

IT-Project Jass

Requirement specification and Test plan

Autoren:

Frank Mauchle
Oliver Mosimann
Pascal Wyser
Leon Xhinovci

Gruppenname: WeightWatchers

Dozenten: Prof. Bradley Richards & Lukas Frey

Ort, Datum: Olten, 24. Mai 2020

Version: **V01.2**

Versionskontrolle

Version	Bearbeitet von	Änderung	Datum
X01.00	Oliver Mosimann	Dokument erstellt	02.03.2020
V01.00	Pascal Wyser Leon Xhinovci Frank Mauchle Oliver Mosimann	Ergänzung MockUps, Flow-Charts, UseCases, Ergänzungen, Überarbeitung und Fertigstellung des Dokuments	06.03.2020
V01.1	Pascal Wyser Leon Xhinovci Frank Mauchle Oliver Mosimann	Klassendiagramm erstellt, Sequenzdiagramm erstellt, Zuteilung Programmiertasks, DB Schema erstellt	30.03.2020
V01.2	Pascal Wyser	Klassendiagramm Update	22.05.2020

1 Inhalt

Versionskontrolle	2
2 Requirement Specification	5
2.1 Einleitung	5
2.2 UseCases	5
2.3 Mockups	15
2.4 FlowChart	17
3 Test Plan.....	18
3.1 Einleitung	18
3.2 Testgegenstand	18
3.3 Test-Scope	18
3.4 Testansatz, gewählte Teststrategien	18
3.5 Kriterien für Testende und -abbruch	19
3.6 Kriterien für Testunterbrechung und -wiederanlauf.....	19
3.7 Erwartete Testergebnisse.....	19
3.7.1 Vor Testbeginn	19
3.7.2 Während Test.....	19
3.7.3 Nach Testende	19
3.8 Testaktivitäten	20
3.9 Erforderliche Voraussetzungen	21
3.9.1 Testumgebung	21
3.10 Zuständigkeiten.....	21
3.11 Besetzung und Trainingsbedarf.....	21
3.11.1 Zeitaufwand.....	21
3.12 Testplan.....	22
3.13 Risiken und Notfallplan	22
Genehmigung / Freigabe	22
4 UML Klassendiagramm.....	23
4.1 Package Client.....	23
4.2 Package Server	23
4.3 Verantwortlichkeiten.....	24
5 Sequenzdiagramm.....	26
6 Datenbank.....	29
6.1 ER-Modell	29

6.2	<i>Relationales Modell</i>	29
6.3	<i>Inhalte der Datenbank</i>	30

2 Requirement Specification

2.1 Einleitung

Im Modul IT-Projekt der FHNW haben wir den Auftrag, ein spielfertiges Jass-Spiel zu programmieren. In den Folgenden Kapiteln werden wir die einzelnen dafür notwendigen Schritte aufzeigen. Als erstes folgen die UseCases bei denen wir uns damit befasst haben, was für Funktionen und Abläufe vorhanden sind. In den Weiteren Kapiteln folgen die Mock-Ups sowie der Test-Plan.

2.2 UseCases

Name Client mit Server verbinden		
Use Case ID	1	
Beschreibung	Das System muss mehreren Clients die Möglichkeit bieten sich mit dem Server zu verbinden.	
Verantwortlicher	System	
Beteiligte	System, Client	
Auslöser	Der User startet das Programm	
Vorbedingungen	Client startet ordnungsgemäss	
Eingaben	-	
Ablauf (Szenario)	Standard	Alternativ
	1. Software starten durch den User	
	2. Client wird mit dem Programmstart geladen und versucht sich anschliessend automatisch mit dem Server zu verbinden	Verbindung gescheitert, der Client versucht sich wiederholt mit dem Server zu verbinden
	3. Verbindung mit dem Server ist gelungen	Verbindung mit Server ist gescheitert, der User wird aufgefordert das Programm neu zu starten
	4. Einstieg in Login-Menü	
Ausgaben	Login-Menü wird angezeigt	
Nachbedingungen	Es kann auf die Dienste des Servers zugegriffen werden/Login View startet	

Name	Sprache wählen	
Use Case ID	2	
Beschreibung	Das System muss dem User die Möglichkeit bieten, im Login-Menü eine Sprache auszuwählen.	
Verantwortlicher	System	
Beteiligte	System, Client, User	
Auslöser	Der Client ist mit dem Server verbunden	
Vorbedingungen	Der Client ist bereits mit dem Server verbunden (UC1) Login-View wurde gestartet	
Eingaben	Passwort, Username, Sprachauswahl	
Ablauf (Szenario)	Standard	Alternativ
	1. Die Standardsprache wird beibehalten.	Der User wählt in der Dropdownliste im Login- Menü eine andere Sprache
	2. Der Client übermittelt dem Server die Sprache.	Andere Sprache wird dem Server übermittelt
	3. Der Server lädt die korrekte Sprache.	
Ausgaben	Meldung: Sprache wurde angepasst	
Nachbedingungen	-	

Name		User registrieren
Use Case ID	3	
Beschreibung	Das System muss dem User die Möglichkeit bieten, sich im Login-Menü neu zu registrieren.	
Verantwortlicher	System	
Beteiligte	System, Client, User	
Auslöser	Das System ist mit dem Client verbunden	
Vorbedingungen	Der Client ist bereits mit dem Server verbunden (UC1)	
Eingaben	Username, Passwort, Register	
Ablauf (Szenario)	Standard	Alternativ
	1. Der User gibt einen Usernamen und ein Passwort ein.	
	2. Der User betätigt den Button Register.	
	3. Der Client sendet das Passwort und den Usernamen an den	
	4. Der Server prüft ob dieses Passwort und Username schon vorhanden ist	Benutzername schon vergeben, das System fordert den User auf einen neuen Benutzername einzugeben. --> Wieder bei Punkt 1
	5. Der Server speichert das Passwort und Username	
	6. In der Login View wird angezeigt "Login	
Ausgaben	Login created	
Nachbedingungen	Login (UC4)	

Name		Login prüfen
Use Case ID	4	
Beschreibung	Das System muss in der Lage sein, zu prüfen ob ein User bereits auf dem Server existiert. Das System muss in der Lage sein einen Usernamen und ein Passwort an den Server zu übermitteln.	
Verantwortlicher	System	
Beteiligte	System, Client, User	
Auslöser	Der Client ist mit dem Server verbunden	
Vorbedingungen	Der Client ist bereits mit dem Server verbunden (UC1)	
Eingaben	Passwort, Username	
Ablauf (Szenario)	Standard	Alternativ
	1. Der User gibt ein Passwort und Usernamen in das Login-Menü ein.	
	2. Der User klickt auf den Button "Login & Enter Lobby"	
	3. Der Client übermittelt dem Server das Passwort sowie den Usernamen.	
	4. Der Server prüft ob bereits ein User mit dem Passwort vorhanden ist.	Login und/oder Passwort falsch - Fehlermeldung wird angezeigt
	5. Spieler wird Zugriff zur Lobby freigegeben	
	6. Einstieg in die Lobby	
Ausgaben	Die Lobby GUI wird angezeigt	
Nachbedingungen	-	

Name	Spiel erstellen und wählen	
Use Case ID	5	
Beschreibung	Das System muss dem User die Möglichkeit bieten, ein Spiel zu wählen und diesem beizutreten oder ein neues Spiel zu erstellen.	
Verantwortlicher	System	
Beteiligte	System, Client, User	
Auslöser	User hat sich in der Lobby angemeldet	
Vorbedingungen	Login erfolgreich (UC4)	
Eingaben	Klick auf ein Spiel oder Klick auf "neues Spiel erstellen"	
Ablauf (Szenario)	Standard	Alternativ
	1. Das System zeigt in der Lobby vorhandene Spiele an	Kein Spiel vorhanden
	2. Der User klickt auf ein vorhandenes Spiel	Der User klickt neues Spiel erstellen
	3. Der Client übermittelt dem Server das gewählte Spiel	Der User gibt dem neuen Spiel einen Namen und wählt die Kartenart aus
	4. Spiel wird auf dem Server als aktuelles Spiel gespeichert	Das neue Spiel wird auf dem Server als Spiel mit der Kartenart festgelegt - zurück zu Punkt 1
	5. Meldung Spiel wurde erstellt/beigetreten	
Ausgaben	Spiel wurde beigetreten	
Nachbedingungen	Server wartet auf weitere Clients, die dem Spiel beitreten	

Name	Spiel starten	
Use Case ID	6	
Beschreibung	Das System muss dem Client die Möglichkeit bieten, ein Spiel zu starten.	
Verantwortlicher	System	
Beteiligte	System, Client, User	
Auslöser	Spiel starten	
Vorbedingungen	Ein Spiel wurde bereits erstellt und gewählt (UC5)	
Eingaben	klick auf Spiel starten	
Ablauf (Szenario)	Standard	Alternativ
	1. Der Server wartet bis genügend Spieler dem Spiel beigetreten sind	
	2. Spiel wird gestartet	Der Server übergibt dem Client eine Meldung, zu wenig Spieler vorhanden, bitte warten bis mindestens 4 Spieler vorhanden sind.
Ausgaben	Spiel-GUI wird gestartet	
Nachbedingungen	Spiel gestartet	

Name	Kartendeck generieren und Clients zuweisen	
Use Case ID	7	
Beschreibung	Das System muss in der Lage sein ein Kartendeck von 36 Karten zu generieren und diese den jeweiligen Clients zufällig zu zuweisen	
Verantwortlicher	System	
Beteiligte	System, Client	
Auslöser	Spiel starten in der Lobby	
Vorbedingungen	Spiel starten (UC6)	
Eingaben	-	
Ablauf (Szenario)	Standard	Alternativ
	1. Das Kartendeck wird auf dem Server generiert	
	2. Das Kartendeck wird zufällig auf dem Server gemischt	
	3. Der Server weist den mitspielenden Clients/Usern die Karten zu.	
Ausgaben	Anzeige der Karten in der View	
Nachbedingungen	Nächster Spieler wird durch System bestimmt	

Name	nächsten Spieler bestimmen	
Use Case ID	8	
Beschreibung	Das System muss in der Lage sein den nächsten Spieler der an den Zug kommt zu bestimmen.	
Verantwortlicher	System	
Beteiligte	System, Client	
Auslöser	Kartendeck wurde generiert	
Vorbedingungen	Alle Karten wurden verteilt (UC7)	
Eingaben		
Ablauf (Szenario)	Standard	Alternativ
	1. Das System bestimmt per Zufall den ersten Spieler	Spiel befindet sich nicht in der ersten Runde, nächster Spieler wird festgelegt (gegen Uhrzeigersinn)
	2. Der Spieler der als nächstes am Zug ist, wird angezeigt (gegen den Uhrzeigersinn)	
Ausgaben	Spieler der am Zug ist, wird auf in der View angezeigt	
Nachbedingungen	Nächster Spieler bestimmt	

Name		Trumpf bestimmen und anzeigen	
Use Case ID	9		
Beschreibung	Das System muss dem Spieler der am Zug ist, die Möglichkeit bieten einen Trumpf zu wählen		
Verantwortlicher	System		
Beteiligte	System, User		
Auslöser	Karten wurden fertig ausgeteilt		
Vorbedingungen	Spieler der am Zug ist wurde bestimmt (UC8)		
Eingaben	Klick auf eine eigenen Karte des Spielers der am Zug ist		
Ablauf (Szenario)	Standard	Alternativ	
	1. Eine Meldung in der Main GUI wird ausgegeben --> Bitte Trumpf auswählen		
	2. Der Spieler klickt auf eine beliebige Karte aus seiner Hand	Der Spieler entscheidet sich zu schieben, somit wird automatisch der Partner aufgefordert den Trumpf zu klicken	
	3. Der Server übermittelt dem Client die Farbe des Trumpfes, die er dann in der View anzeigt		
Ausgaben	Anzeige der Trumpffarbe in der View		
Nachbedingungen	Spielzug wird getätigt		

Name		Prüfen ob erster Zug eines Spielers	
Use Case ID	10		
Beschreibung	Das System muss in der Lage sein zu prüfen, ob der Spieler, der am Zug ist seinen ersten Zug ausführt		
Verantwortlicher	System		
Beteiligte	System		
Auslöser	Trumpf wurde ausgewählt		
Vorbedingungen	Die Farbe des Trumpfs ist bekannt (UC9) Nächster Spieler ist bekannt (UC8)		
Eingaben	-		
Ablauf (Szenario)	Standard	Alternativ	
	1. Das System überprüft wie viel Karten der Spieler in der Hand hat		
	2. Der Spieler hat alle Karten in der Hand --> erster Zug	Der Spieler hat nicht mehr alle Karten in der Hand --> nicht erster Zug	
	3. Programm wird aufgrund der Informationen aus Punkt		
Ausgaben	-		
Nachbedingungen	Programmfluss kann fortgesetzt werden		

Name		Weis erkennen und ansagen	
Use Case ID	11		
Beschreibung	Das System muss erkennen, ob ein Spieler ein Weis in seinen Karten hat und dem Spieler das ersichtlich machen.		
Verantwortlicher	System, User		
Beteiligte	System, Client, User		
Auslöser	Spieler ist am Zug		
Vorbedingungen	Kartendeck generieren und Clients zugewiesen (UC7) Trumpf wurde bestimmt (UC9)		
Eingaben	Höhe des Weis		
Ablauf (Szenario)	Standard	Alternativ	
	1. Der Spieler sieht ob er ein Weis hat	Kein Weis vorhanden.	
	2. Der Spieler sagt die Höhe des Weis an	Spieler verzichtet auf den Weis	
	3. Der Client zeigt in der View die angegebene Höhe des Weis an		
	4. Die Höhe des höchsten Weis (Team) wird dem Team dazugezählt zum Zwischenresultat		
Ausgaben	System zeigt Spieler mit Weis an		
Nachbedingungen	Punkte des Weis bei Gewinn der Punkte dazu zählen		

Name		Karten noch vorhanden	
Use Case ID	13		
Beschreibung	Das System muss erkennen, ob noch Karten im Spiel sind. Wenn das zutrifft, geht das Spiel weiter, ansonsten ist die Runde beendet		
Verantwortlicher	System		
Beteiligte	System, User		
Auslöser	Spiel in progress		
Vorbedingungen	Jeder Spieler hat noch mindestens eine Karte in der Hand		
Eingaben	-		
Ablauf (Szenario)	Standard	Alternativ	
	1. Der Spieler hat noch Karten auf der Hand	Der Spieler hat keine Karten mehr in der Hand	
	2. Spielverlauf wird fortgesetzt	Spielverlauf wird beendet	
Ausgaben	Karten werden angezeigt, solange die Spieler noch Karten auf der Hand haben		
Nachbedingungen	Die Spieler haben keine Karten mehr und die Runde wird beendet (UC14)		

Name		Runde beenden	
Use Case ID	14		
Beschreibung	Das System erkennt, wenn keine Karten mehr vorhanden sind und beendet die Runde. Nach jeder beendeten Runde wird die gewonnene Punktezahl zum jeweiligen Team dazugezählt und vom Client angezeigt		
Verantwortlicher	System, Client		
Beteiligte	System, Client		
Auslöser	Letzte Karte wird gespielt und Runde wird beendet		
Vorbedingungen	UC13 (Karten noch vorhanden?)		
Eingaben	Letzte Karte wird gespielt		
Ablauf (Szenario)	Standard	Alternativ	
	1. Die Runde wird vom System beendet		
	2. Der Client zeigt an welches Team wieviele Punkte gemacht hat		
	3. Das System zählt die Punkte zum jeweiligen Zwischentotal jeder Runde dazu bis 1000 Punkte erreicht sind		
	4. Das Spiel geht in die nächste Runde	1000 Punkte wurden erreicht (UC15)	
Ausgaben	Die jeweiligen Teams erhalten die in der Runde erreichten Punkte in ihr Zwischentotal		
Nachbedingungen	Das Spiel wird weitergespielt bis 1000 Punkte erreicht sind (UC15)		

Name		1000 Punkte erreicht	
Use Case ID	15		
Beschreibung	Das System muss erkennen, ob eins der Teams 1000 Punkte erreicht hat		
Verantwortlicher	System		
Beteiligte	System, Client		
Auslöser	1000 Punkte erreicht		
Vorbedingungen	UC14 (Runde beenden)		
Eingaben	-		
Ablauf (Szenario)	Standard	Alternativ	
	1. Das System zählt nach jeder Runde die Punkte zusammen (UC14)		
	2. Das System beendet das Spiel wenn die 1000 Punkte erreicht sind		
	3. Das System zählt alle Punkte zusammen der jeweiligen Spieler		
	4. Der Client zeigt in der View die jeweilig gewonnen Punkte der		
	5. Das System zeigt den Gewinner-Team an (mehr gewonnene Punkte - <= 1000 Punkte)		
Ausgaben	Alle Spieler werden angezeigt mit jeweiligen Punkten und das Gewinnerteam wird angezeigt		
Nachbedingungen	Weiteres Spiel? (UC16)		

Name		Möglichkeit neues Spiel zu starten	
Use Case ID	16		
Beschreibung	Das System gibt dem User die Möglichkeit geben, dass er ein neues Spiel beginnen möchte oder nicht		
Verantwortlicher	System, Client		
Beteiligte	System, Client, User		
Auslöser	Vorheriges Spiel ist beendet		
Vorbedingungen	1000 Punkte erreicht (UC15) Runde beendet (UC14)		
Eingaben	Entscheidung ob neues Spiel oder nicht		
Ablauf (Szenario)	Standard	Alternativ	
	1. Der Client zeigt den Usern an, ob er ein neues Spiel beginnen möchte		
	2. Alle User wählen "Ja", um ein neues Spiel zu starten und die Karten werden neu ausgeteilt --> UC7	Mind. ein User wählt "nein" und das Spiel wird beendet und die User landen in der Lobby	
Ausgaben	Die Karten werden ausgeteilt, sollte das Spiel erneut beginnen, ansonsten landen die User in der Lobby		
Nachbedingungen	Neues Spiel oder Lobby		

2.3 Mockups

Jassen - The Game - Login

Welcome to "Jassen - The Game"!
Please choose Language and Login to play...

English


Username

Password

Login & Enter Lobby

Not registered yet?

Register



Jassen - The Game - Lobby

Games

HierFliegenKarten (1/4)

JassRaum7 (0/4)

Create new

Chat

You have joined the Lobby.

UrsJasst1Spiel joined the Lobby.

Send

Player online

Seppli_123 (You)

UrsJasst1Spiel

SwissBoy88

JustinSchieber


DonaldTrumpf

Selected Game: HierFliegenKarten

Players waiting... (1/4)

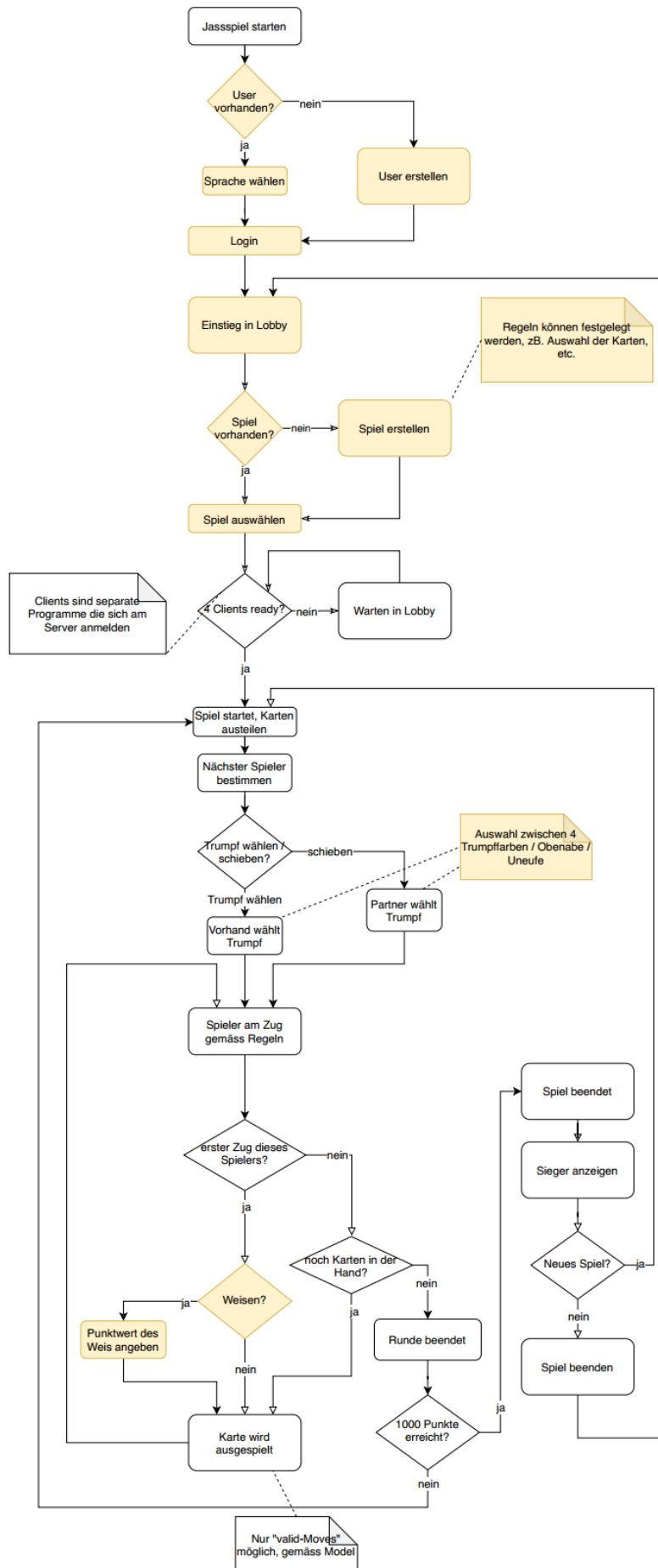
Seppli_123 (You)

Leave Game





2.4 FlowChart



3 Test Plan

3.1 Einleitung

Mit dem Testplan wollen wir sicherstellen, dass die gewünschten Minimalanforderungen an das Programm fehlerfrei und mit den gewünschten Funktionalitäten laufen. Auch eventuelle zusätzliche Features werden getestet. In den folgenden Kapiteln wird definiert, welche Punkte, wie und mit welchen Ressourcen getestet werden. Nach Möglichkeit sollen am Ende der Entwicklung Benutzertests mit nicht an der Entwicklung beteiligten Personen stattfinden. Der Testverlauf sowie die Testergebnisse werden dokumentiert.

3.2 Testgegenstand

Testgegenstand ist das von der Gruppe für das IT-Projekt der FHNW programmierte Jass-Spiel.

3.3 Test-Scope

3.3.1.1 Was wird getestet

Getestet werden sämtliche Aktivitäten/Eingaben, die durch den Benutzer vorgenommen werden können. Hierbei werden die definierten Funktionen validiert. Das Programm muss bei Benutzereingaben die gewünschten Aktivitäten ausführen und Resultate liefern. Mittels absichtlichen Fehleingaben werden Fehlertests durchgeführt.

Mittels Modultests werden die Methoden die für das korrekte Auswerten und zusammenzählen der gespielten Karten getestet. Diese Tests sollen möglichst mittels automatisierten Tests durchgeführt werden.

3.3.1.2 Was wird nicht getestet

Es werden keine Performancetests durchgeführt. Da es sich um ein sehr kleines und kein kritisches Programm handelt, sind keine Performancetest notwendig.

Es werden ebenfalls keine Tests in Bezug auf Security durchgeführt, da keine sensiblen Daten verarbeitet werden.

3.4 Testansatz, gewählte Teststrategien

Da die Tests in erster Linie durch die Entwickler selbst durchgeführt werden, handelt es sich grösstenteils um Entwicklertests. Die Tests finden modular, meist auf Klassenebene statt. Speziell zur Prüfung der zu implementierenden Methoden für die Kartenauswertung, wird der Ansatz der Testgetriebenen Entwicklung angewendet.

3.5 Kriterien für Testende und -abbruch

Testende	Abbruch
Einzeltest erfolgreich und fehlerfrei durchgelaufen	Test nicht erfolgreich/mit Fehler
Alle Tests erfolgreich und fehlerfrei durchgelaufen	Test läuft nicht erwartungsgemäss infolge Fehler im Test
	Projekt wird abgebrochen
	Zu testende Funktionalität wird nicht umgesetzt

3.6 Kriterien für Testunterbrechung und -wiederanlauf

Unterbrechung	Wiederanlauf
Test läuft nicht wunschgemäss durch	Fehler im Test behoben
Testresultat mit Fehler	Fehler im Programmcode behoben
System steht nicht zur Verfügung	Sobald System wieder zur Verfügung steht
Ressourcen der Testpersonen	Sobald Ressourcen ausreichend zur Verfügung stehen

3.7 Erwartete Testergebnisse

3.7.1 Vor Testbeginn

- Testpläne erstellt
- Einzeltests definiert/spezifiziert
- Einzeltests dokumentiert/vorbereitet

3.7.2 Während Test

- Funktionierendes Test-Tool/JUNIT
- Logeinträge bei erfolgreichen Prozessschritten
- Dokumentierte Testergebnisse
- Fehlerberichte mit Massnahmen

3.7.3 Nach Testende

- Report der Testresultate
- Dokumentierte Fehlerbehebung
- Freigabe für Auslieferung Software

3.8 Testaktivitäten

Nr.	UseCase	Testcase	Vorbedingung/Test-Task	Erwartetes Verhalten	Test mittels
1	1 Verbindung Client - Server	Client/ Programm starten	Server läuft, Netzwerkverbindung vorhanden	Client muss sich automatisch mit dem Server verbinden, Login Menü erscheint	Tester
			Keine Netzwerkverbindung zum Server vorhanden; Server ausschalten, Verbindung trennen	Client muss mehrmals versuchen eine Verbindung zum Server herzustellen, bei wiederholtem Scheitern, muss eine Fehlermeldung erscheinen	
2	2 Sprache ändern	Sprache im Login Menü ändern	Voreingestellte Sprache, als User anmelden, in allen folgenden Menüs/Buttons muss der Text in der entsprechenden Sprache angezeigt werden	Alle Views werden in der entsprechenden Sprache angezeigt.	Tester
			Im Login-Menü Sprache ändern	Die Sprache wird in allen Elementen sofort umgestellt, Prüfung nach Anmeldung ob in folgenden Views ebenfalls die gewählte Sprache angezeigt wird.	
3	3	User registrieren	Im Login-Menü ein Username und ein Passwort eingeben, «Register» klicken	Hinweisfenster erfolgreich registriert erscheint, mit OK bestätigen	Tester
			Mit bereits vergebenem Username versuchen	Fehlermeldung muss erscheinen, danach erneute Registrierung möglich	
4	4	Login	In Login-View mit bereits registriertem User anmelden, Login & Enter Lobby klicken	Login-View schliesst, Lobby-View erscheint	
			Falscher Benutzername eingeben	Fehlermeldung "Benutzer nicht vorhanden" erscheint	
			Falsches Passwort eingeben	Fehlermeldung «Passwort falsch» erscheint	
5		Anzeige Spieler	Mehrere Spieler anmelden	Spielernamen werden in der Lobby-View angezeigt	Tester
6	5	Spiel erstellen	In Lobby-View auf «neues Spiel erstellen» klicken, Spielname eintragen, Kartentypen auswählen	Fenster zur Eingabe für den Name des neuen Spiels erscheint, nach Bestätigung muss Spiel in der Spielkarte vorhanden sein	Tester
			Neues Spiel mit einem bereits verwendeten Namen erstellen	Warnung «Spielname wird bereits verwendet» erscheint	
7		Anzeige Spiele	Mehrere Spiele erstellen	Erstellte Spiele werden in der Lobby-View angezeigt	Tester
8	5	Spiel beitreten	In der Lobby-View mittels Doppelklicks ein Spiel auswählen	Ausgewähltes Spiel sowie bereits beigetretene Spieler werden in der Lobby-View angezeigt	Tester
9	6/7/8	Spielstart	4 Spieler haben das gleiche Spiel gewählt	Lobby-View schliesst, Jassen-View öffnet, Karten werden den Spielern verteilt und angezeigt, beginnender Spieler wird angezeigt	Tester
10	9	Trumpf bestimmen	Bei Spielstart muss erster Spieler Trumpfkarte auswählen; entsprechender Spieler auf eine Karte klicken	Trumpffarbe wird in der View angezeigt, Spieler muss nächste Karte spielen	Tester
11	12	Karte spielen	Der Spieler, der an der Reihe ist auf eine Karte klicken	Gewählte Karte wird von der Hand entfernt und erscheint auf der Mitte des Tisches, Spiel wechselt zum nächsten Spieler	Tester
12		Auf Spielbare Karten prüfen	Karten spielen nach Regeln	Spielbare Karten werden angezeigt	Tester
13	13/14/15	Karten spielen bis keine mehr vorhanden oder 100 Punkte erreicht	Spiele bis keine Karten mehr vorhanden sind	Die beim Zug gewählte Karte wird von der Hand entfernt, sobald alle Spieler keine Karten mehr in der Hand haben wird die Runde beendet, die Spielergebnisse werden zusammengezählt und angezeigt, neue Runde wird gestartet.	Tester
			Spiele bis erster Spieler 1000 Punkte erreicht	Spiel wird beendet, Gewinner wird angezeigt, Jeder Spieler wird gefragt ob er noch eine Runde spielen möchte	Tester JUNIT

14	16	Spiel beenden oder neue Runde	Test Nr. 12, spielen bis erster Spieler 100 Punkte erreicht, alle User wählen neues Spiel	Neue Runde/Spiel wird gestartet	Tester
			Test Nr. 12, spielen bis erster Spieler 100 Punkte erreicht, ein Spieler wählt kein neues Spiel	Spiel-View wird geschlossen, Lobby-View erscheint	
15	Modultest	Kartenwert	Trumpf bestimmen	Kartenwerte werden entsprechend hinterlegt; Listenausgabe für Test	Tester
16	Modultest	Normaler Kartenwert	Verschiedene vorgegebene Kartenwerte	Karten werden richtig gezählt	JUNIT, Tester
17	Modultest	Auf Stich prüfen	Sicht wird gespielt	Korrekte Werte gezählt	JUNIT, Tester
18	Modultest	Letzter Stich in Runde	Letzter Gewinner der einen Stich macht	Zusätzlich 5 Punkte für den Spieler	JUNIT, Tester
19	Modultest	Gewinner pro Runde ermitteln	Punkte nach gespielten Karten pro Runde	Punkte werden korrekt zusammengezählt und angezeigt	JUNIT, Tester
20	Modultest	Auf Match prüfen	Spieler/Team macht Durchmarsch	Zusätzlich 100 Punkte	JUNIT, Tester

3.9 Erforderliche Voraussetzungen

Mit den Tests wird begonnen, sobald folgende Bedingungen erfüllt sind:

- Software bzw. Teilbereiche soweit fertiggestellt, dass Tests durchgeführt werden können
- Die Testumgebung ist bereit und lauffähig
- Die Testpersonen haben freie Kapazitäten die Tests durchzuführen
- Automatisierte Tests sind programmiert und lauffähig

3.9.1 Testumgebung

- Server
- Client
- Netzwerkverbindung

3.10 Zuständigkeiten

Da es sich hier um ein relativ kleines Projekt handelt, werden die Tests durch die am Projekt beteiligten Personen durchgeführt. Auf externe Testpersonen wird während der Entwicklungsphase verzichtet.

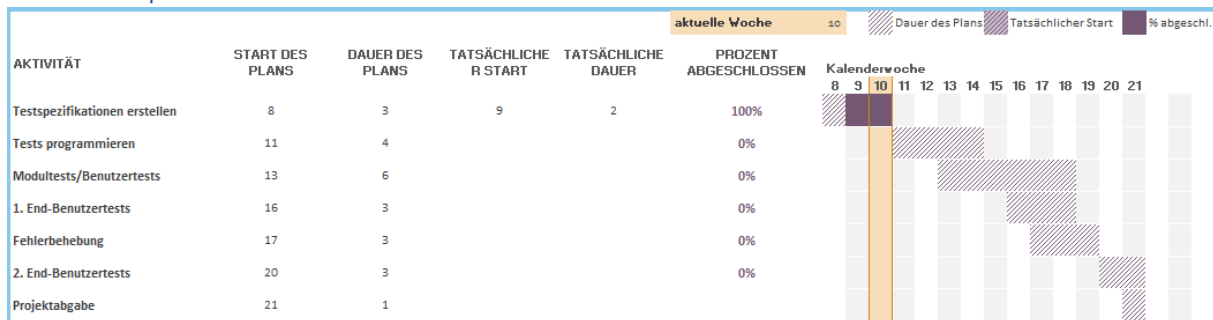
3.11 Besetzung und Trainingsbedarf

Alle Gruppenmitglieder die an der Entwicklung beteiligt sind.

3.11.1 Zeitaufwand

Aufgabe/Task	Tester	Geschätzter Aufwand
Testspezifikationen erstellen	Test-Designer	5 h
Automatische Tests programmieren	Entwickler	12 h
Tests durchführen	Tester, Gruppenmitglieder	8 h
Tests dokumentieren	Tester, Gruppenmitglieder	2 h
Total		27 h

3.12 Testplan



3.13 Risiken und Notfallplan

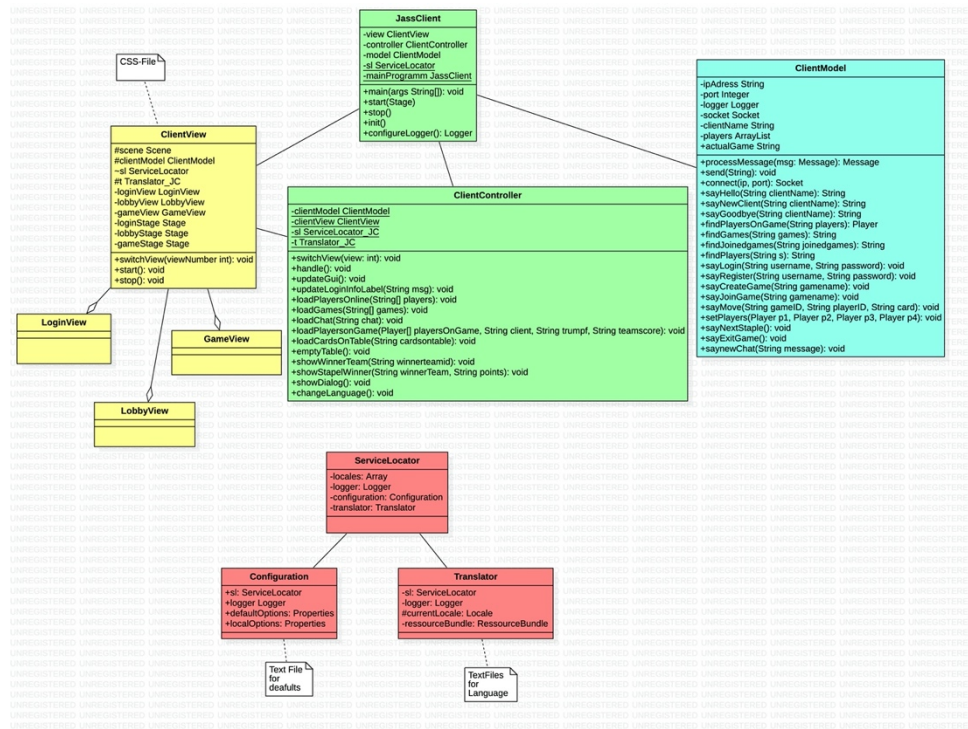
Risiko	Intervention
Modul nicht termingerecht fertiggestellt	Besprechung in der Gruppe, wie das Modul schnellstmöglich fertiggestellt werden kann, ev. Aufteilung auf mehrere Gruppenmitglieder
Ausfall Gruppenmitglied	Task wird von anderem Gruppenmitglied übernommen oder auf mehrere Gruppenmitglieder aufgeteilt.
Zu komplizierter/aufwändiger Task	Hilfe durch die anderen Gruppenmitglieder.
Automatisierter Test kann nicht wie gewünscht erstellt/durchgeführt werden	Überprüfen ob der Test manuell durch Benutzer durchgeführt werden kann.
Testergebnis nicht erfolgreich	Mögliche Fehlerbehebung in der Gruppe besprechen. Notfalls prüfen, ob Funktionalität anderweitig umgesetzt oder ganz weggelassen werden kann.
Zeitverzug des Projektes	Noch zu erledigende Tasks priorisieren, ebenfalls die Testfälle dementsprechend priorisieren.
Tests können nicht bis zur Projektabgabe erfolgreich abgeschlossen werden.	Eingabe Projekt trotz vorhandener Fehler, nicht erfolgreichen Tests.

Genehmigung / Freigabe

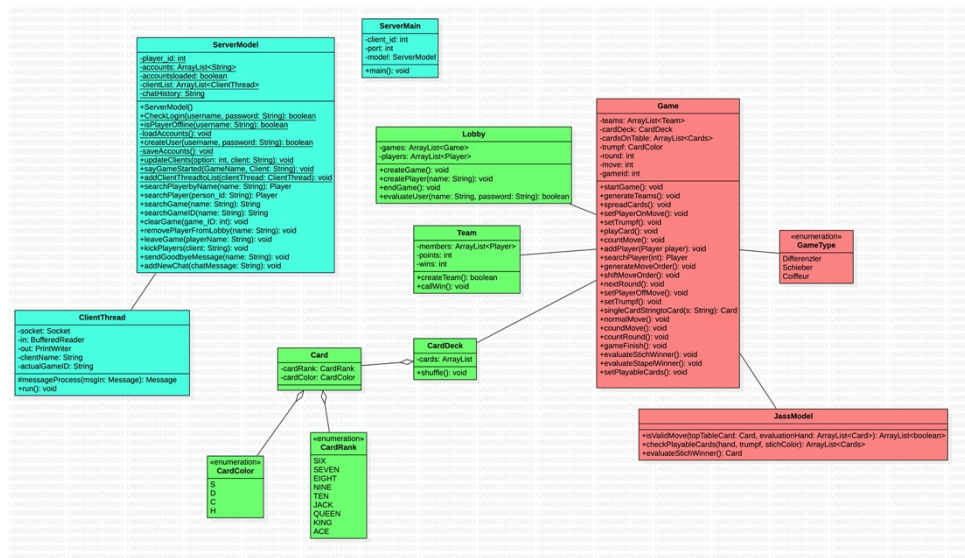
Die Freigabe erfolgt durch die Bewertung durch die Dozenten.

4 UML Klassendiagramm

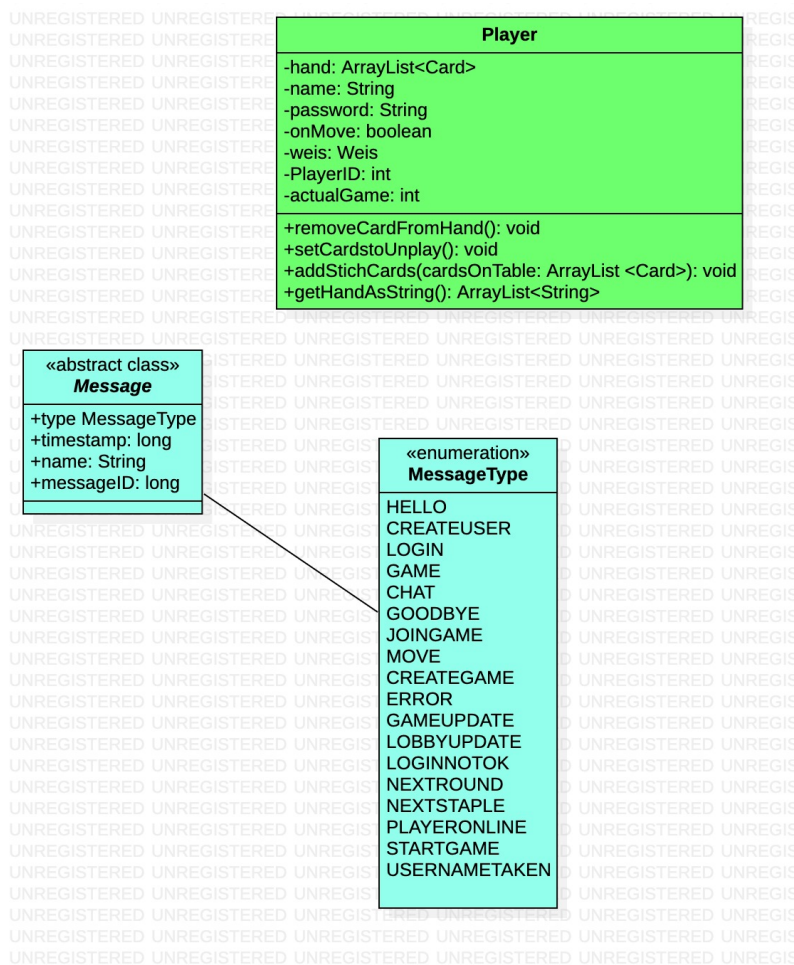
4.1 Package Client



4.2 Package Server



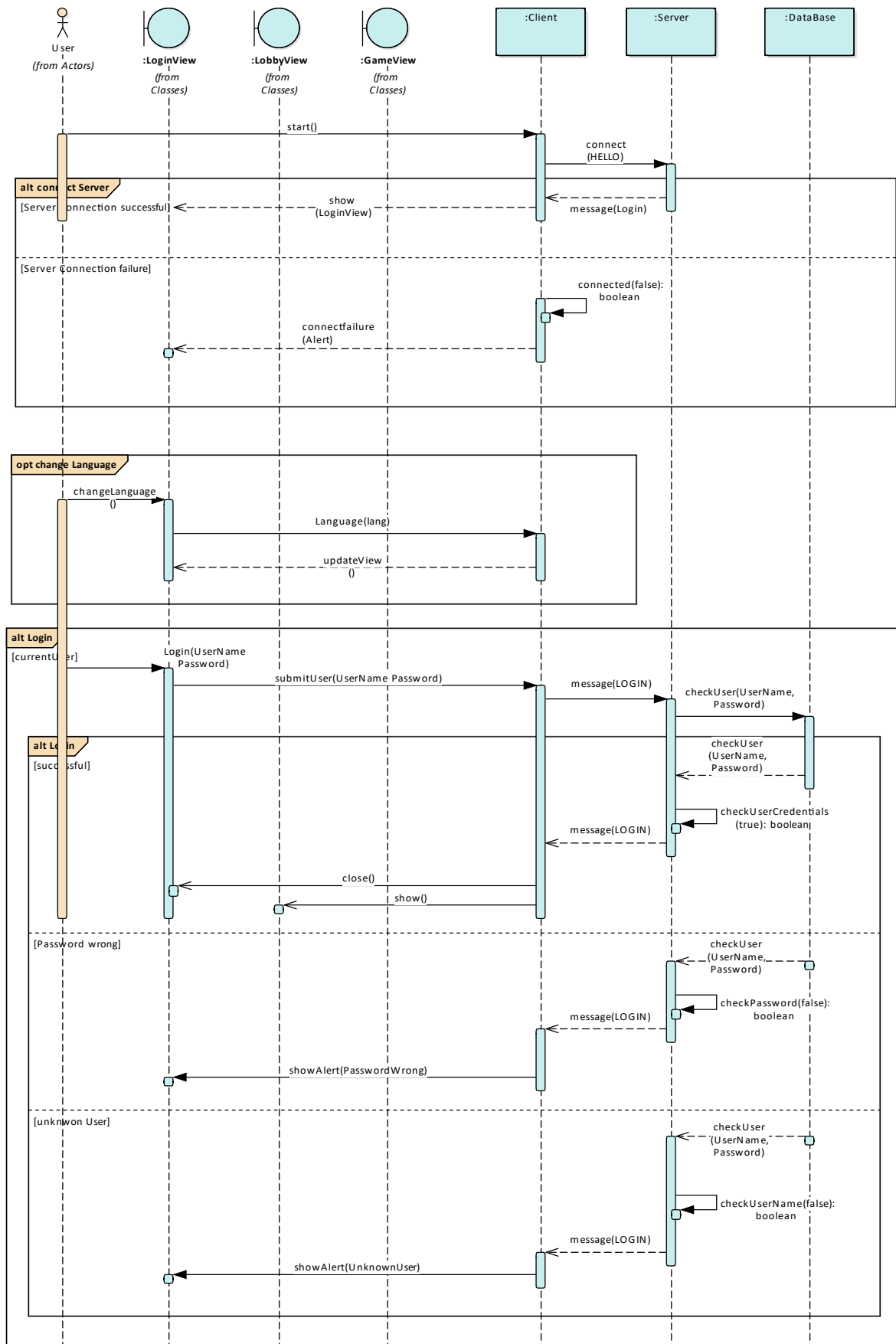
Package Commons

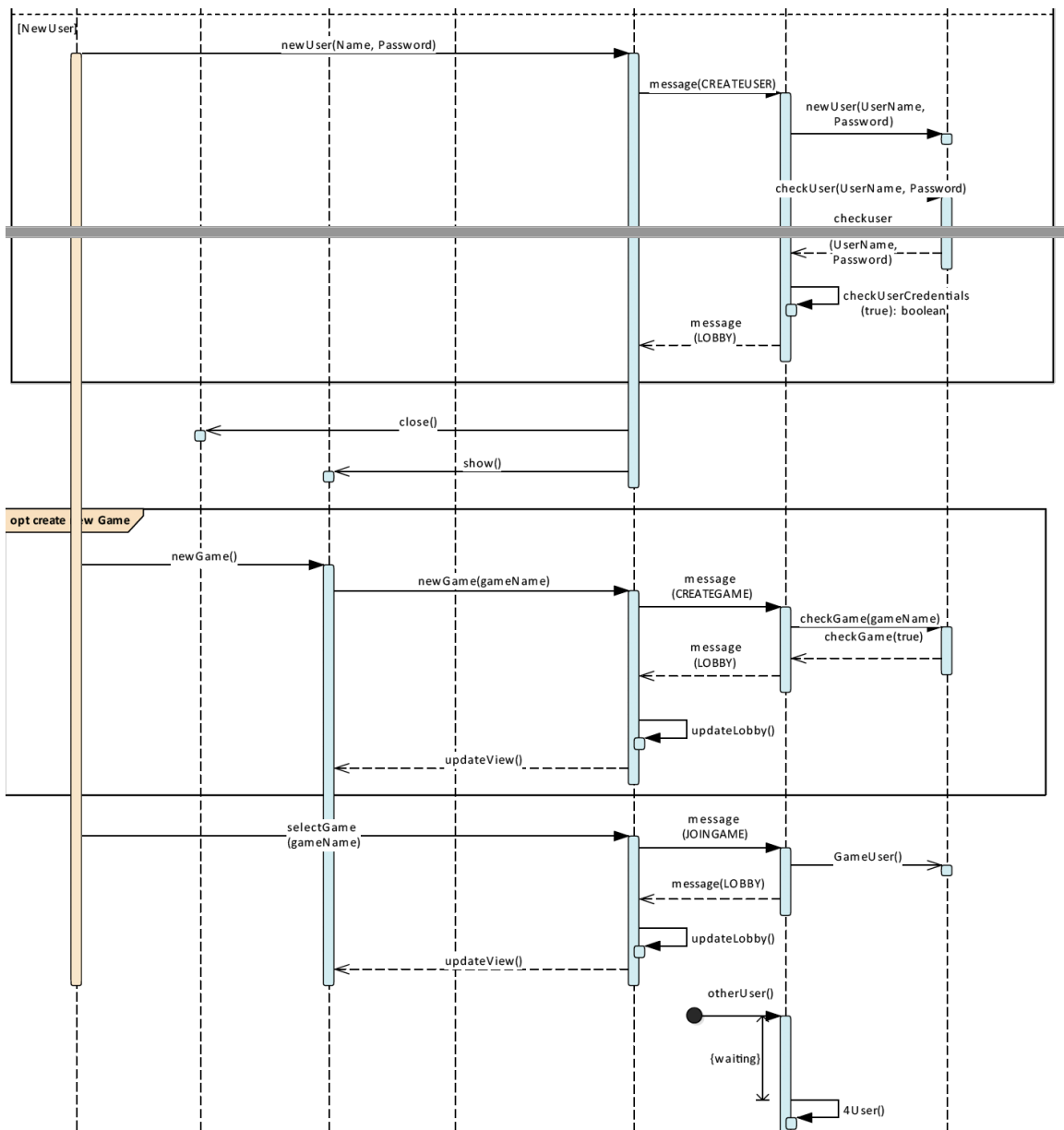


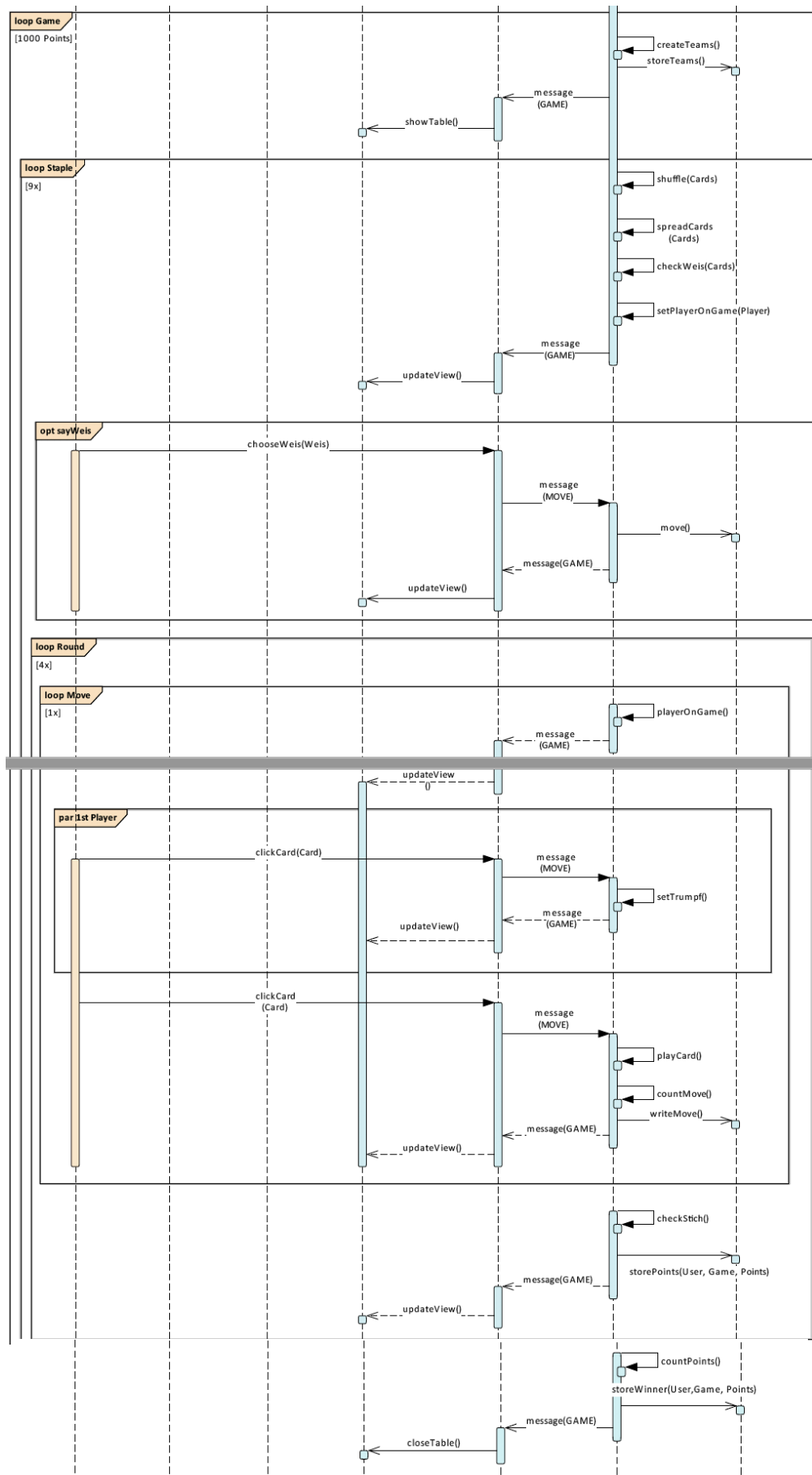
4.3 Verantwortlichkeiten

Zuständig	Verantwortlich für
Leon	Views, Database
Frank	Spiellogik, Spielablauf
Pascal	Server <-> Client, Messaging
Oliver	Spiellogik, Spielablauf

5 Sequenzdiagramm

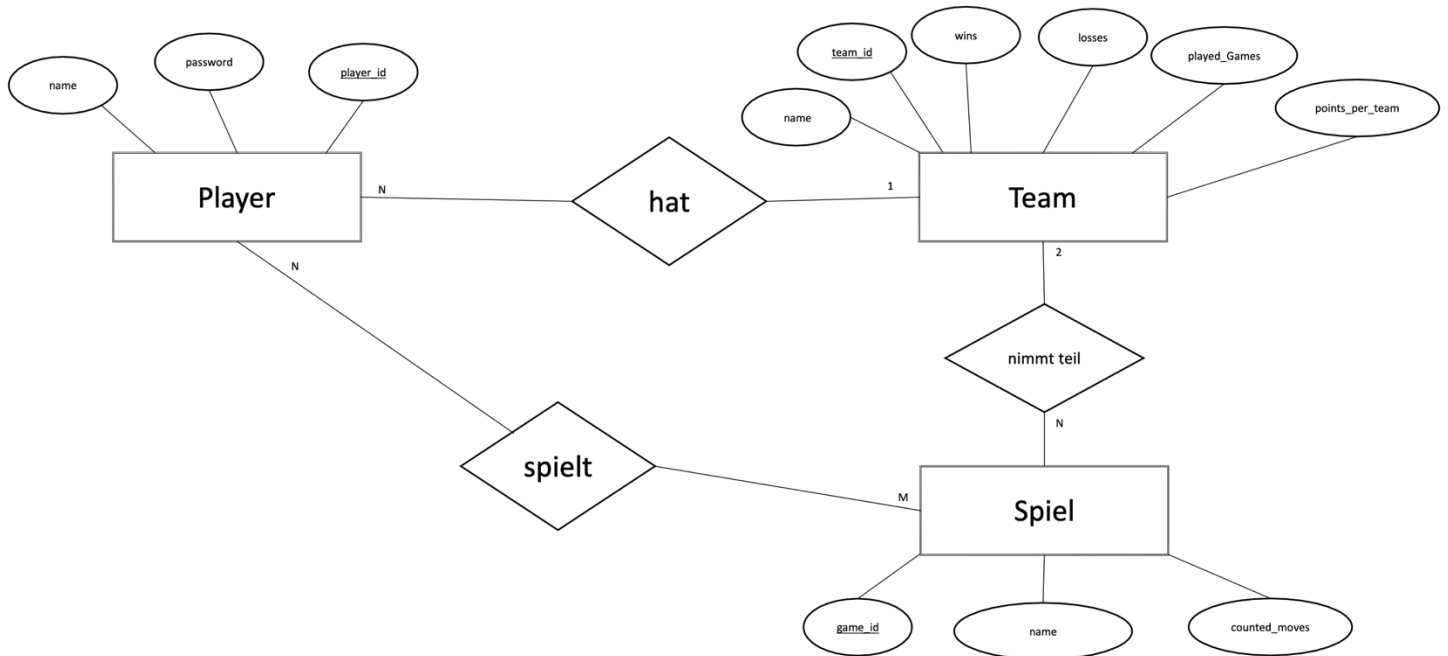




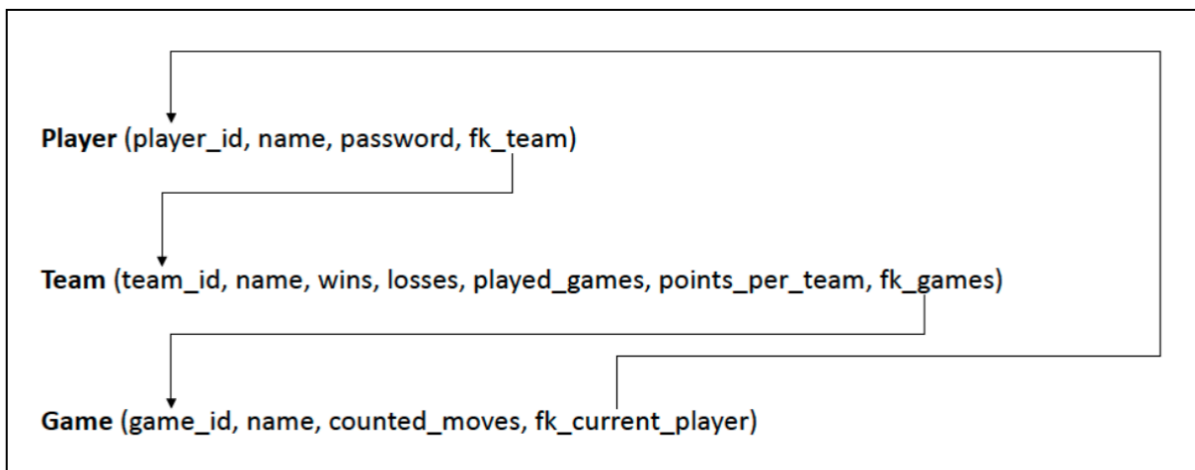


6 Datenbank

ER-Modell



6.1 Relationales Modell



6.2 Inhalte der Datenbank

1. Player:
 - player_id: int
 - name: varchar (30)
 - password: varchar (30)
2. Team:
 - team_id: int
 - name: varchar (30)
 - wins: int
 - losses: int
 - played_games: int
 - points_per_team: int
3. Spiel:
 - game_id: int
 - name: varchar (30)
 - counted_moves: int

