden soll, müssen beide erfasst sein UC 101 -> Der Benutzer muss im System angemeldet sein UC 501 -> Verbindung muss hergestellt sein			
Eingaben	Benutzername, Passwort				
	Standard	Alternati	v		
	Benutzer suchen < <includes case<="" td="" use=""><td></td><td>"Datei" und dann "Ende" anklicken.</td></includes>		"Datei" und dann "Ende" anklicken.		
Ablauf (Szenario)	203>>		Ausstieg aus dem Use Case und zurück zur Startseite		
	2. Altes Benutzerpasswort				
	eingeben und "Löschen" anklicken				
	3. Benutzer gelöscht				
			·		
Ausgaben					
Nachbedingung	Benutzer ist gelöscht.				

Tabelle 11: UC Nr. 204: Benutzer löschen



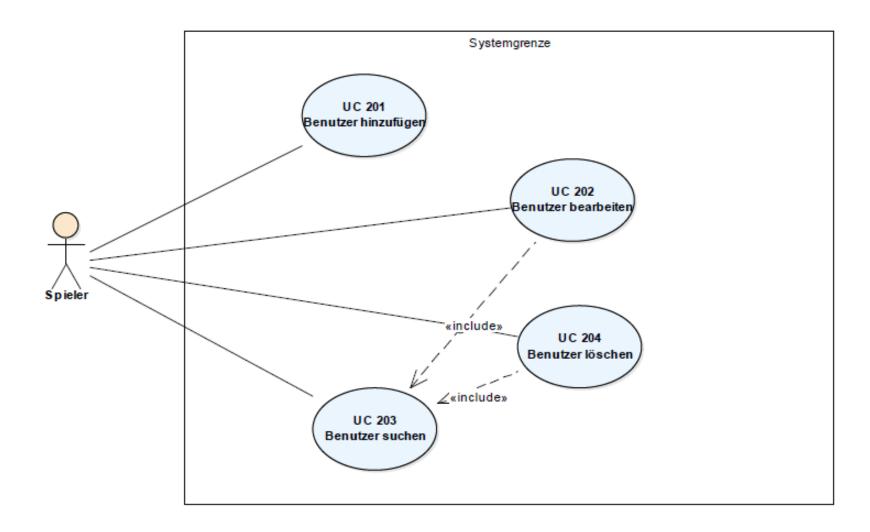


Abbildung 7: UC Diagramm: 201 - 204



5.3 300 - Spielumgebung

UC Nr. 301	Spiel starten			core
Beschreibung	Das System muss dem Spiele	er ermöglic	hen ein neues S	piel zu starten
Verantwortlicher	Spieler			
Beteiligte				
Auslöser	Der Spieler erstellt ein neue	s Spiel.		
Vorbedingungen	UC 201 -> Benutzer muss im	System er	fasst sein	
	UC 501 -> Verbindung muss	hergestellt	sein	
Eingaben	Name und Vorname für die	Benutzersu	iche	
	Standard Alternativ			
	1. Einloggen im Login	1.a.1 "Oh	ne Server"	"Datei" und dann
	Menu	spielen (a	ıls Gast)	"Ende" anklicken.
Ablauf (Szenario)				Ausstieg aus dem
Abiaui (Szellallo)				Use Case und zurück
				zur Startseite
	2. Auswahl eines			
	Spielmodus			
Ausgaben	-			
Nachbedingung	Das Spiel wurde gestartet			

Tabelle 12: UC Nr. 301: Spiel starten



UC Nr. 302	Spiel beenden		core	
Beschreibung	Das System muss dem Spiele	r ermöglic	chen das Spiel zu beenden	
Verantwortlicher	Spieler			
Beteiligte				
Auslöser	Der Benutzer beendet das Sp	iel.		
Vorbedingungen	UC 201 -> Benutzer muss im System erfasst sein			
	UC 301 -> Das Spiel muss gestartet sein			
	UC 501 -> Verbindung muss I	nergestellt	sein	
Eingaben				
	Standard	Alternati	v	
Ablauf (Szenario)	1. "Datei" und dann			
	"Ende" anklicken.			
Ausgaben	-			
Nachbedingung	Das Spiel wurde beendet			

Tabelle 13: UC Nr. 302: Spiel beenden



UC Nr. 303	Punktestand aufzeid	chnen	optional		
Beschreibung	Das System muss das Aufzei	chnen der	erzielten Punkte ermöglichen.		
Verantwortlicher	Spieler				
Beteiligte					
Auslöser	Der Benutzer hat das Spiel e	rfolgreich	peendet		
Vorbedingungen	UC 201 -> Benutzer muss im	UC 201 -> Benutzer muss im System erfasst sein			
	UC 301 -> Das Spiel muss gestartet sein				
	UC 501 -> Verbindung muss hergestellt sein				
Eingaben					
	Standard	Alternati	v		
Ablauf (Szenario)	1. System schreibt nach				
Abladi (Szellálló)	Beendigung eines Spiels	Beendigung eines Spiels			
	den Punktestand nieder				
Ausgaben	Punktestand				
Nachbedingung	Der Spieler kann den neuen	Punktesta	nd im User Menu abrufen		

Tabelle 14UC Nr. 303: Punktestand aufzeichnen

UC Nr. 304	Spielfigur anwähle	n	core		
Beschreibung	Das System muss dem Spie auszuwählen	ler ermöglic	hen die Spielfig	gur im Spiel	
Verantwortlicher	Spieler				
Beteiligte					
Auslöser	Der Benutzer wählt die Figi	ur aus.			
Vorbedingungen	UC 201 -> Benutzer muss ir	n System er	fasst sein		
	UC 301 -> Das Spiel muss g	estartet seir	ı		
Eingaben					
	Standard	Alternati	v		
Ablauf (Szenario)	1. "Spielfigur" anklicken.			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite	
	2. Die Spielfigur ist	2.a.1 Die	Spielfigur		
	ausgewählt	kann nich	nt ausgewählt		
		werden			
		2.a.2 Kon	itaktieren des		
		Supports			
Ausgaben	Angewählte Spielfigur	Angewählte Spielfigur			
Nachbedingung	Ein Spielzug wird gemacht	Ein Spielzug wird gemacht			

Tabelle 15 UC Nr. 304: Spielfigur anwählen



UC Nr. 305	Spielfigur bewegen			core
Beschreibung	Das System muss dem Spieler ermöglichen die Spielfigur im Spiel zu bewegen			
Verantwortlicher	Spieler			
Beteiligte				
Auslöser	Der Benutzer hat die Figur a	ngewählt u	ınd klickt auf eir	n freies Feld.
Vorbedingungen	UC 201 -> Benutzer muss im	System er	fasst sein	
	UC 301 -> Das Spiel muss ge	startet seir	1	
	UC 304 -> Die Figur muss an	_		
	UC 501 -> Verbindung muss	hergestellt	sein	
Eingaben				
		ı		
	Standard	Alternati	v	
Ablant (Cranavia)	1. "Freies Feld" anklicken.			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
Ablauf (Szenario)	2. Die Spielfigur verschiebt sich auf das angeklickte Feld	verschieb	Spielfigur ot sich nicht ngeklickte	
	2.a.2 Kontaktieren des Supports			
Ausgaben	Neuer Standort der Spielfigur			
Nachbedingung	Der andere Spieler ist am Zug			

Tabelle 16 UC Nr. 305: Spielfigur bewegen

UC Nr. 306	Gewinner ermitteln				core
Beschreibung	Das System muss den Gewir	ner ermitte	eln		
Verantwortlicher	Spieler				
Beteiligte					
Auslöser	Der festgelegte Punktestand	l ist erreich	t.		
Vorbedingungen	UC 201 -> Benutzer muss im	System erf	asst sein		
	UC 301 -> Das Spiel muss ge	startet sein			
	UC 307 -> Punktestand einge	eben			
	UC 501 -> Verbindung muss	hergestellt	sein		
Eingaben					
	Standard Alternativ				
	1. Gewinner wird	1.a.1 Der	Gewinner		
	ausgegeben nachdem die	wird nich	t ausgegeben		
Ablauf (Szenario)	Punktezahl von einem				
Abiaui (Szeriario)	Spieler erreicht wurde				
	2. Das Spiel terminiert	2.a.1 Kon	taktieren des	2.a.2 Neuer	
	-> UC 302 Supports Punktestand				
	eingeben -> UC 307				
Ausgaben	Gewinner				
Nachbedingung	Das Spiel ist beendet oder wird mit dem neuen Punktestand weitergeführt				

Tabelle 17 UC Nr. 306: Gewinner ermitteln

UC Nr. 307	Gewinnpunkte eingeben	core				
Beschreibung	Das System muss dem Spieler ermöglic an Gewinnpunkten zu spielen	Das System muss dem Spieler ermöglichen auf eine unterschiedliche Anzahl an Gewinnpunkten zu spielen				
Verantwortlicher	Spieler					
Beteiligte						
Auslöser	Spielmodus wird ausgewählt oder Spie	l gegen Freund gestartet.				
Vorbedingungen	UC 101 -> Login	UC 101 -> Login				
	UC 201 -> Benutzer muss im System er	fasst sein				
	UC 501 -> Verbindung muss hergestell	t sein				
Eingaben						
	Standard	Alternativ				
	1. Gewinn-Punkte werden je nach	1.a.1 Die Gewinnpunkte können				
Ablauf (Szenario)	Spielmodus gesetzt (vordefinierte	manuell eingegeben werden bei				
	Standards) einem Spiel gegen einen Freund					
	2. Das Spiel startet -> UC 301 2.a.1 Kontaktieren des Supports					
Ausgaben	Logging, dass Verbindung hergestellt wurde					
Nachbedingung	Das Spiel kann gestartet werden, weiter mit UC 301					

Tabelle 18 UC Nr. 307: Gewinnpunkte eingeben



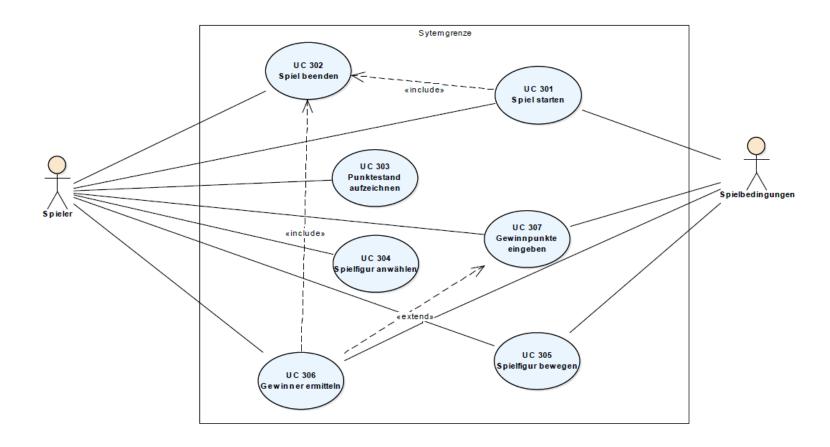


Abbildung 8: UC Diagramm: 301 - 307



5.4 400 - Rangliste

UC Nr. 401	Rangliste	optional	
Beschreibung	Das System soll dem Spieler ermöglich haben.	hen eine Punkteübersicht der Spieler zu	
Verantwortlicher	Spieler		
Beteiligte	-		
Auslöser	Der Benutzer möchte die Punkte der	Benutzer sehen.	
Vorbedingungen	UC 101 -> Login UC 201 -> Benutzer muss im System erfasst sein UC 501 -> Verbindung muss hergestellt sein		
Eingaben	Erstellung eines Reports		
	Standard	Alternativ	
Ablauf (Szenario)	1. Benutzer suchen < <includes 203="" case="" use="">></includes>	"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite	
	2. Das System gibt die Punktezahl		
	des gesuchten Benutzers aus		
Ausgaben	Punkte		
Nachbedingung	Die Anzahl Punkte ist ausgegeben.		

Tabelle 19: UC Nr. 401: Rangliste



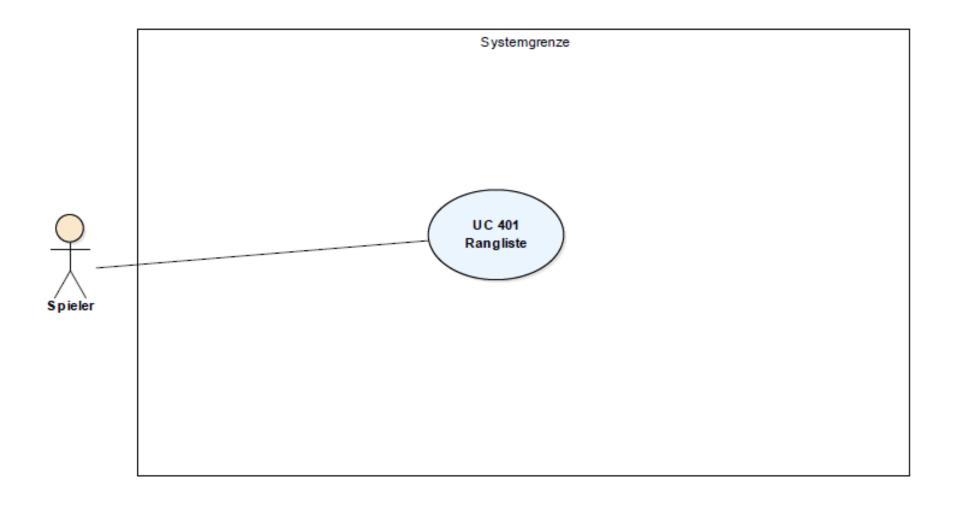


Abbildung 9: UC Diagramm 401



5.5 500 - Technisch

UC Nr. 501	Verbindung Server-	Client		core
Beschreibung	Das System muss dem Spiel Server herzustellen	ermögliche	en eine Verbind	ung vom Client zum
Verantwortlicher	Spieler			
Beteiligte				
Auslöser	Der Benutzer will den Client	mit dem S	erver verbinden	١.
Vorbedingungen				
Eingaben	Netzwerkadresse, Port			
	Standard	Alternati	v	
Ablauf (Szenario)	1. "Verbinden" anklicken			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
Abiaui (Szenario)	2. Der Client verbindet sich über das Netzwerk zum Server	sich nicht Server ve		
		2.a.2 Erne anderer A versuche	Adresse	
Ausgaben	-			
Nachbedingung	Client ist mit Server verbunden			

