

**ChessTime**  
**Spezifikation der Softwareanforderungen**  
*Version 2*

## Bearbeitungshistorie

Datum	Version	Beschreibung	Autor
01.08.2018	1.0	Format & Inhalt	Mika Stamm
06.09.2018	2.0	Verbesserungen	Mika Stamm
02.11.2018	2.1	Änderungen Ergänzt	Björn Crombach

# Inhaltsverzeichnis

<b>1 - Vorwort</b>	<b>5</b>
1.1 - Ziel des Dokuments	5
1.2 - Geltungsbereich	5
1.3 - Definitionen, Akronyme, und Abkürzungen	5
1.4 - Referenzen	5
<b>2 - Anforderungen aus App (Client) Perspektive</b>	<b>6</b>
2.1 - Flow of Events	6
Schach Spielregeln	6
Ziel	6
Spielfeld	6
Züge	6
2.2 – Use-Case Modell	7
2.3 – Anwendungsfall Beschreibungen	8
2.3.1 - Spielvorbereitung	8
2.3.1.1 - Spielregeln anzeigen	8
2.3.1.2 - Anmelden	8
2.3.1.3 - Gegenspieler Suchen	8
2.3.1.4 - Spiel fortsetzen	9
2.3.2 - Spielablauf	9
2.3.2.1 - Spielzug durchführen	9
2.3.2.2 - Remis beantragen	10
2.3.2.3 - Figur auswählen	10
2.3.2.4 - Figur bewegen	10
Erweiterungen: Figur bewegen	12
2.3.2.3.1 - Springer bewegen	12
2.3.2.3.2 - Läufer bewegen	12
2.3.2.3.3 - Turm bewegen	12
2.3.2.3.4 - Dame bewegen	13
2.3.2.3.5 - Bauer bewegen	13
2.3.2.3.6 - König bewegen	14
2.3.2.4 - Spielzustand aktualisieren	15
2.4 - Zusätzliche Anforderungen	16
2.4.1 - Remis	16
2.4.2 - Sicherheit	16
2.4.3 - Fehler aufzeichnung	16
2.4.4 - Benutzbarkeit	16
2.4.5 - Performance	16
2.4.6 - Verwendung von Technologien	16
2.4.7 - Plattform	16

2.4.8 - Abgabedatum	16
2.4.9 - Hardwareanforderungen	16
2.5 - User Interface Prototype	18
<b>3 - Requirements aus Server Perspektive</b>	<b>19</b>
3.1 - Flow of Events	19
3.2 - Use-Case Modell	19
3.3 - Anwendungsfall Beschreibungen	19
3.3.1 - Gegenspieler Suchen	19
3.3.2 - Elo-Zahl abrufen	20
3.3.3 - Elo-Zahl anpassen	20
3.3.4 - Spielzug weiterleiten	21
3.3.5 - Spielzug validieren	21
3.3.6 - Spieler sperren	22
3.4 Zusätzliche Anforderungen	22
3.4.1 - Sicherheit	22
3.4.2 - Fehler aufzeichnung	22
3.4.3 - Performance	22
3.4.4 - Verfügbarkeit	22
3.4.5 - Datenpersistenz	22
3.4.6 - Verwendung von Technologien	23
3.4.7 - Plattform	23
3.4.8 - Abgabedatum	23
3.4.9 - Hardwareanforderungen	23

# **1 - Vorwort**

## **1.1 - Ziel des Dokuments**

Ziel dieses Dokumentes ist es die funktionalen Anforderungen, nicht funktionale Anforderungen, Randbedingungen und anderen Faktoren, die für die vollständige Anforderungserfassung für die Software ChessTime notwendig sind, zu Dokumentieren.

## **1.2 - Geltungsbereich**

Android-App für das Spiel Schach mit einer Server-Komponente, die das Online spielen ermöglicht.

## **1.3 - Definitionen, Akronyme, und Abkürzungen**

Siehe Glossar.

## **1.4 - Referenzen**

- Glossar
- Vision
- Software Development Plan
- Risk List

## **2 - Anforderungen aus App (Client) Perspektive**

### **2.1 - Schach Spielregeln**

Um die in diesem Dokument beschriebenen Anwendungsfälle in einen Kontext zu setzen beschreibt dieser Abschnitt die Spielregeln von Schach.

#### **Ziel**

Das Ziel des Schachspiels ist es, den Gegenspieler matt zu setzen. Matt bedeutet, dass der König im nächsten Spielzug geschlagen werden könnte und es keinen Spielzug gibt der dies verhindern könnte.

#### **Spielfeld**

Das Spiel wird auf einem Spielfeld bestehend aus 64, in einem 8x8 Raster angeordneten, Feldern gespielt.

