

# PROJEKTDOKUMENTATION

## Release III

UNIVERSITÄT KASSEL  
SOFTWARE ENGINEERING I SS 19

**Projekt: Rundenbasiertes Strategiespiel  
RBSG - Enhanced Wars**

GRUPPE G



EnGineerinG  
<Problem/>

SCRUM MASTER: JAN MÜLLER  
PRODUCT OWNER: KEANU STÜCKRAD

8.7.2019 - 4.8.2019

## Vorwort

Das Release III von Gruppe G des Projekts RBSG, ein rundenbasiertes Strategiespiel, soll im Rahmen des Moduls „Software Engineering I“ im Fachbereich „Software Engineering“ im Sommersemester 2019 mit dieser Dokumentation festgehalten werden.

Das Vorgehen in diesem Modul ist so gegliedert, dass es vier Releases mit jeweils zwei Sprints über vier Monate gibt. Das Projekt begann am 13.5.2019 und wird bis zum 1.9.2019 laufen. Ein Client für einen Klon vom Klassiker „Advanced Wars“ soll entwickelt werden. Dabei ist das Team über die vier Releases immer in einen Scrum Master, einen Product Owner und mehrere Developer eingeteilt.

Die Dokumentation wird eine Einleitung in das Release, danach die Zusammenfassung der beiden Sprints und eine abschließende Releaseanalyse enthalten. In der Einleitung sollen die Anforderungen des Kunden, der letzte Standpunkt und die neuen MockUps erläutert werden. In den Sprintdokumentationen wird jeweils kurz das Ziel dargelegt und darauf sollen alle User Stories, Tasks oder Bugs aufgelistet werden. Am Ende wird es immer eine Zeitübersicht mit einer Analyse des Sprints geben. Das letzte Kapitel mit der Releaseanalyse wird aufzeigen, ob das Resultat den Anforderungen entspricht und ob das Team effektiv arbeitete.

## Inhaltsverzeichnis

<b>1 Vorwort</b>	<b>1</b>
<b>2 Release III</b>	<b>4</b>
2.1 Releaseanforderungen . . . . .	4
2.2 Standpunkt des letzten Releases . . . . .	5
2.2.1 Login . . . . .	5
2.2.2 Lobby . . . . .	7
2.2.3 Army Manager . . . . .	8
2.2.4 Waiting Room (Gamelobby) . . . . .	10
2.2.5 Ingame . . . . .	12
2.2.6 Weitere Features . . . . .	14
2.3 MockUps . . . . .	15
2.3.1 Lobby . . . . .	15
2.3.2 Waiting Room (Gamelobby) . . . . .	15
2.3.3 Ingame . . . . .	17
2.3.3.1 Einheit auswählen . . . . .	18
2.3.3.2 Einheit bewegen . . . . .	20
2.3.3.3 Einheit angreifen . . . . .	21
2.3.3.4 Spielende . . . . .	22
2.3.3.5 Weitere Features . . . . .	23
2.4 Domain Stories . . . . .	24
2.4.1 Spielstart . . . . .	24
2.4.2 Bewegung . . . . .	26
<b>3 Sprint V</b>	<b>29</b>
3.1 Sprintziel . . . . .	29
3.2 User Stories . . . . .	29
3.2.1 Waiting Room - Spielbeitritt anpassen . . . . .	29
3.2.2 Ingame - Spielfeld . . . . .	30
3.2.3 Ingame - Phase beenden . . . . .	30
3.2.4 Ingame - Einheit auswählen . . . . .	30
3.2.5 Ingame - Einheit bewegen . . . . .	31
3.2.6 Ingame - Einheit angreifen . . . . .	32
3.2.7 Ingame - Spielstatus anzeigen . . . . .	32
3.2.8 Ingame - Minikarte anzeigen . . . . .	33
3.2.9 Ingame - Chatintegration . . . . .	33
3.2.10 Ingame - Game Over . . . . .	34
3.2.11 Ingame - Game Won . . . . .	34
3.2.12 Ingame - Spectator Over . . . . .	34
3.2.13 Lobby - Spectating . . . . .	35
3.2.14 Ingame - Spectating . . . . .	35
3.2.15 Ingame - Einheiteninformationen . . . . .	35
3.2.16 Ingame - Musiksteuerung . . . . .	36
3.3 Tasks . . . . .	36
3.3.1 Servernachrichten . . . . .	36
3.3.2 Einheiteninformationen im Datenmodell . . . . .	36

3.4 Zeitübersicht . . . . .	37
3.5 Analyse . . . . .	37
3.5.1 Burndown . . . . .	37
3.5.2 Abgeschlossene Stories . . . . .	38
3.5.2.1 Waiting Room - Spielbeitritt anpassen . . . . .	38
3.5.2.2 Ingame - Spielfeld . . . . .	38
3.5.2.3 Ingame - Phase beenden . . . . .	39
3.5.2.4 Ingame - Spielstatus anzeigen . . . . .	39
3.5.3 Angefangene Stories . . . . .	39
3.5.3.1 Ingame - Einheit auswählen . . . . .	39
3.5.3.2 Ingame - Einheit bewegen . . . . .	39
3.5.3.3 Ingame - Minikarte anzeigen . . . . .	39
3.5.4 Nicht abgeschlossene Stories/Tasks . . . . .	39
3.5.5 Fazit . . . . .	40
<b>4 Sprint VI</b>	<b>41</b>
4.1 Sprintziel . . . . .	41
4.2 Geänderte User Stories vom Sprint V . . . . .	41
4.2.1 Ingame - Minikarte anzeigen . . . . .	41
4.2.2 Ingame - Einheiteninformationen . . . . .	42
4.3 Übernommene User Stories aus Sprint V . . . . .	42
4.4 User Stories . . . . .	43
4.4.1 Ingame - Einheit bewegen nach Bewegung . . . . .	43
4.5 Geänderte Tasks vom Sprint V . . . . .	43
4.5.1 Einheiteninformationen im Datenmodell . . . . .	43
4.6 Übernommene Tasks aus Sprint V . . . . .	43
4.7 Tasks . . . . .	43
4.7.1 Ingame - Bestützungsbox in der Sidebar . . . . .	44
4.8 Bugs . . . . .	44
4.8.1 Terminierung sicherstellen . . . . .	44
4.8.2 Ingame - Einheit mehrmals auswählen . . . . .	44
4.8.3 Spectating - Automatischer Wechsel zum Spielfeld . . . . .	44
4.8.4 Lobby - Spielbeitritt blockieren . . . . .	44
4.9 Zeitübersicht . . . . .	45
4.10 Analyse . . . . .	45
4.10.1 Burndown . . . . .	46
4.10.2 Ausreißer . . . . .	46
4.10.3 Abgeschlossen . . . . .	46
4.10.4 Angefangen . . . . .	46
4.10.5 Nicht abgeschlossen . . . . .	46
4.10.6 Entfernt . . . . .	46
4.10.7 Fazit . . . . .	46
<b>5 Abschluss Release III</b>	<b>47</b>
5.1 Neue MockUps im Verlauf des Releases III . . . . .	47
5.2 Vergleich MockUps mit aktueller Implementation . . . . .	47
<b>6 Quellenangaben</b>	<b>48</b>

## Release III

Das Release III erstreckte sich über die vier Wochen vom 8.7.2019 bis zum 4.8.2019. Die zwei Sprints gingen vom 8.7.2019 bis zum 21.7.2019 und vom 22.7.2019 bis zum 4.8.2019.

Das Team war in diesem Release wie gefolgt aufgeteilt:

- Scrum Master:  Jan Müller
- Product Owner:  Keanu Stückrad
- Developer:
  1.  Georg Siebert
  2.  Juri Lozowo
  3.  Omar Sood
  4.  Tobias Klipp

## Releaseanforderungen

Folgende Mindestanforderungen wurden vereinbart:

1. Client: Waiting Room (Gamelobby)
  - Chatfunktion
  - Armeeauswahl
  - Senden eines Bereit-Signals
2. Client: Ingame
  - Bewegen und Angreifen von Einheiten
  - Minikarte
  - Chatfunktion
  - Anzeige aller Spieler
  - Anzeige der aktuellen Runde und Phase
  - Anzeige des Gewinners
  - Spiele verlassen
  - Beobachtermodus
3. Client: Lobby
  - Spielbeitritt als Beobachter
4. Qualitätssicherung
  - C0 Testabdeckung von 75%

## Standpunkt des letzten Releases

Das Release II erstreckte sich vom 10.6.2019 bis zum 7.7.2019. Nach dem 7.7. waren folgende Teile des Clients fertig gestellt:

- Login
  - \* Anmelden
  - \* Registrieren
- Lobby
  - \* Spiel erstellen/beitreten
  - \* Anzeige der angemeldeten Spieler
  - \* Anzeige der aktiven Spiele
  - \* Chatfunktion
  - \* Ausloggen
- Army Manager
  - \* Konfigurieren eigener Armeen
  - \* Speichern von Armeen lokal und auf dem Server
- Waiting Room (Gamelobby)
  - \* Chatfunktion
  - \* Anzeige der beigetretenen Spieler
  - \* Zurück zur Lobby
- Ingame
  - \* Anzeige des initialen Spielgeschehens
  - \* Zurück zur Lobby

Weiterhin wurde bereits eine C0 Testabdeckung von mindestens 60% erreicht.

## Login

Im Login konnte der Nutzer sich anmelden oder registrieren (siehe Abbildung 1).

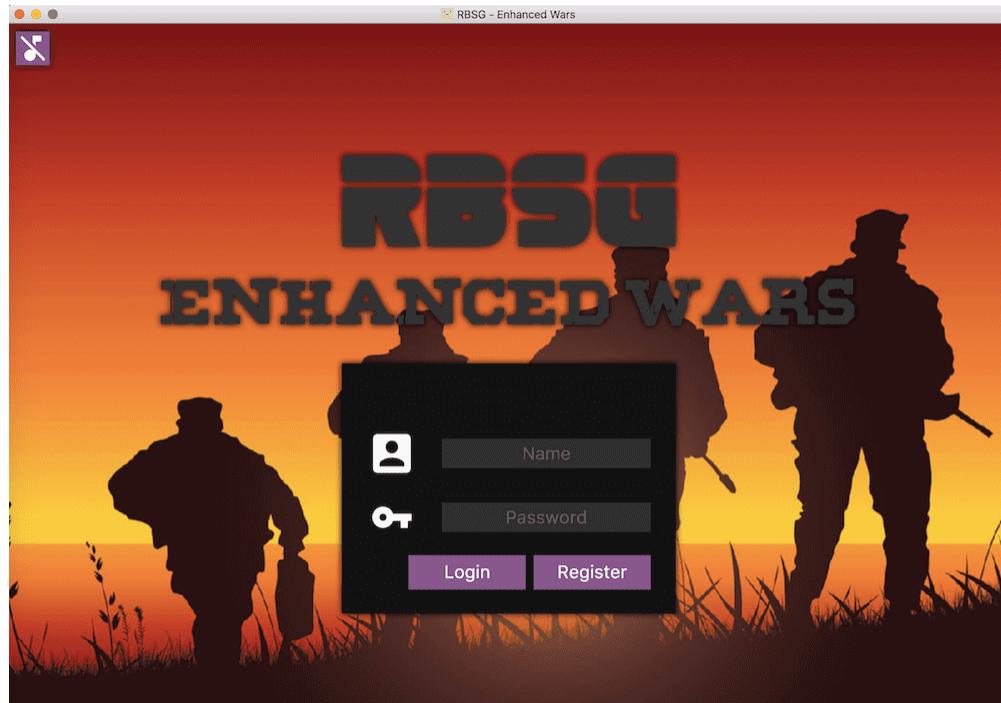


Abbildung 1: Release II: Login Szene

Versuchte der Nutzer sich anzumelden oder zu registrieren, erschien ein Lade-Indikator (siehe Abbildung 2) und die Buttons und Textfelder wurden disabled. Machte der Nutzer dabei einen Fehler oder konnte keine Verbindung zum Server hergestellt werden, erschien eine Fehlermeldung (siehe Abbildung 2). Der Login war internationalisiert und der Untertitel „Enhanced Wars“ war animiert. Weiterhin konnte der Nutzer die Musik über den Musik Button in der Ecke ein- und ausschalten.

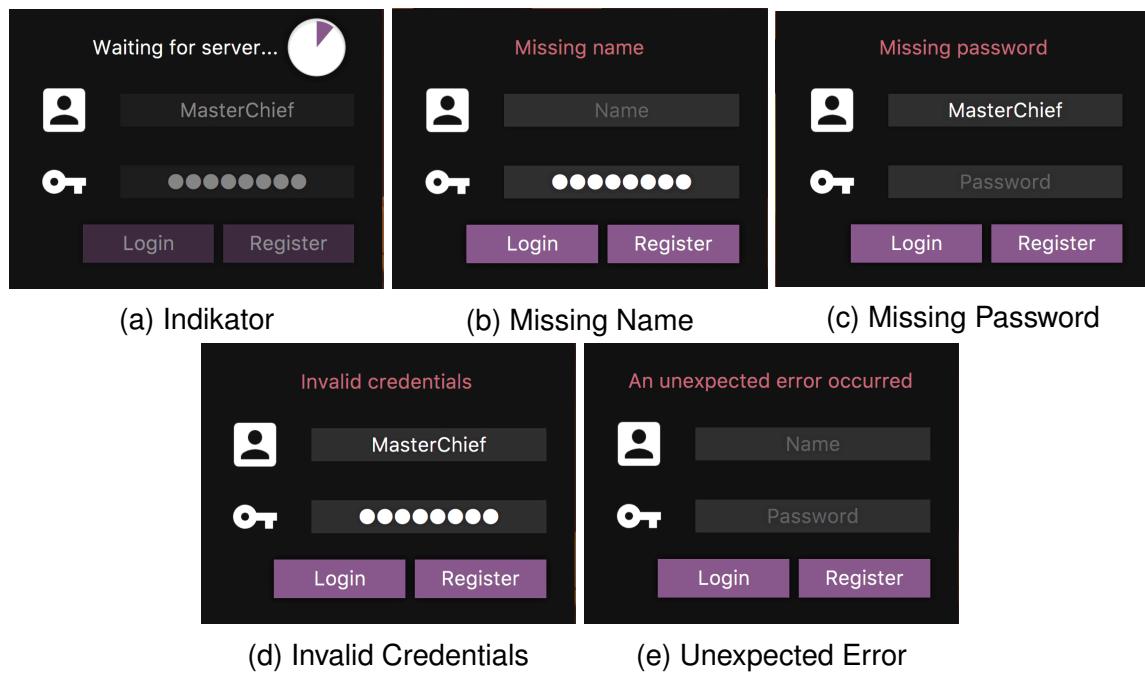


Abbildung 2: Release II: Login Formular

## Lobby

Die Lobby zeigte eine Liste von allen Spielern (die nicht im Spiel waren) und allen Spielen an. Diese aktualisierten sich, wenn der Server die entsprechende Nachricht schickte. Der Nutzer konnte sich über die Buttons oben rechts ausloggen, die Sprache ändern oder die Musik an- und ausschalten. Der Chat wurde unten links angezeigt. Rechts sah der Nutzer eine Liste mit allen vollständigen Armeen. Hatte der Nutzer keine vollständige Armee ausgewählt, konnte er kein Spiel erstellen (siehe Abbildung 3) und auch keinem Spiel beitreten.