Tabelle 20 UC Nr. 501: Verbindung Server-Client

UC Nr. 502	Computerspieler (KI)	optional			
Beschreibung	Das System muss dem Spiel ermöglichen mittels programmierter Logik selbständig Spielzüge zu berechnen und vorzunehmen				
Verantwortlicher	Spieler				
Beteiligte					
Auslöser	Der Benutzer will ein Single-Player-Spie	el starten.			
Vorbedingungen	UC 101 -> Login				
	UC 201 -> Benutzer muss im System er	fasst sein			
Eingaben					
	Standard	Alternativ			
Ablauf (Szenario)	1. Einloggen und Spielmodi auswählen.	"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite			
	2. Weiter mit UC 301, mit Computer- Gegner				
Ausgaben					
Nachbedingung	Spiel wird gegen den Computer gespielt				

Tabelle 21 UC Nr. 502: Computerspieler (KI)

UC Nr. 503	Mehrsprachig	optional			
Beschreibung	Das System muss dem Benutzer ermö ändern	Das System muss dem Benutzer ermöglichen die Sprache des Spiels zu ändern			
Verantwortlicher	Spieler				
Beteiligte					
Auslöser	Der Benutzer will eine andere Sprach	e auswählen.			
Vorbedingungen					
Eingaben	Gewünschte Sprache	Gewünschte Sprache			
	Standard	Alternativ			
	Im Spiel "Datei" und dann "Sprache" anklicken				
Ablauf (Szenario)	2. Gewünschte Sprache auswählen	2.a.1 Die Sprache kann nicht geändert werden			
		2.a.2 Kontaktieren des Supports			
	3. Sprache ist geändert				
	·				
Ausgaben	Text erscheint in gewünschter Sprach	Text erscheint in gewünschter Sprache			
Nachbedingung	Sprache geändert				

Tabelle 22 UC Nr. 503: Mehrsprachig

UC Nr. 504	Chat	optional		
Beschreibung	Das System muss dem Benutzer ermöglichen mit dem anderen Spieler via Chat zu kommunizieren			
Verantwortlicher	Spieler			
Beteiligte				
Auslöser	Der Benutzer will mit dem anderen Sp	ieler kommunizieren.		
Vorbedingungen	UC 101 -> Login			
	UC 201 -> Benutzer muss im System e	rfasst sein		
	UC 501 -> Verbindung muss hergestell	t sein		
Eingaben				
	Standard	Alternativ		
	Den gewünschten Text im Text- Feld eingeben			
Ablauf (Szenario)	2. "Absenden" drücken			
	3. Der andere Benutzer kann den	3.a.1 Der Text kann nicht übermittelt		
	Text sehen und antworten	werden		
		3.a.2 Kontaktieren des Supports		
Ausgaben	Der Übermittelte Text wird auf dem anderen Client angezeigt			
Nachbedingung	Antwort schreiben			

Tabelle 23 UC Nr. 504: Chat



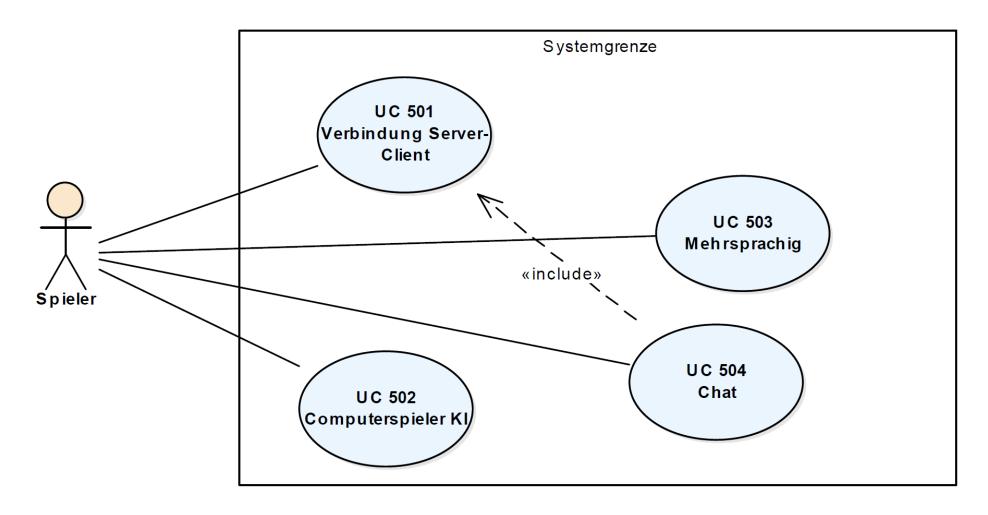


Abbildung 10 UC Diagramm 501 - 504



5.6 600 - Zusätzliche optionale Use Cases

UC Nr. 601	Lobby			optional
Beschreibung	Das System muss dem Benutzer ermöglichen, mit einem Freund direkt vor dem Spiel zu chatten, ohne dass dies die anderen Benutzer sehen und seine Bereitschaft zum Spielen anzumelden			
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Der Benutzer möchte ein Freu	nde-Spie	l starten	
Vorbedingungen	UC 201 -> Benutzer muss im S UC 501 -> Verbindung muss h	•		
Eingaben	Spiel starten			
	Standard	Alterna	ativ	
	1. Aufruf der Lobby			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
	2. Den Button "Go" drücken			
Ablant (Connecia)	3. Das Spiel mit dem Freund startet	3.a.1 D startet	as Spiel nicht	
Ablauf (Szenario)		nochm	en Button als abwählen neut drücken	
		6.a.1 E korrekt	ingaben nicht	
		zur ern	enutzer wird euten e aufgefordert	
Ausgaben	Log Daten für das Freunde-Spiel werden geschrieben			
Nachbedingung	Das Freunde Spiel wurde gestartet			

Tabelle 24: UC Nr. 601: Lobby



UC Nr. 602	Starting the Game w options	ith		optional
Beschreibung	Das System muss dem Benutz Modi zu starten	Das System muss dem Benutzer ermöglichen, das Spiel mit verschiedenen Modi zu starten		
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Der Benutzer startet ein Spiel			
Vorbedingungen		UC 201 -> Benutzer muss im System erfasst sein UC 501 -> Verbindung muss hergestellt sein / Bei dem Modus "Ohne Server"		
Eingaben	Spiel starten			
	Standard	Alternativ		
Ablauf (Szenario)	1. Aufruf des Haupt-Menus			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
	2. Spiel-Variante auswählen und den entsprechenden Button drücken			
	3. Spiel wird gestartet	3.a.1 Das Sp nicht gestar		
		3.a.2 Kontak Sie den Administrat		
Ausgaben	Logdaten für das starten des	Logdaten für das starten des Spiels werden geschrieben		
Nachbedingung				

Tabelle 25: UC Nr. 602: Starting the Game with options



UC Nr. 603	Friends			optional
Beschreibung	Das System muss dem Benutzer ermöglichen, seine Freunde zu verwalten. Dies bedeutet Freunde hinzufügen, diese löschen, Anfragen zu versenden und nachzuschauen wer alles im Spiel registriert ist.			
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Der Benutzer verwaltet seine F	reunde		
Vorbedingungen	UC 201 -> Benutzer muss im Sy UC 501 -> Verbindung muss he	•		
Eingaben	 Add Friends -> hinzufügen eines Freundes Name in Add Friends eingeben -> Freund suchen Remove drücken -> Freund entfernen Play drücken -> Freunde-Spiel starten Accept drücken -> Anfrage annehmen Refuse drücken -> Anfrage abweisen 			
	Standard	Alternat	i.,	
Ablauf (Szenario)	1. Aufruf des Freunde- Menus	Atternat		"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
	2. Die entsprechende Eingabe tätigen – siehe Eingabe			24. 344. 400.00
	3. Die Aktion wie in Eingabe beschrieben wird ausgelöst.	3.a.1 Die nicht aus	e Aktion wird sgelöst	
		3.a.2 Kor Sie den Administ	ntaktieren trator	
Ausgaben	Logdaten für das Einloggen werden geschrieben			
Nachbedingung	Freunde wurden verwaltet			

Tabelle 26: UC Nr. 603: Friends



UC Nr. 604	Tutorial			optional
Beschreibung	Das System muss dem Benutzer ermöglichen, sich über die möglichen Spiele zu orientieren			
Verantwortlicher	Spieler			
Beteiligte	-			
Auslöser	Der Benutzer informiert sich ü	ber die S	piel-Möglichkei	ten
Vorbedingungen	UC 201 -> Benutzer muss im System erfasst sein UC 501 -> Verbindung muss hergestellt sein			
Eingaben	Help >> Tutorial drücken			
	Standard Alternativ			
	1. Aufruf des Tutorial Grundfensters im Spiel unter "Hilfe" – "Einführung"			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite
Ablauf (Szenario)	2. Basic Rules auswählen	2.a.1 Lo auswäl	ong Play nlen	2.b.1 Marathon Play auswählen
	3. Basic Rules werden erklärt		ong Play Rules n erklärt	2.b.2 Marathon Play Rules werden erklärt
	4. "Ok" anklicken	2.a.3 "	Ok" anklicken	2.b.3 "Ok" anklicken
	5. Zurück kehren ins Spiel	2.a.4. Z ins Spie	Zurück kehren el	2.b.4 Zurück kehren ins Spiel
Ausgaben	Logdaten für das Tutorial werden geschrieben			
Nachbedingung	Der Benutzer hat das Tutorial gelesen			

Tabelle 27: UC Nr. 604: Tutorial



UC Nr. 605	Player-Number			optional	
Beschreibung	Das System muss dem Benutzer anzeigen, welcher Spieler er ist				
Verantwortlicher	Spieler				
Beteiligte	-				
Auslöser	Der Benutzer startet ein Spiel				
Vorbedingungen	UC 201 -> Benutzer muss im Sy UC 501 -> Verbindung muss he				
Eingaben	Spiel-Start				
	Standard	Alterna	nativ		
Ablauf (Szenario)	1. Aufruf der Spieloberfläche			"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite	
	2. Auf der rechten Seite des Spielbretts wird angezeigt welcher Spieler der User ist	angeze Spieler 2.a.2 D	s wird nicht igt welcher der User ist as Spiel starten		
				•	
Ausgaben	Anzeige welcher Spieler der Benutzer ist				
Nachbedingung	-				

Tabelle 28: UC Nr. 605: Player-Number

UC Nr. 606	Database		optional	
Beschreibung	Das System muss ermöglichen,	, die Dat	en ordnungsgemäss zu verwalten.	
Verantwortlicher	Programm			
Beteiligte	-			
Auslöser	Benutzerverwaltung, Freundev	verwaltu	ng, Spiel gewonnen	
Vorbodingungen	UC 201 -> Benutzer muss im Sy	/stem er	fasst sein	
Vorbedingungen	UC 501 -> Verbindung muss hergestellt sein			
Eingaben	-			
	Standard	Alterna	ativ	
	1. Die Daten werden richtig			
Ablauf (Szenario)	in der Datenbank			
Ablaul (32ellallo)	gespeichert			
	2. Die Daten können wieder			
abgerufen werden				
Ausgaben	Logdaten für das abrufen oder speichern der Daten in der Datenbank werden			
	geschrieben			
Nachbedingung	Daten sind persistent gespeichert			

Tabelle 29: UC Nr. 606: Database



UC Nr. 607	Al-Different levels		optional		
Beschreibung	· ·	Das System muss dem Benutzer ermöglichen, den KI-Gegner in verschiedenen stärken einzustellen			
Verantwortlicher	Spieler	Spieler			
Beteiligte	-	-			
Auslöser	Der Benutzer startet ein Al-Ga	Der Benutzer startet ein Al-Game			
Vorbedingungen		UC 201 -> Benutzer muss im System erfasst sein UC 501 -> Verbindung muss hergestellt sein			
Eingaben	Spiel start	Spiel start			
	Standard	Alternativ	ternativ		
	1. Aufruf des Haupt-Menus		"Datei" und dann "Ende" anklicken. Ausstieg aus dem Use Case und zurück zur Startseite		
	2. Auswahl vom Gamemodi				
	3. AI Controller auswählen				
Ablauf (Szenario)	4. Player two, resp. Player one einstellen				
	5. Single Player Button resp. Double AI Button anklicken				
	6. Die stärke des Al- Gegners wurde angepasst	6.a.1 Die Stä Al-Gegners k nicht angepa	connte		
		werden			
		6.a.2 Verbin	dung		
		zum Server			
		überprüfen			
Ausgaben	-	Logdaten für den Al-Gegner werden geschrieben			
Nachbedingung	Der Benutzer spielt gegen den	Der Benutzer spielt gegen den Al-Gegner			

Tabelle 30: UC Nr. 607: AI-Different levels



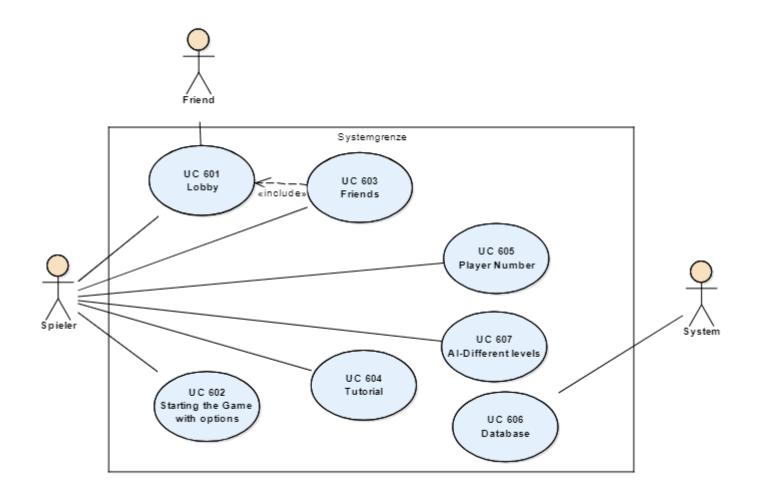


Abbildung 11: Diagramm 601 - 607



6 Endprodukt Use Cases

6.1 Einzel- oder Mehrplayer

Dieses Storyboard zeigt die derzeitige Situation auf, dass ein Spiel zu zweit begonnen werden könnte.