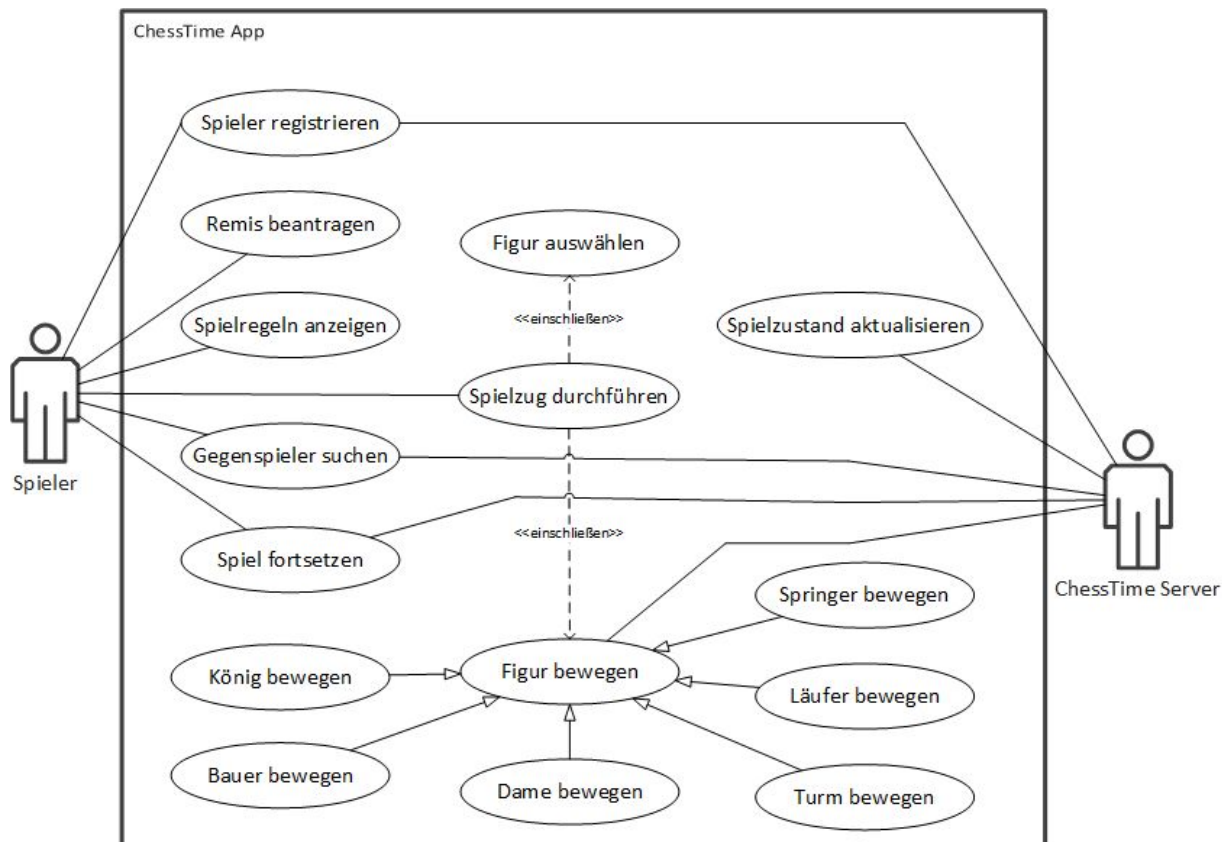
#### **Züge**

Beginnend mit dem Spieler mit den Weißen Figuren sind die Spieler abwechselnd am Zug. Ein Zug beinhaltet die Bewegung genau einer eigenen Figur. Die Ausnahme dazu bildet die Rochade, bei der sich der König und Turm in einem Zug bewegen.

Ein Spieler kann Figuren des Gegenspielers schlagen, indem er eine seiner Figuren auf ein Feld bewegt, das von einer gegnerischen Figur besetzt ist. Dies entfernt die geschlagene Figur aus dem Spiel.

Es gibt 6 unterschiedliche Figuren, die verschiedene Bewegungsmuster aufweisen (Anwendungsfall 2.3.2.3 und Erweiterungen).

## 2.2 – Use-Case Modell



### Beispielhafter Ablauf

- Der Spieler startet das Spiel und wird eingeloggt / Registriert sich. (Anwendungsfall: *Client 1.1*).
  - Ein Spiel wird gestartet.
    - Der Spieler sucht einen Gegenspieler (Anwendungsfall *Client 1.3*).
    - Der Spieler setzt eines der Laufenden Spiele fort (Anwendungsfall *Client 1.4*).
  - Spiel-Ansicht wird angezeigt (*UI Konzept 2*).
  - Die Spieler tätigen abwechselnd Spielzüge (Anwendungsfälle in 2.3.2 - *Spielablauf*).
- Der König eines Spielers wird matt gesetzt.

## 2.3 – Anwendungsfall Beschreibungen

### 2.3.1 - Spielvorbereitung

#### 2.3.1.1 - Spielregeln einsehen

Name des Use-Case	Spielregeln einsehen
ID	Client 1.1
Beschreibung	Es öffnet sich ein Fenster, das die Spielregeln anzeigt (User Interface Prototype 4)
Beteiligte Akteure	Spieler
Priorität	Niedrig

#### 2.3.1.2 - Spieler registrieren

Name des Use-Case	Spieler registrieren
ID	Client 1.2
Beschreibung	<p>Beim ersten Starten der App wird der Spieler aufgefordert einen Benutzernamen anzugeben.</p> <p>Dieser wird an den Server übermittelt, der prüft ob der Name bereits vergeben ist. Falls der Name bereits vergeben ist wird dem Nutzer eine Fehlermeldung angezeigt.</p> <p>Beim anmelden wird ein HTTP-POST request an <i>*serverurl*/register</i> gesendet. Der Request hat folgende Parameter (URL-Encoded):</p> <ul style="list-style-type: none"><li>• Name: "name"</li><li>• Wert: den gewünschten Benutzernamen.</li> <li>• Name: "firebase_instance_id"</li><li>• Wert: die Firebase Instance Id der installierten App-Instanz</li></ul> <p>Der Server antwortet darauf mit einem einer Firebase Json Nachricht mit dem Parameter namens "password_token", was das generierte Passwort des Benutzernamens enthält. Wenn der name bereits vergeben ist wird kein passwordtoken mitgeschickt.</p> <p>Bei jedem folgenden Start der App wird der Nutzer automatisch mit seinem gewählten namen angemeldet.</p>
Beteiligte Akteure	Spieler, Server
Vorbedingungen	
Nachbedingungen	Spieler ist angemeldet
Priorität	Mittel



### 2.3.1.3 - Gegenspieler Suchen

Name des Use-Case	Gegenspieler Suchen
ID	Client 1.3
Kurzbeschreibung	Spieler startet Suche nach einem Gegenspieler mit ähnlichem Fähigkeitsgrad
Beteiligte Akteure	Spieler, Server
Vorbedingungen	Spieler ist angemeldet
Nachbedingungen	Spiel-Ansicht mit laufendem Spiel ist geöffnet
Detaillierter Ablauf	<p>Sobald der Spieler die Gegnersuche startet wird eine Anforderung an den Server gestellt, einen Gegenspieler ähnlichen Fähigkeitsgrades zu Finden.</p> <p>Wenn ein Gegenspieler gefunden wurde startet das Spiel und wird geöffnet.</p> <p>Die Anforderung wird als POST Request an <code>*serverurl*/findGame</code> mit dem Parameter <code>password_token</code> gesendet. Das Password Token wurde bei der registrierung erhalten.</p> <p>Das gefundene Spiel wird über eine Firebase Message als Json Objekt empfangen. Das Json Objekt enthält folgende Schlüssel:</p> <ul style="list-style-type: none"><li>• <code>"kind"</code> - Enthält den string <code>"new_game"</code></li><li>• <code>"opponent_name"</code></li><li>• <code>"opponent_elo"</code></li><li>• <code>"is_opponent_white"</code> : boolean; Is der gegenspieler Weiß?</li><li>• <code>"game_id"</code></li></ul> <p>Die Gameld wird verwendet um dem Server zu sagen, für welches Spiel man einen Spielzug tätigen möchte.</p> <p>Hat der Spieler die Farbe Weiß, so kann er seinen Spielzug durchführen (Anwendungsfall: Client 2.1). Andernfalls kann er nur den aktuellen Zustand des Spiels sehen.</p>
Priorität	Hoch

### 2.3.1.4 - Spiel fortsetzen

Name des Use-Case	Spiel fortsetzen
ID	Client 1.4
Kurzbeschreibung	Der Spieler wählt eines seiner laufenden Spiele aus, das fortgesetzt wird.
Beteiligte Akteure	Spieler
Vorbedingungen	Spieler ist angemeldet, Spieler hat laufende Spiele
Nachbedingungen	Spiel-Ansicht mit laufendem Spiel ist geöffnet

Detaillierter Ablauf	<p>Nachdem ein laufendes Spiel ausgewählt wurde, wird das Spielfeld des Spiels in der Spiel-Ansicht angezeigt.</p> <p>Ist der Spieler am Spielzug kann der Spieler diesen Sofort tätigen. Andernfalls kann er nur den aktuellen Zustand des Spiels sehen.</p>
Priorität	Mittel

## 2.3.2 - Spielablauf

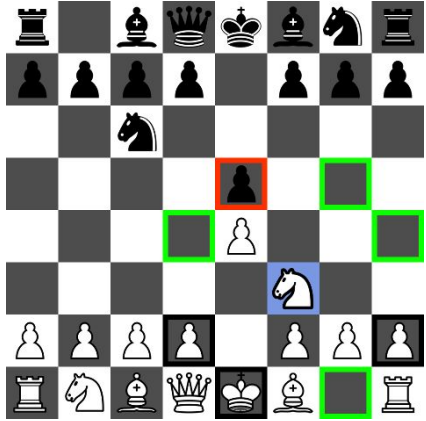
### 2.3.2.1 - Spielzug durchführen

Name des Use-Case	Spielzug durchführen
ID	Client 2.1
Kurzbeschreibung	Spieler führt seinen Spielzug durch
Beteiligte Akteure	Spieler
Vorbedingungen	Ein Spiel wurde gestartet
Nachbedingungen	
Detaillierter Ablauf	<p>Wenn die App nicht geöffnet ist und der Spielzug des Spielers beginnt wird dieser durch einen Ton und Push-Benachrichtigung darauf aufmerksam gemacht.</p> <p>Sein zug endet wenn er eine Figur bewegt hat (<i>Anwendungsfall: Client 2.3</i>). Der Spielzug ist nicht zeitlich begrenzt.</p> <p>Der Spielzug wird als Post-Request an <i>*serverurl*/move</i> gesendet. Der Server überprüft diesen Zug und sendet daraufhin dem Gegenspieler eine Push benachrichtigung das es nun am Zug ist.</p> <p>Der Post-Request enthält folgende Parameter (URL-Encoded):</p> <ul style="list-style-type: none"> <li>• "password_token" - Bei Registrierung erhalten</li> <li>• "game_id" - Bei Spielstart erhalten</li> <li>• "from" - Position von der Figur die bewegt werden soll (z.B. "e3")</li> <li>• "to" - Position auf die die Figur bewegt werden soll (z.B. "e4")</li> </ul> <p>Das Spiel endet wenn ein Spieler seinen Gegenspieler matt gesetzt hat.</p>
Alternativer ablauf	Das Spiel endet im unentschieden wenn ein Spieler patt gesetzt wurde.
Priorität	Hoch

### 2.3.2.2 - Remis beantragen

Name des Use-Case	Remis beantragen
ID	Client 2.2
Beschreibung	<p>Spieler beantragt das Spiel als unentschieden zu beenden. Wenn beide Spieler Remis beantragt haben, oder 50 Züge lang keine Figur geschlagen worden ist, kein Bauer bewegt worden ist und ein Spieler Remis beantragt hat endet das Spiel als unentschieden.</p> <p>Ein Remis wird als Post-Request an <code>*serverurl*/request_remis</code> beantragt.</p> <p>Die Parameter sind:</p> <ul style="list-style-type: none"><li>• <code>"password_token"</code> - Bei Registrierung erhalten</li><li>• <code>"game_id"</code> - Bei Spielstart erhalten</li></ul> <p>Informationen über Remis werden über eine Firebase Message als Json Objekt empfangen.</p> <p>Wenn der gegenspieler Remis beantragt, man selbst jedoch noch kein Remis beantragt hat wird folgendes Json Objekt empfangen:</p> <ul style="list-style-type: none"><li>• <code>"kind"</code> - Enthält den String <code>"remis_request"</code></li></ul> <p>Wenn der Gegenspieler Remis beantragt hat wird dies bei seinen Spielerinformationen angezeigt.</p> <p>Der Spieler kann durch einen Tap auf den Remis button, Remis anfordern.</p> <p>Wenn das Spiel durch ein Remis beendet wird, wird folgende Nachricht als Json Objekt empfangen</p> <ul style="list-style-type: none"><li>• <code>"kind"</code> - Enthält den String <code>"game_over"</code></li><li>• <code>"reason"</code> - Enthält einen der folgenden Strings<ul style="list-style-type: none"><li>○ <code>"remis_consent"</code> Beide spieler haben Remis beantragt</li><li>○ <code>"remis_stalemate"</code> Spielende durch Patt</li><li>○ <code>"50turns"</code> 50 Züge keine Figur geschlagen &amp; kein Bauer bewegt &amp; gegenspieler hat Remis beantragt</li><li>○ <code>"repition"</code> Die selbe Stellung des Spielfelds zum dritten mal und ein Spieler hat Remis beantragt</li><li>○ <code>"invalid_move"</code></li></ul></li></ul>
Beteiligte Akteure	Spieler
Vorbedingungen	
Nachbedingungen	
Priorität	Mittel

### 2.3.2.3 - Figur auswählen

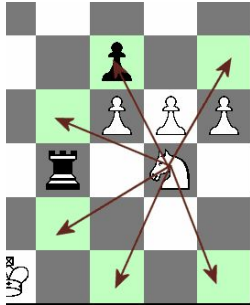
Name des Use-Case	Figur auswählen
ID	Client 2.3
Kurzbeschreibung	Spieler wählt eine seiner Figuren aus, die noch auf dem Spielfeld steht
Beteiligte Akteure	Spieler
Vorbedingungen	Spieler ist am Spielzug
Nachbedingungen	
Detaillierter Ablauf	<p>Wenn ein Spieler eine Figur auswählt, werden alle Felder hervorgehoben, auf die die ausgewählte Figur gehen kann. Steht eine gegnerische Figur auf einem dieser Felder wird dieses auf eine andere Weise hervorgehoben (z.B. andere Farbe). Das gleiche gilt für Felder, die von eigenen, anderen Figuren belegt sind.</p> <p>Eine Figur wird ausgewählt, indem auf sie getippt wird (Touchscreen). Wird getippt und gehalten, wird der Bereich um die Position des Tipps vergrößert dargestellt. Wenn der Tipp losgelassen wurde wird die Figur ausgewählt auf der er losgelassen wurde.</p>
Priorität	Hoch
Bild	

#### 2.3.2.4 - Figur bewegen

Name des Use-Case	Figur bewegen
ID	Client 2.4
Kurzbeschreibung	Spieler bewegt die ausgewählte Figur auf ein mögliches Feld
Beteiligte Akteure	Spieler
Vorbedingungen	Spieler ist am Spielzug und hat eine Figur ausgewählt, die Bewegungsmöglichkeiten hat
Nachbedingungen	Figur steht auf einem neuen Feld und der Spielzug wird beendet
Detaillierter Ablauf	<p>Felder auf die sich die ausgewählte Figur bewegen kann sind hervorgehoben. Der Spieler wählt ein Feld aus, auf das sich die Figur bewegen soll.</p> <p>Ist das Feld nicht durch eine eigene Figur belegt, bewegt sich die ausgewählte Figur auf das ausgewählte Feld, andernfalls passiert nichts.</p>
Alternativer Ablauf	Auf dem ausgewählten Feld befindet sich eine gegnerische Figur. Die ausgewählte Figur des Spielers bewegt sich auf das Feld und die gegnerische Figur wird aus dem Spiel entfernt.
Alternativer Ablauf	<p>Einmal pro Spiel kann jeder Spieler die Rochade durchführen. Dabei bewegt sich der König zwei felder auf einen seiner beiden Türme zu. Der Turm, auf den sich der König zubewegt hat springt jetzt über den König auf das Nachbarfeld des Königs. Die Rochade kann nur ausgeführt werden wenn</p> <ul style="list-style-type: none"> <li>• der König noch nicht gezogen wurde,</li> <li>• der beteiligte Turm noch nicht gezogen wurde,</li> <li>• zwischen dem König und dem beteiligten Turm keine andere Figur steht,</li> <li>• der König über kein Feld ziehen muss, das durch eine feindliche Figur bedroht wird,</li> <li>• der König vor und nach Ausführung der Rochade nicht im Schach steht.</li> </ul>
Priorität	Hoch
Bild	

## Erweiterungen: Figur bewegen

### 2.3.2.3.1 - Springer bewegen

Name des Use-Case	Springer bewegen
Beschreibung	Der Springer bewegt sich immer zwei Felder in eine Richtung und ein Feld in die andere Richtung. Er kann andere Figuren überspringen
Bild	

### 2.3.2.3.2 - Läufer bewegen

Name des Use-Case	Läufer bewegen
Kurzbeschreibung	Der Läufer kann sich nur diagonal in eine Richtung bewegen. Er kann sich so weit bewegen, bis er den Spielfeldrand erreicht.
Bild	

### 2.3.2.3.3 - Turm bewegen

Name des Use-Case	Turm bewegen
Kurzbeschreibung	Der Turm kann sich entweder Horizontal oder Vertikal in eine Richtung bewegen. Er kann sich so weit bewegen, bis er den Spielfeldrand erreicht.
Bild	

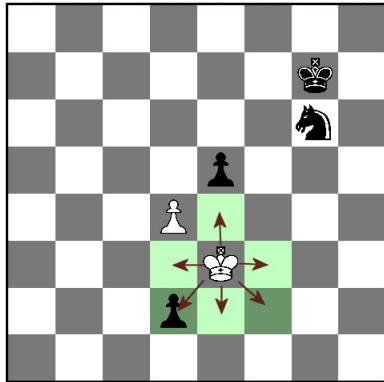
#### 2.3.2.3.4 - Dame bewegen

Name des Use-Case	Dame bewegen
Kurzbeschreibung	Die Dame kann sich Horizontal, Vertikal oder Diagonal bewegen. Sie kann sich so weit bewegen, bis sie den Spielfeldrand erreicht.
Bild	

#### 2.3.2.3.5 - Bauer bewegen

Name des Use-Case	Bauer bewegen
Beschreibung	<p>Im normalfall kann der Bauer sich pro Spielzug nur ein Feld nach vorne bewegen.</p> <p>Wenn er im Spiel noch nicht bewegt wurde, kann er 2 Felder nach vorne rücken.</p> <p>Der Bauer kann keine Figuren schlagen, die direkt vor ihm stehen. Er kann nur Figuren schlagen die vorne links, oder vorne rechts von ihm stehen. Wenn er eine Figur schlägt, rückt er auf die Position der geschlagenen Figur anstatt nach vorne zu rücken.</p>
Alternativer Ablauf	Wenn ein Bauer 2 Felder nach vorne rückt und sich danach direkt neben ihm ein gegnerischer Bauer befindet, so kann der gegnerische Bauer handeln als wäre der Bauer nur 1 Feld anstatt 2 Feldern nach vorne gerückt und ihn somit schlagen.
Alternativer Ablauf	Sollte ein Bauer die gegnerische seite des Spielfelds erreicht haben (achte Reihe für weiße Bauern, erste Reihe für schwarze Bauern), so wird dieser in eine andere Figur umgewandelt. Hierbei wird der Spieler aufgefordert sich zwischen Turm, Läufer, Springer und Dame zu entscheiden.

#### 2.3.2.3.6 - König bewegen

Name des Use-Case	König bewegen
Beschreibung	Der König kann sich ein Feld Horizontal, Vertikal oder Diagonal bewegen. Er kann sich auf kein Feld bewegen, in dem er von einer gegnerischen Figur geschlagen werden könnte.
Bild	



#### 2.3.2.4 - Spielzustand aktualisieren

Name des Use-Case	Spielzustand aktualisieren
ID	Client 2.4
Beschreibung	<p>Nach einem Spielzug des Gegenspielers wird der Zustand des Spiels vom Server aktualisiert.</p> <p>Die Spielzustand aktualisierung wird von als Firebase Message empfangen. Sie enthält ein Json Objekt mit folgenden Parametern:</p> <ul style="list-style-type: none"><li>• "kind" - Enthält den String "opponent_turn"</li><li>• "game_id"</li><li>• "from"</li><li>• "to"</li></ul>
Beteiligte Akteure	Server
Vorbedingungen	
Nachbedingungen	
Priorität	Hoch

## **2.4 - Zusätzliche Anforderungen**

### **2.4.1 - Remis**

Schachpartien können unentschieden enden. Dieses sogenannte Remis tritt ein, wenn:

- Ein Spieler patt gesetzt wurde (Anwendungsfall Client 2.1)
- Beide Spieler sich auf das unentschieden einigen.
- Zum dritten Mal die gleiche Aufstellung der Figuren auftritt und der gleiche Spieler am Spielzug ist.
- 50 Züge lang keine Figur geschlagen worden ist und kein Bauer sich bewegt hat und ein Spieler remis beantragt hat.

### **2.4.2 - Sicherheit**

Alle Nutzerdaten müssen vor unbefugtem Zugriff geschützt sein. Es werden nur für das ausführen des Spiels relevante Daten und anonymisierte Fehlermeldungen an den Server übermittelt.

### **2.4.3 - Fehleraufzeichnung**

Sollte beim Ausführen der ChessTime App ein Fehler auftreten, wird die Fehlermeldung, die Stelle, an der der Fehler auftrat, das verwendete Gerät und die verwendete Android-Version an den Server übermittelt um aufgezeichnet zu werden.

### **2.4.4 - Benutzbarkeit**

Das System soll, für Spieler die mit den Spielregeln von Schach vertraut sind, intuitiv nutzbar sein. Die Benutzeroberfläche soll simpel und übersichtlich sein.

### **2.4.5 - Performance**

Die Performance-Anforderungen beziehen sich auf das Gerät *Google Nexus 5* mit der Android-Version 6.0.1.

Die App soll im durchschnitt mit weniger als 5 übersprungenen Frames pro Sekunde ausgeführt werden.

Das Starten der App darf 10s nicht überschreiten.

Die Reaktionszeit auf Benutzereingaben muss unter 0,5 Sekunden liegen.

### **2.4.6 - Verwendung von Technologien**

Die App wird mit Java 1.8 im zusammenhang mit dem Android-SDK umgesetzt.

### **2.4.7 - Platform**

Das System wird als Android-App mit minSdkVersion 21 (entspricht Android 5.0) und targetSdkVersion 24 (entspricht Android 7.0) entwickelt. Geräte unter Api-Level 20 werden nicht unterstützt.

### **2.4.8 - Abgabedatum**

Das System muss bis zum 8. Oktober 2018 fertiggestellt sein.

### **2.4.9 - Hardwareanforderungen**

- Android Gerät
- Netzwerkverbindung möglich
- Leistungs schwächstes Testgerät: *Google Nexus 5*

## 2.5 - User Interface Prototype

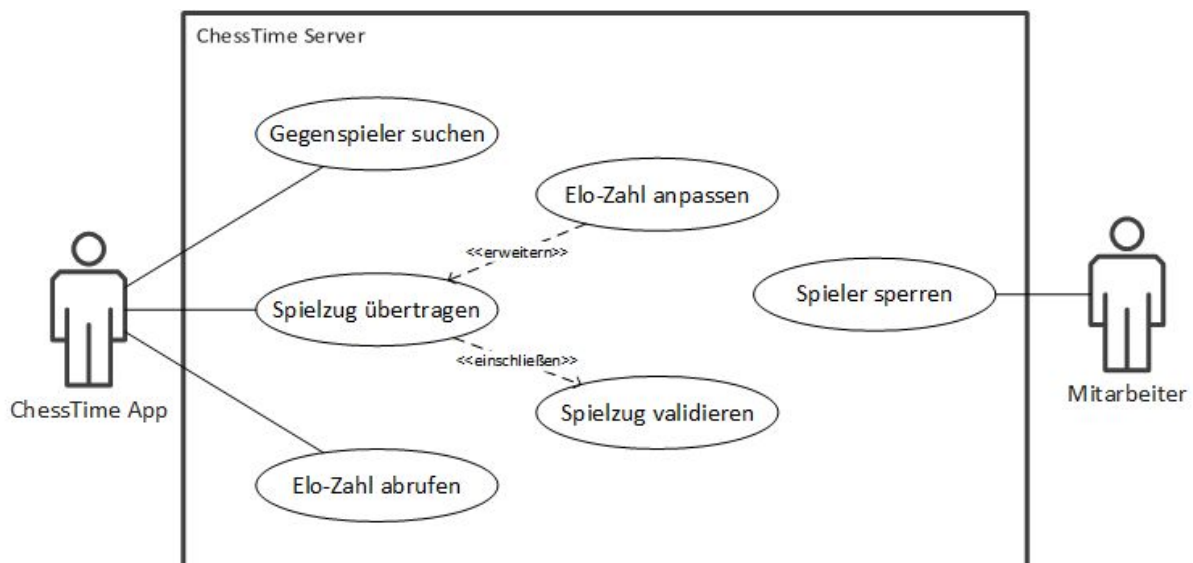
1 - Hauptmenü	2 - Spiel-Ansicht	3 - Navigations-Seitenleiste
 <p>The main menu features a dark grey background with the 'ChessTime' logo at the top. Below the logo are two buttons: 'Gegner Suchen' and 'Spielregeln Anzeigen'. A section titled 'Deine Partien' lists two ongoing games: 'ChessPro' (Elo: 2234, 2 Stunden, 18 Minuten) and 'User1701' (Elo: 2011, 8 Stunden, 7 Minuten).</p>		 <p>The navigation side bar is a vertical overlay on the right side of the screen. It contains a back arrow at the top, followed by the user's nickname 'MyNickname13' and Elo '2121'. Below this are two buttons: 'Spielregeln anzeigen' and 'Neues Spiel suchen'. A section titled 'Deine Partien' lists two ongoing games: 'ChessPro - Aktuelle Partie' (Elo: 2234, 2 Stunden, 18 Minuten) and 'User1701' (Elo: 2011, 8 Stunden, 7 Minuten). The side bar is positioned over a chessboard background.</p>
Setzt alle Anwendungsfälle aus der Spielvorbereitung um. Für jedes laufende Spiel wird der Name, die Elo-Zahl und die Spielzeit angezeigt.	Setzt alle Anwendungsfälle aus dem Spielablauf um. Für jeden Spieler wird der Name, Elo-Zahl und gesamte Spielzeit angezeigt.	Die Navigations-Seitenleiste hat dieselben Funktionen wie das Hauptmenü.
4 - Spielregeln-Dialog		
 <p>The rules dialog is a white box with a dark header containing the title 'Spielregeln' and a close button. The body contains placeholder text 'Lorem ipsum' and 'At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren.' The dialog is positioned over a chessboard background.</p>	 <p>The rules dialog is a white box with a dark header containing the title 'Spielregeln' and a close button. The body contains placeholder text 'Lorem ipsum' and 'At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren.' The dialog is positioned over a chessboard background.</p>	

### 3 - Requirements aus Server Perspektive

#### 3.1 - Flow of Events

- Der Server erhält eine Anfrage eines Clients ihm einen Gegenspieler zu vermitteln
- Server vergleicht Elo-Bewertungen mit anderen Spielern, die ein Spiel suchen
- Wenn ein anderer Spieler mit ähnlicher Elo-Zahl gefunden wurde, beginnt das Spiel, ansonsten wird die Spanne in der gesucht wird geweitet bis ein Gegenspieler gefunden wird.
- Sobald ein Gegenspieler gefunden wurde wird ein Spiel in der Datenbank angelegt welche mit dem Server verknüpft ist.
- Immer wenn ein Spieler eine Figur bewegt, wird dies über den Server an den Gegenspieler weitergeleitet. Der Server überprüft dabei auch ob der Zug valide ist. Der Gegenspieler erhält dann eine Benachrichtigung über Firebase das er nun am Zug ist.
- Am ende des Spiels wird für beide Spieler die neue Elo-Zahl berechnet und in der Datenbank gespeichert.

#### 3.2 - Use-Case Modell



#### 3.3 - Anwendungsfall Beschreibungen

##### 3.3.1 - Gegenspieler Suchen

Name des Use-Case	Gegenspieler Suchen
ID	Server 1.1
Kurzbeschreibung	Server bekommt eine Anfrage eines Clients ihm einen Gegenspieler zu vermitteln
Beteiligte Akteure	Server, Client
Vorbedingungen	Client und Server sind über ein Netzwerk verbunden

Nachbedingungen	Ein Spiel zwischen zwei Clients wird gestartet
Detaillierter Ablauf	<p>Ein Client, der ein Spiel starten möchte, teilt dies dem Server mit. Der Server sucht unter den anderen Clients, die ein Spiel starten möchten nach einem Gegenspieler mit ähnlichen Fähigkeitsgrad. Der Fähigkeitsgrad eines Spielers wird durch seine Elo-Zahl bestimmt (Siehe Use-Case 2)</p> <p>Wenn kein Gegenspieler mit ähnlicher Elo-Zahl gefunden wurde, wird die höchstmögliche Elo-Zahl-Differenz erhöht.</p> <p>Die maximale Elo-Zahl-Differenz ist erreicht, wenn die Gewinnwahrscheinlichkeit 25% unterschreitet bzw. 75% überschreitet, was einer Elo-Zahl-Differenz von ca. 200 entspricht. Wenn dies eintritt, wird erst dann ein Spiel begonnen, wenn ein Spieler, dessen Elo-Zahl-Differenz mit dem suchenden Spieler unter die maximale Elo-Zahl-Differenz fällt.</p> <p>Sobald ein Gegenspieler gefunden wurde wird beiden Spielern mitgeteilt welche Farbe sie haben (zufällig zugeteilt) und wer ihr gegenspieler ist mit seiner Elo-Zahl.</p> <p>Die Anforderung wird von Client als POST Request an <code>*serverurl*/findGame</code> mit dem Parameter <code>password_token</code> gesendet.</p> <p>Das gefundene Spiel wird über eine Firebase Message als Json Objekt empfangen. Das Json Objekt enthält folgende Schlüssel:</p> <ul style="list-style-type: none"> <li>• <code>"kind"</code> - Enthält den string <code>"new_game"</code></li> <li>• <code>"opponent_name"</code></li> <li>• <code>"opponent_elo"</code></li> <li>• <code>"is_opponent_white"</code> : boolean; Is der gegenspieler Weiß?</li> <li>• <code>"game_id"</code></li> </ul> <p>Die Gameld wird verwendet um dem Server zu sagen, für welches Spiel man einen Spielzug tätigen möchte.</p>
Priorität	Hoch

### 3.3.2 - Elo-Zahl abrufen

Name des Use-Case	Elo-Zahl abrufen
ID	Server 1.2
Beschreibung	Der Server überträgt einem Client seine Elo-Zahl
Beteiligte Akteure	Server, Client
Ablauf	Der Client sendet eine HTTP-GET Anforderung an <code>*serverurl*/getElo?name=*name*</code> . Der Parameter <code>"name"</code> enthält

	den Namen des Spielers, dessen Elo abgefragt wird. Die Antwort enthält den parameter "elo", der die Elo-Zahl enthält.
Vorbedingungen	
Nachbedingungen	
Priorität	Niedrig

### 3.3.3 - Elo-Zahl anpassen

Name des Use-Case	Elo-Zahl anpassen
ID	Server 1.3
Kurzbeschreibung	Nach einem Spiel wird die Elo-Zahl beider Spieler, abhängig vom Ergebnis und Fähigkeitsgrad unterschied, angepasst.
Beteiligte Akteure	Server
Vorbedingungen	Ein Spiel zwischen zwei Clients ist zu einem Ergebnis gekommen
Nachbedingungen	Elo-Zahlen beider Clients wurden angepasst
Detaillierter Ablauf	<p>Um die Elo-Zahl zu berechnen muss erst der Erwartungswert berechnet werden. Dieser gibt die erwartete Punktzahl, unter Berücksichtigung der Elo-Zahlen beider Spieler, an. Hierbei entspricht ein Sieg einem Punkt, ein Unentschieden einem halben Punkt und eine Niederlage keinem Punkt.</p> <p>Der Erwartungswert berechnet sich folgendermaßen:  <math display="block">E_A = 1 / (1 + 10^{(R_B - R_A)/400})</math> - <math>E_A</math> : Erwarteter Punktestand für Spieler A  - <math>R_A</math> : Elo-Zahl von Spieler A  - <math>R_B</math> : Elo-Zahl von Spieler B</p> <p>Mit dem Erwartungswert kann nun nach dem Spiel die angepasste Elo-Zahl errechnet werden.  <math display="block">R'_A = R_A + k \cdot (S_A - E_A)</math> - <math>R'_A</math> : Angepasste Elo-Zahl  - <math>E_A</math> : Erwarteter Punktestand für Spieler A  - <math>S_A</math> : Tatsächlich gespielter Punktestand von Spieler A  - <math>R_A</math> : Bisherige Elo-Zahl von Spieler A  - <math>k</math> : Faktor der bestimmt wie stark sich ein Spiel auf die Gesamtwertung auswirkt.</p> <p>In ChessTime wird die Elo-Zahl mit <math>k = 32</math> berechnet.</p>
Priorität	Mittel

### 3.3.4 - