TSCHAU SEPP JAVA-EDITION

PROJEKTDOKUMENTATION MODUL 326

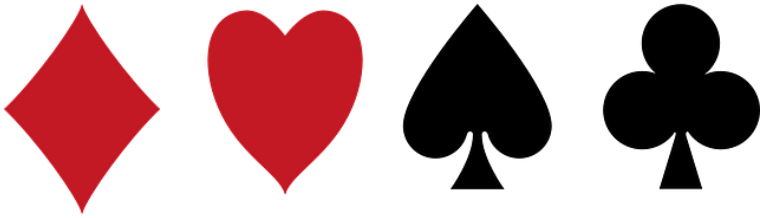
Auftraggeber: Rene Probst / Daniel Fahrni

Autorin: Mia Gudelj

gudeljm@bzz.ch / mia.gudelj@gmx.ch

Klasse: IMS18a

Projekttitel: Tschau Sepp



INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

PROJEKTBESCHREIBUNG

Mein Projekt ist es ein «Tschau Sepp» Spiel mit Java zu entwerfen und dazu eine Dokumentation zu erstellen. In diesem Programm werden die Regeln von «Tschau Sepp» implementiert. Die graphische Oberfläche mit der Technologie *Swing* realisiert. Die vier Spieler wechseln sich der Reihe nach am selben Computer ab.

ZIELE

- Es muss auf allen Geräten funktions- und lauffähig sein
- Das Programm muss den Spielregeln entsprechen
- Die Grafik und die Userfreundlichkeit müssen angemessen sein
- Alle einzelnen Projektschritte werden genau aufgeteilt und terminiert
- Der Benutzer sollte die Möglichkeit haben, die Regeln nachzulesen

ANWENDUNGSENTWURF

ANWENDUNGSFALLDIAGRAMME

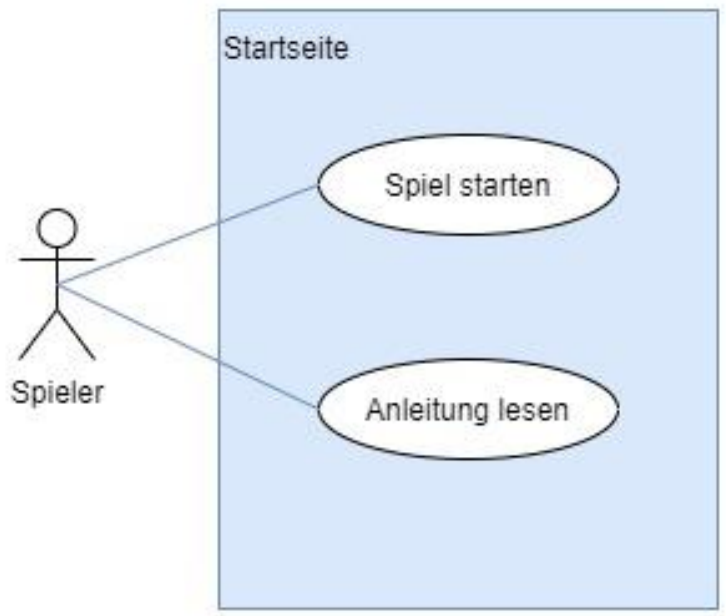


Abbildung 1: Startseite UseCase

Dem Benutzer steht es frei direkt mit dem Spiel zu beginnen, indem er auf «start» drückt, oder erstmals die Spielregeln durchzulesen. Will der Spieler die Spielregeln durchlesen, so hat er von dort aus auch die Möglichkeit das Spiel zu starten.

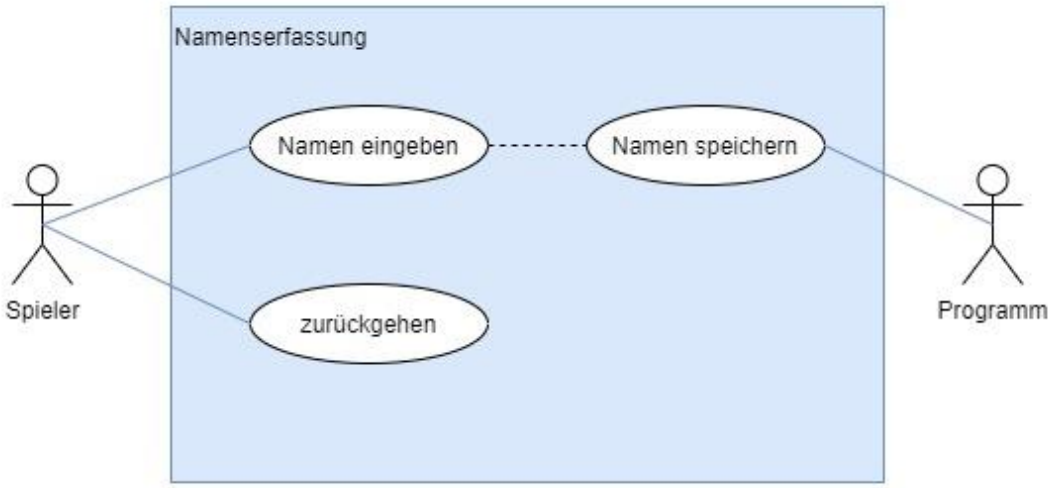


Abbildung 2: Namenserfassung UseCase

Bevor das Spiel überhaupt startet, müssen die Namen der vier Spieler erfasst werden, welche das Programm anschliessend speichert.

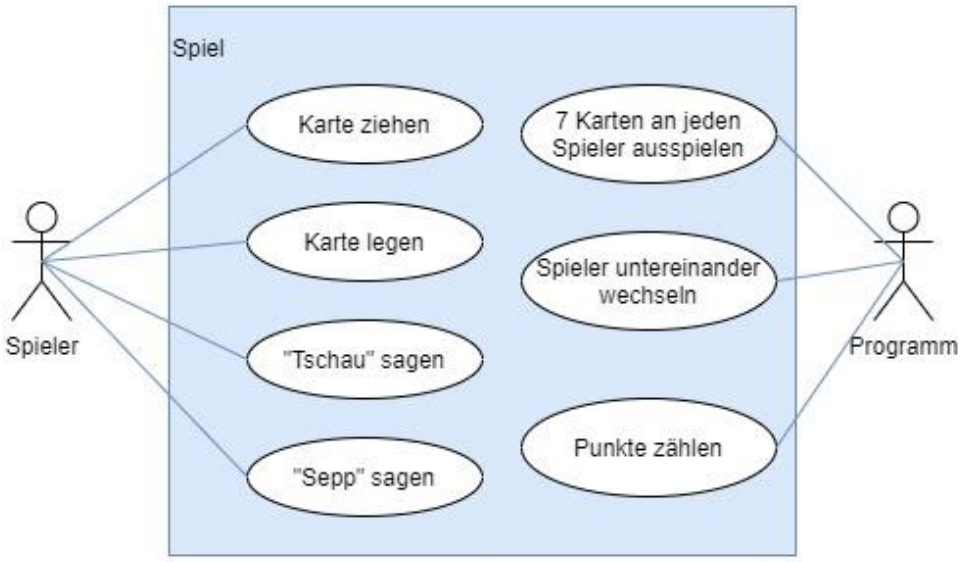


Abbildung 3: Spiel UseCase

Hat das Spiel begonnen, so verteilt das Programm jedem Spieler 7 zufällige Karten. Ausserdem wird eine Karte vom Kartenstapel offen nebendrann gelegt, woraufhin der Spieler, welcher gerade an der Reihe ist eine gültige Karte spielen muss. Gültig wird hier definiert als gleiche Zahl, Figur oder gleiches Symbol. Kann oder will der Spieler nicht legen, so muss er eine Karte ziehen. Kann er immer noch nicht legen, so wechselt das Programm zum nächsten Spieler. Dies geht solange weiter, bis einer der Spieler nurnoch eine einzige Karte besitzt. Dieser muss Dann «Tschau» rufen. Vergisst er dies, muss er zur Strafe zwei Karten ziehen. Hat der Spieler keine Karten mehr, so muss er «Sepp» rufen, andernfalls muss er vier Karten zur Strafe ziehen. Hat ein Spieler keine Karten mehr und «Sepp» gerufen, so zählt das Programm die Punkte der Karten der anderen Spielern zusammen und rechnet diese dem Gewinner der Runde zu. Hat noch kein Spieler 200 Punkte erreicht, so wird eine weitere neue Runde gestartet.

Abbildung 4: Spielende UseCase

Sobald ein Spieler die Runde, oder auch das Spiel gewonnen hat, so rangiert das Programm die Spieler nach Punkten absteigend. Des Weiteren steht dem Spieler jederzeit die Option zur Verfügung das Spiel zu beenden.

ANWENDUNGSFALLSPEZIFIKATIONEN

Tabelle 1: Startseite Anwendungsfallspezifikation

ID	1
Name	Startseite
Kurzbeschreibung	Beim Starten des Programmes, taucht als erstes die Startseite auf, wobei der Spieler die Wahl hat das Spiel zu starten oder die Anleitung zu lesen.
Akteur(e)	Spieler
Auslöser	Programmstart
Vorbedingung(en)	Min. Java-Version 12
Ergebnis	Startseite Maske erscheint
Nachbedingung(en)	Benutzer drückt auf eine Schaltfläche
Ablauf	1. Programm starten 2. «spielen» oder «Anleitung» drücken
Alternativen	❖ «Spielen» ❖ «Anleitung»
Fehlerfälle	-
Testfälle	3

Tabelle 2: Anleitung lesen Anwendungsfallspezifikation

ID	1.1
Name	Anleitung
Kurzbeschreibung	Die Spielregeln werden hier erklärt
Akteur(e)	Spieler
Auslöser	Schaltfläche «Anleitung»
Vorbedingung(en)	Schaltfläche «Anleitung» wurde getätigt
Ergebnis	Spieler wird zu einer Maske, in der die Spielregeln als Fliesstext stehen, weitergeleitet.
Nachbedingung(en)	Spieler tätigt Schaltfläche «spielen»
Ablauf	1. «spielen» drücken
Alternativen	-
Fehlerfälle	-
Testfälle	2

Tabelle 3: Namenserfassung Anwendungsfallspezifikation

ID	2
Name	Namenserfassung
Kurzbeschreibung	Hier werden die Namen der Spieler erfasst
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4
Auslöser	Schaltfläche «spielen»
Vorbedingung(en)	Schaltfläche «spielen» wurde getätigt
Ergebnis	Neue Maske mit Textfeldern, um die Namen einzugeben
Nachbedingung(en)	Alle Namen wurden erfasst Schaltfläche «Spiel starten» wird getätigt
Ablauf	1. Name von Spieler 1 wurde erfasst 2. Name von Spieler 2 wurde erfasst 3. Name von Spieler 3 wurde erfasst 4. Name von Spieler 4 wurde erfasst 5. «Spiel starten» wird gedrückt
Alternativen	«zurück» → hier kann er wieder zurück gehen zur Anleitungs-Maske um die Spielregeln (nochmals) durchzulesen.
Fehlerfälle	❖ Name bereits vergeben ❖ Name zu lang ❖ Name leer ❖ Name ungültig
Testfälle	7