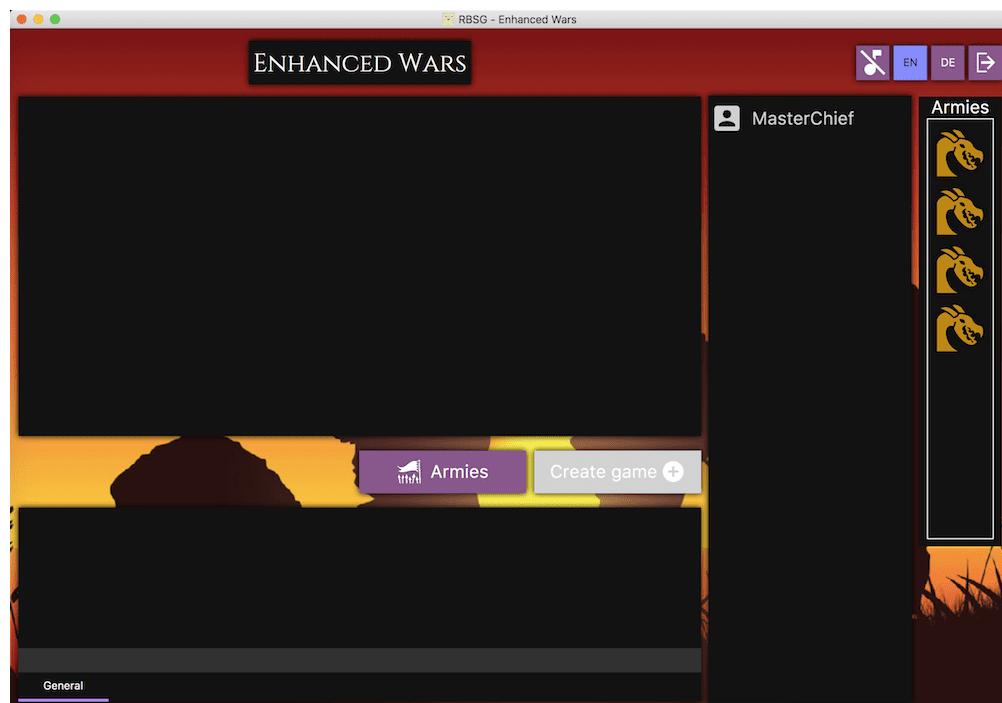


Abbildung 3: Release II: Lobby Szene: Keine Armee ausgewählt

Hatte der Nutzer aber eine Armee ausgewählt, konnte er ein Spiel erstellen oder einem Spiel beitreten (siehe Abbildung 4 oder 6). Spiel erstellen funktionierte über den Create Game Button (siehe Abbildung 5). Der Nutzer hatte die Auswahl zwischen einem 2-Spieler-Spiel und einem 4-Spieler-Spiel. Danach wurde ein Autojoin gerufen und der Nutzer trat dem Spiel automatisch bei. Die Lobby war auch internationalisiert (vgl. Abbildung 5 mit 6).

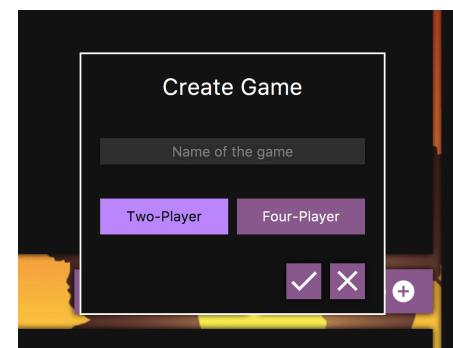


Abbildung 4: Release II: Create Game Formular

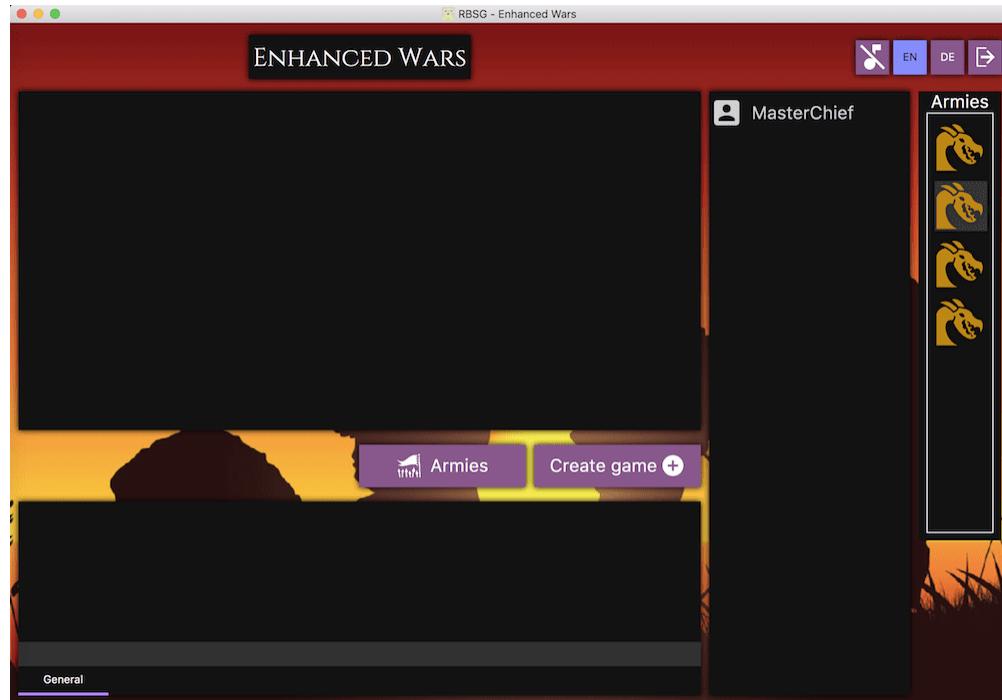


Abbildung 5: Release II: Lobby Szene: Armee ausgewählt

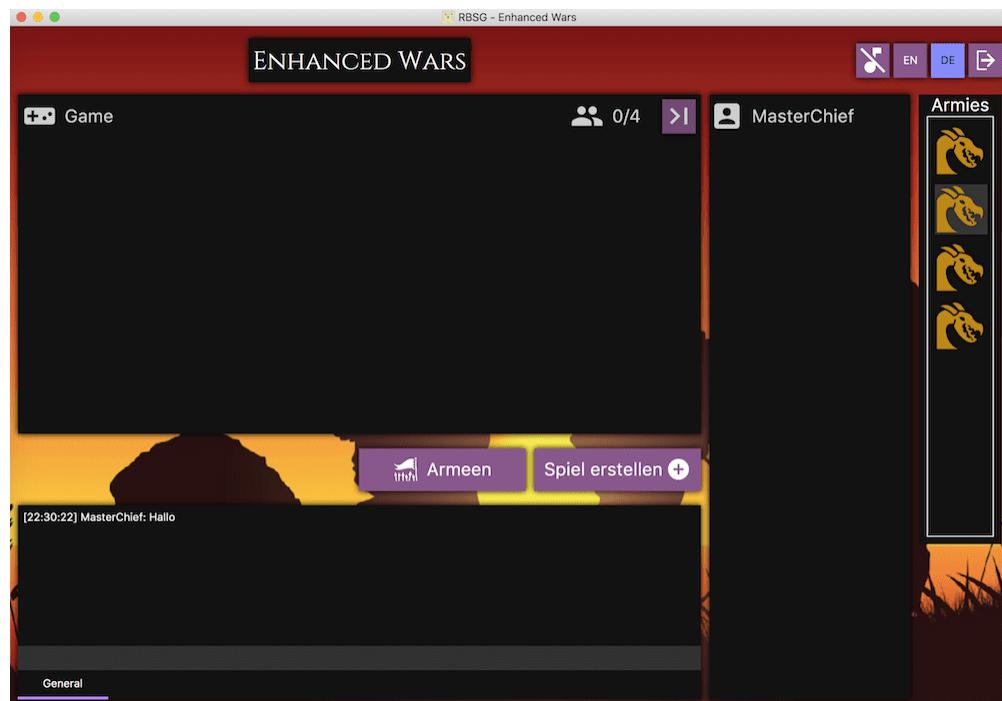


Abbildung 6: Release II: Lobby Szene: Sprache DE

### Army Manager

Im Army Manager gab es eine Liste der Einheiten, die der Nutzer wählen konnte, und eine Liste der Einheiten, die in der gerade rechts ausgewählten Armee waren. In der Mitte stand der Name der ausgewählten Armee, ein Plus (+) und ein Minus (-) Button, um Einheiten der Armee hinzuzufügen oder um sie zu löschen und ein Zähler, der anzeigte,

wie viele Einheiten bereits in der Armee waren. Neben der Einheitenliste war eine Vorschau der gerade ausgewählten Einheit. Neben diesem Feld wurden die Properties der Einheit angezeigt und welche Einheiten die ausgewählte Einheit angreifen konnte. Hier stand bei keiner Auswahl in jedem Feld ein Fragezeichen (siehe Abbildung 7), ansonsten Zahlenwerte (siehe Abbildung 8).

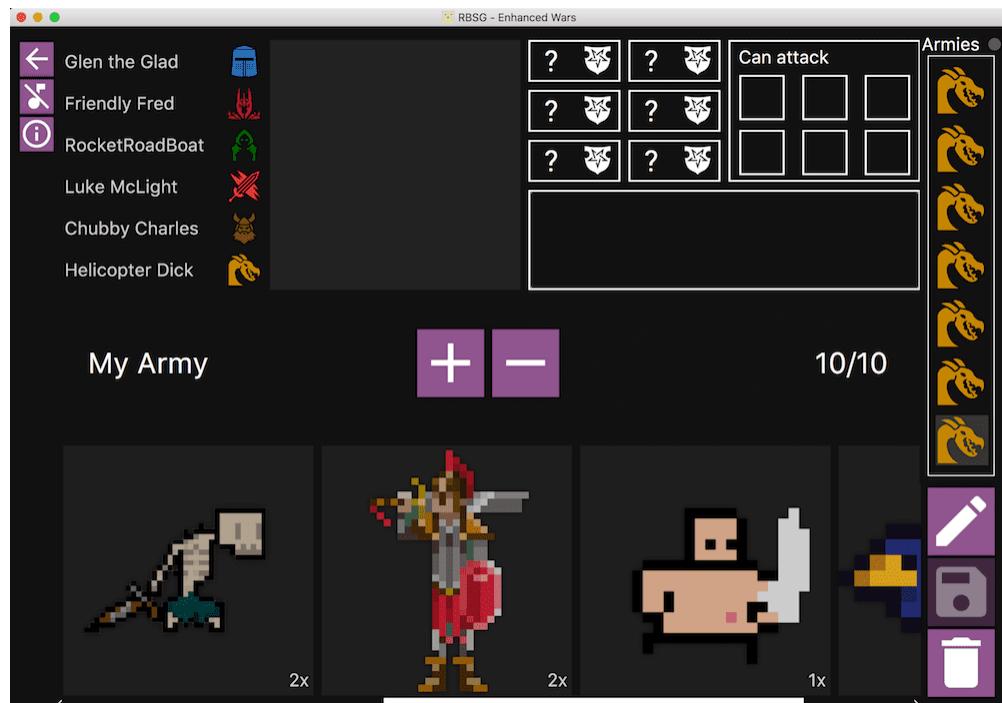


Abbildung 7: Release II: Army Manager Szene: Keine Einheit ausgewählt

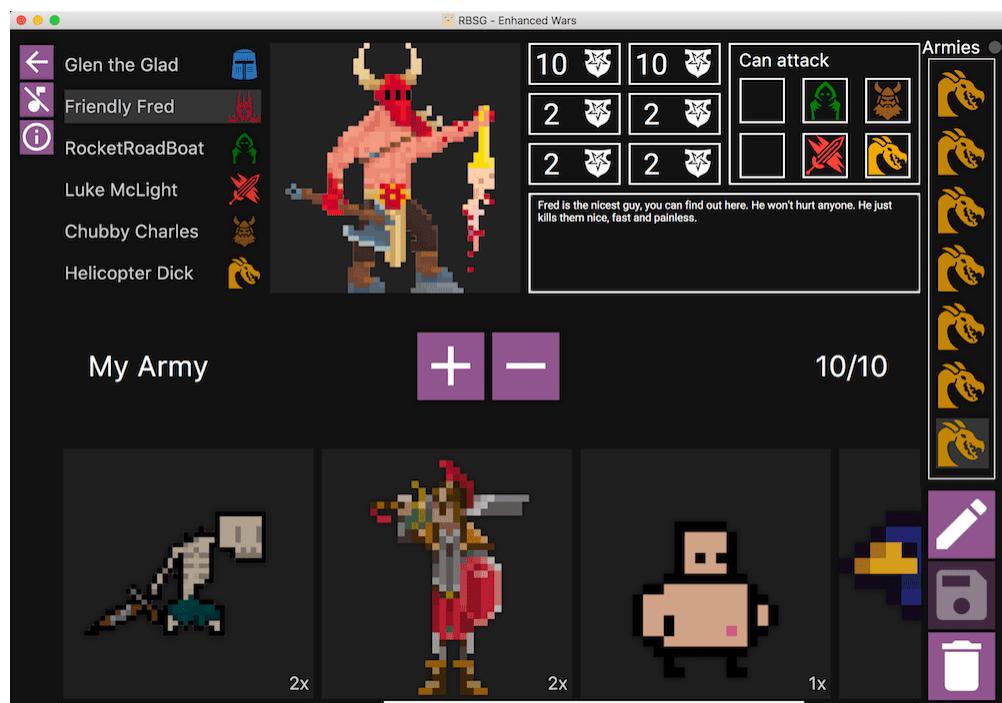


Abbildung 8: Release II: Army Manager Szene: Einheit ausgewählt

Darunter war der Lore Text der ausgewählten Einheit zu sehen. Oben rechts waren Buttons, um zurück zur Lobby zu gehen, um die Musik ein- oder auszuschalten und ein Button, der die Properties in einem Overlay erklärte (siehe Abbildung 9). Unten rechts befanden sich Buttons, um eine Armee komplett zu löschen,



Abbildung 9: Release II: Property Info

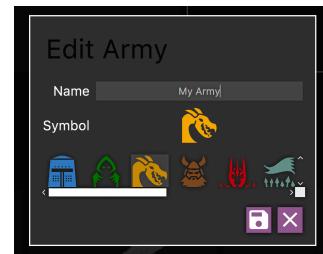


Abbildung 10: Release II: Armee bearbeiten

um sie zu speichern (wurde disabled, wenn sich nichts änderte (siehe Abbildung 7). Wenn der Nutzer aber etwas änderte wurde der Button wieder enabled (siehe Abbildung 11)). Es gab dort in der Ecke einen weiteren Button, um den Armeenamen zu ändern oder das Icon zu wechseln. Dazu öffnet sich ein weiteres Overlay (siehe Abbildung 10).

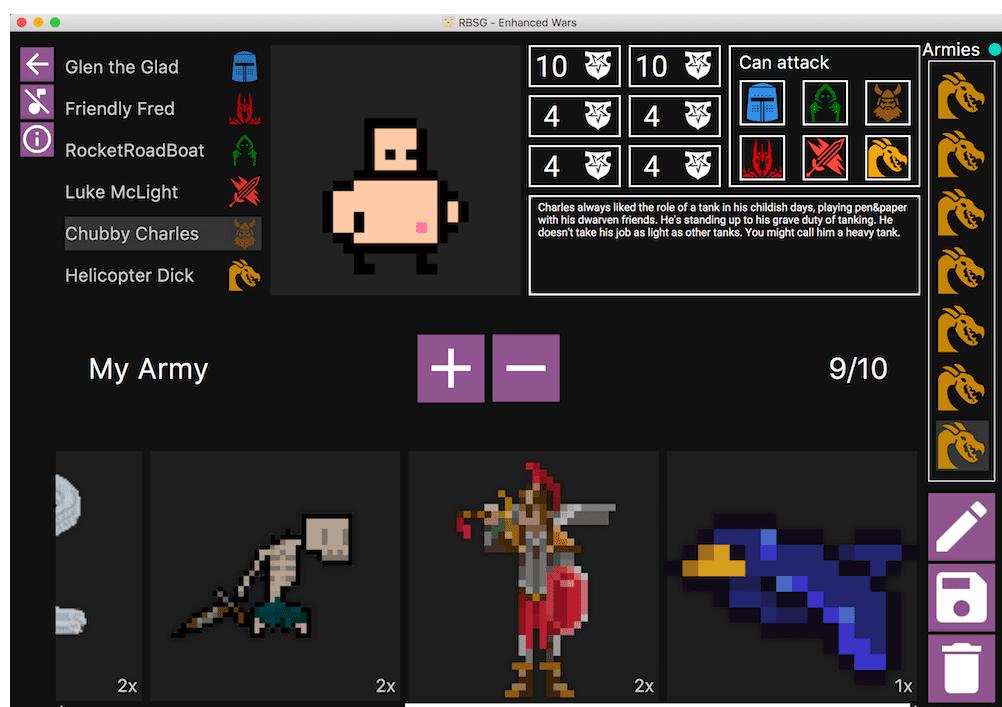


Abbildung 11: Release II: Army Manager Szene: Speicherbar

### Waiting Room (Gamelobby)

Trat der Nutzer dem Warteraum bei, dann sah er oben rechts verschiedene Buttons. Einmal einen Button zum Verlassen des Spiels in die Lobby, einen Button, um die Musik ein-

und auszuschalten, und einen Button, um Informationen anzuzeigen. Da dieser Button noch keine spezielle Funktion hatte, wechselte der Nutzer mit diesem testweise auf das Spielfeld (Ingame). Oben in der Mitte wurde die Kartenvorschau angezeigt.