Sollte dies aber mal nicht möglich sein, kann ein Spiel auch im Singleplayer-Modus gespielt werden.

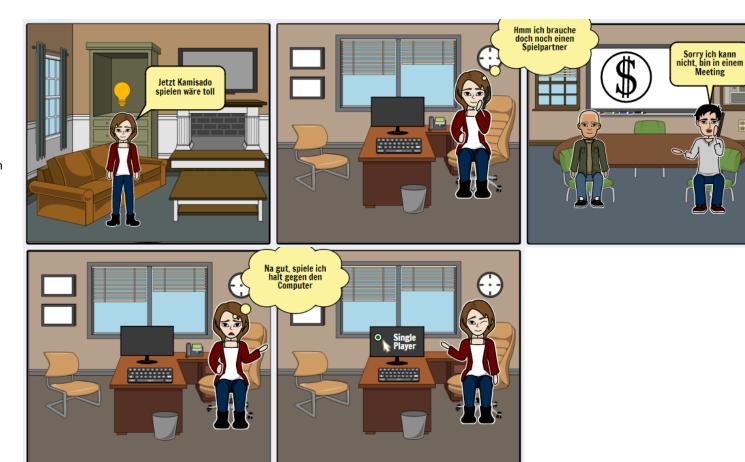


Abbildung 12: StoryBoard möglicher Ablauf



7 Klassendiagramm

Das Klassendiagramm musste aufgrund der Übersichtlichkeit und auch der Grösse auseinander genommen werden. Dies in die verschiedenen einzelnen Ansichten, welche programmiert wurden. Sämtliche Klassen sind mit dem GameMenu_Model als Model selbst verbunden. Die Beziehungen hierzu können in dieser Aufsplittung nicht mehr angezeigt werden.

7.1 Abstract MVC

Abgeleitet sind die Klassen mit Controller, View oder Model jeweils von den abstrakten Klassen

«class abstract»

Model

Model()

«class abstract»
Client

Controller(M, V)

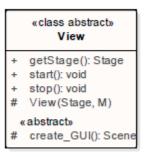


Abbildung 13: Klassendiagramm - abstrakte Klassen



7.2 Allgemeine Klassen

Klassen, welche überall im ganzen Programm gebraucht werden, vor allem aber auch im Ladeprozess des Programmes.

«class» Configuration

- + Configuration()
- + getOption(): String
- save(): void

«class» ServiceLocator

- configuration: Configuration
- logger: Logger
- port: String
- startClient: String
- translator: Translator
- webServer: String

«final»

- APP_CLASS: Class<?>
- APP_NAME: String
- locales: Locale[]

«static»

- serviceLocator. ServiceLocator
- + getAPP_CLASS(): Class<?>
- + getAPP_NAME(): String
- + getConfiguration(): Configuration
- + getLocales(): Locale[]
- + getLogger(): Logger
- + getPort(): String
- + getStartClient(): String
- + getTranslator(): Translator
- + getWebServer(): String
- + setConfiguration(Configuration): void
- + setLogger(Logger): void
- + setPort(String): void
- + setStartClient(String): void
- + setTranslator(Translator): void
- + setWebServer(String): void
- . .

«static»

+ getServiceLocator(): ServiceLocator

«class» Translator

- currentLocale: Locale
- logger: Logger
- ResourceBundle: resourceBundle
- sl: ServiceLocator
- + getCurrentLocale(): Locale
- + getString(): Strng
- + Translator(Translator)

Abbildung 14: Klassendiagramm - allgemeine Klassen



7.3 Splash-Screen

Beim Lade-Prozess des Programmes wird eine Splash-Screen angezeigt, bis alles vollständig geladen wurde.

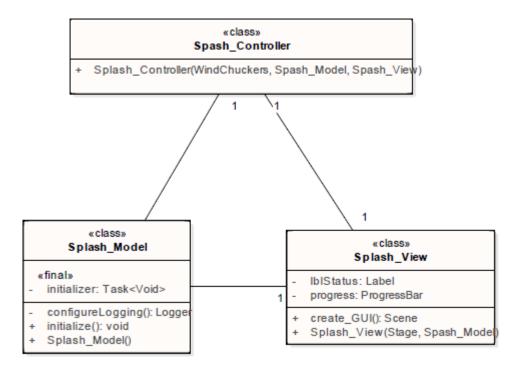


Abbildung 15: Klassendiagramm - Splash Screen



7.4 Client

Damit sich der Client mit dem Server verbinden wird der Client benötigt.

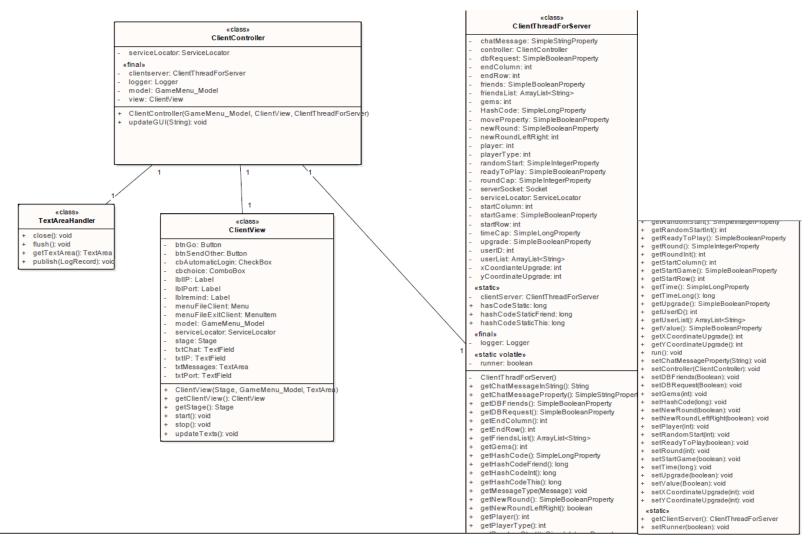


Abbildung 16: Klassendiagramm - Client



7.5 Message

Für den Meldungsaustausch zwischen Client und Server wird die Message-Klasse gebraucht. Diese wird vom Server, sowie vom Client selbst gebraucht.

«class» Message - chat: String - DB: Integer - friendld: int - friendName: String gems: int - hash: long - hashFriend: long - Id: int - id: long - lastName: String MessageType: enum nameList: ArrayList<String> password: String - player: int - preName: String - singlePlayer: boolean something: String time: long - timestamp: long type: MessageType - update: Boolean - userName: String value: Value Value: enum - weightBlockOne: double - weightBlockTwo: double - weightMovesOne: double weightMoveTwo: double weightProgressOne: double weightProgressTwo: double weightSumoBlockOne: double weightSumoBlockTwo: double - weightSumoWinOne: double - weightSumoWinTwo: double weightWinOne: double weightWinTwo: double win: Boolean - xCoordinate1: Integer - xCoordinate2: Integer yCoordinate1: Integer yCoordinate2: Integer «static» messageID: long



```
    createStandardMessage(MessageType): void

    getBlockONE(): double

+ getBlockTWO(): double
+ getChatMessage(): String
+ getDB(): int
+ getFriendId(): int
+ getFriendName(): String
+ getGems(): int
+ getHash(): long
  getHashFriend(): long
  getld(): String
+ getLastName(): String
+ getMessageType(): MessageType
+ getMovesONE(): double
+ getMovesTWO(): double
+ getNameList(): ArrayList<String>
+ getPassword(): String
+ getPlayer(): int
+ getPreName(): String
+ getProgressONE(): double
  getProgressTWO(): double
+ getSinglePlayer(): boolean
+ getSomething(): String
+ getSumoBlockONE(): double
+ getSumoBlockTWO(): double
+ getSumoWinONE(): double
+ getSumoWinTWO(): double
+ getTime(): long
+ getUpdate(): int
+ getUserName(): String
  getValue(): Value
  getWin(): int
+ getWinONE(): double
+ getWinTWO(): double
+ getXCoordinate1(): int
  getXCoordinate2(): int
  getYCoordinate1(); int
+ getYCoordinate2(): int

    Message(MessageType, int, int, int, int, int)

+ Message(MessageType, boolean)
  Message(MessageType, int, int, int, int)
+ Message(MessageType, double, double, double, double, double)
+ Message(MessageType, String)
+ Message(MessageType, String, String)
+ Message(MessageType, double, double,
  Message(MessageType, int)
+ Message(MessageType, int, ArrayList<String>)
+ Message(MessageType, int, int, String, String, String, String)
+ Message(MessageType, int, String, String)
+ Message(MessageType, long)
  Message(MessageType)
+ Message(MessageType, int, int)
+ Message(MessageType, int, int, int)
+ Message(MessageType, long, String)
+ Message(MessageType, long, long)
+ send(Socket): void
+ toString(): String
 «static»
  nextMessageID(): long
   receive(Socket): Message
```

Abbildung 17: Klassendiagramm - Message



7.6 Login

Damit der Benutzer sich einloggen kann wird das Login aufgerufen. In diesem kann ebenfalls auch ein lokales Spiel, ohne Server gestartet werden.

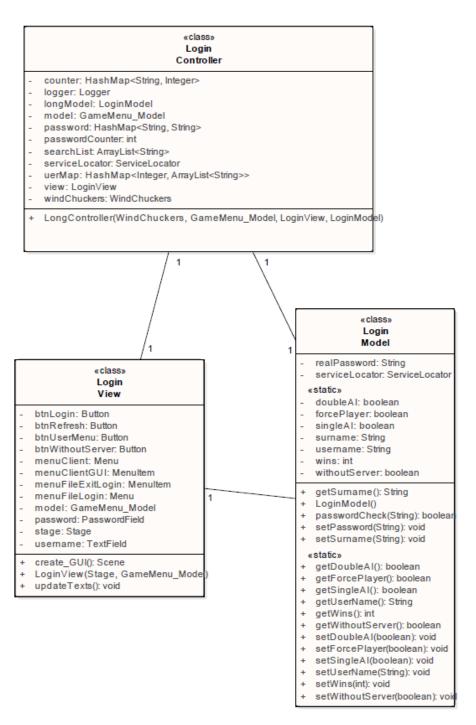


Abbildung 18: Klassendiagramm – Login



7.7 Main Menu

Für die Auswahl des Spiels wird das Main Menu aufgerufen.

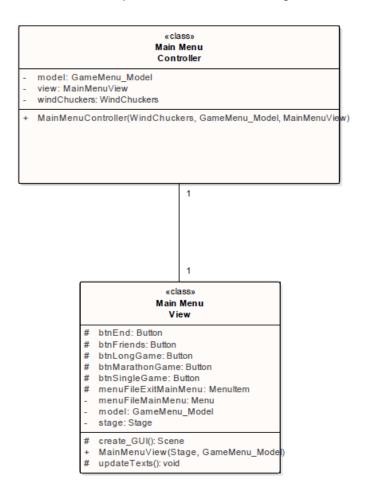


Abbildung 19: Klassendiagramm - Main Menu



7.8 Game Menu

Das Spiel selbst läuft auf dem Game Menu, hierzu werden diverse Klassen gebraucht.

7.8.1 Game Menu View und Controller

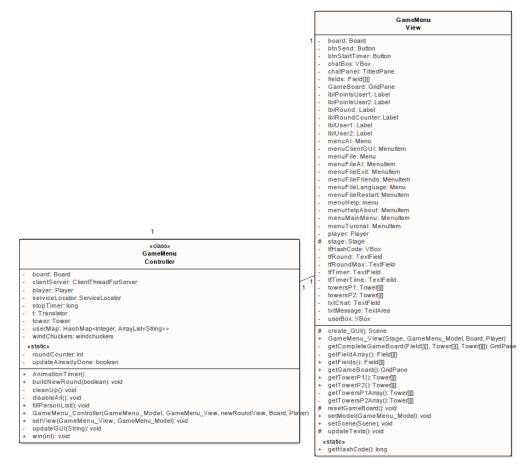


Abbildung 20: Klassendiagramm - Game Menu Client und View

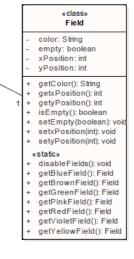


7.8.2 Game Menu Tower, Field, und Player

Diese Klassen sind ebenfalls am GameMenu Model angehängt, als Erweiterung dessen

«class» Tower color: String gems: int model: GameMenu_Model playerNumber: int saveSumoMove: int sumoTower, boolean xPosition: int yPosition: int «static» firstMove: boolean lastField: Field + changeTurn(Field[][], Tower[][], Tower[][], Field, int, Player, Player): void + checkPat(Field[][], Tower[][], Tower[][]): void + checkPatServer(Field[][], Tower[][], Tower[][]): void + checkWin(Field[][], GridPane, Tower[][], Tower[][], Field, Player, Player): void + checkWinSituation(Field[][], GridPane, Tower[][], Tower[][], Field, Player, Player): voi + enableTowerAfterSumoHit(Field[][], Tower[][], Field): void + enableTowers(Field[][], Tower[][], Field): void + getColor(): String + getGems(): int + getPlayerNumber(): int + getSumoTower(): boolean + getxPosition(); int + getyPosition(): int + move(Field[]], GridPane, Tower[][], Tower[][], Field, Player, Player): void + move(Field[][], GridPane, Tower[][], Tower[][], int, int, int, int, int): void + notChangeTurnSumo(Field[][, Tower[][], Tower[][], Field, Player, int, Player): void + setGems(int): void + setPlayerNumber(int): void + setSumoTower(boolean): void + setxPosition(int): void + setyPosition(int): void + showMoves(Field[][], GridPane, Tower[][], Tower[][]): void + showSumoMoves(Field[][], GridPane, Tower[][], Tower[][]): void + sumo1Move(Field[][], GridPane, Tower[][], Tower[][], Field, Player, Player): void + sumo2Move(Field[][], GridPane, Tower[][], Tower[][], Field, Player, Player): void + sumo3Move(Field[][], GridPane, Tower[][], Tower[][], Field, Player, Player): void upgradeTower(Field[][], Tower, int, int, int, Tower[][], Tower[][]): void upgradeTowerFromServer(Field[][], Tower, int, int, int, Tower[][], Tower[][]): void «static» + disableTowers(Tower[][]): void + getBlueTower(): Tower + getBrownTower(): Tower + getGreenTower(): Tower + getOrangeTower(): Tower + getPinkTower(): Tower + getRedTower(): Tower + getVioletTower(): Tower + getYellowTower(): Tower

¹Abbildung 22: Klassendiagramm - Tower



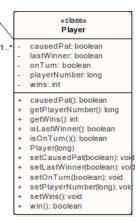


Abbildung 21: Klassendiagramm -Field und Player



7.8.3 GameMenu Model

Diese Klasse ist bei allen Klassen auf dem Client eingebunden, muss aufgrund der grösse aber alleine Angezeigt werden.

«class» GameMenu_Model

- dbRequest: SimpleBooleanProperty
- + disableSingleGame: SimpleBooleanProperty
- + enableFriendsGame: SimpleBooleanProperty
- endColumn: int
- endRow: int
- gems: int
- + HashCode: SimpleLongProperty
- + moveProperty: SimpleBooleanProperty
- player1: Player
- player2: Player
- playerList: ArrayList<Player>
- PlayerType: int
- # serviceLocator: ServiceLocator
- socket Socket
- startColumn: int
- startRow: int
- + thisUsersNameProperty: SimpleStringProperty
- userID: int
- userList: ArrayList<String>
- userMap: HashMap<Integer, ArrayList<String>>

«static final»

- + BLUE: String
- + BROWN: String
- + DIMENSIONS: int
- + GREEN: String
- + MAX_FIELDS: int
- + ORANGE: String
- + PINK: String
- + RED:String
- + VIOLET: String
- + YELLOW: String

«static

- friends: Hashtable<Integer, String>
- friendsRequest: Hashtable<Integer, Boolean>
- gameMode: int
- gameModel: GameMenu_Model
- + gameStart: boolean
- newRoundLeftRight boolean
- otherHash: long
- playerCausedPat: int
- users: Hashtable<Integer, String>
- + Winner: IntegerProperty
- + acceptFriendsRequest(String, String): void
- + addFriend(String, String); void
- + buildNewPlayer(long): void
- + connect(String, Integer): boolean
- GameMenu_Model()
- + getDBRequest(): SimpleBooleanProperty
- + getDisableSingeleGame(): SimpleBooleanProperty
- + getEnableFriendsGame(): SimpleBooleanProperty
- + getEndColumn(): int
- + getEndRow(): int
- + getFriends(): ArrayList<String>
- + getFriendsRequests(): ArrayList<String>
- + getFriendsSearchResults(): ArrayList<String>
- + getGameMode(): int
- + getGems(): int



- + getHashCode(): SimpleLongProperty
- getMoveProperty(): SimpleBooleanProperty
- + getPlayer1(): Player
- + getPlayer2(): Player
- + getPlayerList(): ArrayList<Player>
- + getPlayerType(): int
- + getStartColumn(): int
- + getStartRow(): int
- + getUserID(): int
- + getUserList(): ArrayList<String>
- + getUserMap(): HashMap<Integer, ArrayList<String>>
- + getuserName(): SimpleStringProperty
- + getUserNameString(): String
- + makeMove(int, int, int, int, int): void
- + makeUserList(ArrayList<String>): void
- + messageConstructorForBuildBinom(String, String): void
- + messageConstructorForBuildCapsule(long, String): void
- + messageConstructorForCoordinate(int, int, int, int, int): void
- + messageConstructorForDB(int): void
- + messageConstructorForDBInsertUpdate(int, int, String, String, String, String): void
- + messageConstructorForDBUpdate(int, int): void
- + messageConstructorForEnd(): void
- + messageConstructorForError(): void
- + messageConstructorForFriendsInsert(int, int, int): void
- + messageConstructorForName(long, String): void
- + messageConstructorForNewRound(boolean): void
- + messageConstructorForTime(long): void
- + messageConstructorForUpdate(int, int, int, int): void
- + messageConstructorForWaiter(long): void
- + messageConstructorForWin(int, String): void
- + messageContructorForChat(String): void
- + messageContstructorForAlDouble(double, double, double, double, double, double, double, double, double, double)
- + messageContstructorForAlSingle(double, double, double, double, double, double, double): void
- + refuseFriendsRequest(String): void
- removeFriend(String): void
- sendMessaeg(Message): void
- + setDBRequest(Boolean): void
- + setDisableSingeleGame(boolean): void
- + setEnableFriendsGame(boolean): void
- + setFriends(ArrayList<String>): void
- + setGameMode(int): void
- + setGems(int): void
- + setHashCode(int): void
- + setHashCode(long): void
- + setMoveProperty(Boolean): void
- + setPlayer(long, long): void
- setUserID(int): void
- setUserList(ArrayList<String>): void
- + setUserName(String): void

«static»

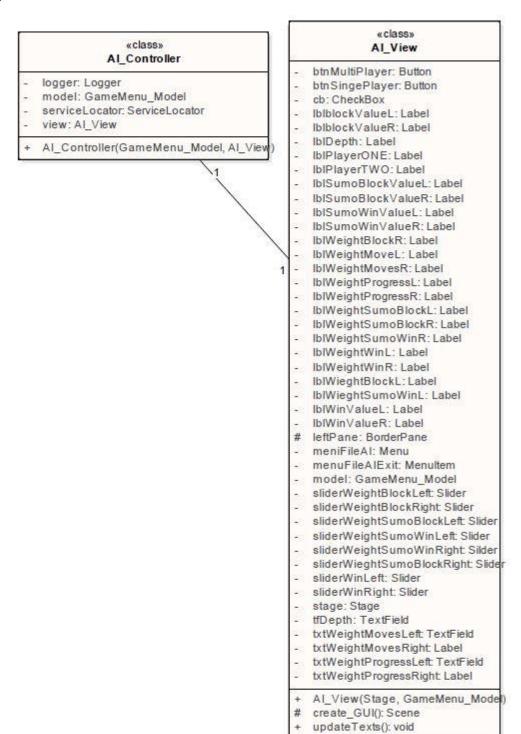
- + getGameModel(): GameMenu_Model
- + getNewRoundLeftRight(): boolean
- getOtherHash(): long
- + setNewRoundLeftRight(boolean): void
- + setOtherHash(long): void

Abbildung 23: Klassendiagramm - GameMenu_Model



7.9 Al Steuerung

Für die Steuerung der Al aus dem Game Menu.





7.10 Friends

Für das Spiel mit Freunden wird das Friends-Menu benötigt.

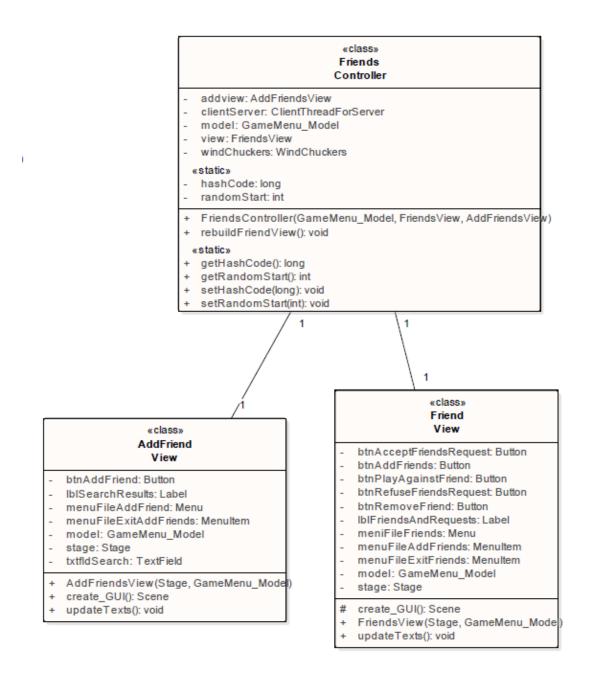


Abbildung 25: Klassendiagramm - Friends



7.11 Lobby

Sobald sich zwei Freunde entschieden haben miteinander zu spielen, wartet das Programm das "Go" beider Spieler in der Lobby ab.

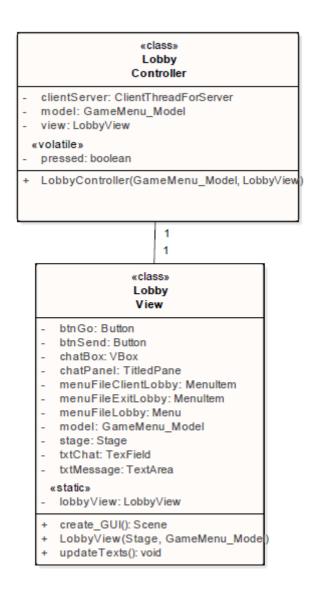
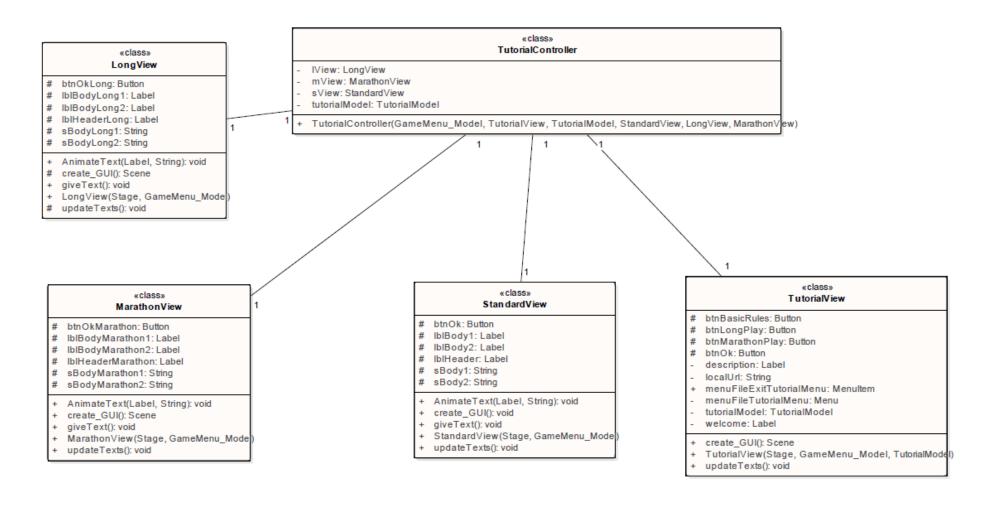


Abbildung 26: Klassendiagramm - Lobby



7.12 Turorial

Möchte der Spieler wissen was den einzelnen Spielen für Regeln zu Grunde liegt, öffnet er das Tutorial.





7.13 User Menu

Damit neue Users angelegt, oder bestehende Users verwaltet werden können benötigt es das User Menu.

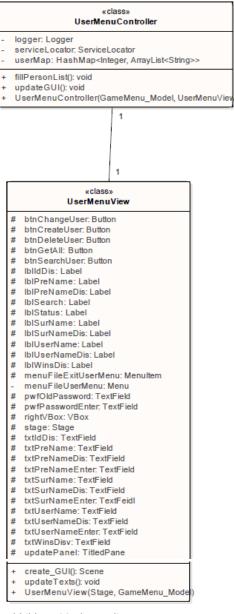


Abbildung 28:Klassendiagramm - User Menu



7.14 Aufforderung zur neuen Runde

Wenn eine Runde zu Ende ist, wird der Spieler aufgefordert zu wählen wie er die neue Runde gestalten möchte.

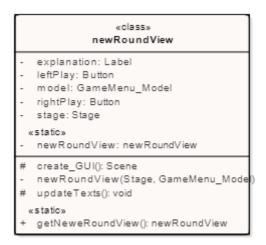


Abbildung 29: Klassendiagramm - neue Runde



7.15 Steuerung der Menus

Für die Steuerung der Menu's, sowie das Start-Programm des Clients.

«class» WindChuckers

- addFriendsView: AddFriendsView
- aiController: Al_Controller
- aiView: Al_View
- board: Board
- clientController: ClientController
- clientserver: ClientThreadForServer
- clientView: ClientView
- controller: GameMenu_Controller
- friendscontroller: FreindsController
- friendsView: FriendsView
- lobbyView: LobbyView
- loginModel: LoginModel
- loginView: LoginView
- mainMenuView: mainMenuView
- model: GameMenu_Model
- newRoundView: newRoundView
- player: Player
- serviceLocator: ServiceLocator
- splashView: Splash_View
- userMenuView: userMenuView
- view: GameMenu_View

«static»

- mainProgram: WindChuckers
- getGameMenuView(): GameMenu_View
- + getLoginView(): LoginView
- + getStartetClient(): void
- + init(): int
- + main(String[]): void
- + start(Stage): void
- + startAI(): void
- + startApp(): void+ startClient(): void
- + startFriends(): void
- + startGameMenuViewe(): void
- + startLobby(): void
- + startLogin(): void
- + startMainMenu(): void
- + startMainMenuView(): void
- + startNewRound(): void
- + startTutorial(): void
- + startUserMenu(): void
- + stop(): void
- WindChuckers()

«static»

- # getMainProgram(): WindChuckers
- getWindChuckers(): WindChuckers

Abbildung 30: Klassendiagramm - Steuerung Programm



7.16 Server

Der Server als Gegenseite verwaltet das Spiel auf dem Server selbst.

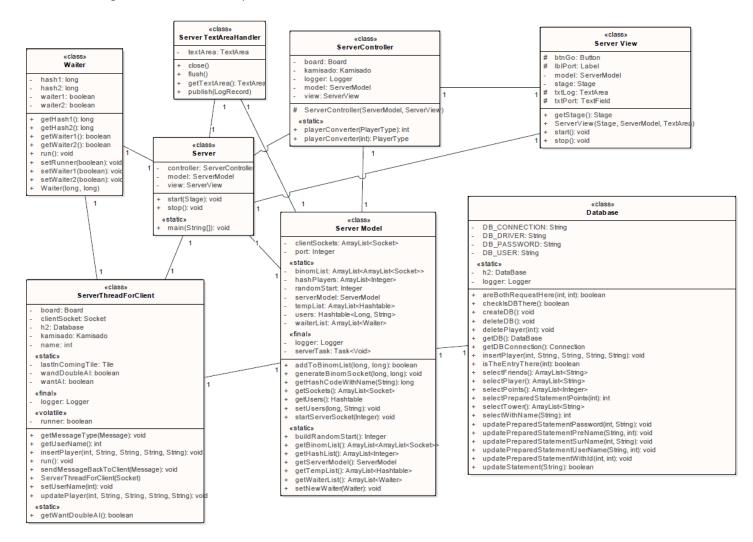


Abbildung 31: Klassendiagramm - Server



Die Kommunikation zwischen Client und Server teilt sich die Klasse "Message". Das Format für die Kommunikation ist XML, dazu wird die "Simple XML"-API verwendet.

In der Datenbank werden hauptsächlich die Spieler-Daten gespeichert. Also Id, Name, Vorname, Punkte, SumoTower und wie viele Spiele bereits gewonnen wurden für das Ranking.

7.17 AI

Der Computer-Gegner selbst berechnet sich in der AI. Verbunden sind die beiden Teile durch die Klassen PlayerType und MiniMaxAlphaBeta.

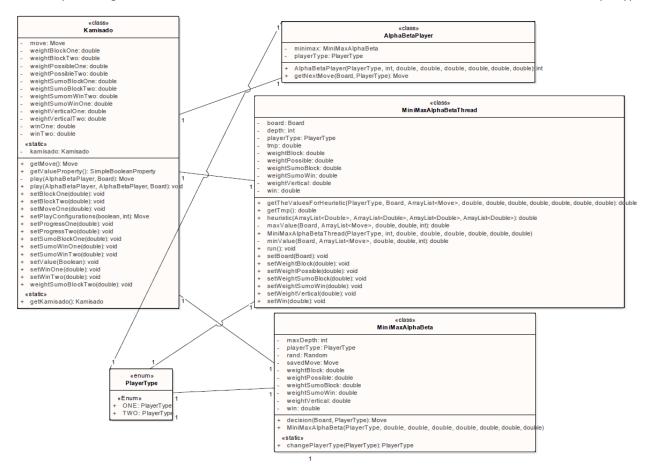


Abbildung 32: Klassendiagramm - Al Teil 1

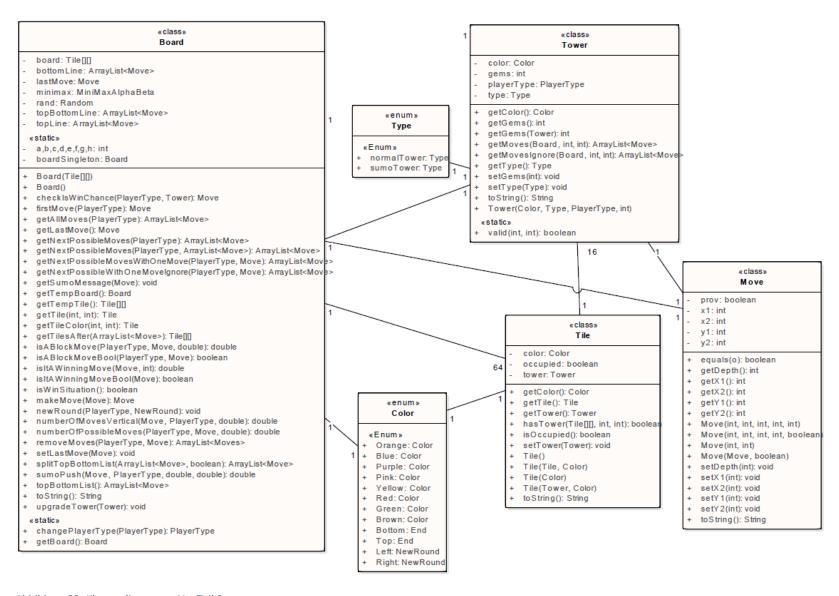


Abbildung 33: Klassendiagramm AI – Teil 2



8 Entity-Relationship-Modell

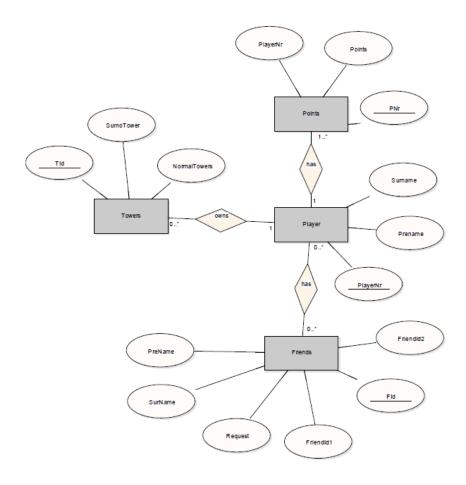


Abbildung 34: ERM-Model



9 Relationales-Modell

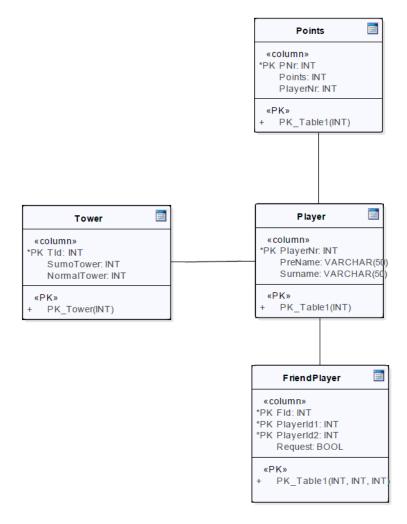


Abbildung 35: Relationales Modell



10 Verteilungsdiagramm

Das System benötigt eine 3-Tier-Aufteilung, damit sämtliche Funktionen korrekt ausgeführt werden. Auf dem Client selbst läuft die View, welcher die Rückgabe des Webservers anzeigt. Auf dem Webserver wird die gesamte Logik ausgeführt. Der Server kann die Ansicht an mehrere Clients senden, sprich mehrere User können gleichzeitig auf das Programm zugreifen.

Es gibt eine interne Datenbank. In der internen werden die Benutzerdaten und der Punktestand abgelegt.

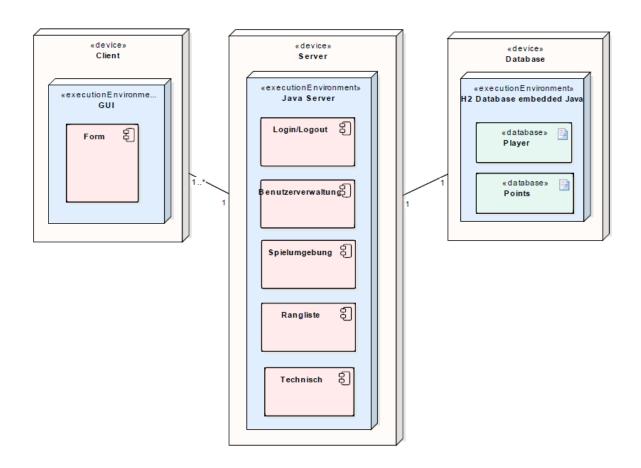


Abbildung 36: Verteilungsdiagramm



11 Qualitätsanforderungen

Zur Ergänzung der funktionalen Anforderungen von Kapitel 5 ergeben sich folgende nicht funktionale Anforderungen:

Anforderungen	Messkriterien
Verfügbarkeit	Das System muss eine jährliche Verfügbarkeit von 90% aufweisen
Abänderbarkeit	Eine neue Komponente soll in weniger als einem Monat designt, entwickelt und getestet werden können.
Effizienz	Die Reaktion des Systems soll unter 5 Sekunden sein. Eine Änderung resp. ein Update soll in weniger als 10 Sekunden erledigt sein.
Erlernbarkeit	Der Benutzer soll das System innert 10 Minuten erlernt haben.
MTTF (Mean time to failure)	Das System soll min. 50 Stunden laufen, bis der erste Fehler auftritt.
MTTR (Mean time to repair)	Der aufgetretene Fehler muss innert 5 Arbeitstagen repariert sein.

Tabelle 31: Qualitätsanforderungen

12 GUI-Bestandteile

12.1 Login



Abbildung 37: GUI Login

12.2 Lobby

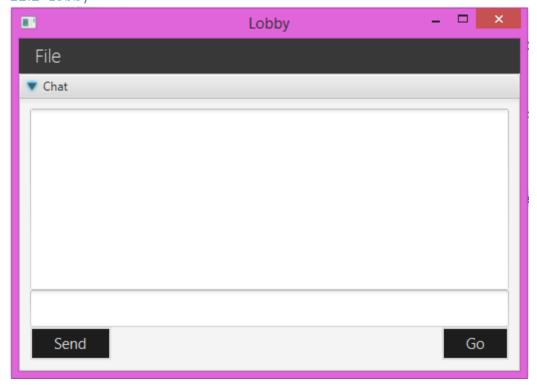


Abbildung 38: GUI Lobby



12.3 Main Menu



Abbildung 39: GUI Main Menu

12.4 Friends Menu

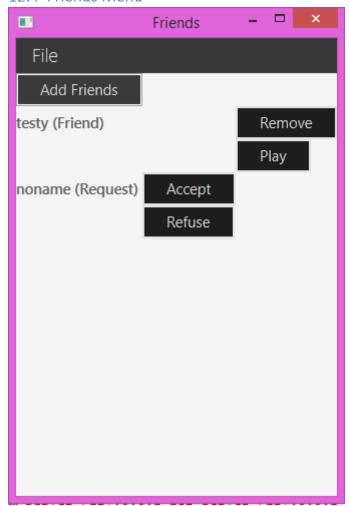


Abbildung 40: GUI Friends Menu



12.5 Add Friends

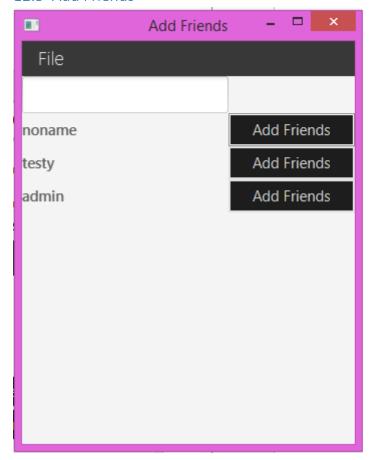


Abbildung 41: GUI Add Friends

12.6 Game Menu

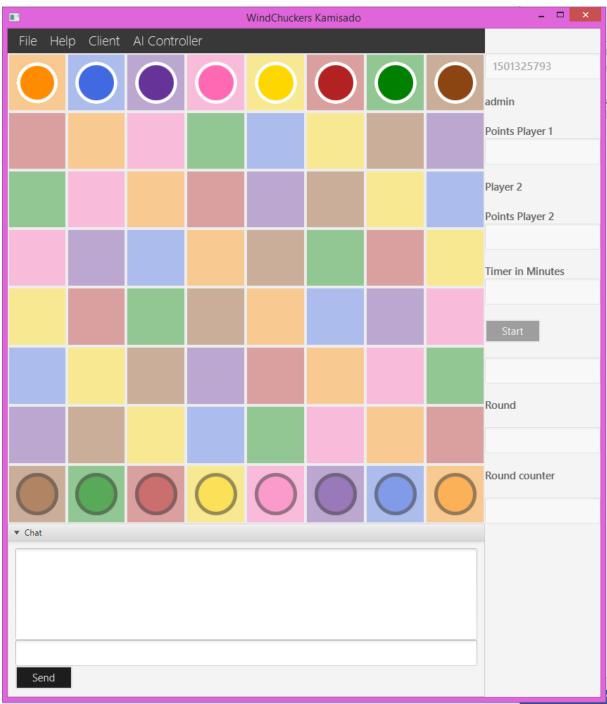


Abbildung 42: GUI Game Menu



13 Testkonzept

13.1 Einführung

Bei dem vorliegenden Dokument handelt es sich um den Testplan der Gruppe WindChuckers, welcher im Rahmen des IT-Projektes das an der FHNW durchgeführt wird, erstellt wird.

13.2 Zweck

Zweck des Testplans ist es, ein organisiertes und systematisches Testen des IT-Projektes zu ermöglichen. Es werden die zu testenden Komponenten des Systems benannt und dazu adäquate Testfälle entworfen. Auch werden Kriterien gesetzt, wann der Test als erfolgreich durchgeführt gilt und wann er abgebrochen werden muss.

13.3 Bezeichnen der Testfälle

Die Testfälle sind nach folgendem Prinzip aufgebaut:

- Die Reihenfolge der Testfälle stimmt mit der Reihenfolge der Use Cases überein
- Sämtliche Ausprägungen werden aufgelistet
- Die Testfälle erhalten immer eine Testfallnummer

13.4 Abnahme- und Testkriterien

Wann wird das Testen beendet? Die angegebenen Kriterien müssen objektiv prüfbar sein.

Die "wahre" Fehlerzahl einer Software ist unbekannt. Daher müssen im Vorfeld Kriterien bestimmt werden, die festlegen, in welchem Umfang das Testen stattfindet und wann die Tests beendet werden können.

Es gibt für jeden Test ein erwartetes Ergebnis. Trifft dieses ein gilt der Test als erfolgreich und ist abgeschlossen. Bei einem unerwarteten Ergebnis wird der Test zu einem späteren Zeitpunkt wiederholt.

13.4.1 Erfassung Testfälle und Anweisungsabdeckung

Anweisungsabdeckung: Tests wird die Anweisungsabdeckung gemessen. Wir streben eine Anweisungsabdeckung von 90 Prozent an. Dies wird bestimmt dadurch, dass von jedem Use Case sämtliche aufgeführte Ausprägungen als Testfall erfasst werden. Ebenfalls kann damit festgestellt werden, welche Zeilen des Codes vom Testfall durchlaufen wurden und welche nicht. Die Tests müssen die angegebene Anweisungsabdeckung erreichen, vorher darf nicht abgebrochen werden.

13.4.2 Unit-Tests

Zusätzlich werden durch Unit-Tests mit jUnit die Testfälle aufgebaut und eingerichtet. Folgende Ausprägungen werden pro Klasse erfasst:

- @BevorClass (Ausführung vor den Tests, Vorbereitung der Daten)
- @BevorMethods (Ausführung vor den einzelnen Tests)
- @AfterMethods (Nach jedem einzelnen Testfall)
- @AfterClass (Wenn alle Testfälle durchgelaufen sind)

Dies stellt sicher, dass die Klasse in sich komplett ist und funktioniert.

Die einzelnen Testfälle werden alle als separierte Tests mit jUnit erfasst, dies nach den oben beschriebenen Ausprägungen, welche bereits schriftlich festgehalten wurden.

Diese Methode stellt sicher, dass bereits geschriebener und implementierter Code nicht später Fehler auswirft, welche nicht erkannt werden bei neuen Implementierungen.



13.4.3 Testergebnisse

Testergebnisse: Erfolgreiche Tests (d.h. Tests, die einen Fehler gefunden haben), werden in Fehlerklassen entsprechend Tabelle kategorisiert. Pro 100 Zeilen Code dürfen nur Fehler mit einer Wertigkeitssumme von 100 auftreten. Das Testen darf nicht beendet werden, solange noch Fehler der Klassen 3 oder 4 auftreten. Die zur Verfügung stehende Zeit ist begrenzt. Es ist sinnvoll, frühzeitig und regelmassig den Fortschritt der Testphase zu beurteilen.

Fehlerklasse	Beschreibung	Wertigkeit
1	unwesentlich: Fehler, der keine sichtbaren Schäden	1
	verursacht, aber trotzdem korrigiert werden muss	
2	gering: Fehler, der das System in seiner Leistung	10
	einschränken oder dessen Verfügbarkeit reduzieren könnte	
3	kritisch: Fehler, der schwere Schäden am System	20
	verursachen kann	
4	katastrophal: Fehler, der die Unterbrechung des Systems	100
	verursachen kann bzw. das System unbrauchbar macht	

13.5 Vorgehen

13.5.1 Beginn Projekt

Ein Projekt wird auf GitHub.com eröffnet. Dazu wird ein leerer Model-View-Controller erstellt und direkt "gepusht", sodass die Entwickler, welche am Projekt arbeiten einen Branch von der Repository auf ihren Laptop holen können und von dort aus arbeiten. So starten alle von derselben Basis.

13.5.2 Aufbau Testing

Wie in Abschnitt 16.4.1 beschrieben werden sämtliche Ausprägungen der erfassten Use Cases als Testfälle erfasst und festgehalten.

13.5.3 Unit-Tests

Wie in Abschnitt 16.4.2 beschrieben werden zu allen erfassten Ausprägungen, parallel zum entstehenden Code für die Umsetzung der Use Cases, Unit-Tests geschrieben. Somit wird am Schluss jede Methode einen Unit-Test haben, welcher garantiert, dass keine Fehler im Programm vorhanden sind.

13.5.4 Implementierung des neuen Codes

Jeweils bevor ein "commit" gemacht wird, muss mittels Unit-Tests sichergestellt werden, dass das Programm läuft und sich so verhält wie es die Anforderungen definieren. Es darf kein Unit-Test auf "Fehler" gelaufen sein.

Erst dann darf das Programm gepusht werden und in die Repository aufgenommen werden.

13.5.5 Neuster Stand

Sobald dies der Fall ist, sollen die anderen Benutzer nun die aktuellste Repository via "fetch" abholen, sodass ihre Version auf dem neusten Stand ist und dass auch bezüglich dem Testing keine Lücken entstehen.

13.5.6 Freigabe Komplettpacket

Die einzelnen Use Cases werden mittels Testprotokoll, sowie Unit-Tests getestet und auch geführt. Erst wenn das Testprotokoll angibt, dass sämtliche Testfälle ok sind kann eine Version des Programms freigegeben werden.



13.6 Testprotokoll

Auftraggeber FHNW, Olten Projektleiter Weber Lukas Autor Weber Lukas

Klassifizierung Intern
Status In Arbeit

Änderungsverzeichnis

Datum	Version	Änderung	Autor
08.03.2017	1.0	Erstellung	Wel
28.03.2017	1.1	Komplettierung der Ausprägungen	Wel
02.08.2017	1.2	Testdurchführung	Wel



Testfallsammlung:

UseCase-Gruppe	UC Nr Bezeichnung	Testfall	Kurzbeschreibung	Resultat
100 Login/Logout				
	101 Login	Standard	Einloggen des Users im System	Ok
		Alternativ 1	Passwort / Username falsch eingeben	Ok
		Alternativ 2	Abbruch	Ok
	102 Logout	Standard	Abmelden des Users	Ok
		Alternativ 1	Abbruch	Ok
200 Benutzerverwaltung				
	201 Benutzer hinzufügen	Standard	Benutzer hinzufügen	Ok
		Alternativ 1	Benutzerdaten unvollständig	Ok
		Alternativ 2	Benutzer ist bereits erfasst	Ok
		Alternativ 3	Abbruch	Ok
	202 Benutzer bearbeiten	Standard	Bearbeiten des Benutzers	Ok
		Alternativ 1	Keine oder unvollständige Daten eingeben	<mark>Ok</mark>
		Alternativ 2	Abbruch	Ok
	203 Benutzer suchen	Standard	Benutzer suchen	Ok
		Alternativ 1	Benutzer wird nicht gefunden (nicht erfasst)	OK
		Alternativ 2	Nicht gefundener Benutzer direkt hinzufügen	OK
		Alternativ 3	Verbindungsfehler	Ok
		Alternativ 4		Ok
	204 Benutzer löschen	Standard	Benutzer löschen	<mark>Ok</mark>
		Alternativ 1	Abbruch nach bereits eingebenen Daten	<mark>Ok</mark>
		Alternativ 2		Ok



UseCase-Gruppe	UC Nr	Bezeichnung	Testfall	Kurzbeschreibung	Resultat
300 Spielumgebung					
	301	Spiel starten	Standard	Spiel starten	<mark>Ok</mark>
			Alternativ 1	Benutzer wird nicht gefunden	<mark>Ok</mark>
			Alternativ 2	Als Gast spielen	Ok
			Alternativ 3	Abbruch	Ok
	302	Spiel beenden	Standard	Spiel beenden	Ok
		•	Alternativ 1	·	Ok
			Alternativ 2		Ok
		Punktestand			
	303	aufzeichnen	Standard	Nach Beendigung wird Punktestand aufgezeichnet	Ok
	304	Spielfigur auswählen	Standard	Spielfigur kann angewählt werden	Ok
		· -	Alternativ 1	Spielfigur kann nicht ausgewählt werden	Ok
			Alternativ 2	Abbruch	Ok
	305	Spielfigur bewegen	Standard	Freies Feld kann angeklickt werden und Spielfigur verschiebt sich	Ok
			Alternativ 1	Spielfigur verschiebt sich nicht	Ok
			Alternativ 2		Ok
	306	Gewinner ermitteln	Standard	Der Gewinner wird festgelegt	Ok
	000	Gowiii i o o o o o o o o o o o o o o o o	Alternativ 1	Der Gewinner wird nicht ausgegeben	Ok
				Weiterspielen nach Gewinnerausgabe	Ok
	207	Cowinnnunkto oingchan	Standard	Cowinn Bunkto singshon	Ok
	307	Gewinnpunkte eingeben	Standard	Gewinn-Punkte eingeben	Ok
			Alternativ 1	Gewinn-Punkte können nicht eingegeben werden	Ok



UseCase-Gruppe	UC Nr	Bezeichnung	Testfall	Kurzbeschreibung	Resultat
400 Rangliste					
	401	Rangliste	Standard	Gesammelte Punkte der Spieler ersichtlich	Ok
			Alternativ 1	Abbruch	Ok
500 Technisch					
		Verbindung Server-			
	501	Client	Standard	Die Verbindung kann hergestellt werden	Ok
			Alternativ 1	Die Verbindung kann nicht hergestellt werden	<mark>Ok</mark>
			Alternativ 2	Abbruch	Ok
	502	Computerspieler (KI)	Standard	Das Spiel wird mit Gegener gestartet	Ok
			Alternativ 1	Abbruch	Ok
	503	Mehrsprachig	Standard	Die Sprache kann geändert werden	Ok
			Alternativ 1	Die Sprache kann nicht geändert werden	Ok
	504	Chat	Standard	Die Chatnachrichten können gesendet und emfpangen werden	Ok
			Alternativ 2	Die Chatnachrichten können nicht gesendet und empfangen werden	Ok



UseCase-Gruppe	UC Nr	Bezeichnung	Testfall	Kurzbeschreibung	Resultat
600 Zusätzlich optionale Use Cases					
	601	Lobby	Standard	Chat Funktion verwenden können	<mark>Ok</mark>
			Alternativ 1	Abbruch	Ok
		Starting the game with			_
	602	options	Standard	Spielvariante auswählen und Button drücken	Ok
			Alternativ 1	Das Spiel wird nicht gestartet	Ok
			Alternativ 2	Abbruch	Ok
	603	Friends	Standard	Friendsmenu aufrufen und entsprechende Eingabe tätigen	Ok
			Alternativ 1	Die Aktion wird nicht ausgelöst	Ok
			Alternativ 2	Abruch	Ok
	604	Tutorial	Standard	Aufrufen des Tutorial Fensters	Ok
			Alternativ	Entsprechender Modi auswählen	Ok
	605	Player-Number	Standard	Anzeige welche Nummer der User ist	Ok
			Alternativ 1	Nummer vom User wird nicht angezeigt	Ok
				Die Daten werden in der Datenbank gespeichert und sind wieder	
	606	Database	Standard	abrufbar	Ok
	607	AI-Different levels	Standard	Al Controller aufrufen und Stärke einstellen	Ok Ok
			Alternativ 1	Stärke nicht anpassbar	Ok



Testdurchführung und Testergebnis

Tester	L.Weber
Datum Testdurchführung	02.08.2017
Fehlerklasse (Testergebnis)	Nun alles i.O
Fehlerbeschreibung	Keine Fehler mehr gefunden



14 Abbildungsverzeichnis

ABBILDUNG 1: STAKEHOLDER IN DER SYSTEMGRENZE	6
ABBILDUNG 2: STAKEHOLDER: WICHTIGKEIT UND EINFLUSS	8
ABBILDUNG 3: STAKEHOLDER: WICHTIGKEIT UND MOTIVATION	9
ABBILDUNG 4: SYSTEMABGRENZUNG	10
ABBILDUNG 5: USE CASE MODELLIERUNG ÜBERSICHT	12
ABBILDUNG 6: UC DIAGRAMM 101 - 102	15
ABBILDUNG 7: UC DIAGRAMM: 201 - 204	20
ABBILDUNG 8: UC DIAGRAMM: 301 - 307	26
ABBILDUNG 9: UC DIAGRAMM 401	28
ABBILDUNG 10 UC DIAGRAMM 501 - 504	31
ABBILDUNG 11: DIAGRAMM 601 - 607	38
ABBILDUNG 12: STORYBOARD MÖGLICHER ABLAUF	39
ABBILDUNG 13: KLASSENDIAGRAMM - ABSTRAKTE KLASSEN	40
ABBILDUNG 14: KLASSENDIAGRAMM - ALLGEMEINE KLASSEN	41
ABBILDUNG 15: KLASSENDIAGRAMM - SPLASH SCREEN	42
ABBILDUNG 16: KLASSENDIAGRAMM - CLIENT	43
ABBILDUNG 17: KLASSENDIAGRAMM - MESSAGE	45
ABBILDUNG 18: KLASSENDIAGRAMM – LOGIN	46
ABBILDUNG 19: KLASSENDIAGRAMM - MAIN MENU	47
ABBILDUNG 20: KLASSENDIAGRAMM - GAME MENU CLIENT UND VIEW	48
ABBILDUNG 21: KLASSENDIAGRAMM - FIELD UND PLAYER	49
ABBILDUNG 22: KLASSENDIAGRAMM - TOWER	49
ABBILDUNG 23: KLASSENDIAGRAMM - GAMEMENU_MODEL	51
ABBILDUNG 24: KLASSENDIAGRAMM - AI CONTROLLING	52
ABBILDUNG 25: KLASSENDIAGRAMM - FRIENDS	53
ABBILDUNG 26: KLASSENDIAGRAMM - LOBBY	54
ABBILDUNG 27: KLASSENDIAGRAMM – TUTORIAL	55
ABBILDUNG 28:KLASSENDIAGRAMM - USER MENU	56
ABBILDUNG 29: KLASSENDIAGRAMM - NEUE RUNDE	57
ABBILDUNG 30: KLASSENDIAGRAMM - STEUERUNG PROGRAMM	58
ABBILDUNG 31: KLASSENDIAGRAMM - SERVER	59
ABBILDUNG 32: KLASSENDIAGRAMM - AI TEIL 1	60
ABBILDUNG 33: KLASSENDIAGRAMM AI – TEIL 2	61
ABBILDUNG 34: ERM-MODEL	62
ABBILDUNG 35: RELATIONALES MODELL	63
ABBILDUNG 36: VERTEILUNGSDIAGRAMM	64
ABBILDUNG 37: GUI LOGIN	66
ABBILDUNG 38: GUI LOBBY	66
ABBILDUNG 39: GUI MAIN MENU	67
ABBILDUNG 40: GUI FRIENDS MENU	67
ABBILDUNG 41: GUI ADD FRIENDS	68
ABBILDUNG 42: GUI GAME MENU	69
ABBILDUNG 43: SINGLE PLAYER	81
ABBILDUNG 44: DOUBLE AI	81
ABBILDUNG 45: CLIENT-MENU, ANSICHT ALS "ADMIN-USER" MIT ALLEN MESSAGE-TYPEN	82
ABBILDUNG 46: SUCHEN NACH DEM SPIELER "TESTY"	83
ABBILDUNG 47: ALLE SPIELER ANZEIGEN	83
ABBILDUNG 48: DAS MENU DER FELDER FÜR ÄNDERUNGEN ODER NEUERZEUGUNGEN IST AUFGEKLAPPT	84
ABBILDUNG 49: LOGIN MENU MIT KORREKTER EINGABE EINES AKTIVEN BENUTZERS	85
ABBILDUNG 50: LOGIN MENU MIT EINGABE EINES BENUTZER, DEN ES NICHT GIBT	85
ABBILDUNG 51: SPIEL MIT CHAT	86



ABBILDUNG 52: TUTORIAL HAUPT MENU	88
ABBILDUNG 53: TUTORIAL BASIC RULES	88
ABBILDUNG 54: HAUPT-MENU	89
ABBILDUNG 55: FRIENDS-MENU MIT FREUND UND PENDENTER ANFRAGE	90
ABBILDUNG 56: ADD FRIENDS ZEIGT DIE ANGEMELDETEN SPEILER	91
ABBILDUNG 57: ZWEI LOBBY'S IN DENEN MITEINANDER GECHATTET WIRD. DER EINTE SPIELER HAT BEREITS	
DAS GO GEGEBEN	92
ABBILDUNG 58: EINFACHES SERVER-FELD	93
ABBILDUNG 59: GUI-MOCKUP LOGIN OLD	94
ABBILDUNG 60: GUI-MOCKUP LOBBY OLD	94
ABBILDUNG 61: GUI-MOCKUP MAIN MENU OLD	95
ABBILDUNG 62: GUI-MOCKUP FRIENDS MENU OLD	95
ABBILDUNG 63: GUI-MOCKUP ADD FRIENDS OLD	96
ABBILDUNG 64: GUI-MOCKUP GAME MENU OLD	96
15 Tabellenverzeichnis	
TABELLE 1: UMSETZUNGS- UND PROJEKTZIELE	5
TABELLE 2: STAKEHOLDERGRUPPE: SPIELER	6
TABELLE 3: STAKEHOLDERGRUPPE: ENTWICKLER	7
TABELLE 4: STAKEHOLDERGRUPPE: FHNW	7
TABELLE 5: STAKEHOLDERGRUPPE: SPIELBEDINGUNGEN	7
TABELLE 6: UC NR 101: LOGIN	13
TABELLE 7: UC NR. 102: LOGOUT	14
TABELLE 8: UC NR. 201: BENUTZER HINZUFÜGEN	16
TABELLE 9: UC NR. 202: BENUTZER BEARBEITEN	17
TABELLE 10: UC NR. 203: BENUTZER SUCHEN	18
TABELLE 11: UC NR. 204: BENUTZER LÖSCHEN	19
TABELLE 12: UC NR. 301: SPIEL STARTEN	21
TABELLE 13: UC NR. 302: SPIEL BEENDEN	22
TABELLE 14UC NR. 303: PUNKTESTAND AUFZEICHNEN	23
TABELLE 15 UC NR. 304: SPIELFIGUR ANWÄHLEN	23
TABELLE 16 UC NR. 305: SPIELFIGUR BEWEGEN	24
TABELLE 17 UC NR. 306: GEWINNER ERMITTELN	25
TABELLE 18 UC NR. 307: GEWINNPUNKTE EINGEBEN	25
TABELLE 19: UC NR. 401: RANGLISTE	27
TABELLE 20 UC NR. 501: VERBINDUNG SERVER-CLIENT	29
TABELLE 21 UC NR. 502: COMPUTERSPIELER (KI)	29
TABELLE 22 UC NR. 503: MEHRSPRACHIG	30
TABELLE 23 UC NR. 504: CHAT	30
TABELLE 24: UC NR. 601: LOBBY	32
TABELLE 25: UC NR. 602: STARTING THE GAME WITH OPTIONS	33
TABELLE 26: UC NR. 603: FRIENDS	34
TABELLE 27: UC NR. 604: TUTORIAL	35
TABELLE 28: UC NR. 605: PLAYER-NUMBER	36
TABELLE 29: UC NR. 606: DATABASE	36
TABELLE 30: UC NR. 607: AI-DIFFERENT LEVELS	37
TABELLE 31: QUALITÄTSANFORDERUNGEN	65



16 Anhang

16.1 AI

Für dieses Spiel wurde eigens eine künstliche Intelligenz geschaffen, damit das Spiel auch im Single-Player-Modus gespielt werden kann. Ebenfalls möglich ist es zwei künstliche Intelligenzen gegen einander antreten zu lassen.

Die Parameter für die hier erhobene Heuristik können angepasst werden auf die eigenen Bedürfnisse. Ebenfalls können die Parameter für beide künstliche Intelligenzen geändert werden. So könnte z.B. auch im Feldversuch evaluiert werden welche der Einstellungen in welcher Situation stärker wäre.

Da die Türme im Spiel Kamisado immer gleich wertig respektive gleich stark sind, können nicht die Türme selbst mit Punkten gewichtet werden, sondern es muss jeweils die gesamte Situation betrachtet und die daraus entstehenden Möglichkeiten gewichtet werden.

16.1.1 Heuristik

Dazu wurden sich folgende Heuristiken überlegt:

- 1. Umso weiter vorne sich ein Turm befindet (mit diesem Zug), desto besser für seine Gewinnchancen
- 2. Umso mehr Züge ein Turm noch zur Verfügung hat (nach diesem Zug), desto besser steht der Turm
- 3. Wenn der Zug ein Gewinn erzielt werden kann, muss dieser Zug gemacht werden
- 4. Wenn der Zug die Möglichkeit bietet den Gegner in seinem nächsten Zug an einem Gewinn zu hindern, wird der Zug besser bewertet
- 5. Wenn der Turm ein Sumo ist können folgende Situationen zum Vorteil führen:
 - a. Der Sumo-Push ermöglicht für den nächsten Zug einen Sieg
 - b. Der Sumo-Push ermöglicht für den nächsten Zug einen Block zu erzielen

16.1.2 Gewichtung

Die Standard-Gewichtung:

Nummer	Gewichtung	Einheit	Was wird gemessen
1	100	Prozent	Anzahl Schritte von der Home-Line
2	100	Prozent	Anzahl mögliche Schritte
3	Maximal	Punkte	Kann dieser Zug direkt Siegen
4	15	Punkte	Kann dieser Zug blocken
5.a	25	Punkte	Kann dieser Sumo-Push einen Sieg im nächsten Zug
			erringen
5.b	15	Punkte	Kann dieser Sumo-Push einen Block im nächsten Zug
			erringen



16.1.3 Steuerung

Single Player

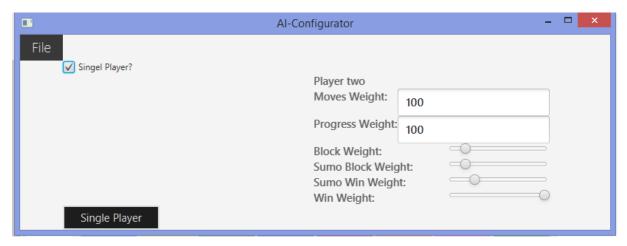


Abbildung 43: Single Player

Soll ein Spiel gestartet werden, indem der Spieler gegen einen Computer-Gegner spielen soll, kann der Single-Player-Modus aktiviert werden, indem die Parameter beim Spieler zwei eingestellt werden und der Button "Single Player" gedrückt wird.

Double AI - BetaVersion



Abbildung 44: Double Al

Sollen zwei künstliche Intelligenzen gegen einander antreten, dann kann das Häkchen bei "Single Player" entfernt werden, die Parameter bei Spieler eins und Spieler zwei eingestellt werden und auf den Button "Double Al" gedrückt werden.



16.2 Client

Der "Client" wird direkt zu Beginn des Spiels aufgerufen, damit eine Verbindung zum Server hergestellt werden kann.

Das Client-Menu bietet folgende Funktionen:

- 1. Eingabe der Host-Adresse
- 2. Eingabe des Portes
- 3. Log-Einträge werden für die Nachvollziehbarkeit der Übertragung im Fenster angezeigt
- 4. Chat-Nachrichten können gesendet werden, dies auch ohne Login, falls Probleme auftreten und man jemanden, welcher bereits im Spiel ist um Hilfe bitten möchte
- 5. Test-Modus für den "admin-User". Dieser User kann diverse Nachrichten via Knopfdruck an den Server übermitteln um die Übertragungen nachzuverfolgen (Troubleshooting). Solange man nicht als "admin-User" eingeloggt ist, werden diese Funktionen nicht freigeschalten
- 6. Die Verbindung kann sich "gemerkt" werden, sodass beim nächsten Start des Programmes direkt auf eine Verbindung aufgebaut wird.

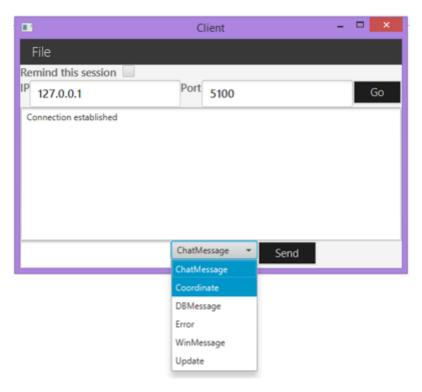


Abbildung 45: Client-Menu, Ansicht als "admin-User" mit allen Message-Typen



16.3 User Menu

Das User Menu kann aus dem Login Menu aufgerufen werden und ermöglicht die Abfrage und Erfassung von Spielern. Man kann Spieler mit dem Vornamen, Nachnamen und Usernamen suchen. Wenn ein User Fokussiert wurde kann dieser mit dem Button "Change" geändert werden, oder mit dem Button "Delete" gelöscht werden. Um Änderungen beim User vorzunehmen müssen die Änderungen entsprechend in die Felder eingetragen werden.

Der Button "Search All" zeigt alle Einträge in der Datenbank an.

Damit etwas geändert werden kann, muss zwingend das Passwort (Old Password) richtig eingegeben sein, ansonsten dürfen keine Änderungen vorgenommen werden.

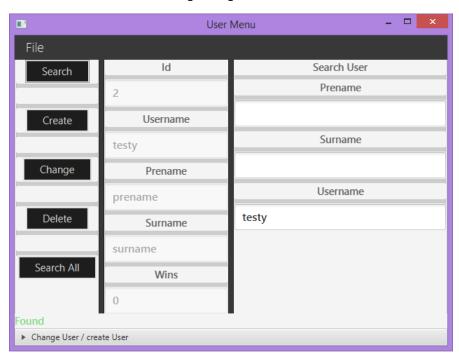


Abbildung 46: Suchen nach dem Spieler "testy"

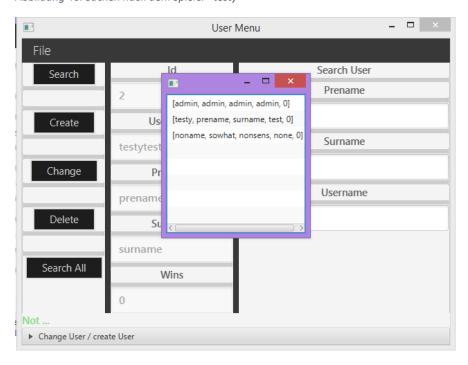


Abbildung 47: Alle Spieler anzeigen



Sollte ein neuer User erzeugt werden, kann dies mit dem Button "Create" gemacht werden. Dafür müssen die Felder ausgefüllt sein.

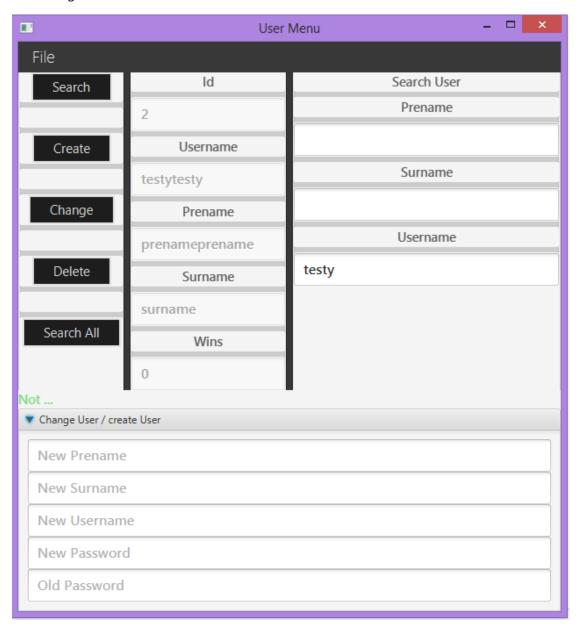


Abbildung 48: Das Menu der Felder für Änderungen oder Neuerzeugungen ist aufgeklappt



16.4 Login

Das Login Menu ist dazu hier sich mit seinem Benutzer anzumelden, damit der Benutzer identifiziert werden kann und auch seine Punkte entsprechend vergeben werden können.

Die Funktionen des Login Menu:

- 1. Überprüft ob der Benutzer bereits in der Datenbank vorhanden ist oder nicht. Vorhandene werden grün angezeigt, nicht vorhandene werden rot angezeigt. Ein Login ist nur möglich, wenn der Benutzer auch angelegt ist
- 2. Der Button "Refresh" ermöglicht eine komplette Aktualisierung der User-Daten von der Datenbank, sollte etwas nicht funktionieren respektive nicht aktuell sein
- 3. Der Button "User Menu" ruft das User Menu auf
- 4. Im Menu kann das Client Menu wieder aufgerufen werden, sofern dieses gebraucht wird
- 5. Nach drei falschen Versuchen das Passwort einzugeben, wird der Account gesperrt. Um diesen wieder zu aktivieren, muss der Administrator kontaktiert werden

Über den Button "Without Server" kann ein Spiel-Modus eröffnet werden indem zwei Spieler auf demselben Computer gegeneinander antreten können. Hierzu sollte keine Server-Verbindung hergestellt sein, so wie es der Name des Buttons bereits verrät.

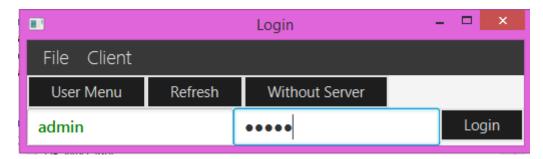


Abbildung 49: Login Menu mit korrekter Eingabe eines aktiven Benutzers



Abbildung 50: Login Menu mit eingabe eines Benutzer, den es nicht gibt



16.5 Spiel

Das Hauptmenu beinhaltet das Spielbrett, sowie einen Chat. Hier treten die Spieler gegeneinander an. Im Grund-Modus wird auf demselben Computer gegeneinander gespielt. Dieses wird gestartet indem der Button "Without Server" im Login gedrückt wird. Es gibt weitere Spiel-Modi's, welche im Haupt-Menu genauer erklärt werden.

In den Menus können folgende weitere Menus aufgerufen werden:

- 1. File
- a. Restart Startet das Spiel neu
- b. Language Ermöglicht eine Änderung der Sprache, die Einstellung wird sich gemerkt
- c. Friends Startet das Friends Menu
- d. Main Menu Startet das Main Menu
- e. Exit Zurück kehren zum Haupt-Menu
- 2. Help
 - a. Help about Version und Hersteller Informationen
 - b. Tutorial Startet das Tutorial
- 3. Client
 - a. Get Client View Holt die Client GUI
- 4. Al Controller
 - a. Al Controller Startet das Al Menu

Auf der Seite werden die jeweiligen Spieler mit ihrem Punktestand angezeigt. Es ist möglich ein Timer einzustellen in Minuten. Sollte der Timer abgelaufen sein, wird das Spiel beendet. Dies ist vor allem für kurze Spiele gedacht, sodass auf Zeit gespielt wird.

Klassisch wird, wie im Tutorial angegeben, auf Runden gespielt. Sobald die angegebene Zahl Runden erreicht wurde, wird der Gewinner ermittelt.

Diese Möglichkeit besteht nur im Freunde-Modus, da in den anderen Spiel-Modi's die Rundenanzahl vorgegeben wird.

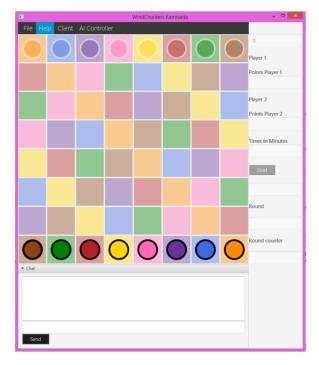


Abbildung 51: Spiel mit Chat



Das Spiel selbst bietet folgende Möglichkeiten:

- Nur ein Spieler ist am Zug und kann wählen.
- Nur der Turm mit der Farbe, auf dem der gegnerische Turm stehen geblieben ist darf den nächsten Zug machen
- Es werden die möglichen Züge anzeigt, indem die Felder aufgeschaltet werden
- Ist es eine Patt-Situation wird dies dem Spieler anzeigt und der andere Spieler ist am Zug (mit derselben Farbe)
- Gewinne werden angezeigt und der Spieler hat beim Spielstart die Wahl zwischen einem links Auffüllen oder einem rechts Auffüllen
- Die Sumo Towers werden mit Zahlen gekennzeichnet, sodass sofort erkennbar ist wie "stark" der Sumo Tower ist
- Die Sumo Tower können nur entsprechend ihrer "stärke" bewegt werden (limitierter Bewegungsumfang)



16.6 Tutorial

Das Tutorial führt den Spieler durch die verschiedenen Regeln des Kamisados. Dazu können die Spielvarianten

- Basic Rules
- Long Play
- Marathon Play

angeschaut werden.

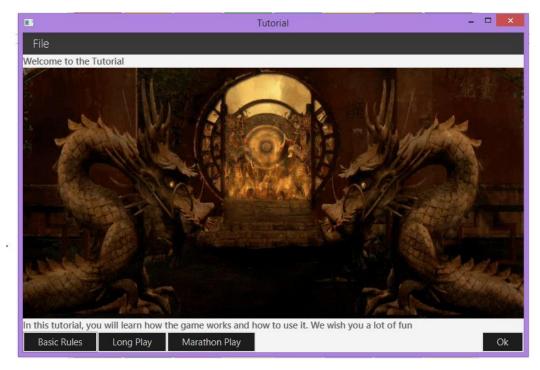


Abbildung 52: Tutorial Haupt Menu

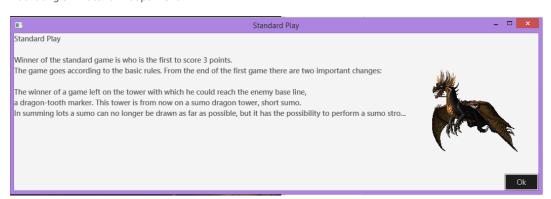


Abbildung 53: Tutorial Basic Rules



16.7 Haupt-Menu

Das Haupt-Menu ist das Menu, aus dem die verschiedenen Game-Modis gestartet werden können. Ebenfalls ist das Haupt-Menu der zentrale Punkt des Spiels, hier kehr man auch zurück, wenn man das Brettspiel selbst verlässt.

Es gibt hier folgende Auswahl:

- Single Game: Eine einfache Runde gegen den KI-Spieler
- Standard Game: 3 Runden werden gespielt gegen den KI-Spieler
- Long Game: 7 Runden werden gespielt gegen den KI-Spieler
- Marathon Game: 15 Runden werden gespielt gegen den KI-Spieler
- Friends: Ein Spiel gegen einen Freund wird gestaret
- End Game: Beendet das Spiel komplett.

Durch das Menu File >> Exit kehr man zurück zum Login, sollte man 2 Users erfasst haben ist es so möglich diesen so zu ändern, ohne aus dem Spiel auszusteigen.



Abbildung 54: Haupt-Menu



16.8 Friends-Menu

Im Friends-Menu kann der Spieler erfasste Users als Freunde anfragen oder sollte bereits eine Anfrage gekommen sein, diese annehmen, sodass gemeinsame Speile gestartet werden können.

Unter dem Menu File können folgende Punkte ausgewählt werden:

- Add Friends: Startet die Add Friends Ansicht um neue Freunde anzufragen
- Exit: Zurück zum Hauptmenu

Wenn ein Freund hinzugefügt ist hat der Spieler zwei Möglichkeiten:

- Remove: Entfernt den Freund bei beiden Spielern. Es ist möglich diesen Freund erneut anzufragen, dies muss aber wieder quittiert werden
- Play: Ein Spiel im Freunde-Modus zu starten

Wenn eine Anfrage eines Freundes besteht hat der Spieler zwei Möglichkeiten:

- Accept: Nimmt die Anfrage an und dieser Spieler wird jetzt als Freund angezeigt. Ebenso geschieht dies beim Spieler, welcher die Anfrage gesendet hat
- Refuse: Die Anfrage wird abgelehnt.

Anfragen sieht nur der Spieler, welcher angefragt wurde.

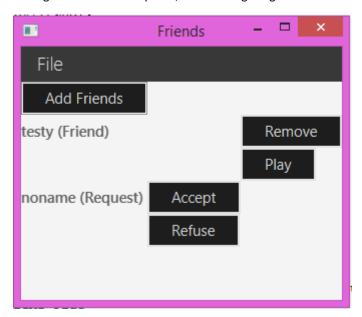


Abbildung 55: Friends-Menu mit Freund und pendenter Anfrage



16.8.1 Add Friends

Add Friends zeigt die Spieler, welche sich einen Account angelegt haben. Diese können durch das Suchfeld gefiltert werden.

Mit dem Add Friends-Button wird eine Anfrage an den entsprechenden Spieler gesendet.

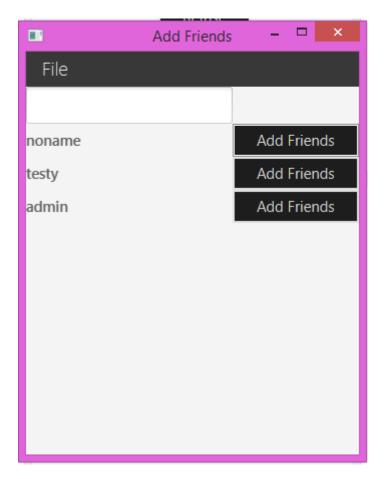


Abbildung 56: Add Friends zeigt die angemeldeten Speiler



16.9 Lobby

Die Lobby ist hier um auf einen Freund zu warten, mit dem man ein Spiel spielen möchte.

Die Lobby bietet folgende Funktionen:

- Wenn beide Spieler in der Lobby sind, kann in der Lobby untereinander gechattet werden. Dieser Chat wird nicht in den allgemeinen Chat übertragen. Der allgemeine Chat hingegen wird im Lobby-Chat angezeigt, sodass die beiden Spieler weiterhin die Informationen der anderen erhalten. Sollten die beiden Spieler in den allgemeinen Chat schreiben wollen, müssten sie via Menu den Client aufrufen und über diesen chatten.
- Sind beide Spieler in der Lobby angelangt (wichtig), und drücken dann beide Spieler auf dem Button "Go", wird das Spiel gestartet. Es ist möglich sich wieder um zu entscheiden und die Bereitschaft zurück zu nehmen, sofern der Button "Go" erneut gedrückt wird. Der Go-Button sollte erst gedrückt werden, wenn beide Spieler auch in der Lobby angekommen sind.
- Durch den Menu-Punkt "Exit" wird das Haupt-Menu aufgerufen

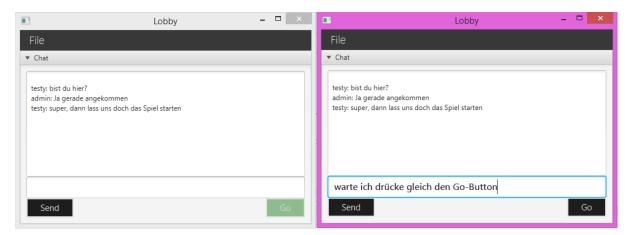


Abbildung 57: Zwei Lobby's in denen miteinander gechattet wird. Der einte Spieler hat bereits das Go gegeben



16.10 Server

Der Server wird als einfaches Feld anzeigt und in diesem wird das Log angezeigt.

Mit dem Button "Go" wird die Verbindung auf dem angegebenen Port (TextFeld neben Port) geöffnet. Der Server wartet nun auf einen Client, welcher sich verbinden will und handelt die Verbindung.

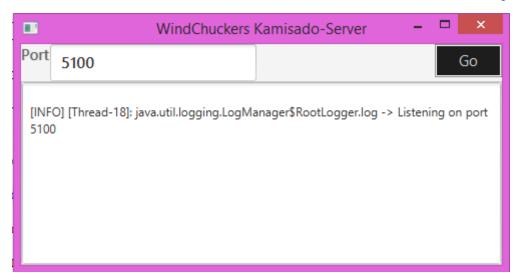


Abbildung 58: Einfaches Server-Feld

Der Server selbst verwaltet die eingehenden Meldungen in Form von XML und sendet Meldungen im Format von XML an den Client zurück. Weiteres was durch den Server verwaltet wird:

- Komplette Al
- Komplette Datenbank
- Die Koordination für das Starten eines Freunde-Spiels aus der Lobby

17 Old GUI-Mockup's

Da dieses Dokument sowohl als Requirement, als auch als Dokumentation dient, wurden die alten GUI-Mockup's wie sie ganz am Anfang in der Planungsphase angedacht wurden hier in den Anhang eingefügt, damit der Leser die Entwicklung des Spiels verfolgen kann.

17.1 Login

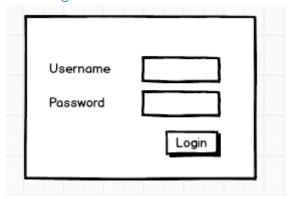


Abbildung 59: GUI-Mockup Login old

17.2 Lobby

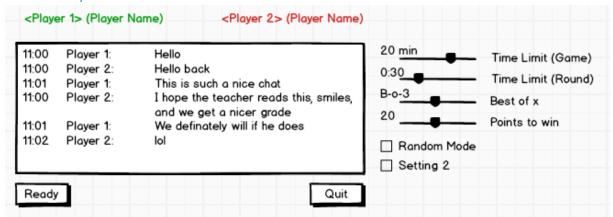


Abbildung 60: GUI-Mockup Lobby old



17.3 Main Menu

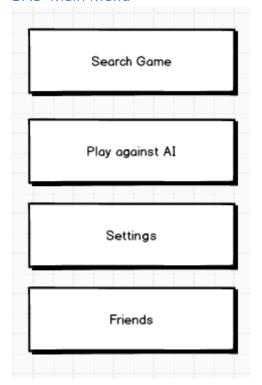


Abbildung 61: GUI-Mockup Main Menu old

17.4 Friends Menu

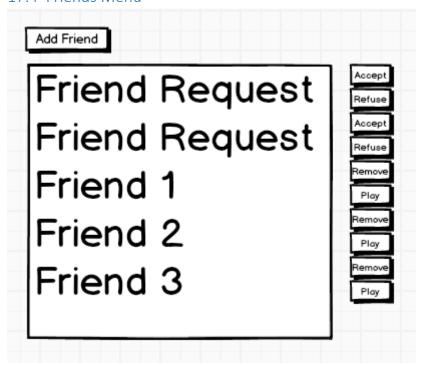


Abbildung 62: GUI-Mockup Friends Menu old



17.5 Add Friends

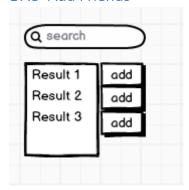


Abbildung 63: GUI-Mockup Add Friends old

17.6 Game Menu

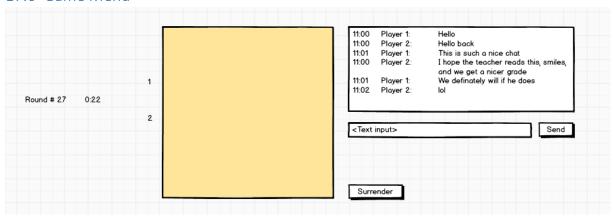


Abbildung 64: GUI-Mockup Game Menu old