Tabelle 4: Spielstart Anwendungsfallspezifikation

ID	3
Name	Spiel - Spielstart
Kurzbeschreibung	Neue Runde des Spiels beginnt
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4 ❖ Programm
Auslöser	«Spiel starten»
Vorbedingung(en)	❖ Alle Namen wurden erfasst und sind gültig ❖ Schaltfläche «Spiel starten» wurde getätigt
Ergebnis	Neue Maske für das Spiel erscheint mit allen 4 Spielern, je Spieler 7 Karten, einem umgedrehten Kartenstapel und einer offengelegten Karte neben dem Kartenstapel.
Nachbedingung(en)	❖ Jeder Spieler besitzt 7 Karten ❖ Ein Spieler wurde zufällig gewählt, der anfangen darf ❖ Eine Karte wurde offen neben dem Kartenstapel gelegt
Ablauf	1. 7 zufällige Karten werden jedem Spieler verteilt 2. Ein Spieler wird zufällig gewählt, der anfangen kann
Alternativen	❖ Spiel beenden ❖ Anleitung
Fehlerfälle	-
Testfälle	5

Tabelle 5: Spielerzug Anwendungsfallspezifikation

ID	3.1
Name	Spiel - Spielerzug
Kurzbeschreibung	Hier spielen nun die Spieler das Spiel
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4
Auslöser	-
Vorbedingung(en)	❖ Jeder Spieler besitzt 7 Karten. ❖ Ein Spieler wurde zufällig gewählt, der anfangen darf. ❖ Eine Karte wurde offen neben dem Kartenstapel gelegt.
Ergebnis	Maske für das Spiel
Nachbedingung(en)	❖ Spieler n hat eine gültige Karte ausgewählt oder eine Karte gezogen.
Ablauf	1. Falls Spieler n eine spielbare Karte besitzt, legt er diese, indem er sie anklickt. 2. Falls 1. nicht erfüllt, zieht der Spieler eine Karte. a. Kann er immer noch nicht Spielen, tritt 3. ein. b. Kann er diese Karte jedoch spielen, so tritt 1. ein und danach 3. 3. Nächster Spieler ist dran.
Alternativen	❖ Karte legen ❖ Karte ziehen ❖ Spiel beenden ❖ Anleitung
Fehlerfälle	❖ Zahl ist nicht identisch zur Karte auf dem Ablagestapel. ❖ Symbol ist nicht identisch zur Karte auf dem Ablagestapel. ❖ Figur ist nicht identisch zur Karte auf dem Ablagestapel.
Testfälle	12

Tabelle 6: "Tschau" Anwendungsfallspezifikation

ID	3.2
Name	Spiel – «Tschau»
Kurzbeschreibung	Der Spieler hat noch 2 Karten
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4
Auslöser	Schaltfläche «Tschau» wurde getätigt.
Vorbedingung(en)	Spieler hat 2 Karten, hat aber mindestens eine spielbare Karte, die er legen wird.
Ergebnis	
Nachbedingung(en)	Spieler hat eine Karte.
Ablauf	1. Spieler n drückt Schaltfläche «Tschau». a. Hat der Spieler mehr als 2 Karten oder keine spielbare Karte von den 2 Karten, so muss er 2 Strafkarten ziehen. 2. Der Spieler legt eine spielbare Karte. 3. Spieler wird gewechselt.
Alternativen	❖ Karte legen ❖ Karte ziehen ❖ Spiel beenden ❖ Anleitung
Fehlerfälle	❖ Spieler drückt «Tschau» und hat mehr als 2 Karten ❖ Spieler drückt «Tschau» aber kann keine Karte spielen ❖ Spieler vergisst «Tschau» drücken
Testfälle	4

Tabelle 7: "Sepp" Anwendungsfallspezifikation

ID	3.3
Name	Spiel – «Sepp»
Kurzbeschreibung	Der Spieler hat noch 2 Karten
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4
Auslöser	Schaltfläche «Sepp» wurde getätigt.
Vorbedingung(en)	Spieler hat eine Karte, die spielbar ist.
Ergebnis	Runde beendet
Nachbedingung(en)	Spieler hat noch keine 200 Punkte erreicht
Ablauf	1. Spieler n drückt Schaltfläche «Sepp». a. Hat der Spieler mehr als eine Karte oder ist diese nicht spielbar, so muss er 4 Strafkarten ziehen. 2. Der Spieler legt seine spielbare Karte. 3. Spieler wird gewechselt.
Alternativen	❖ Karte legen ❖ Karte ziehen ❖ Spiel beenden ❖ Anleitung
Fehlerfälle	❖ Spieler drückt «Sepp» und hat mehr als eine Karte ❖ Spieler drückt «Sepp» aber kann keine Karte spielen ❖ Spieler vergisst «Sepp» drücken
Testfälle	4

Tabelle 8: Scorebord Anwendungsfallspezifikation

ID	4
Name	Scorebord
Kurzbeschreibung	Punkte werden am Ende jeder Runde zusammengezählt und dem Gewinner der Runde zugerechnet. Dies wird anhand einer Rangliste dann dargestellt. Hat jemand 200 Punkte erreicht, so ist das Spiel beendet.
Akteur(e)	❖ Spieler 1 ❖ Spieler 2 ❖ Spieler 3 ❖ Spieler 4 ❖ Programm
Auslöser	Spieler n spielt seine letzte Karte aus.
Vorbedingung(en)	Ein Spieler hat keine Karten mehr.
Ergebnis	❖ Die Punkte der Karten der Mitspieler werden zusammengezählt und dem Gewinner der Runde gutgeschrieben. ❖ Eine Rangliste ist ersichtlich
Nachbedingung(en)	Spieler haben 200 Punkte noch nicht erreicht.
Ablauf	1. Punkte der Mitspieler, die noch Karten haben, werden zusammengezählt und dem Gewinner der Runde gutgeschrieben. 2. Rangliste erscheint 3. a) Hat ein Spieler 200 Punkte erreicht, so ist das Spiel zu ende. b) Sonst wird eine neue Runde gestartet (Spezifikation 3 – 3.x)
Alternativen	❖ Nächste Runde ❖ Spiel beenden
Fehlerfälle	❖ Nach Erreichen von 200 Punkten geht das Spiel weiter
Testfälle	4

ANFORDERUNGEN

FUNKTIONALE ANFORDERUNGEN

- Möglichkeit für den Spieler haben, die Spielregen jederzeit nachzulesen
- Spielername setzen
- Spiel starten
- Spiel beenden
- «Tschau Sepp» spielen
  - Karten mischen
  - jedem Spieler 7 Karten verteilen
  - Karten legen
  - Karten ziehen
  - zum nächsten Spieler wechseln, sobald der eine gespielt hat
  - «tschau» rufen
  - «sepp» rufen
- Den Spielregeln entsprechende Züge erlauben
- Punkte am Ende zusammenzählen
- Rangliste ausgeben

NICHT FUNKTIONALE ANFORDERUNGEN

- muss auf allen Betriebssystemen und Geräten mit Java lauffähig sein.
- Darstellung/Programm muss für den Benutzer übersichtlich sein. Das heisst, der Benutzer wird nicht von irrelevantem Content abgelenkt oder verwirrt.
- Selbstbeschreibungsfähig

AKTIVITÄTSDIAGRAMM

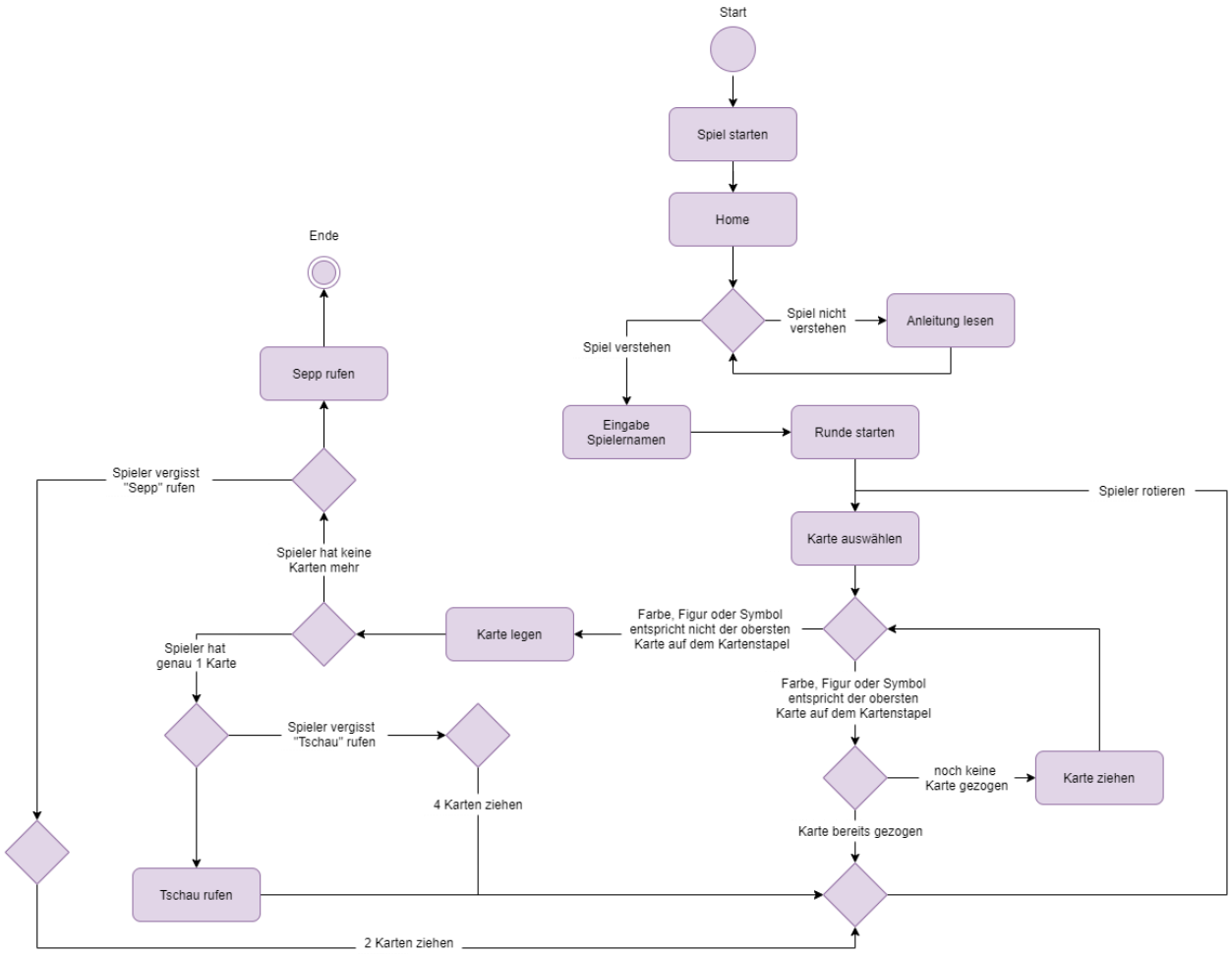


Abbildung 5: Sequenzdiagramm Aktivitätsdiagramm

CRC-KARTEN

Tabelle 9: CRC-Karte Card

Class: Card *Spielkarte mit der gespielt wird und eine Zahl/Figur/Farbe enthält oder ist eine Strafkarte.*	
Responsibilities: ❖ Hat eine Zahl ❖ Hat eine Figur ❖ Hat eine Farbe ❖ Hat eine Aktion (z.B. 2 Karten ziehen) ❖ Vergleichbar mit anderer Karte ❖ Hat Punktwert	**Collaborators**: ❖ CardBank ❖ CardStack ❖ CardDeck