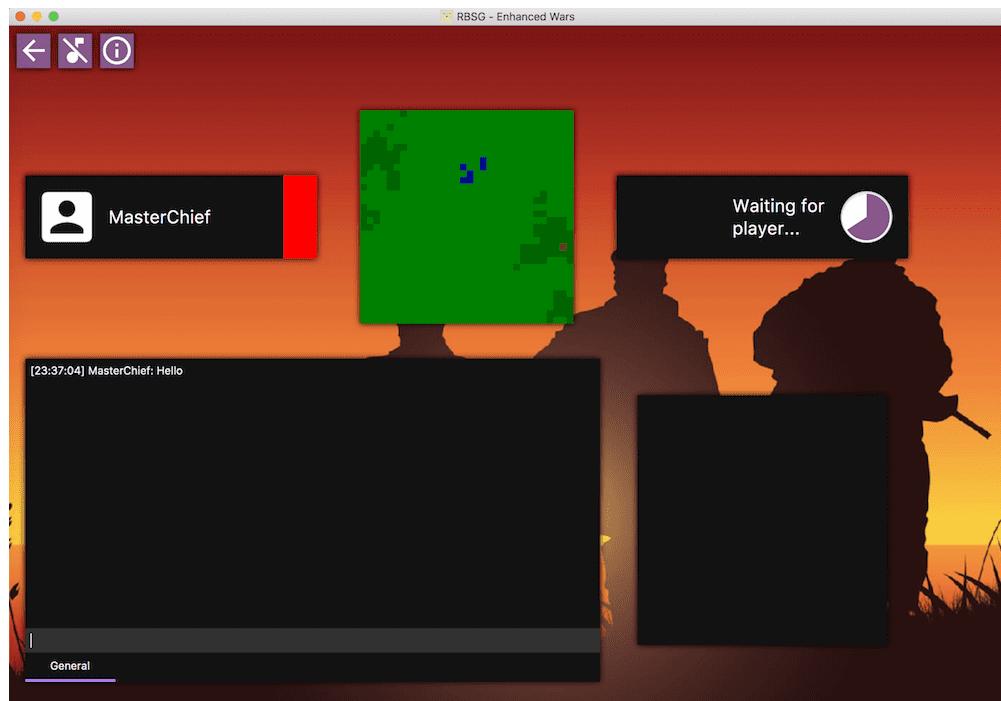


Abbildung 12: Release II: Waiting Room Szene: 2 Spieler

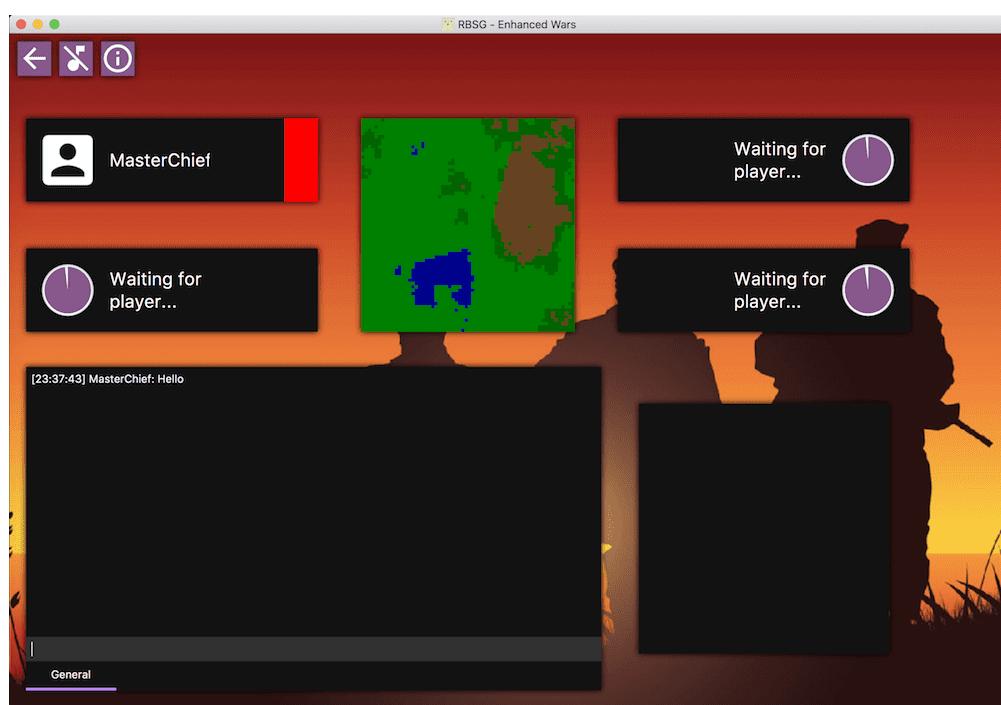


Abbildung 13: Release II: Waiting Room Szene: 4 Spieler

Links und rechts daneben waren die Spielerkarten zu sehen. Je nachdem ob der Nutzer

ein 2-Spieler-Spiel oder ein 4-Spieler-Spiel startete, befanden sich dort zwei oder vier Spielerkarten (siehe Abbildung 12 und 13). Auf den Spielerkarten fanden sich der Name und die Farbe des jeweiligen Spielers vor. Waren noch nicht genug Spieler dem Spiel beigetreten, wurden Lade-Indikatoren in den Spielerkarten angezeigt. Unten war der Chat und daneben eine freie Pane für ein Minigame, dass noch implementiert werden sollte.

### Ingame

Die Ingameszene zeigte bisher nur die Karte und das initiale Spielgeschehen. Die initialen Einheiten wurden auch angezeigt. Weiterhin gab es einen Button, der zurück in die Lobby navigierte. Zwei weitere Buttons kontrollierten das Zoomen auf dem Spielfeld (siehe Abbildung 14, 15 und 16).



Abbildung 14: Release II: Ingame Szene

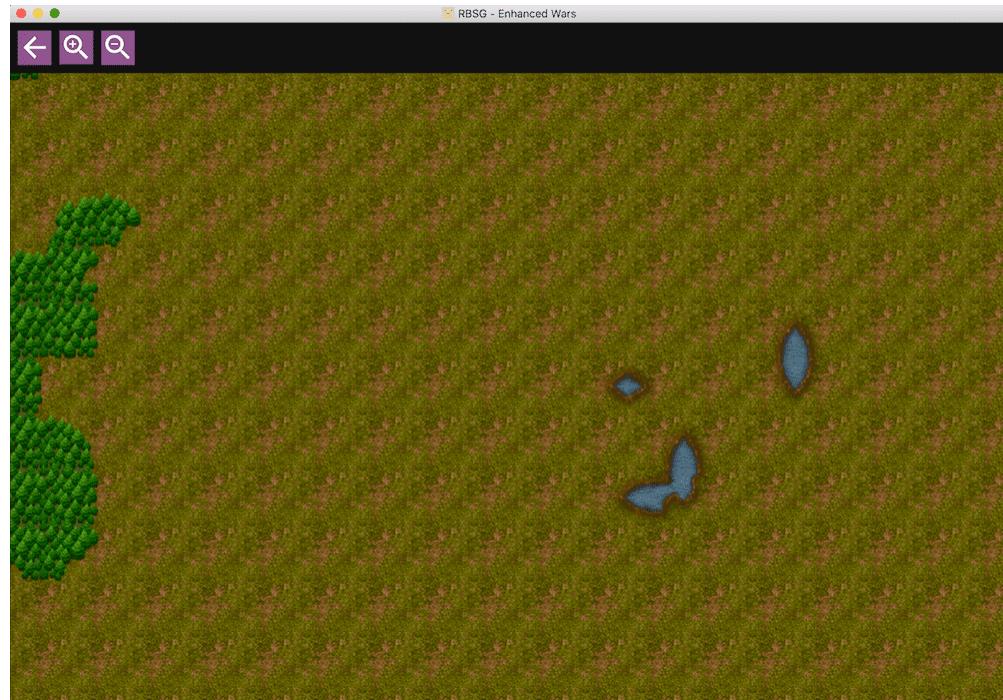


Abbildung 15: Release II: Ingame Szene: Zoom In

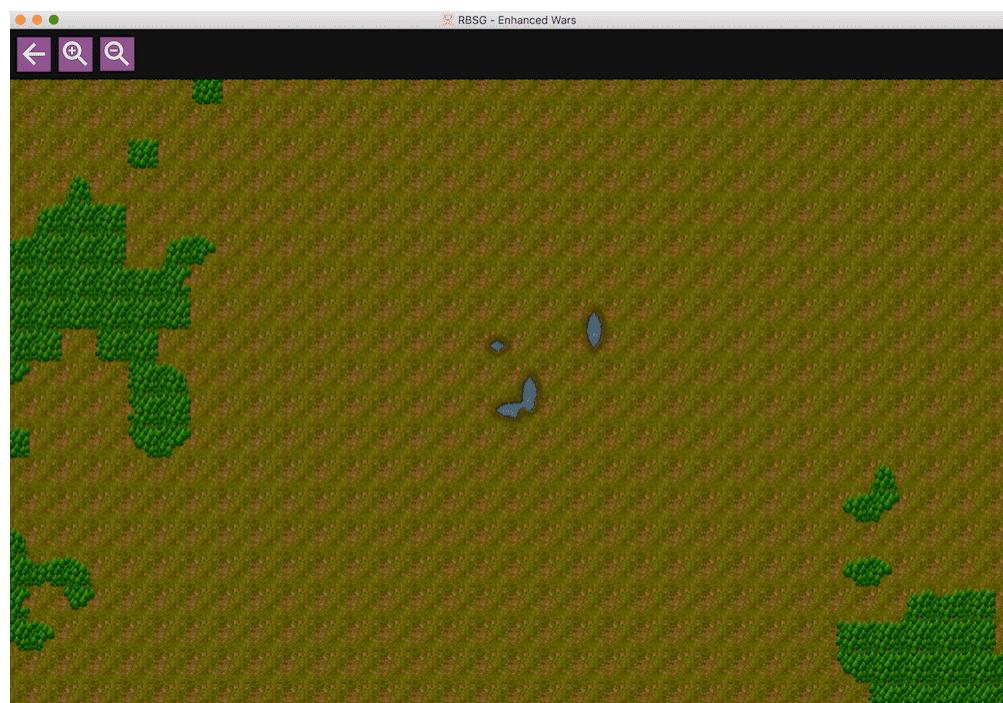


Abbildung 16: Release II: Ingame Szene: Zoom Out

## Weitere Features

Weiterhin wurde seit den ersten beiden Releases ein Dark Theme Styling implementiert, wie auf allen vorangegangenen Abbildungen zu sehen ist. Musik und Internationalisierung waren zwar optional, aber funktionierten in jeder Szene. Der Nutzer konnte auch nach Belieben Chuck Norris Zitate über den Chat-Befehl „\chuckMe“ im Chat versenden und es wurde auch mit einem Zeitstempel vor jeder Nachricht angezeigt, wann sie verschickt wurde. Wenn der Nutzer eine private Nachricht bekam, wurde dies auf dem neu aufgegängenem Tab mit der secondary Farbe signalisiert. Die Kartenvorschau im Warteraum war auch optional, genauso wie das Zoomen Ingame. Den Armeenamen, sowie das Icon dazu ändern zu können, war auch optional. Der Autojoin nach dem Create Game wurde auch nicht verlangt, funktioniert aber bestens. Ein weiteres sehr wichtiges Feature war, dass der Client den Spieler Fehlermeldungen (siehe Abbildung 17 und 18) schickte, wenn der Server einen WebSocket unerwartet schloßs. Der Nutzer wurde je nach Fehler in den Login oder die Lobby zurückgeschickt. Außerdem gab es ein Fenster, in dem sich der Nutzer noch einmal entscheiden konnte, ob er sich wirklich ausloggen oder ob er wirklich das Spiel verlassen möchte (siehe Abbildung 19 und 20).

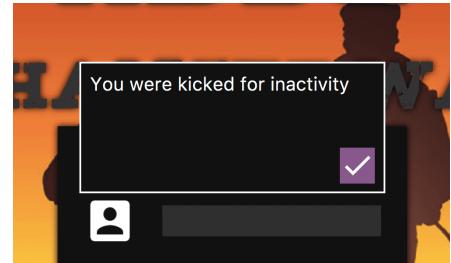


Abbildung 17: Release II: Login Fehler

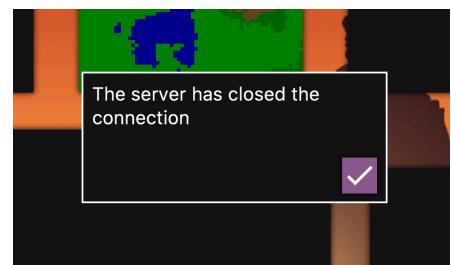


Abbildung 18: Release II: Spielfehler

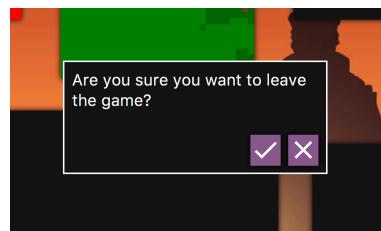


Abbildung 19: Release II: Spiel verlassen

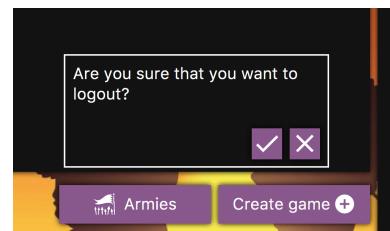


Abbildung 20: Release II: Logout

Die C0 Testabdeckung betrug zum Ende des Releases II 77%.

97% classes, 77% lines covered in package 'de.uniks.se19.team_g'			
Element	Class, %	Method, %	Line, %
project_rbsg	97% (380/389)	80% (1586/1967)	77% (7915/10223)

Abbildung 21: Release II: C0 Testabdeckung (Siehe Line, %)

## MockUps

Aufgrund der Releaseanforderungen wurden für das Kundentreffen am 1.7.2019 MockUps zu der neuen Ingame Szene erstellt. Weiterhin gab es auch noch MockUps zur Lobby und zum Waiting Room. Diese wurden nach dem Erscheinen der Serverdokumentation noch erweitert. Im sechsten Sprint wurden die MockUps auch noch einmal angepasst, da sich grundlegende Änderungen ergeben haben (Ein Kontextmenü wurde durch die Sidebar ersetzt. Die MockUps waren im Stil des Dark Theme und sollten an die UI aus den vorherigen Releases anknüpfen).

### Lobby

Es wurde gefordert, in der Lobby ein Button hinzuzufügen, der es dem Nutzer erlaubt einem Beobachtungsmodus beizutreten. Des Weiteren sollte die Armeeauswahl aus der Lobby entfernt werden (und in den Waiting Room verlegt werden).

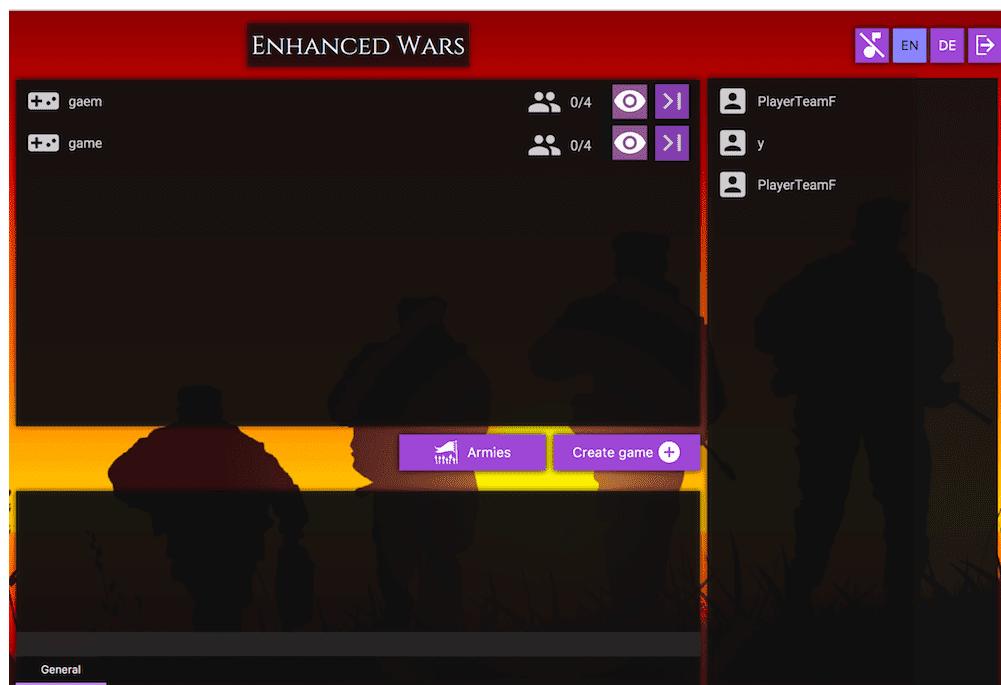


Abbildung 22: MockUp: Beobachtungsmodus Lobby

### Waiting Room (Gamelobby)

Da der Waiting Room im Release II (siehe Kapitel 2.2.4) schon größtenteils vorhanden war, musste nur noch das Ready geben hinzugefügt werden. Dies sollte bei uns passieren, wenn man eine Armee auswählt (siehe Abbildung 24). Die Armeeauswahl musste deswegen von der Lobby in den Waiting Room umgezogen werden. Spieler, die bereits bereit waren, sollten in ihrer Playercard in der secondary Farbe hinterlegt werden und die Icons sollten sich schwarz färben. Wenn alle Spieler das Ready gaben, wurde ein Overlay angezeigt, in dem ein Indikator lud, bis zur Ingame Szene gewechselt wurde (siehe Abbildung 25). Auf den MockUps waren auch zwei optionale Features zu sehen. Dies war einmal das TicTacToe Spiel unten rechts, das gegebenenfalls implementiert werden sollte. Und außerdem sollte oben in der Mitte der Name des Spiels angezeigt werden.

(siehe auf den MockUps mit Beispielnamen „Enhanced Wars“).

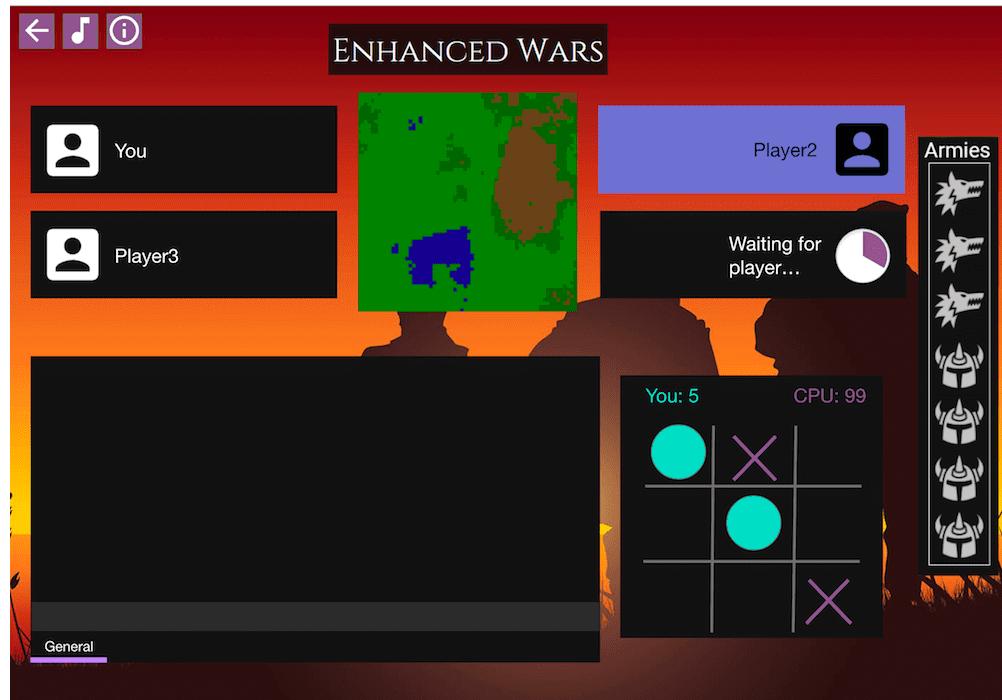


Abbildung 23: MockUp: Nutzer nicht bereit

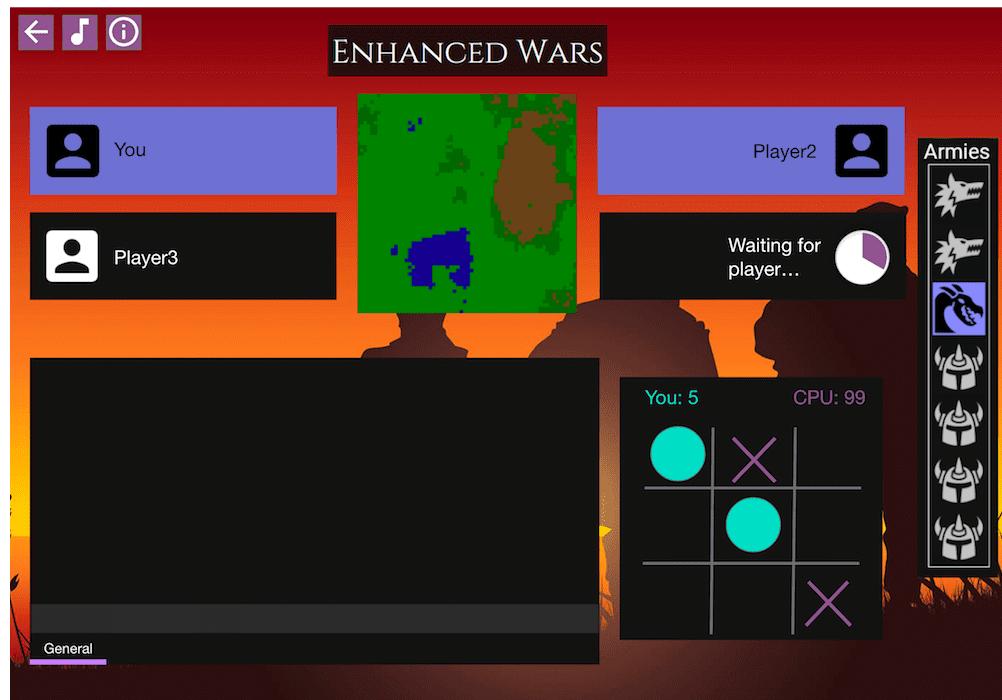


Abbildung 24: MockUp: Nutzer bereit

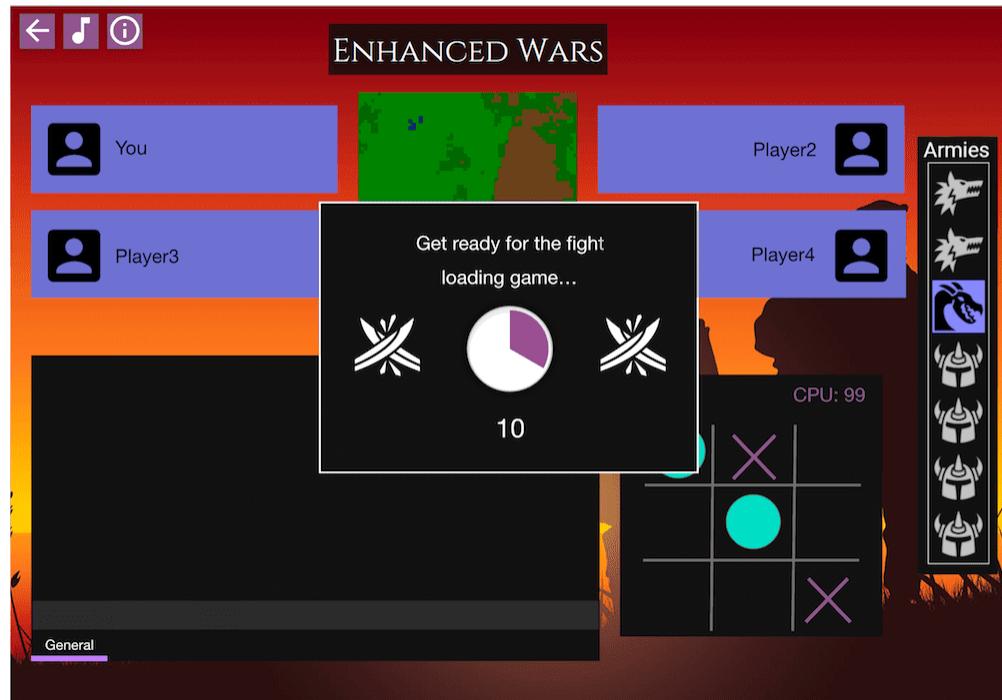


Abbildung 25: MockUp: Spielstart

### Ingame

Bisher zeigte die Ingame Szene nur das initiale Spielgeschehen. In diesem Release sollte diese Szene hauptsächlich erweitert werden. Chat, die Minikarte und die Spieler mussten angezeigt werden.

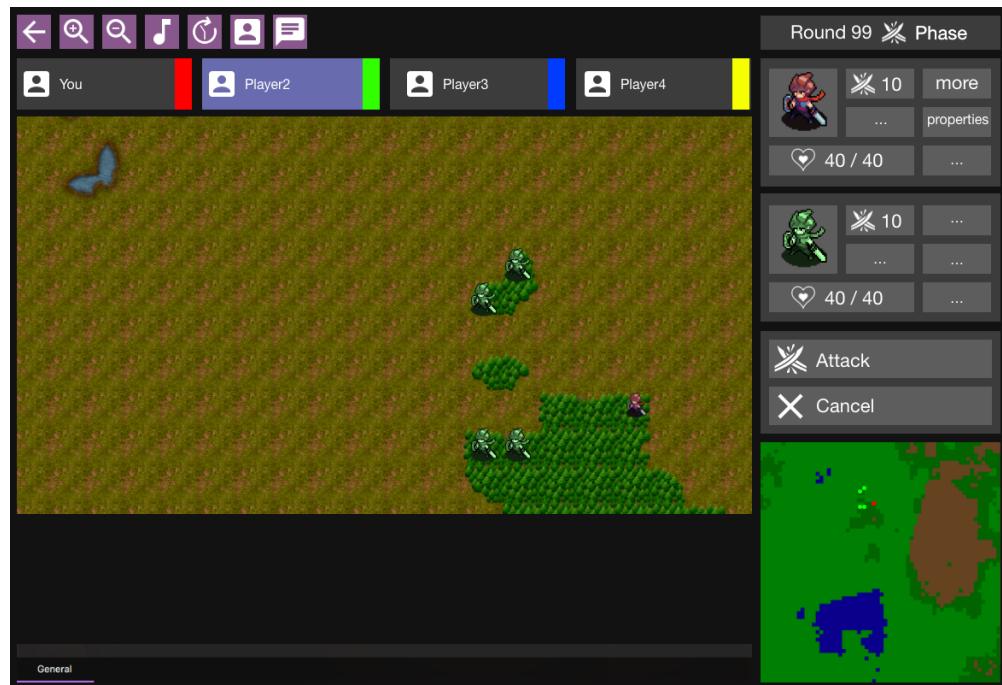


Abbildung 26: MockUp: Ingame

Der Chat und die Spielerkarten sollten auch über die jeweiligen Buttons oben in der Leiste eingeklappt werden können und Overlays über dem Spielfeld sein. Die Minikarte sollte unten rechts unter der Sidebar angezeigt werden. Die Sidebar war eine neue feste Box rechts neben dem Spielfeld. Diese sollte die Einheiteninformationen und die Bestätigen/Abbrechen Buttons enthalten. Die aktuelle Runde und die aktuelle Phase sollten oben über der Sidebar angezeigt werden. Der aktuelle Spieler sollte in seiner Playercard mit der secondary Farbe hinterlegt werden. In der oberen Leiste kam außerdem der Musik Button zum ein- und ausschalten der Musik hinzu.

### 2.3.3.1 Einheit auswählen

Der Server stellte drei Phasen zur Verfügung, wenn ein Spieler an der Reihe war. Eine Bewegungsphase, in der mindestens eine Einheit bewegt werden musste, eine Angriffsphase, die übersprungen werden konnte und eine restliche Bewegungspunktephase, die ebenfalls übersprungen werden konnte.



Abbildung 27: MockUp: Phase 1: Auswählen

Nachdem alle drei Phasen vom aktiven Spieler beendet wurden, war die Runde des Spielers vorbei. Wenn der Nutzer selbst an der Reihe war und in einer seiner drei Phasen auf eine Einheit klickte, sollte der Bewegungsradius oder Angriffsradius angezeigt werden. Klickte der Nutzer wieder auf die bereits ausgewählte Einheit, auf eine andere Einheit oder auf ein freies Feld, dann sollte die Einheit nicht mehr ausgewählt sein.



Abbildung 28: MockUp: Phase 2: Auswählen



Abbildung 29: MockUp: Phase 3: Auswählen

In einer gegnerischen Phase sollte die Einheit, die vom Gegner in seinem Client ausgewählt wurde, auch beim Nutzer, währenddessen der Nutzer dem Gegner zuschaute, auf die selbe Art und Weise ausgewählt werden. In Phase 1 sollten zusätzlich zum Bewegungsradius auch alle möglichen Angriffsziele markiert werden (siehe Abbildung 27). In Phase 2 sollten nur alle möglichen Angriffsziele angezeigt werden (siehe Abbildung 28).

In Phase 3 sollte ein verkürzter Bewegungsradius ohne Angriffsziele angezeigt werden (siehe Abbildung 29), je nachdem wie weit sich eine Einheit noch bewegen konnte.

### 2.3.3.2 Einheit bewegen

Nachdem der Nutzer in Phase 1 oder 3 eine Einheit ausgewählte und auf ein Feld klickte, sollte die Einheit auf das Feld bewegt werden, sobald der kürzeste Pfad berechnet und zum Server geschickt wurde. Falls nach der Bewegung noch Bewegungspunkte übrig waren, sollte die Einheit weiterhin ausgewählt bleiben und die Bewegungsreichweite dementsprechend verkleinert werden. War ein Gegner an der Reihe und der Server sendete eine bewegte Einheit, dann sollte auch zuerst die Bewegungsreichweite angezeigt werden und danach (kurz verzögert) die Einheit bewegt werden. Später sollte der Nutzer über eine Box in der Sidebar die Möglichkeit haben, seinen Zug doch noch zu widerrufen. Dabei sollte auf dem geklicktem Feld ein Icon angezeigt werden, bis der Nutzer die Aktion bestätigte oder abbruch (siehe Abbildung 30 oder 31). War ein Gegner an der Reihe, wurde der Text und das Icon in der Box nicht angezeigt.



Abbildung 30: MockUp: Bewegen bestätigen (Phase 1)



Abbildung 31: MockUp: Bewegen bestätigen (Phase 3)

### 2.3.3.3 Einheit angreifen

Nachdem der Nutzer in Phase 2 eine Einheit ausgewählte und alle möglichen Angriffsziele angezeigt wurden, konnte er eine gegnerische Einheit neben ihm auswählen (insofern eine vorhanden war) und sie angreifen.



Abbildung 32: MockUp: Angreifen bestätigen

War ein Gegner an der Reihe und der Server sendete eine angegriffene Einheit, dann sollten auch zuerst die möglichen Angriffsziele angezeigt werden und danach (kurz verzögert) sollte die Einheit angreifen. Später sollte dies auch noch über ein Box in der Sidebar bestätigt werden, um dem Nutzer die Möglichkeit zu geben, seinen Angriff doch nicht durchzuführen. Dabei sollte auf dem geklicktem Feld ein Icon angezeigt werden, bis der Nutzer die Aktion bestätigte oder abbruch (siehe Abbildung 32). War ein Gegner an der Reihe, wurde der Text und das Icon in der Box nicht angezeigt.

### 2.3.3.4 Spielende

Wenn ein Spieler verlor, da er keine Einheiten mehr hatte, sollte sich das Icon und der Name in seiner Playercard schwarz färben. Bei diesem Nutzer sollte ein Overlay angezeigt werden, in dem stand, dass er verloren hatte. Außerdem konnte er über das Overlay entscheiden, ob er sich zurück in die Lobby ging oder dem Spiel als Spectator beitrat (siehe Abbildung 33). Wenn nur noch ein Spieler übrig war, gewann er das Spiel. Diesem Nutzer sollte auch ein Overlay angezeigt werden, durch das er zurück zur Lobby wechselte (siehe Abbildung 34). Bei anderen Nutzern, die sich im Spectatormodus befanden, sollte auch ein ähnliches Overlay mit der gleichen Funktionalität angezeigt werden (siehe Abbildung 35).

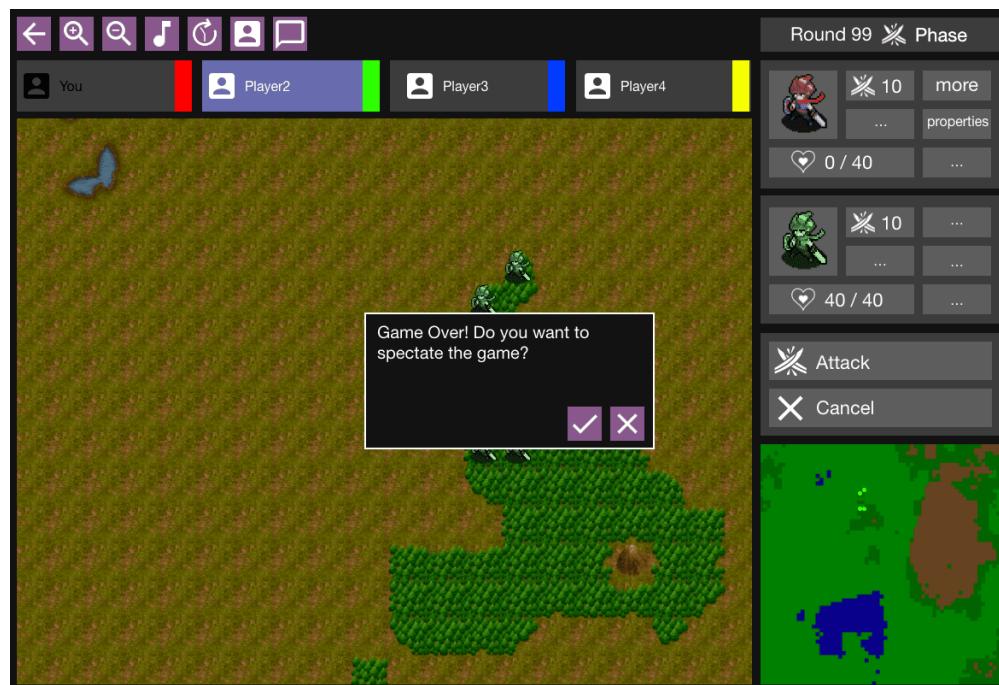


Abbildung 33: MockUp: Spiel verloren



Abbildung 34: MockUp: Spiel gewonnen



Abbildung 35: MockUp: Beobachtungsmodus: Spiel vorbei

### 2.3.3.5 Weitere Features

Es sollte für den Nutzer möglich sein, über einen Button die Phase beenden zu können und in die nächste zu wechseln. Nach Beenden aller drei Phasen, musste auch die Runde beendet werden. Wenn ein Gegner an der Reihe war, musste der Phase beenden Button disabled sein. Beim Beenden der Phasen sollte immer ein Overlay angezeigt werden

(siehe Abbildung 36).

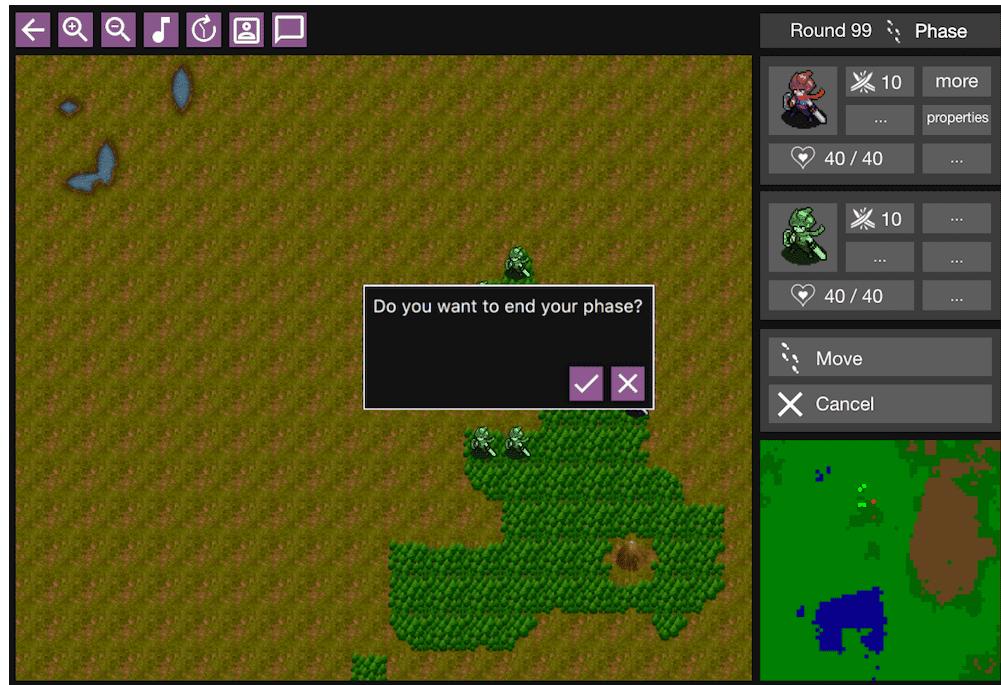


Abbildung 36: MockUp: Phase beenden

Es gab ein weiteres Feature, bei dem der Nutzer über eine Einheit hovern konnte, und je nachdem über welche Einheit der Nutzer hoerte, aktualisierten sich alle Properties und das Bild der gehoerten Einheit in der Sidebar. Die Einheiten des Spielers, der an der Reihe war, sollten in der oberen Box der Sidebar angezeigt werden. Die Einheiten der restlichen Spieler, die nicht an der Reihe waren, sollten in der unteren Box angezeigt werden. Zu Beginn jeder neuen Runde und sollten die beiden Boxen immer wieder leer sein und sich erst wieder füllen, wenn über eine Einheit gehoert bzw. wenn eine Aktion mit einer Einheit ausgeführt wurde.

### Domain Stories

Im Folgenden werden die Domain Stories für den Spielstart und das Ausführen einer Aktion, konkret die Bewegung einer Einheit, erläutert. Die Domain Stories wurden vor der Veröffentlichung der Serverdokumentation erstellt. Deshalb weichen die tatsächlichen Nachrichten leicht von den hier Aufgeführten ab.

### Spielstart

Der Nutzer tritt einem Spiel über die Lobby bei. Der Client zeigt nun die initiale Spielsituation, die Armeeauswahl und alle Spieler, welche sich im Spiel befinden (siehe Abbildung 37).

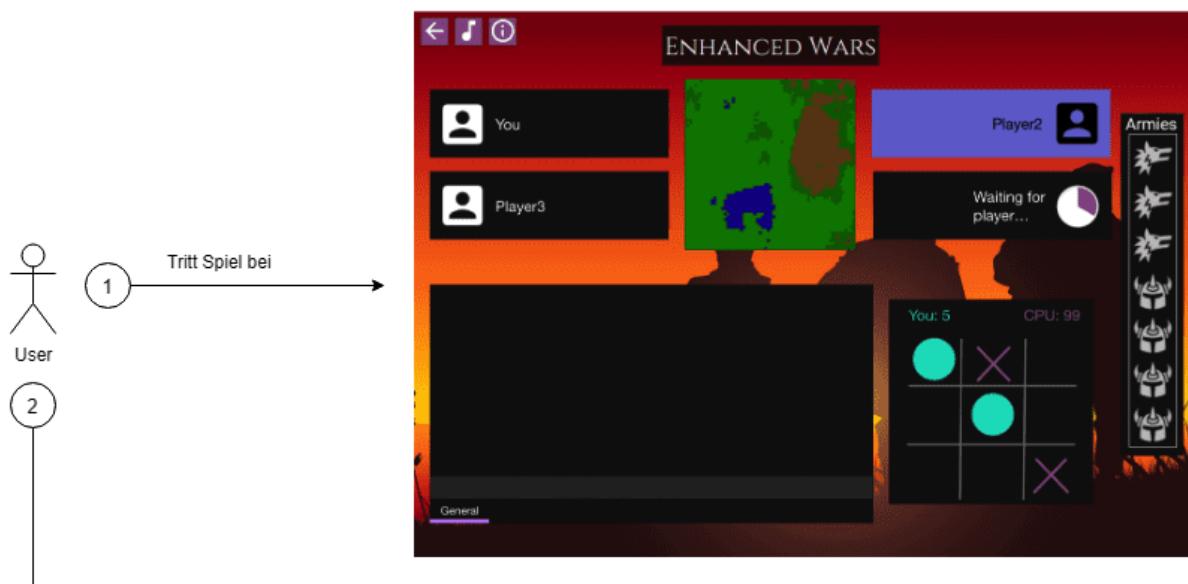


Abbildung 37: Domain Story: Ready 1

Der Nutzer wählt eine Armee aus. Die Spielerkarte des Nutzers wird aktualisiert und zeigt nun, dass er bereit ist (siehe Abbildung 38).

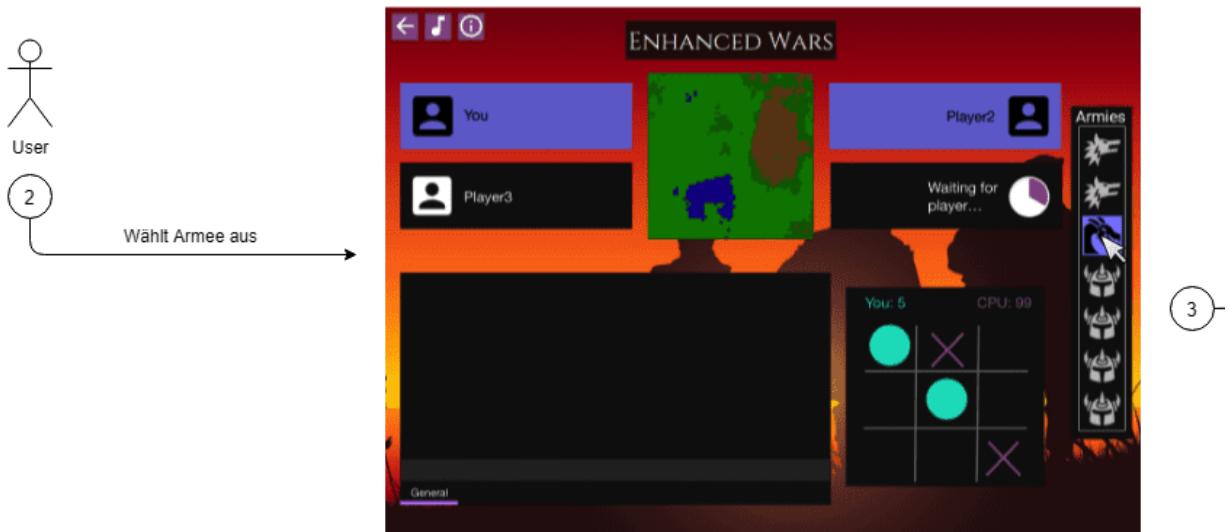


Abbildung 38: Domain Story: Ready 2

Die Anwendung teilt dem Server mit, dass der Nutzer nun bereit ist. Der Server informiert die Anwendung über den Spielstart. Das Spiel startet (siehe Abbildung 39).

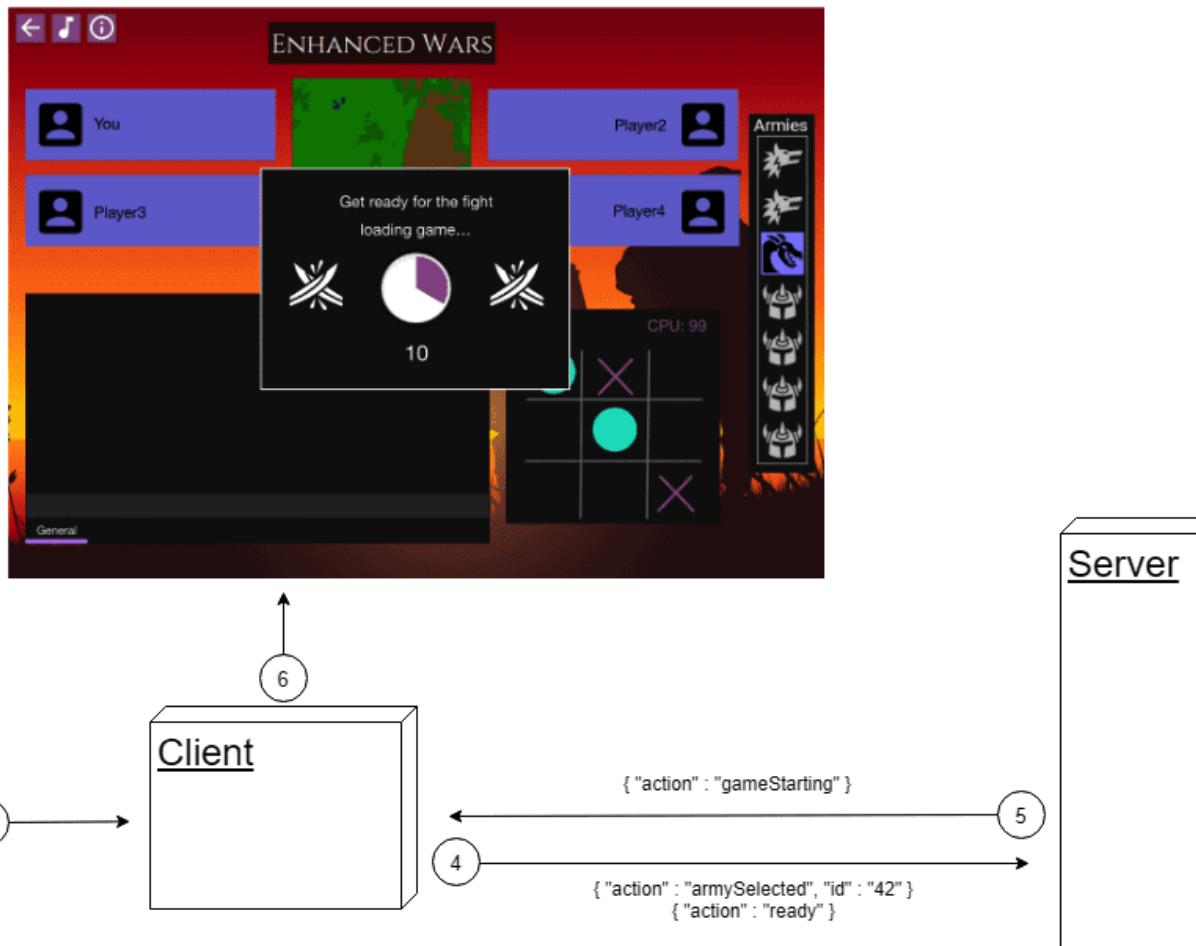


Abbildung 39: Domain Story: Ready 3

### Bewegung

Der Nutzer klickt auf eine Einheit, wodurch diese ausgewählt wird (siehe Abbildung 40).

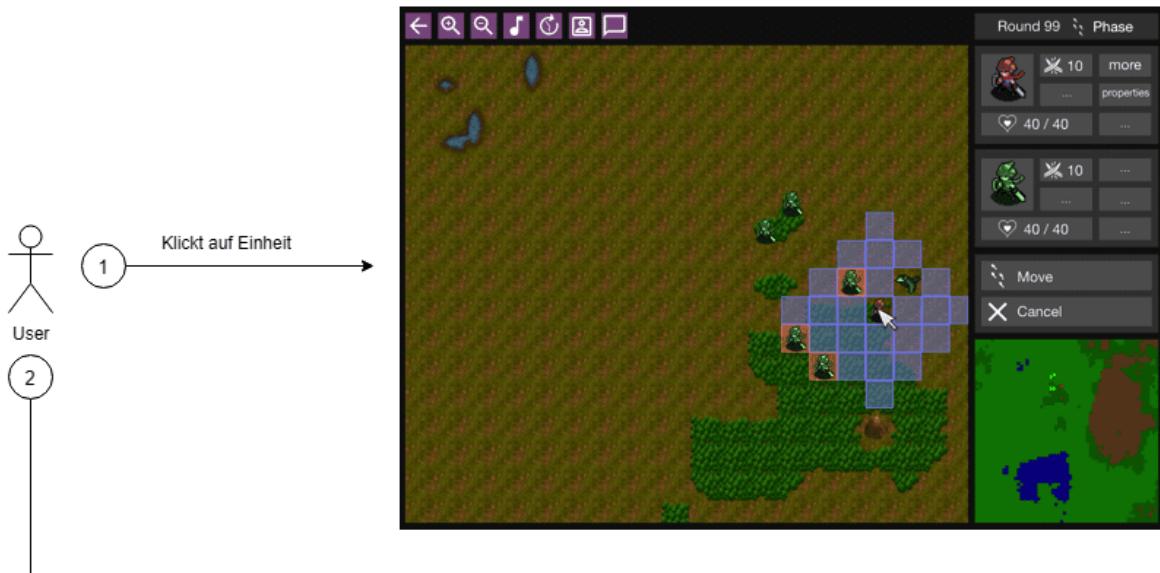


Abbildung 40: Domain Story: Move 1

Der Nutzer klickt auf gültiges Zielfeld und bestätigt die Aktion (siehe Abbildung 41).



Abbildung 41: Domain Story: Move 2

Die Anwendung sendet eine Nachricht, welche die Bewegung beschreibt, an den Server. Dieser informiert die Anwendung über die Zulässigkeit der Aktion. Die Anwendung aktualisiert das Datenmodell und das Benutzerinterface zeigt die neue Position der Einheit (siehe Abbildung 42).

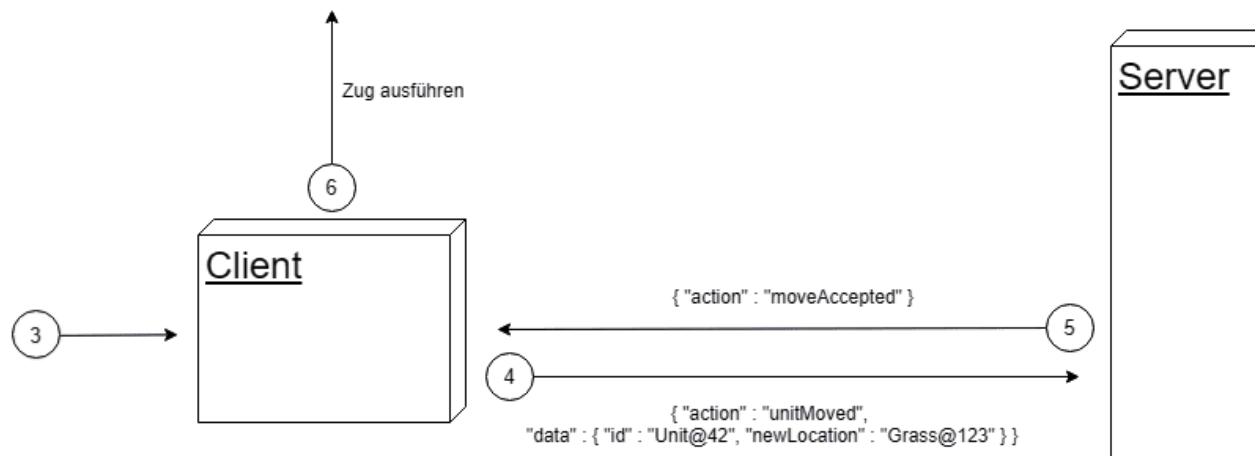


Abbildung 42: Domain Story: Move 3

## Sprint V

Der fünfte Sprint erstreckte sich über dem Zeitraum vom 8.7.2019 bis zum 21.7.2019. Es wurden 116 Storypoints für alle User Stories geschätzt.

### Sprintziel

Es sollte nach dem fünften Sprint möglich sein:

- Einem Spiel beizutreten
- Die Einheiten auf dem Spielfeld und der Minikarte zu sehen
- Einheiten auszuwählen, zu bewegen und anzugreifen
- Den Spielstatus zu sehen und Phasen zu beenden

Falls die Entwicklung schneller als erwartet voranschreiten sollte, würden der Ingamechat und das Spielende noch implementiert werden. Sollten alle Ziele abgeschlossen werden, würden über 80 Storypoints abgearbeitet werden, was bei vier Entwicklern als ein guter Fortschritt gewertet werden könnte.

### User Stories

Die User Stories, die vom Product Owner erstellt wurden, sind wie gefolgt aufgebaut: Dem Titel kann entnommen werden zu welcher Szene die Story gehört. Anschließend werden die Story Points und der zugeteilte Entwickler genannt. Danach folgt die Story und das Ziel. Das Ziel fasst die Subtasks, die der Scrum Master erstellt hat, zusammen. Danach werden alle Subtasks aufgelistet.

#### Waiting Room - Spielbeitritt anpassen

**Zuteilung** Diese User Story wurde auf 5 Storypoints geschätzt und an Omar Sood zugeordnet.

**Story** Karli befindet sich in der Wartreraumszene. Alle anderen Spieler sind bereit. Karli wählt eine Armee aus. Da nun alle Spieler bereit sind, startet das Spiel (und es findet ein Szenenwechsel statt). TODO: Verbindung zur Domain Story

**Ziel und Subtasks** Der Spielbeitritt sollte an die Serveränderungen angepasst werden. Das Auswählen einer Armee sollte dem Server ein Bereit-Signal senden und das Spiel sollte starten, sobald alle Spieler bereit sind.

**Armeeauswahl im Warteraum** Die Armeeauswahl soll nur im Warteraum möglich sein. Dafür muss sie aus der Lobby entfernt und von der Armeeauswahl des Army Managers getrennt werden. Das Deaktivieren der Join/Create Game Buttons muss so angepasst werden, dass diese nur deaktiviert sind, wenn der Spieler keine einzige gültige Armee besitzt.

**Ready-Funktionalität** Spieler, welche bereit sind, sollen im Warteraum hervorgehoben werden (siehe Mockup). Wählt ein Spieler eine Armee aus, signalisiert er gleichzeitig auch, dass er nun bereit ist.

**Spielstart** Sobald alle Spieler bereit sind (oder der Server eine entsprechende Nachricht sendet), soll das Spiel starten und ein entsprechender Szenenwechsel erfolgen. Der WebSocket darf dabei nicht mehr terminiert werden, sondern soll für den Ingame Controller zur Verfügung stehen. (Bei einem Wechsel zur Lobby oder beim Schließen der Anwendung muss der Socket natürlich weiterhin terminiert werden.)

#### Ingame - Spielfeld

**Zuteilung** Diese User Story wurde auf 13 Storypoints geschätzt und an Georg Siebert zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli sieht die Einheiten auf dem Spielfeld. Karli klickt auf ein Feld (einer Einheit). Das Feld wird ausgewählt.

**Ziel und Subtasks** Das Spielfeld sollte die Einheiten anzeigen. Jedes Feld musste anklickbar sein.

**Einheiten anzeigen** Die Einheiten sollen angezeigt werden. Ändert sich die Position einer Einheit (im Datenmodell), soll sich auch das Spielfeld entsprechend aktualisieren.

**OnClick-Extensionpoints** Es soll möglich sein, beim Anklicken eines Feldes Methoden mit der Referenz der darunterliegenden Datenstruktur (der Cell Instanz) aufzurufen.

#### Ingame - Phase beenden

**Zuteilung** Diese User Story wurde auf 3 Storypoints geschätzt und an Juri Lozowoj zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli klickt auf den Phase-Beenden Button (siehe Mockup). Die Phase wird beendet und die nächste tritt ein (bzw. der nächste Spieler ist an der Reihe).

**Ziel und Subtasks** Es sollte einen Button geben, welcher bei Betätigung die aktuelle Phase beendete. Dies sollte nur möglich sein, wenn der Nutzer auch seine Phase beenden konnte, da er an der Reihe war.

**Button und Funktionalität hinzufügen** Bei der Betätigung des Buttons soll die aktuelle Phase beendet und der Server darüber informiert werden. Es soll nicht möglich sein, den Button zu betätigen, wenn der Benutzer nicht an der Reihe ist.

#### Ingame - Einheit auswählen

**Zuteilung** Diese User Story wurde auf 13 Storypoints geschätzt und an Juri Lozowoj zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli wählt eine Einheit durch Klicken auf ihre Graphik aus. Gültige Bewegungsziele werden blau und gültige Angriffsziele rot markiert (siehe Mockups).

**Ziel und Subtasks** Durch Anklicken einer Einheit sollte diese Ausgewählt werden. Danach musste die Bewegungs- und Angriffsreichweite sichtbar sein, je nachdem in welcher Phase der Nutzer sich befand.

**Ausgewählte Einheit setzen** Klickt der Anwender auf eine Einheit, soll diese ausgewählt werden (soll im Datenmodell bzw. Controller hinterlegt werden). Zudem soll die Zelle der Einheit farblich hervorgehoben werden (siehe Mockups). Dies soll nur möglich sein, wenn der Spieler an der Reihe ist.

**Gültige Ziele markieren** Gültige Bewegungs- und Angriffsziele (Cell, eventuell auch Unit) sollen als solche markiert werden (Property). Zudem sollen sie farblich hervorgehoben werden (siehe Mockup). Weiterhin darf in der Angriffsphase nur der rote Angriffsradius angezeigt werden und in der restlichen Bewegungspunkte Phase nur der blaue Bewegungsradius (siehe Mockups). Blau: #868cfc für den Rahmen, beim Hintergrund 40% Opacity (siehe selected color im Stylesheet). Rot: #cf6679 für den Rahmen, beim Hintergrund 40% Opacity (siehe error color im Stylesheet).

### Ingame - Einheit bewegen

**Zuteilung** Diese User Story wurde auf 13 Storypoints geschätzt und an Omar Sood zugewieilt.

**Story** Karli befindet sich in der Ingameszene. Karli wählt eine Einheit aus, die Karli bewegen möchte. Karli klickt auf ein Feld in Bewegungsreichweite. Die Einheit wird bewegt. TODO: Verbindung zur Domain Story

**Ziel und Subtasks** Für die Bewegung einer Einheit sollte der kürzeste Pfad zum Zielfeld berechnet werden. Anschließend musste eine Nachricht an den Server gesendet und dessen Antwort korrekt verarbeitet werden.

**Berechnung des kürzesten Weges** Der Server erwartet einen gültigen Weg ohne Hindernisse. Dafür soll der Weg berechnet werden, der die geringste Anzahl an MP verbraucht.

**Zug an Server senden** Der Server soll über die Bewegung informiert werden.

**Antwort des Servers verarbeiten** Die Antwort des Servers soll korrekt verarbeitet werden. Das heißt, dass im Erfolgsfall die Bewegung im Datenmodell umgesetzt werden soll und im Fehlerfall der User eine entsprechende Meldung erhalten soll.

**Klickhandler** Beim Klick auf ein als gültiges Bewegungsziel markiertes Feld soll die aktuell ausgewählte Einheit dorthin bewegt werden.

**Aufräumen** Nach der Aktion soll aufgeräumt werden. Das heißt, dass die zuvor ausgewählte Einheit, welche die Aktion ausführt, nicht mehr ausgewählt sein soll. Es sollen keine Felder mehr hervorgehoben sein.

## Ingame - Einheit angreifen

**Zuteilung** Diese User Story wurde auf 5 Storypoints geschätzt und an Omar Sood zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli wählt eine Einheit aus. Karli sieht alle Felder mit Einheiten in Rot, die angegriffen werden können. Karli wählt eine gegnerische Einheit auf diesen Feldern aus. Die Einheit wird angegriffen.

**Ziel und Subtasks** Es sollte möglich sein, gegnerische Einheiten anzugreifen. Entsprechende Nachrichten sollten an den Server gesendet und dessen Antworten korrekt verarbeitet werden.

**Angriff an Server senden** Der Server soll über den Angriff informiert werden (Angreifende Einheit und angegriffene Einheit).

**Antwort des Servers verarbeiten** Die Antwort des Servers soll korrekt verarbeitet werden. Das heißt, dass im Erfolgsfall der Angriff im Datenmodell umgesetzt werden soll und im Fehlerfall der User eine entsprechende Meldung erhalten (zur Zeit sind keine Fehlerfälle bekannt) soll.

**Klickhandler** Beim Klick auf ein als gültiges Angriffsziel markiertes Feld soll, falls dort auch eine gegnerische Einheit positioniert ist, die aktuell ausgewählte Einheit diese angreifen.

**Aufräumen** Nach der Aktion soll aufgeräumt werden. Das heißt, dass die zuvor ausgewählte Einheit, welche die Aktion ausführt, nicht mehr ausgewählt sein soll. Es sollen keine Felder mehr hervorgehoben sein.

## Ingame - Spielstatus anzeigen

**Zuteilung** Diese User Story wurde auf 8 Storypoints geschätzt und an Tobias Klipp zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli sieht eine Runden- und Phasenanzeige. Karli klickt auf den Spieler-Anzeigen Button. Die Anzeige der Spieler wird eingeblendet (siehe Mockup). Der aktive Spieler ist hervorgehoben (in der selected-Color siehe Stylesheet bzw. MockUp).

**Ziel und Subtasks** Es sollte einen Button zum Ein- und Ausblenden der Spielerinformationen implementiert werden. Der aktive Spieler sollte hervorgehoben werden.

**Rundenanzeige** Die Rundenanzahl soll korrekt dargestellt und aktualisiert werden (siehe Mockup). Die aktuelle Phase soll angezeigt werden (Bewegungsphase, dann Angriffsphase, dann restliche Bewegungspunktephase). Die Phasen sollen neben dem Label mit einem Icon dargestellt werden. Laut Server befinden sich in einer Runde die drei verschiedenen Phasen.

**Spielerliste** Die Spielerliste soll alle Spieler, welche sich im Spiel befinden, und deren Farbe anzeigen. Der aktive Spieler soll hervorgehoben sein (in der selected Color). Spieler, welche das Spiel verlassen oder verloren haben, sollen mit einem schwarzen Icon und schwarzem Spielernamen hinterlegt sein.

**Ein-/Ausblenden der Spielerliste** Über einen Button in der oberen Leiste (siehe Mockup) soll sich die Spielerliste ein- und ausblenden lassen.

Ingame - Minikarte anzeigen

**Zuteilung** Diese User Story wurde auf 8 Storypoints geschätzt und an Georg Siebert zugewiesen.

**Story** Karli befindet sich in der Ingameszene. In der rechten oberen Ecke sieht Karli die Minikarte, welche neben dem Terrain auch alle Einheiten anzeigt. Karli drückt den Minikarte Button. Die Minikarte wird ausgeblendet.

**Ziel und Subtasks** Es sollte eine Minikarte implementiert werden, die über einen Button ein- und ausgeblendet werden konnte.

**Minikarte anzeigen** Die Minikarte soll, wie in der Story und den Mockups beschrieben, dargestellt werden.

**Ein-/Ausblenden der Minikarte** Über einen Button in der oberen Leiste (siehe Mockup) soll sich die Minikarte ein- und ausblenden lassen.

Ingame - Chatintegration

**Zuteilung** Diese User Story wurde auf 13 Storypoints geschätzt und an Tobias Klipp zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli drückt den Chat-Einblenden Button. Der Chat wird angezeigt. Karli schreibt eine Nachricht. Die Nachricht wird im Chat für alle Spieler im Spiel angezeigt. Bob schreibt eine Nachricht. Karli kann Bobs Nachricht lesen.

**Ziel und Subtasks** Es sollte einen Button zum Ein- und Ausblenden des Chats implementiert werden.

**Nachrichten senden und empfangen** Hinweis: Es sollte ausreichen, einen Chatnode 1:1 wie im WaitingRoomViewController zu instanzieren. Nachrichten sollen korrekt gesendet und empfangen werden (vgl. Warteraum und Lobbychat, siehe Serverdoku).

**Ein-/Ausblenden des Chats** Über einen Button in der oberen Leiste (siehe Mockup) soll sich der Chat ein- und ausblenden lassen. Der Chat soll bis zur Sidebar rechts gehen. Der Chat ist in einer StackPane über dem Spielfeld (und die Sidebar ist ein anderes Element neben dem Spielfeld).

## Ingame - Game Over

**Zuteilung** Diese User Story wurde auf 2 Storypoints geschätzt und an Tobias Klipp zugeteilt.

**Story** Karli befindet sich in der Ingameszene. Karlis Playerkarte färbt sich schwarz (Name und Icon). Karli sieht ein Overlay aufgehen, auf welchem Karli gefragt wird, ob Karli das Spiel weiter betrachten möchte. Karli drückt den Cancel Button. Karli wechselt in die Lobby.

**Ziel und Subtasks** Wenn der Nutzer verloren hatte, sollte ein Overlay angezeigt werden. Auf diesem konnte er in die Lobby wechseln oder dem Spiel zuschauen.

**Alert anlegen** Der im Mockup zu sehende Alert soll angezeigt werden, falls der Nutzer das Spiel verloren (keine Einheiten) hat UND nicht gleichzeitig ein anderer Spieler dadurch gewinnt (In diesem Fall greift der Spectate Over Alert). Bestätigt der User die Anfrage, soll der Alert geschlossen werden. Tut er dies nicht, soll in die Lobby gewechselt werden. Achtung: Es kann immer nur ein Alert aktiv sein. Um daher sicherzustellen, dass dieser Alert angezeigt wird, müssen zuvor eventuell aktive Alerts geschlossen werden.

## Ingame - Game Won

**Zuteilung** Diese User Story wurde auf 2 Storypoints geschätzt und an Tobias Klipp zugeteilt.

**Story** Karli befindet sich in der Ingameszene. Karli besiegt die letzte gegnerische Einheit. Karli sieht ein Overlay aufgehen (siehe MockUp). Karli wechselt in die Lobby.

**Ziel und Subtasks** Wenn der Nutzer gewann, sollte ein Overlay angezeigt werden, das ihn in die Lobby wechseln ließ.

**Alert anzeigen** Der im Mockup zu sehende Alert soll angezeigt werden, falls der Nutzer das Spiel gewonnen hat. Bei Betätigung des Buttons soll in die Lobby gewechselt werden. Achtung: Es kann immer nur ein Alert aktiv sein. Um daher sicherzustellen, dass dieser Alert angezeigt wird, müssen zuvor eventuell aktive Alerts geschlossen werden.

## Ingame - Spectator Over

**Zuteilung** Diese User Story wurde auf 2 Storypoints geschätzt und an Tobias Klipp zugeteilt.

**Story** Karli befindet sich in der Ingameszene. Bob besiegt die letzte gegnerische Einheit. Karli sieht ein Overlay aufgehen (siehe MockUp). Karli wechselt in die Lobby.

**Ziel und Subtasks** Wenn das Spiel vorbei war, sollte dem Nutzer ein Overlay angezeigt werden, durch das er in die Lobby wechselte.

**Alert anzeigen** Der im Mockup zu sehende Alert soll angezeigt werden, falls ein anderer Spieler (und nicht der Nutzer) das Spiel gewonnen hat. Bei Betätigung des Buttons soll in die Lobby gewechselt werden. Achtung: Es kann immer nur ein Alert aktiv sein. Um daher sicherzustellen, dass dieser Alert angezeigt wird, müssen zuvor eventuell aktive Alerts geschlossen werden.

#### Lobby - Spectating

**Zuteilung** Diese User Story wurde auf 5 Storypoints geschätzt und an Juri Lozowoj zugewiesen.

**Story** Karli befindet sich in der Lobbyszene. Karli sieht die Spielliste und möchte dieses mal nicht selbst spielen. Karli drückt auf den Spectator-Button. Karli wird in den Warterraumszene weitergeleitet.

**Ziel und Subtasks** Es sollte ein Button hinzugefügt werden, der einen Spielbeitritt als Beobachter ermöglichte.

**Button hinzufügen** Jeder Eintrag in der Spielliste soll um den neuen Button erweitert werden (siehe Mockup). Bei Betätigung dieses Buttons soll ein Szenenwechsel in den Warteraum stattfinden, dieses Mal jedoch nur als Zuschauer.

#### Ingame - Spectating

**Zuteilung** Diese User Story wurde auf 13 Storypoints geschätzt und an Juri Lozowoj zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli versucht etwas in den Chat zu schreiben. Karli versucht Aktionen mit den Einheiten auszuführen. Karlis Aktionen funktionieren nicht. Karli merkt aber, dass alles andere funktioniert, holt Popcorn und genießt das Spiel.

**Ziel und Subtasks** Das Spiel sollte so angepasst werden, dass ein Zuschauer weder mit dem Spielfeld interagieren, noch Chatnachrichten schreiben konnte.

**Zuschauereinfunktionalität hinzufügen** Die Szene soll der regulären Ingameszene (ohne Runde-Beenden Button) entsprechen. Der Benutzer soll jedoch keine Aktionen ausführen können oder Nachricht absenden können. Das Lesen von Chatnachrichten soll möglich sein.

#### Ingame - Einheiteninformationen

**Zuteilung** Diese User Story wurde auf 8 Storypoints geschätzt und an Georg Siebert zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli fährt mit seinem Mouse-Courser über eine Einheit. Alle wichtigen Informationen über die Einheit werden in einem Kontextmenü angezeigt.

**Ziel und Subtasks** Bewegte der Nutzer die Maus über eine Einheit, sollten alle verfügbaren Informationen dieser Einheit in einem Kontextmenü angezeigt werden.

**UI Element für Informationen erstellen** Es soll ein UI Element zur Anzeige der Informationen erstellt werden (siehe Mockups). Die Informationen sollten beinhalten: Alle Attribute der Einheit (nur bei den HP eine max/zurzeit Anzeige) und das idle gif der Einheit.

**Informationen bei Mouseover anzeigen** Das bereits erstellte UI Element soll bei einem Mouseover auf einer Einheit eingeblendet werden. Ansonsten soll es wieder ausgeblendet werden.

### Ingame - Musiksteuerung

**Zuteilung** Diese User Story wurde auf 3 Storypoints geschätzt und an Omar Sood zugewiesen.

**Story** Karli befindet sich in der Ingameszene und die Musik spielt. Karli klickt auf den Musik Button (siehe Mockup). Die Musik wird ausgeschaltet.

**Ziel und Subtasks** Es sollte ein Musik Button hinzugefügt werden, mit dem sich die Musik genauso bedienen ließ wie in den restlichen Szenen.

**Button und Funktionalität hinzufügen** Es soll ein Button zur (De-) Aktivierung der Musik hinzugefügt werden. Die Funktionalität soll die gleiche wie bei Login, Lobby und Warteraum sein. Das heißt, dass die Einstellung szenenübergreifend erhalten bleiben soll.

### Tasks

Die Tasks wurden vom Scrum Master erstellt und sind nicht aus Nutzersicht beschreibbar, da sie nur technische Aufgaben waren. Aus diesem Grund unterschieden sich Tasks nur in einer Sache von User Stories. Sie hatten keine Story.

### Servernachrichten

**Zuteilung** Diese Task wurde auf 8 Zeitstunden geschätzt und an Omar Sood zugewiesen.

**Ziel** Der Entwickler sollte alle restlichen Servernachrichten abfangen, die in keiner User Story berücksichtigt wurden. Dazu gehörten unter anderem Änderungen des Datenmodells, welche durch Aktionen der Gegenspieler anfielen.

### Einheiteninformationen im Datenmodell

**Zuteilung** Diese Task wurde auf 5 Zeitstunden geschätzt und an Tobias Klipp zugewiesen.

**Ziel** Der Entwickler sollte die Enums UnitType und UnitTypeInfo zu einer Enum zusammenfügen und musste darauf achten, dass alle Funktionalitäten erhalten blieben.

## Zeitübersicht

Diese Übersicht zeigt alle Stories, welche im Rahmen der Ziele bearbeitet werden sollten. Wie zu erkennen ist, wurden nicht alle Stories abgeschlossen. Es wurden jedoch schon umfangreichere Grundlagen für Features des sechsten Sprints gelegt.

User Story	Soll Zeit	Ist Zeit	Noch Zeit	Entwickler
Waiting Room - Spielbeitritt anpassen	5 h	13 h 9 min	-	Omar Sood
Ingame - Spielfeld	13 h	8 h 22 min	-	Georg Siebert
Ingame - Phase beenden	3 h	15 h 20 min	-	Juri Lozowoj
Ingame - Einheit auswählen	13 h	8 h 46 min	4h 14 min	Juri Lozowoj
Ingame - Einheit bewegen	13 h	11 h 19 min	35 min	Omar Sood
Ingame - Einheit angreifen	5 h	-	5 h	Omar Sood
Ingame - Spielstatus anzeigen	8 h	18 h 45 min	-	Tobias Klipp
Ingame - Minikarte anzeigen	8 h	8 h 25 min	4 h	Georg Siebert
Ingame - Chatintegration	13 h	-	13 h	Tobias Klipp
Ingame - Game Over	2 h	-	2 h	Tobias Klipp
Ingame - Game Won	2 h	-	2 h	Tobias Klipp

Tabelle 1: Zeitübersicht fünfter Sprint

## Analyse

Der fünfte Sprint endete am 21.7.2019. Das Team schaffte 29 der 116 geschätzten Storypoints.

## Burndown

Wie im Burndown-Diagramm zu sehen ist, wurden diesen Sprint nur vier Stories mit einem Umfang von 29 Storypoints abgeschlossen (siehe Kapitel 3.5.2). Wie Tabelle 1 zeigt, betrug die Arbeitszeit trotzdem 84 Stunden. Die Hauptursache war die Unterschätzung der Aufgabenumfänge, wodurch mehrere begonnene Stories in diesem Sprint nicht mehr abgeschlossen werden konnten und andere deutlich mehr Zeit als erwartet in Anspruch nahmen. Zudem führte die in diesem Release höhere Anzahl an blockierenden und relativen Tasks zu einer insgesamt langsameren Entwicklung, da sich die Entwickler häufig miteinander absprechen mussten.

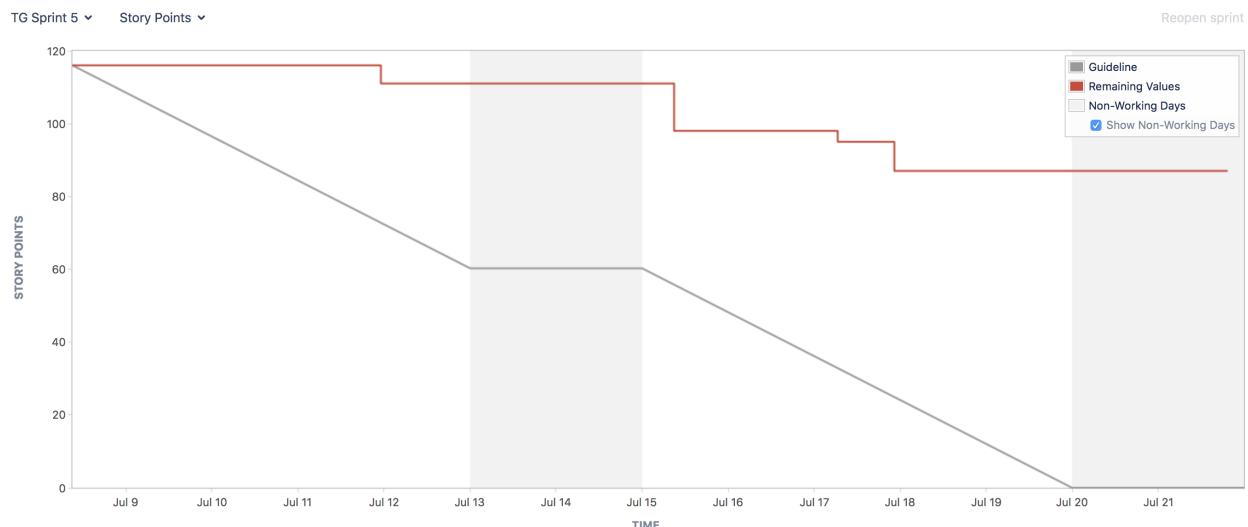


Abbildung 43: Sprint V: Burndown Diagramm

### Abgeschlossene Stories

Es wurden bisher vier Stories abgeschlossen (siehe Abbildung 44). Dies war weniger als erwartet, aber aufgrund komplexer Aufgaben nicht zu vermeiden. Im Nachhinein wird es nun eine genauere Beschreibung der abgeschlossenen Stories geben.

P	T	Key	Summary	Assignee	Status
↗	✚	TG-255	Waiting Room - Spielbeitritt anpassen	 Omar <goatfryed> ...	<span style="background-color: green; color: white; padding: 2px;">DONE</span>
↖	✚	TG-270	Ingame - Spielfeld	 Georg Siebert	<span style="background-color: green; color: white; padding: 2px;">DONE</span>
↖	✚	TG-273	Ingame - Phase beenden	 Juri Lozowoj	<span style="background-color: green; color: white; padding: 2px;">DONE</span>
▬	✚	TG-256	Ingame - Spielstatus anzeigen	 Tobias Klipp	<span style="background-color: green; color: white; padding: 2px;">DONE</span>

Abbildung 44: Sprint V: Abgeschlossene Stories

#### 3.5.2.1 Waiting Room - Spielbeitritt anpassen

**Verlauf** TODO: Schreiben

**Ergebnis** TODO: Schreiben

#### 3.5.2.2 Ingame - Spielfeld

**Verlauf** TODO: Schreiben

**Ergebnis** TODO: Schreiben

### 3.5.2.3 Ingame - Phase beenden

**Verlauf** TODO: Schreiben

**Ergebnis** TODO: Schreiben

### 3.5.2.4 Ingame - Spielstatus anzeigen

**Verlauf** TODO: Schreiben

**Ergebnis** TODO: Schreiben

Angefangene Stories

Wie bereits in der Analyse des Burndowns erwähnt wurde, wurden drei bereits begonnenen Stories zum Ende des Sprints noch nicht abgeschlossen. Im kommenden Abschnitt wird es einen Überblick geben, wie weit diese bearbeitet wurden.

### 3.5.3.1 Ingame - Einheit auswählen

**Verlauf** TODO: Schreiben

**Stand** TODO: Schreiben

### 3.5.3.2 Ingame - Einheit bewegen

**Verlauf** TODO: Schreiben

**Stand** TODO: Schreiben

### 3.5.3.3 Ingame - Minikarte anzeigen

**Verlauf** TODO: Schreiben

**Stand** TODO: Schreiben

Nicht abgeschlossene Stories/Tasks

Des Weiteren wurden neben den drei bereits angefangenen Stories noch weitere elf Stories beziehungsweise Tasks aufgrund von Zeitgründen nicht angefangen und abgeschlossen (siehe Abbildung 45). Diese Aufgaben waren nicht optional und sollten von unserem Team im nächsten Sprint bewältigt werden.

P	T	Key	Summary	Assignee	Status
~	BUG	TG-276	Ingame - Einheit auswählen	 Juri Lozowoj	TO DO
~	BUG	TG-281	Ingame - Einheit bewegen	 Omar <goatfryed> ...	TO DO
~	BUG	TG-282	Ingame - Einheit angreifen	 Omar <goatfryed> ...	TO DO
~	CHECK	TG-311	Einheiteninformationen im Datenmodell	 Tobias Klipp	TO DO
==	BUG	TG-263	Ingame - Minimap anzeigen	 Georg Siebert	TO DO
==	BUG	TG-267	Ingame - Chatintegration	 Tobias Klipp	TO DO
==	BUG	TG-283	Ingame - Game Over	 Tobias Klipp	TO DO
==	BUG	TG-285	Ingame - Game Won	 Tobias Klipp	TO DO
==	BUG	TG-286	Ingame - Spectator Over	 Tobias Klipp	TO DO
==	BUG	TG-287	Lobby - Spectating	 Juri Lozowoj	TO DO
==	BUG	TG-288	Ingame - Spectating	 Juri Lozowoj	TO DO
==	CHECK	TG-289	Servernachrichten	 Omar <goatfryed> ...	TO DO
▼	BUG	TG-266	Ingame - Musiksteuerung	 Omar <goatfryed> ...	TO DO
▼	BUG	TG-275	Ingame - Einheiteninformationen	 Georg Siebert	TO DO

Abbildung 45: Sprint V: Nicht Abgeschlossene Stories/Tasks

## Fazit

Die Ziele des Sprints wurden verfehlt. Um die Mindestanforderungen zu erreichen, müsste das Entwicklerteam seine Produktivität im sechsten Sprint deutlich steigern. Da die C0 Abdeckung (siehe Abbildung 46) zum Ende des fünften Sprints bereits 79% betrug, gab es bei der Qualitätssicherung keine Defizite.

96% classes, 79% lines covered in package 'de.uniks.se19.team_g'			
Element	Class, %	Method, %	Line, %
 project_rbsg	96% (410/423)	81% (1820/2235)	79% (8878/11197)

Abbildung 46: Sprint V: C0 Testabdeckung (Siehe Line, %)

## Sprint VI

Der sechste Sprint erstreckte sich über dem Zeitraum vom 22.7.2019 bis zum 4.8.2019. Er umfasst 87 Storypoints.

### Sprintziel

Es sollte nach dem sechsten Sprint möglich sein:

- Alle Features aus dem Sprint V (siehe Kapitel 3.1) zu verwenden
- Einen Ingame Chat zu benutzen
- Verschiedene Overlays beim Spielende zu sehen
- Einem Spiel als Beobachter beizutreten
- Einheiteninformationen in einer Sidebar einzusehen
- Die Musik auch im Spiel ein- und auszuschalten

Für den Fall, dass die Mindestanforderungen vor Ende des Sprints abgeschlossen werden sollten, sollte eine Sidebar zum Spielfeld hinzugefügt werden. Diese sollte es ermöglichen Aktionen, wie Bewegen oder Angreifen, zu bestätigen und abzubrechen. Falls das Team die Umsetzung der oben genannten Features schaffen würde, wären die Mindestanforderungen jedoch bereits erfüllt.

### Geänderte User Stories vom Sprint V

Die folgenden Stories des fünften Sprints wurden aktualisiert, da sich ihre Mockups änderten.

#### Ingame - Minikarte anzeigen

**Zuteilung** Diese User Story wurde auf 8 Storypoints geschätzt und an Georg Siebert zugeteilt.

**Geänderte Story** Karli befindet sich in der Ingameszene. In der rechten unteren Ecke sieht Karli die Minikarte, welche neben dem Terrain auch alle Einheiten anzeigt. Karli drückt auf eine Stelle in der Minikarte. Der Sichtbereich ändert sich zu dieser Stelle.

**Geändertes Ziel und Subtasks** Es sollte eine Minikarte implementiert werden. Über die Minikarte sollte es möglich sein, den Sichtbereich zu wechseln.

**Minikarte anzeigen** Die Minikarte soll, wie in der Story und den Mockups beschrieben, dargestellt werden.

**Springen über die Minikarte** Durch das Klicken auf eine Stelle auf der Minikarte soll der Sichtbereich auf dem Spielfeld verschoben werden.

## Ingame - Einheiteninformationen

**Zuteilung** Diese User Story wurde auf 8 Storypoints geschätzt und an Georg Siebert zugeteilt.

**Geänderte Story** Karli befindet sich in der Ingameszene. Karli fährt mit Karlis Mouse-Courser über eine Einheit. Alle wichtigen Informationen über die Einheit werden in der Sidebar angezeigt (siehe MockUps).

**Geändertes Ziel und Subtasks** Bewegte der Nutzer die Maus über eine Einheit, sollten alle verfügbaren Informationen dieser Einheit in einer Sidebar angezeigt werden.

**UI Element für Informationen erstellen** Es soll ein UI Element zur Anzeige der Informationen erstellt werden (siehe Mockups). Die Informationen sollen beinhalten: Alle Attribute der Einheit (nur bei den HP eine max/zurzeit Anzeige) und das idle gif der Einheit.

**Informationen bei Mouseover anzeigen** Das bereits erstellte UI Element soll bei einem Mouseover auf einer Einheit eingeblendet werden und eingeblendet bleiben, bis über eine neue Einheit gehovert wird. Die Einheit, die der Spieler ausgewählt hat, soll in der oberen Box angezeigt werden. Die Einheit, über die zuletzt gehovert wurde, soll in der unteren Box angezeigt werden.

## Übernommene User Stories aus Sprint V

Weiterhin gab es auch User Stories aus dem fünften Sprint, welche noch nicht abgeschlossen wurden. Diese wurden mit in den sechsten Sprint übernommen. Es folgt eine Liste dieser Stories:

- Ingame - Einheit auswählen (angefangen in Sprint V)
- Ingame - Einheit bewegen (angefangen in Sprint V)
- Ingame - Einheit angreifen
- Ingame - Spielstatus anzeigen (angefangen in Sprint V)
- Ingame - Minikarte anzeigen (angefangen in Sprint V)
- Ingame - Chatintegration
- Ingame - Game Over
- Ingame - Game Won
- Ingame - Spectator Over
- Lobby - Spectating
- Ingame - Spectating
- Ingame - Einheiteninformationen
- Ingame - Musiksteuerung

Eine genauere Beschreibung zu jeder Story befindet sich im Kapitel 3.2.

## User Stories

Im sechsten Sprint wurden weitere User Stories vom Product Owner erstellt.

### Ingame - Einheit bewegen nach Bewegung

**Zuteilung** Diese User Story wurde auf 5 Storypoints geschätzt und an Omar Sood zugewiesen.

**Story** Karli befindet sich in der Ingameszene. Karli klickt auf eine Einheit. Die Bewegungsreichweite wird angezeigt. Karli klickt auf ein Feld in Bewegungsreichweite. Die Einheit wird bewegt und bleibt weiterhin ausgewählt, aber der blaue Bewegungsradius hat sich verkleinert. Karli bewegt die Einheit noch einmal auf ein weiteres Feld. Die Bewegungspunkte sind noch nicht aufgebraucht und die Einheit bleibt weiter ausgewählt. Karli hat keine Lust mehr die Einheit zu bewegen und wählt sie ab.

**Ziel und Subtasks** Nachdem der Nutzer eine Einheit bewegte, sollte diese erneut ausgewählt sein, falls sie noch verbleibende Bewegungspunkte besaß.

**Einheit auswählen** Nachdem der Nutzer eine Einheit bewegt hat, soll diese erneut ausgewählt werden, falls sie noch verbleibende Bewegungspunkte besitzt.

### Geänderte Tasks vom Sprint V

Die folgenden Tasks des fünften Sprints wurden aktualisiert, da sich ihre Zuteilung änderte.

### Einheiteninformationen im Datenmodell

**Geänderte Zuteilung** Diese Task wurde auf 5 Zeitstunden geschätzt und an Omar Sood zugewiesen.

**Ziel** Der Entwickler sollte die Enums UnitType und UnitTypeInfo zu einer Enum zusammenfügen und musste darauf achten, dass alle Funktionalitäten erhalten blieben.

### Übernommene Tasks aus Sprint V

Weiterhin gab es auch Tasks aus dem fünften Sprint, welche noch nicht abgeschlossen wurden. Diese wurden mit in den sechsten Sprint übernommen. Es folgt eine Liste dieser Tasks:

- Servernachrichten
- Einheiteninformationen im Datenmodell

Eine genauere Beschreibung zu jeder Task befindet sich im Kapitel 3.3.

### Tasks

Im sechsten Sprint wurden weitere Tasks vom Scrum Master erstellt.

Ingame - Bestätigungsbox in der Sidebar

**Zuteilung** Diese Task wurde auf TODO Zeitstunden geschätzt und dan Georg Siebert zugeteilt.

**Ziel** Es musste eine Box in der Sidebar erstellt werden, die es möglich machte, Aktionen zu bestätigen oder abzubrechen (Die Funktionalitäten sollten erst später in den untergeordneten User Stories erstellt werden). War der User nicht an der Reihe oder im Beobachtungsmodus, musste diese Box disabled werden. Beziehungsweise sollte dann auf den beiden Buttons der Text und das Icon nicht mehr sichtbar sein.

### Bugs

Bugtasks zum Beheben von Problemen wurden vom Scrum Master erstellt, da im Verlauf der Entwicklung und beim Testen des Clients der ein oder andere Fehler aufkam. Diese werden hier wie gefolgt beschrieben: Zuerst wird genannt, wer den Bug beheben sollte. Danach wird das Problem beschrieben, das den Bug darstellte. Bearbeitungszeit wurden bei den Bugtasks nicht geschätzt.

Terminierung sicherstellen

**Zuteilung** Dieser Bug sollte von Omar Sood behoben werden.

**Problem** Die Terminierung des GameEventSockets musste angepasst werden. Sie sollte nicht doppelt aufgerufen werden. Weiterhin sollte sie immer stattfinden, wenn der Nutzer die Anwendung schloss oder in die Lobby wechselte.

Ingame - Einheit mehrmals auswählen

**Zuteilung** Dieser Bug sollte von Juri Lozowoj behoben werden.

**Problem** Falls der Nutzer eine Einheit auswählte, abwählte und dann wieder auswählte, ohne vorher eine andere Einheit ausgewählt zu haben, wurde sie danach als Angriffs- bzw. Bewegungsziel farblich markiert. Dies sollte nicht stattfinden.

Spectating - Automatischer Wechsel zum Spielfeld

**Zuteilung** Dieser Bug sollte von Juri Lozowoj behoben werden.

**Problem** Falls der Nutzer einem Spiel als Beobachter beitrat und dieses bereits startete, wechselte der Client nicht zum Spielfeld. Falls also das Spiel bereits startete (sollte an den vorhandenen Einheiten, der aktuellen Phase oder dem aktuellen Spieler am Datenmodell zu erkennen sein), sollte ein automatischer Wechsel zum Spielfeld stattfinden.

Lobby - Spielbeitritt blockieren

**Zuteilung** Dieser Bug sollte von Georg Siebert behoben werden.

**Problem** Falls ein Spiel bereits voll war, sollte kein Spielbeitritt mehr möglich sein. Dafür sollte der entsprechende Button deaktiviert werden (und aktiviert werden, falls wieder ein Platz frei wurde).

## Zeitübersicht

TODO: Intro schreiben

User Story/Task/Bug	Soll Zeit	Ist Zeit	Noch Zeit	Entwickler
Ingame - Einheit auswählen	13 h	16 h 29 min	-	Juri Lozowoj
Ingame - Einheit bewegen	13 h	13 h 35 min	-	Omar Sood
Ingame - Einheit angreifen	5 h	-	5 h	Omar Sood
Ingame - Chatintegration	13 h	2 h	11 h	Tobias Klipp
Ingame - Minikarte anzeigen	8 h	12 h 14 min	-	Georg Siebert
Lobby - Spectating	5 h	-	5 h	Juri Lozowoj
Ingame - Spectating	13 h	-	13 h	Juri Lozowoj
Ingame - Game Over	2 h	-	2 h	Tobias Klipp
Ingame - Game Won	2 h	-	2 h	Tobias Klipp
Ingame - Spectator Over	2 h	-	2 h	Tobias Klipp
Ingame - Einheiteninformationen	8 h	1 h	7 h	Georg Siebert
Ingame - Spielstatus anzeigen	8 h	31 h 30 min	-	Tobias Klipp
Ingame - Musiksteuerung	3 h	20 min	-	Omar Sood
Ingame - Einheit bewegen nach Bewegung	5 h	-	5 h	Omar Sood
Einheiteninformationen im Datenmodell	5 h	1 h 27 min	-	Omar Sood
Servernachrichten	8 h	-	-	Omar Sood
Ingame - Bestätigungsbox in der Sidebar	3 h	-	-	Georg Siebert
Terminierung sicherstellen	-	-	?	Omar Sood
Ingame - Einheit mehrmals auswählen	-	-	?	Juri Lozowoj
Spectating - Automatischer Wechsel zum Spielfeld	-	-	?	Juri Lozowoj
Lobby - Spielbeitritt blockieren	-	-	?	Georg Siebert

Tabelle 2: Zeitübersicht sechster Sprint

## Analyse

TODO: Intro schreiben

Burndown

TODO: Schreiben

Ausreißer

TODO: Schreiben

Abgeschlossen

TODO: Schreiben

Angefangen

TODO: Schreiben

Nicht abgeschlossen

TODO: Schreiben

Entfernt

TODO: Schreiben

Fazit

TODO: Schreiben

Abschluss Release III

TODO: Intro schreiben

Neue MockUps im Verlauf des Releases III

TODO: Schreiben

Vergleich MockUps mit aktueller Implementation

TODO: Schreiben

## Quellenangaben

## Abbildungsverzeichnis

1	Release II: Login Szene . . . . .	6
2	Release II: Login Formular . . . . .	6
3	Release II: Lobby Szene: Keine Armee ausgewählt . . . . .	7
4	Release II: Create Game Formular . . . . .	7
5	Release II: Lobby Szene: Armee ausgewählt . . . . .	8
6	Release II: Lobby Szene: Sprache DE . . . . .	8
7	Release II: Army Manager Szene: Keine Einheit ausgewählt . . . . .	9
8	Release II: Army Manager Szene: Einheit ausgewählt . . . . .	9
9	Release II: Property Info . . . . .	10
10	Release II: Armee bearbeiten . . . . .	10
11	Release II: Army Manager Szene: Speicherbar . . . . .	10
12	Release II: Waiting Room Szene: 2 Spieler . . . . .	11
13	Release II: Waiting Room Szene: 4 Spieler . . . . .	11
14	Release II: Ingame Szene . . . . .	12
15	Release II: Ingame Szene: Zoom In . . . . .	13
16	Release II: Ingame Szene: Zoom Out . . . . .	13
17	Release II: Login Fehler . . . . .	14
18	Release II: Spielfehler . . . . .	14
19	Release II: Spiel verlassen . . . . .	14
20	Release II: Logout . . . . .	14
21	Release II: C0 Testabdeckung (Siehe Line, %) . . . . .	14
22	MockUp: Beobachtungsmodus Lobby . . . . .	15
23	MockUp: Nutzer nicht bereit . . . . .	16
24	MockUp: Nutzer bereit . . . . .	16
25	MockUp: Spielstart . . . . .	17
26	MockUp: Ingame . . . . .	17
27	MockUp: Phase 1: Auswählen . . . . .	18
28	MockUp: Phase 2: Auswählen . . . . .	19
29	MockUp: Phase 3: Auswählen . . . . .	19
30	MockUp: Bewegen bestätigen (Phase 1) . . . . .	20
31	MockUp: Bewegen bestätigen (Phase 3) . . . . .	21
32	MockUp: Angreifen bestätigen . . . . .	21
33	MockUp: Spiel verloren . . . . .	22
34	MockUp: Spiel gewonnen . . . . .	23
35	MockUp: Beobachtungsmodus: Spiel vorbei . . . . .	23
36	MockUp: Phase beenden . . . . .	24
37	Domain Story: Ready 1 . . . . .	25
38	Domain Story: Ready 2 . . . . .	25
39	Domain Story: Ready 3 . . . . .	26
40	Domain Story: Move 1 . . . . .	27
41	Domain Story: Move 2 . . . . .	27
42	Domain Story: Move 3 . . . . .	28
43	Sprint V: Burndown Diagramm . . . . .	38
44	Sprint V: Abgeschlossene Stories . . . . .	38
45	Sprint V: Nicht Abgeschlossene Stories/Tasks . . . . .	40

46	Sprint V: C0 Testabdeckung (Siehe Line, %) . . . . .	40
----	--	----

## Tabellenverzeichnis

1	Zeitübersicht fünfter Sprint . . . . .	37
2	Zeitübersicht sechster Sprint . . . . .	45