Tabelle 10: CRC-Karte CardBank

Class: CardBank *Ist eine Kartensammlung verschiedener, gemischter Karten, doch mit der Rückseite nach oben gedreht.*	
Responsibilities: ❖ Gibt zufällige Karten aus ❖ Gibt nur Karten an aktuellen Spieler aus. ❖ Gibt nur eine gewisse Anzahl Karten aus pro Zug.	**Collaborators**: ❖ Card ❖ CardStack ❖ Player ❖ Game

Tabelle 11: CRC-Karte CardStack

Class: CardStack *Ablage der gespielten Karten, doch Karten sind mit der Vorderseite nach oben gederht.*	
Responsibilities: ❖ Lässt nur Karten zu, die der obersten Karte des Stapels nach einem Kriterium entspricht. ❖ Akzeptiert nur Karten von aktuellem Spieler	**Collaborators**: ❖ Card ❖ Player ❖ Game

Tabelle 12: CRC-Karte CardDeck

Class: CardDeck *Kartensammlung, die jeder Spieler noch in seiner Hand hat.*	
Responsibilities: ❖ Zählt am schluss die Punkte der Karten zusammen	**Collaborators**: ❖ Card ❖ Player

Tabelle 13: CRC-Karte Player

Class: Player *Akteure, die das Spiel spielen und Karten besitzen*	
Responsibilities: ❖ Hat Punkte ❖ Besitzt Karten ❖ Legt Karten ❖ Zieht Karten	**Collaborators**: ❖ CardDeck ❖ CardStack ❖ CardBank

Tabelle 14: CRC-Karte Game

Class: Game *Das Spiel besteht aus mehreren Spiel-Runden und 4 Spielern.*	
Responsibilities: ❖ Setzt die Spieler fest ❖ Legt Karten aus ❖ Zählt Runden ❖ Wechselt Spieler untereinander ab	**Collaborators**: ❖ Player ❖ CardBank ❖ CardStack

KLASSENDIAGRAMME

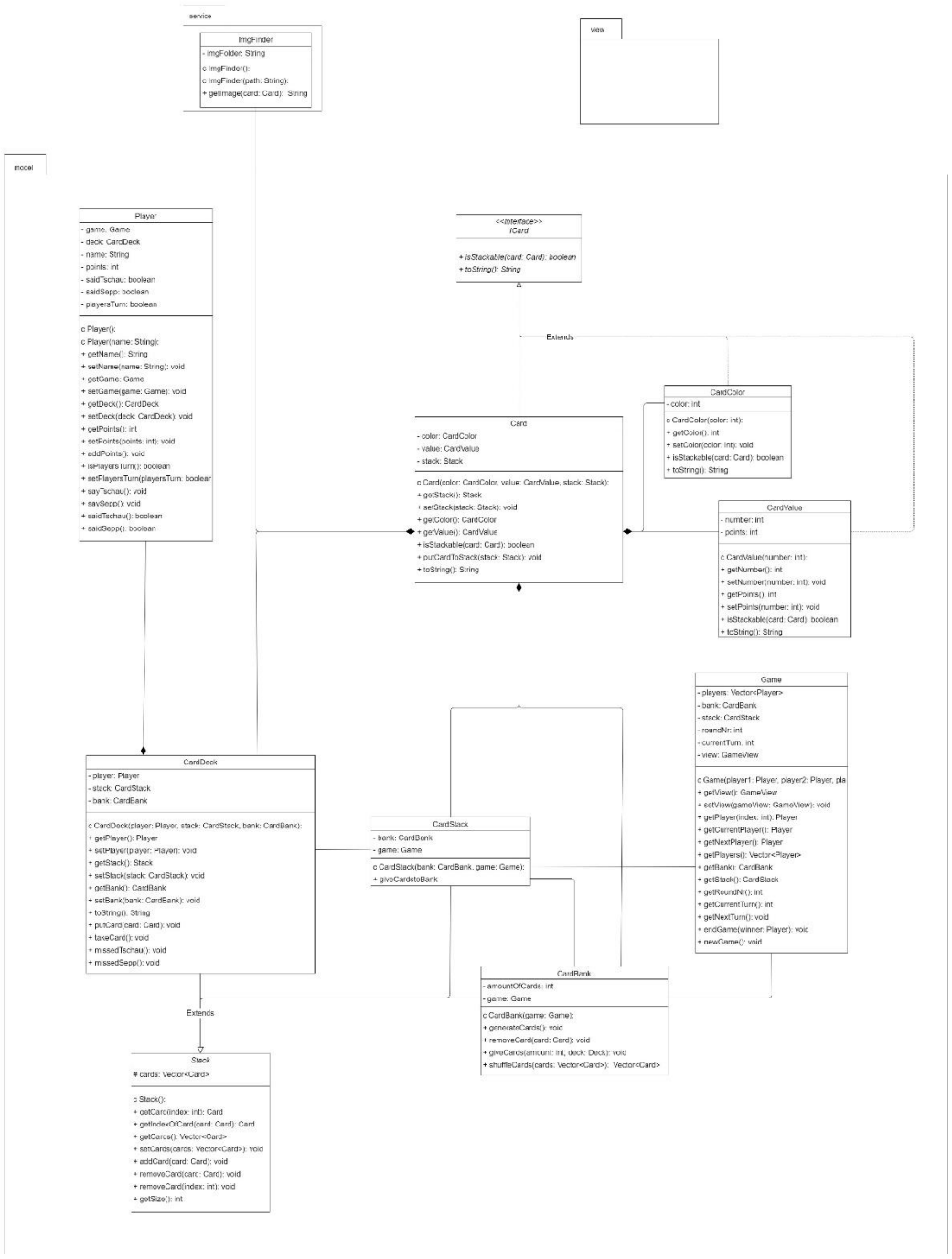


Abbildung 6: Klassendiagramm

ZUSTANDSDIAGRAMME

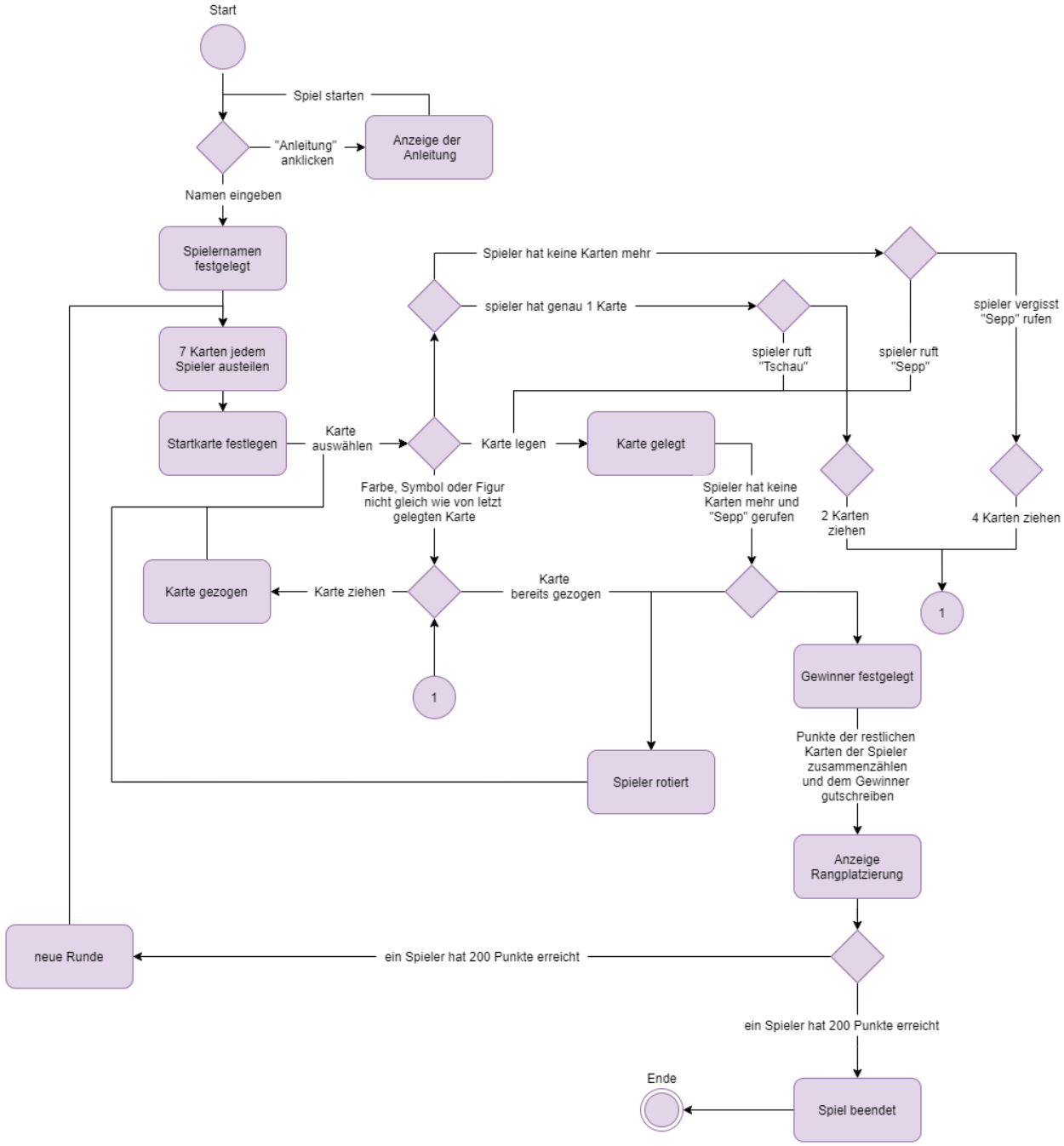


Abbildung 7: Zustandsdiagram

SEQUENZDIAGRAMME

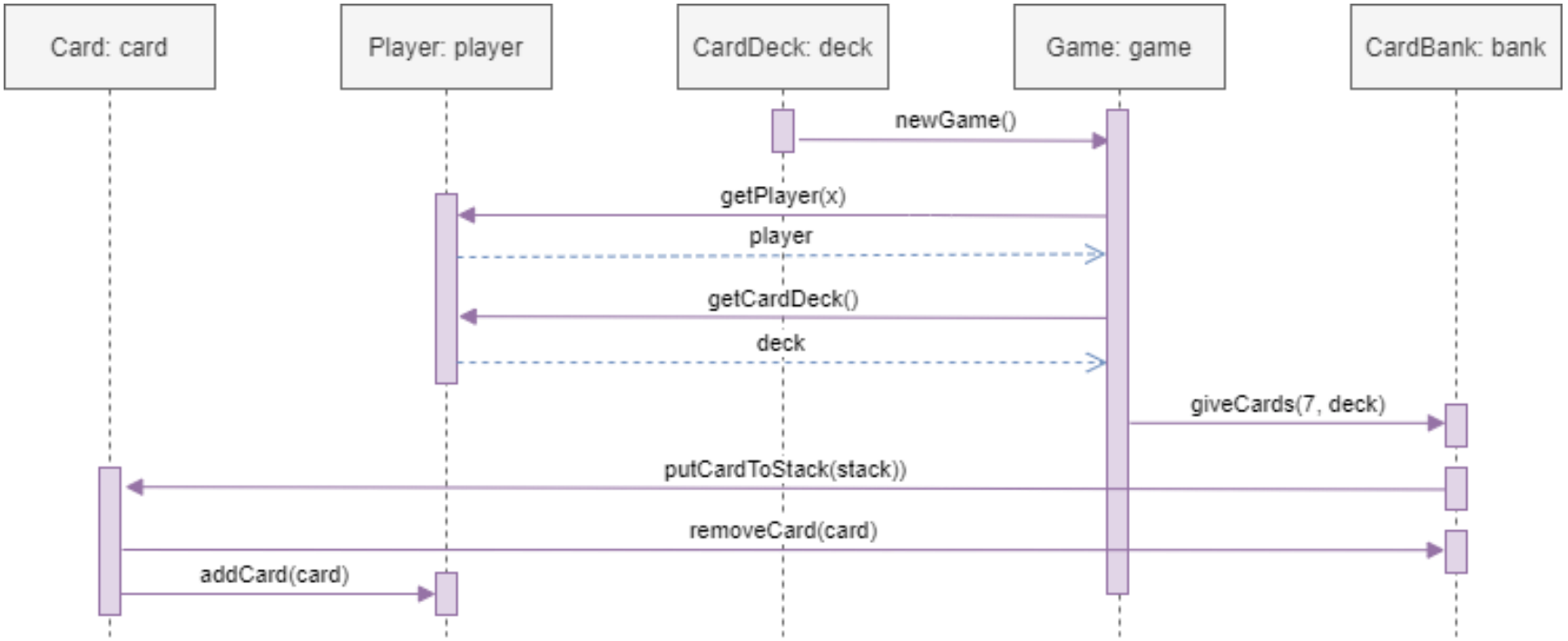


Abbildung 8: Sequenzdiagramm Spielstart

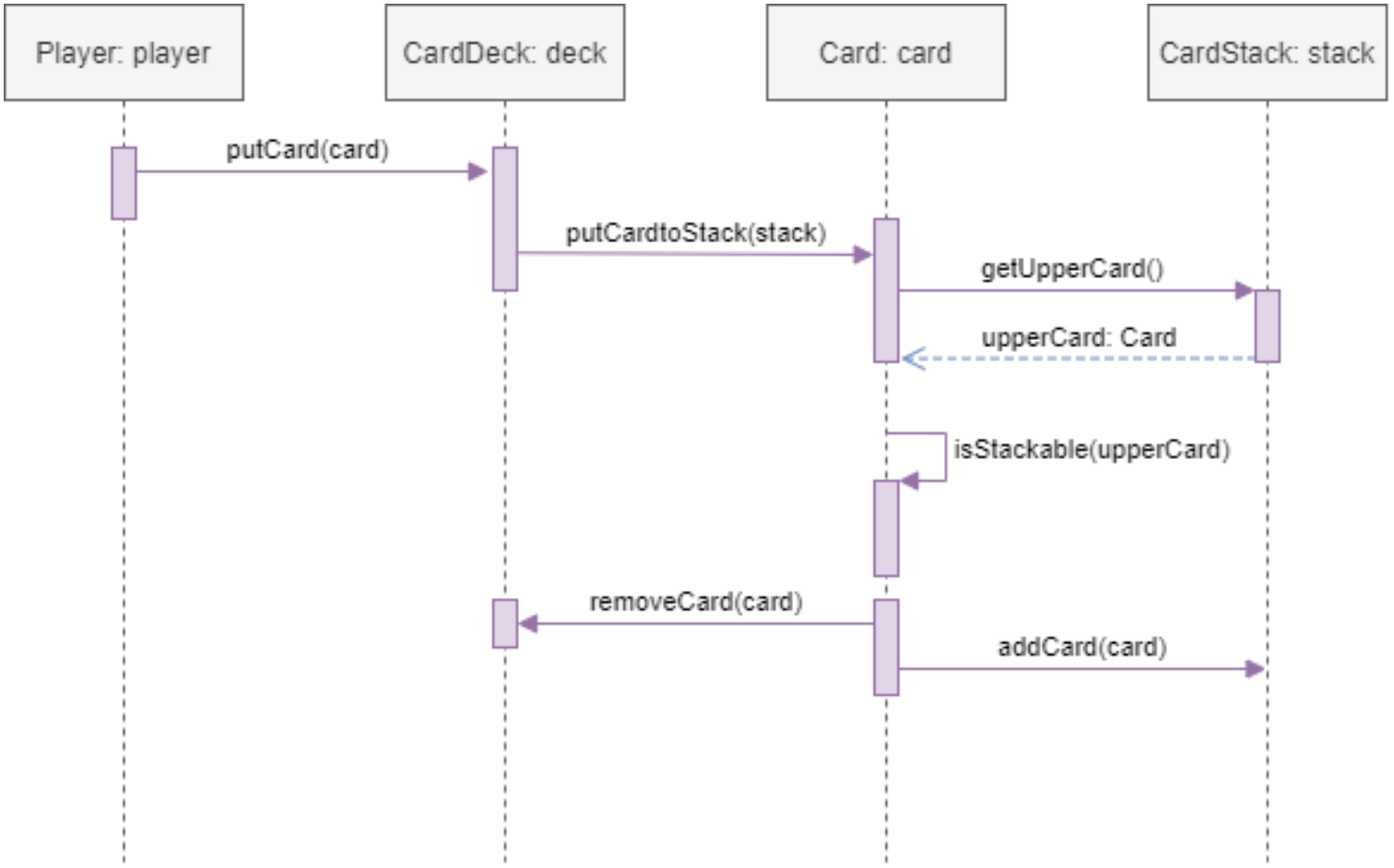


Abbildung 9: Sequenzdiagram Karte legen

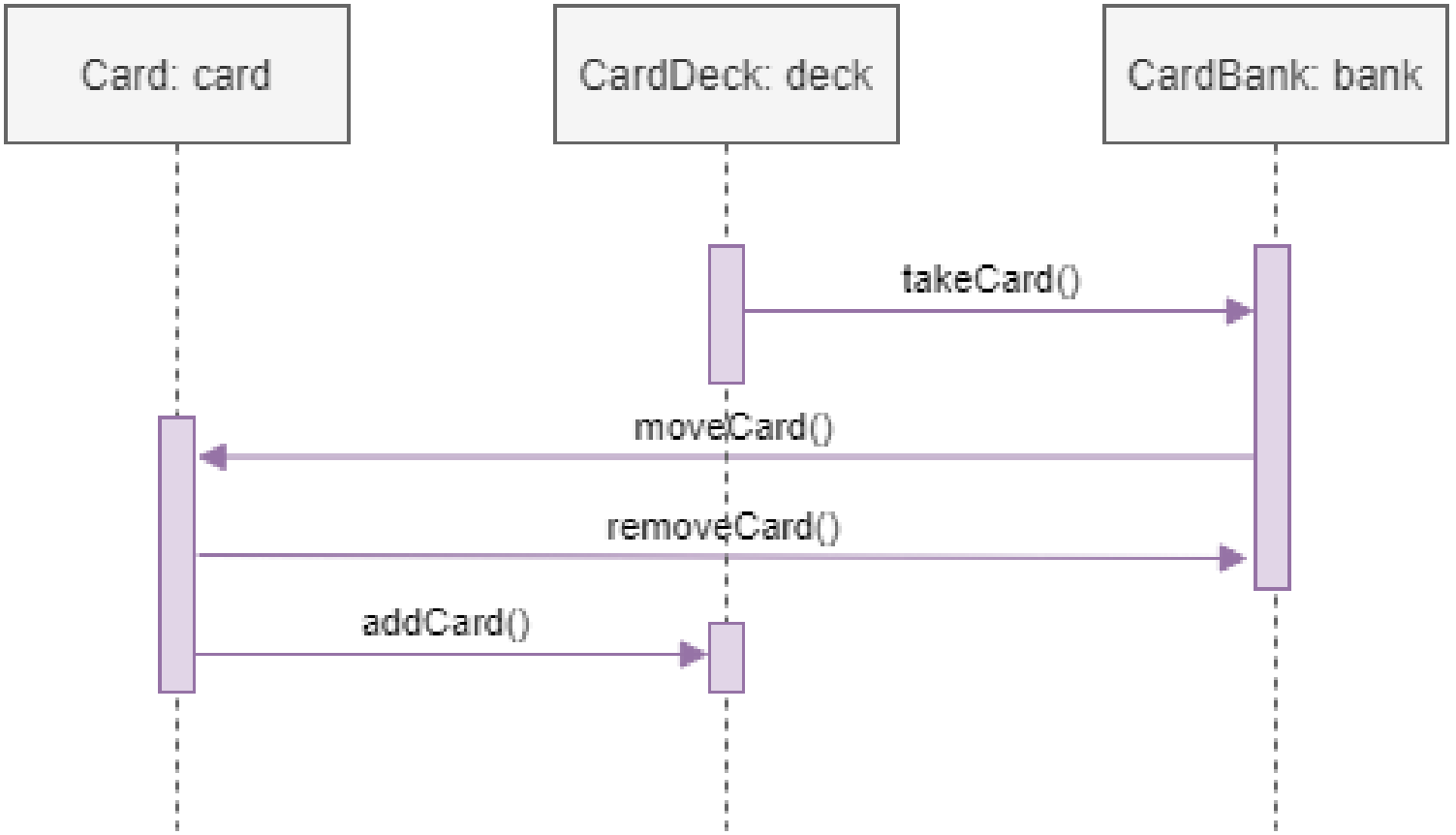


Abbildung 10: Sequenzdiagram Karte aufnehmen

MOCKUPS

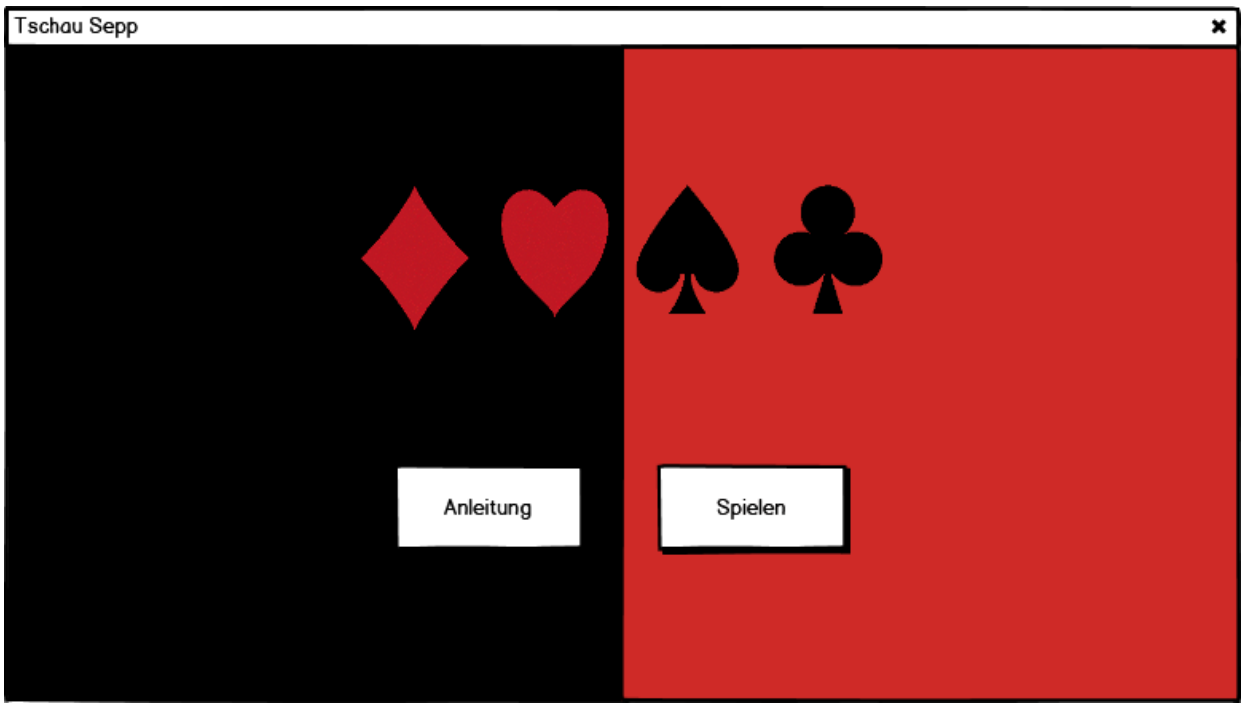


Abbildung 11: Mockup Startseite

Dies ist die Startseite, also die Maske, welches als erstes erscheint, wenn das Programm gestartet wird. Hier hat der Benutzer die Wahl direkt mit dem Spiel zu beginnen und die Spielernamen zu erfassen oder sich zuerst die Spielregeln durchzulesen.

Komponente:

- GridLayout
- BorderLayout
- JButton

Abbildung 12: Mockup Anleitung

Entscheidet sich der Benutzer dafür, die Anleitung zu lesen, erscheint diese Maske, in der das Spiel erklärt wird. Mit dem Tätigen des «Spielen»-Buttons gelangt er zur nächsten Maske, wobei er die Spielernamen erfassen muss.

Komponente:

- GridLayout
- BorderLayout
- JButton
- JScrollpane
- JLabel

Abbildung 13: Mockup Namenserfassung

Hier ist die Maske, wo der Benutzer alle Spieler erfassen muss. Solange nicht alle Spielernamen korrekt erfasst wurden, bleibt die Schaltfläche «START» um das Spiel zu starten nicht anklickbar (disabled). Mit dem Button «zurück» gelangt der Benutzer wieder zur Startseite.

Komponente:

- GridLayout
- BorderLayout
- JButton
- JTextfield
- JLabel

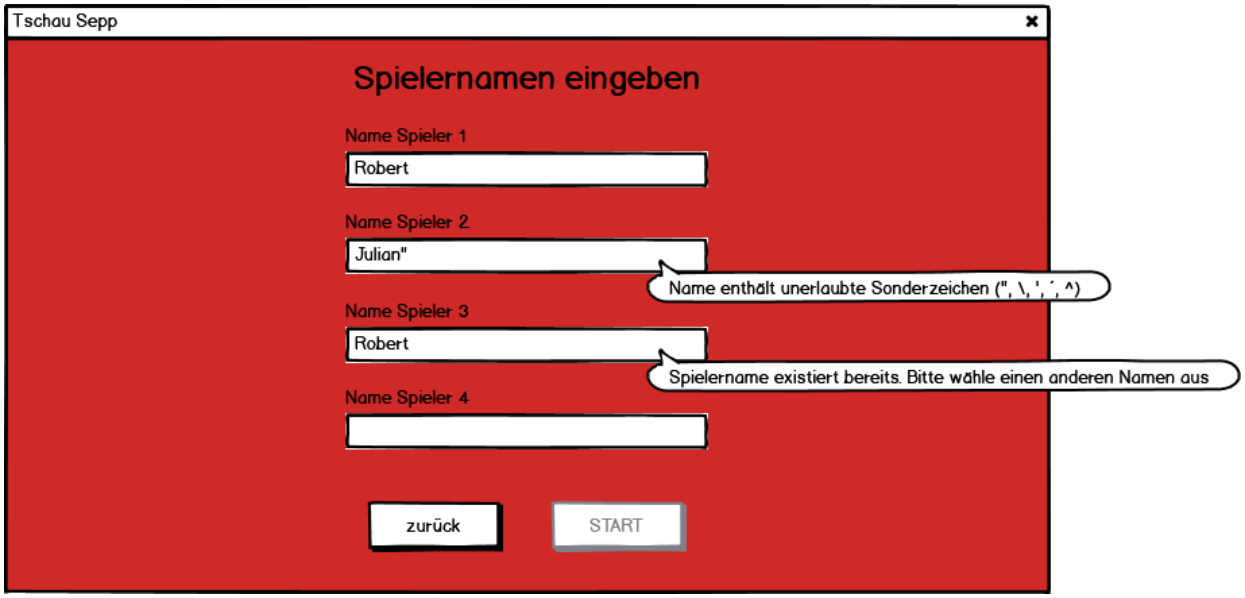


Abbildung 14: Mockup Namenserfassung fehlerhaft

Sollte der Benutzer im Spielernamen nichterlaubte Zeichen eingeben oder einen Namen, welcher bereits existiert, wird durch Tooltips die entsprechende Warnmeldung angezeigt, was einerseits der Fehlerrobustheit als auch der Lernförderlichkeit entspricht.

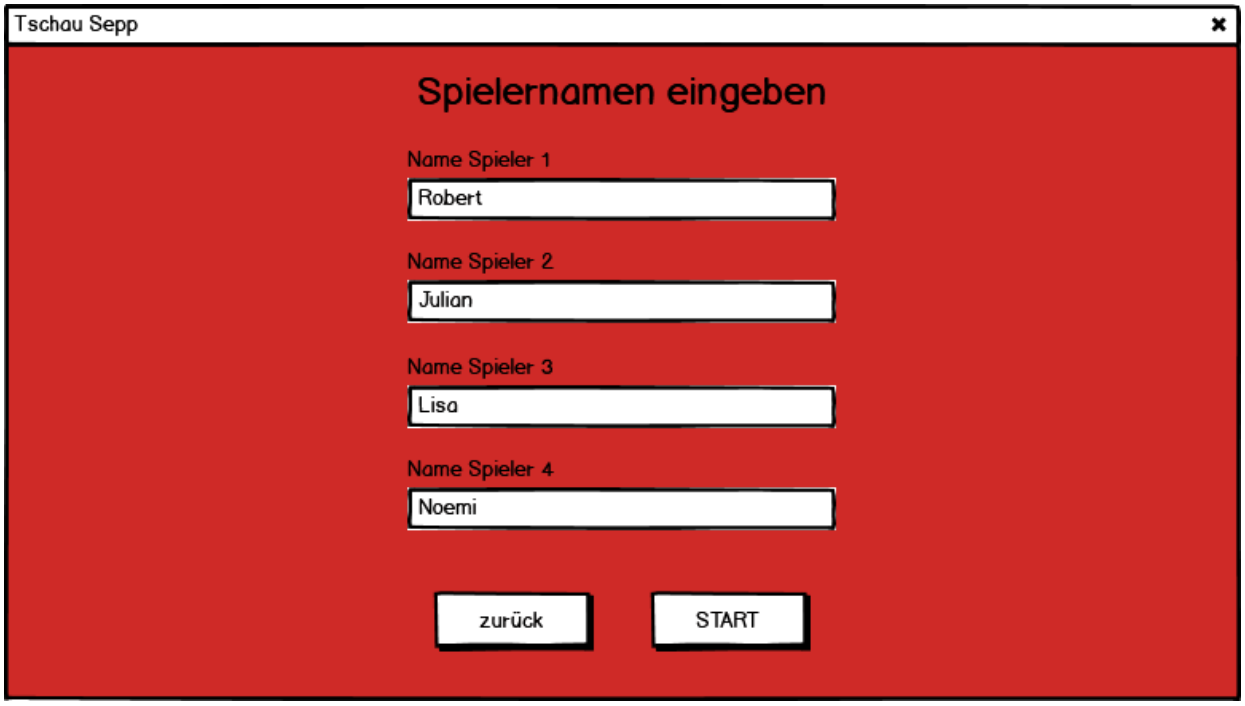


Abbildung 15: Mockup Namenserfassung ausgefüllt

Sind alle Textfelder korrekt ausgefüllt, so wird der «START»-Button wieder aktiviert/anwählbar.

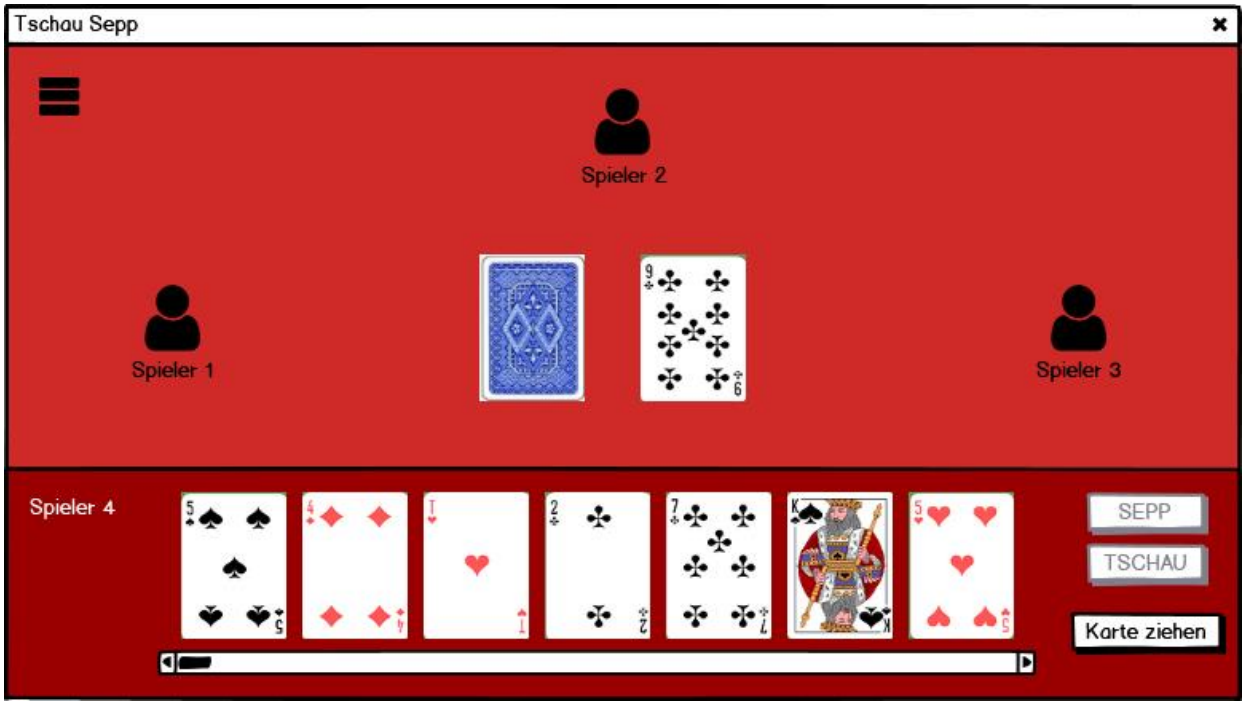


Abbildung 16: Mockup Spiel Start

Dies ist die Maske des Spiels, wenn die Karten verteilt wurden. Die Karten-Bank, als auch die Kartenbeige liegen in der Mitte des Fensters. Die Karten des Spielers, der gerade am Zug ist, werden unten im Fenster angezeigt. Hat der Spieler mehr Karten, so kann er scrollen. Links wird der Spielername angezeigt (*hier jetzt Spieler 4*) und rechts hat er drei Knöpfe. Jedoch ist nur die Schaltfläche «Karte ziehen» aktiv. Die anderen zwei sind nicht anklickbar, da der Spieler noch zu viele Karten besitzt.
Oben links hat der Spieler die Möglichkeit ein Menü aufzumachen. Doch das Menü wird weiter unten beschrieben (*siehe Abbildung 21, S.38*).

Komponente:

- GridLayout
- BorderLayout
- JButton
- JScrollPane
- JLabel

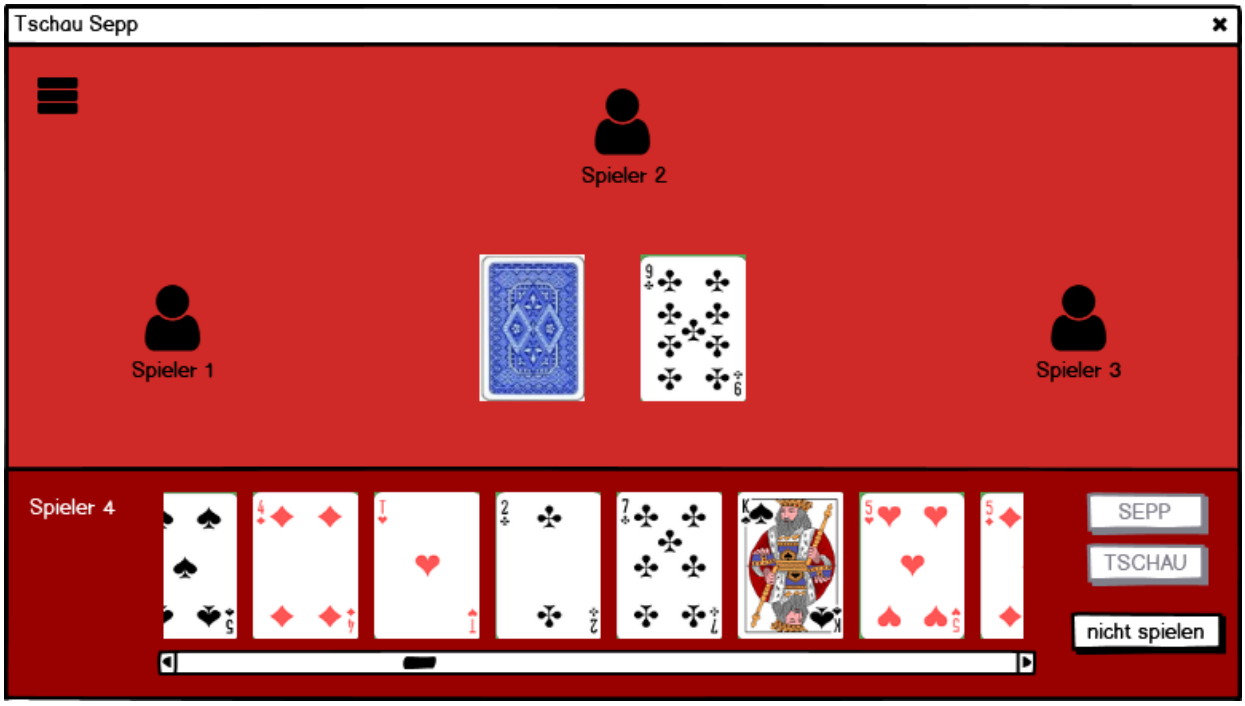


Abbildung 17: Mockup Spiel Karte gezogen

Hier tritt der Fall ein, wo der Spieler mehrmals eine Karte ziehen musste und nun scrollen kann. Auch hat sich die unterste Schaltfläche geändert. Anstelle der Beschriftung «Karte ziehen» steht jetzt «nicht spielen». Der Spieler kann nur ein Mal eine Karte ziehen. Kann oder will der Spieler keine Karte legen, so kommt der nächste Spieler an die Reihe.

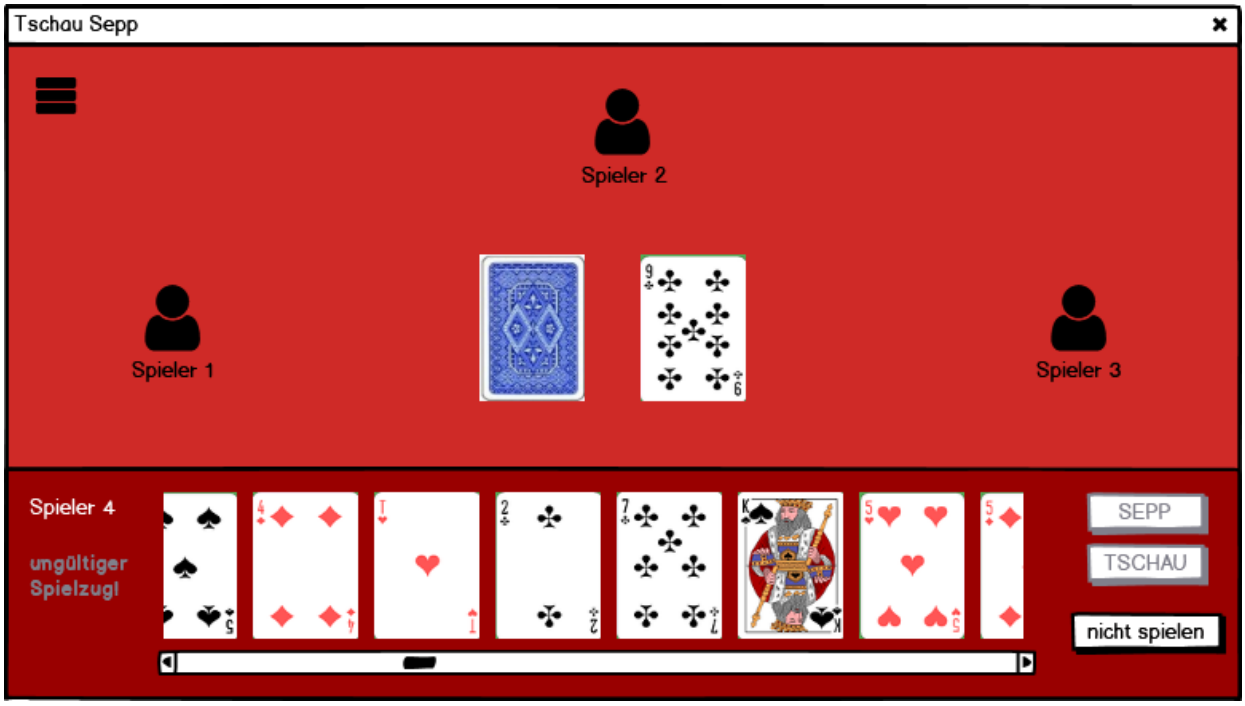


Abbildung 18: Mockup Spiel ungültiger Zug

Will der Spieler eine Karte legen, die nicht der offengelegten Karte auf dem Kartenstapel liegt, entspricht, so wird unten links eine Meldung ausgegeben, was einerseits der Fehlerrobustheit, als auch der Lernförderlichkeit entspricht.

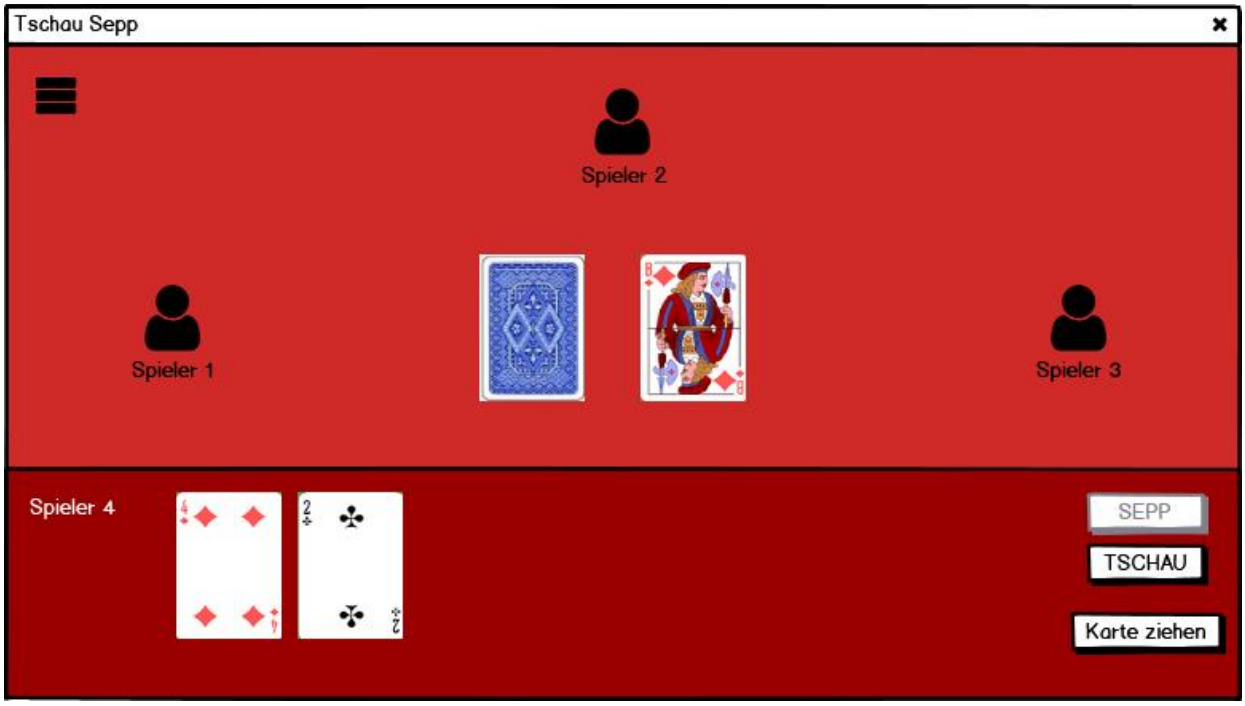


Abbildung 19: Mockup Spiel Tschau

Besitzt der Spieler nur noch zwei Karten, so wird der Knopf «TSCHAU» aktiviert.

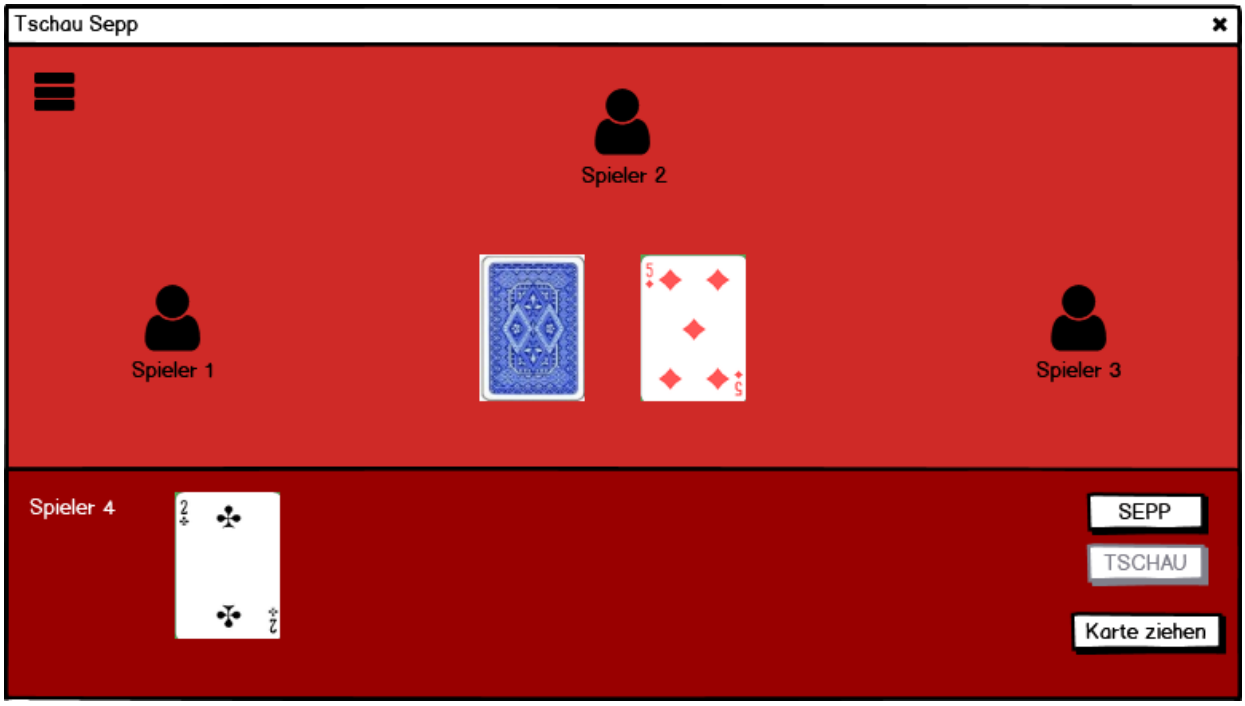


Abbildung 20: Mockup Start Sepp

Besitzt der Spieler nur noch eine Karte, so wird der Knopf «SEPP» aktiviert und «TSCHAU» wieder deaktiviert.

Abbildung 21: Mockup Menü

Drückt der Spieler auf das Menü-Zeichen, so taucht dieses Pop-Up auf, wobei der Spieler jederzeit die Wahl hat das Spiel zu beenden oder die Spielregeln zu lesen. Will er Spieler bzw. der Benutzer wieder zum Spiel zurückkehren, so muss er die Schaltfläche zurück tätigen, oder das Fenster schliessen.

Komponente:

- GridLayout
- BorderLayout
- JButton

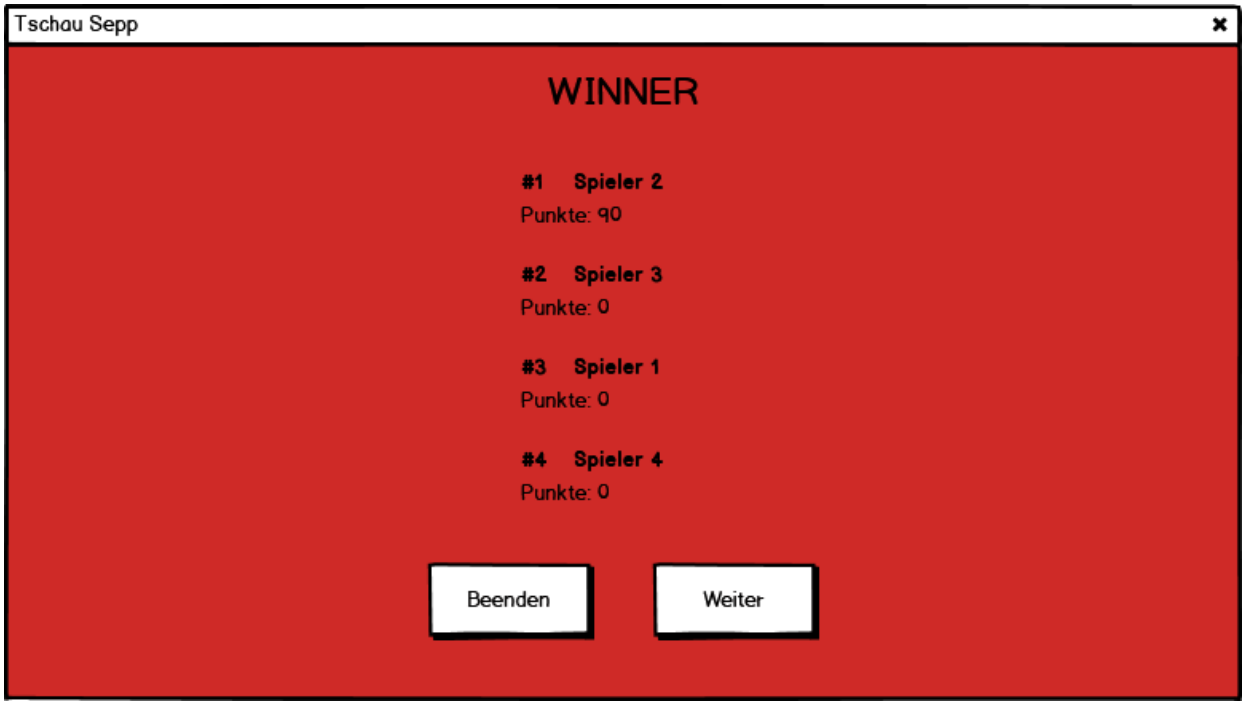


Abbildung 22: Mockup Rangliste

Dies ist die Maske, welche nach jeder Runde, bis zum Ende des Spiels erscheint und die Spieler nach Anzahl Punkten sortiert anzeigt. Wurde das Maximum an möglichen Punkten noch nicht erreicht, so hat der Benutzer die Wahl das Spiel zu beenden oder eine nächste Runde zu starten.

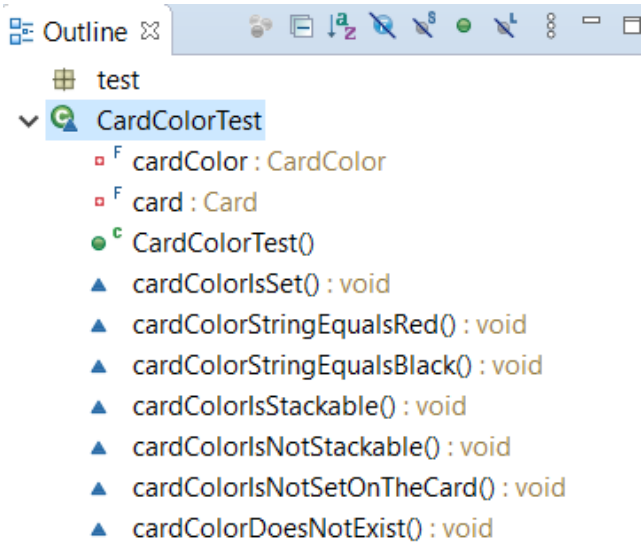
Komponente:

- GridLayout
- BorderLayout
- JButton
- JLabel

TESTFÄLLE

JUNIT-TESTFÄLLE

CARDCOLOR



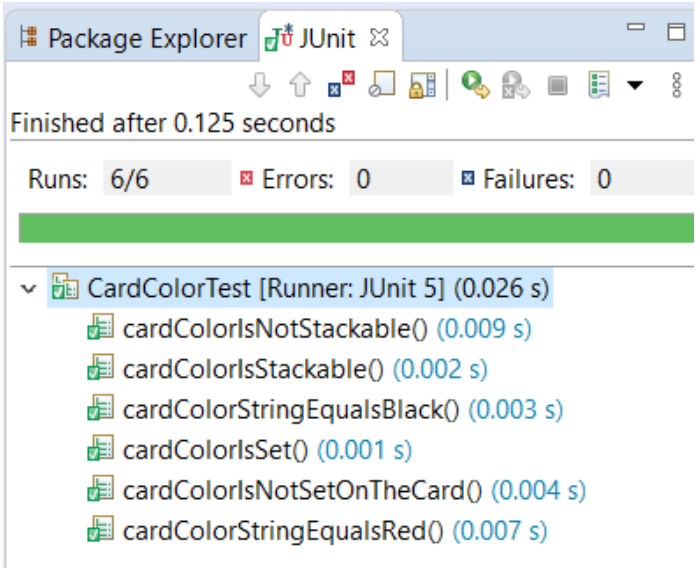


Abbildung 23: JunitTest

..\workspace\M326-TschauSepp\src\test\CardColorTest.java

Leider wurde ich nicht mit dem Programm fertig (GUIs fehlen und die Klasse CardValue muss noch überarbeitet werden). Zeitlich habe ich es nicht geschafft mehr JUnits zu erstellen, da ich zu viel Zeit brauchte, um die Modelklassen und einige Views zu erstellen. Jedoch wird alles noch nachträglich miteingereicht.

MANUELLE TESTFÄLLE

Testprotokoll "Tschau Sepp"

Programm	Tschau Sepp		
Tester	Mia Gudelj	**Version**	1.1
Datum	04.07.2020	**Zeit**	23.15 Uhr

	Nr.	Eingabe	Zustand	Tatsächliches Resultat	Meldung	Bewertung	Bemerkung
Programmstart, Startseite	1	Programmstart	-	Programm startet nicht			Habe ich Jar falsch exportiert?
	2.1	Spieler drückt "Spielstart"	Main Menu				
	2.2	Spieler geht von Name-eingeben zurück	Name-eingeben-screen				
	3.1	Spieler drückt "Anleitung"	Main Menu				
	3.2	Spieler schliesst Anleitung-popup	Anleitung-Popup				

Namen eingeben	4	Beim Klicken auf "zurück" wird wieder die Startseite angezeigt	Spielername-Maske					
	4.1	Im Name wurden Sonderzeichen	Spielername-Maske					
	4.2	Name überschreitet maximale Anzahl erlaubter Zeichen	Spielername-Maske					
	4.3	Name existiert bereits	Spielername-Maske					
	5	Namen eingeben	Spielername-Maske					
	6	Beim Klicken auf "Spiel starten" wird das Spiel gestartet	Spielername-Maske, alle Namen wurden korrekt eingegeben					

Abbildung 24: Manuelle Testfälle Teil 1

UNO	7	7 Karten austeilen	Spiel gestartet						
	8.1	Karte legen (Karte hat gleiche	Spieler n ist dran						
	8.2	Karte legen (Karte hat die gleiche Farbe, andere Nummer)	Spieler n ist dran						
	8.3	Karte legen (Karte hat die gleiche Nummer, andere Farbe)	Spieler n ist dran						
	9.1	Karte legen (Karte hat grössere Nummer, anderes Symbol und andere Farbe)	Spieler n ist dran						
	9.2	Karte hat kleinere Nummer und andere Farbe	Spieler n ist dran						
	10.1	Tschau drücken wenn noch 3 Karten vorhanden sind und **keine** Karte von denen gelegt werden kann	Spieler n ist dran						
	10.2	Sepp drücken wenn noch 2 Karten vorhanden sind und **keine** Karte von beiden gelegt werden kann	Spieler n ist dran						
	10.3	Tschau drücken wenn noch 3 Karten vorhanden sind und **eine** Karte von denen gelegt werden kann	Spieler n ist dran						
	10.4	Sepp drücken wenn noch 2 Karten vorhanden sind und **eine** Karte von beiden gelegt werden kann	Spieler n ist dran						
	10.5	Sepp nicht drücken und einfach eine Karte legen wenn noch **eine** Karte vorhanden sein wird	Spieler n ist dran						
	11.1	Karte nehmen wenn keine ausspielbar	Spieler n ist dran						
	11.2	Karte nehmen wenn eine ausspielbare Karte vorhanden	Spieler n ist dran						
Rangliste	12	Rangliste nach Anzahl Karten, die übriggeblieben sind	Ende --> Spieler hat keine Karten mehr						
	13	Wieder Neustart	Ende						

Fazit Die Abnahme wird nicht freigegeben. Das Programm ist noch nicht fertiggestellt. Ausserdem startet das Programm nicht über das jar-file. Führt man den Code jedoch aus, können viele Testfälle hier positiv abgedeckt werden.

Abbildung 25: Manuelle Testfälle Teil 2

LÖSUNGSBESCHREIBUNG

Nun ist es soweit. Die Abgabe meines Programms. Im Großen und Ganzen bin ich sehr zufrieden mit meinem Endprodukt. Die Views habe ich sehr genau getroffen, was mich sehr freut. Auch die Logik des Spiels (Spielregeln) konnte ich erfolgreich implementieren, sodass das Spiel den Spielregeln entsprechend gespielt werden kann. Vom Starten des Programms bis hin zum Legen der letzten Karte und erscheinen der Rangliste. Jedoch gibt es einige Fehler bzw. Funktionen, die ich nicht mehr geschafft habe, umzusetzen.

Das erste Fehlende Stück ist bereits bei der Maske, wo man die Spielernamen erfasst. Dort habe ich noch keine Validierung der unerlaubten Zeichen als auch redundanten Namen gemacht. Das mit dem Tooltip hatte leider nicht so funktioniert, wie ich es mir vorgestellt hatte. Dies werde ich jedoch anders lösen in Form von *JLabels*. Dazu muss ich dann einiges am Code umstrukturieren.
Was ich aber implementiert habe, ist, dass solange nicht alles Textfelder mindestens ein Zeichen enthalten oder nur aus Leerzeichen besteht, bleibt der Start-Knopf, um das Spiel zu starten, disabled. Also nicht anwählbar.

Das nächste, was nicht mit drin ist, war die Scroll-Bar im Deckbereich (da, wo die Karten des Spielers ersichtlich sind). Jedoch habe ich eine bessere Methode gefunden, welche auch übersichtlicher ist für den Anwender. So muss dieser nicht ständig rumscrollen und nach Karten suchen, sondern hat alle im Blickfeld.

Was noch nicht funktioniert, ist die Logik der Ausgabe der Rangplatzierung. Dafür hatte ich zu wenig Zeit, da die Logik und Anzeige des Spieles sehr zeitintensiv waren. Jedoch werde ich auch dies weiter ausarbeiten. Es fehlt nur ein *Bubble-Sort*, doch ich bekam damit *NullPointerExeptions* und Zeitmangel hatte ich auch.

Ansonsten entspricht das Programm den Spielregeln, als auch den Vorgaben.

Spezialregeln und Karten wurden hier nicht miteinbezogen bzw. nicht berücksichtigt.

FUNKTIONALE ANFORDERUNGEN

- ✓ Möglichkeit für den Spieler haben, die Spielregen jederzeit nachzulesen
- ✓ Spielername setzen
- ✓ Spiel starten
- ✓ Spiel beenden
- ✓ «Tschau Sepp» spielen
  - ✓ Karten mischen
  - ✓ jedem Spieler 7 Karten verteilen
  - ✓ Karten legen
  - ✓ Karten ziehen
  - ✓ zum nächsten Spieler wechseln, sobald der eine gespielt hat
  - ✓ «tschau» rufen
  - ✓ «sepp» rufen
- ✓ Den Spielregeln entsprechende Züge erlauben
- ✗ Punkte am Ende zusammenzählen
- ✗ Rangliste ausgeben

NICHT FUNKTIONALE ANFORDERUNGEN

- ✓ muss auf allen Betriebssystemen und Geräten mit Java lauffähig sein.
- ✓ Darstellung/Programm muss für den Benutzer übersichtlich sein. Das heisst, der Benutzer wird nicht von irrelevantem Content abgelenkt oder verwirrt.
- ✓ Selbstbeschreibungsfähig

Anhand dem, bin ich sehr zufrieden damit, dass ich ca. 88% der Anforderungen erfüllen konnte. Die letzten 22% werden bis zum Wettbewerb ebenso erfüllt werden.

ANHANG

TESTPROTOKOLL

Testfälle:

Anhang\Testing\Testfälle_TschauSepp_MiaGudelj.pdf

TESTPROTOKOLL

Anhang\Testing\Testprotokoll_TschauSepp_MiaGudelj.pdf

API-DOKUMENTATION

Anhang\M326-TschauSepp\doc\allclasses-index.html

BENUTZERHANDBUCH

Anhang\Handbuch\Tschau-Sepp_Handbuch.pdf

PROGRAMM INKL. SOURCE CODE

PROGRAMM

Anhang\TschauSepp_MiaGudelj.jar

Muss noch überarbeitet werden

CODE

Anhang\M326-TschauSepp\src

Muss noch überarbeitet werden

DEKLARATION EIGENLEISTUNG

Anhang\Deklaration Eigenleistung M326 Mia Gudelj.pdf

MOCKUPS

Anhang\MockUps

KLASSENDIAGRAMM

Anhang\UML\Klassendiagramme\Klassendiagramm.jpg

Muss noch überarbeitet werden

SEQUENZDIAGRAMME

Anhang\UML\Sequenzdiagramm

ZUSTANDSDIAGRAMME

Anhang\UML\Zustandsdiagramm\Zustandsdiagramm.jpg

AKTIVITÄTSDIAGRAMME

Anhang\UML\Aktivitätsdiagramm\Aktivitätsdiagramm.jpg

ANWENDUNGSFALLDIAGRAMME

Anhang\UML\Anwendungsfalldiagramm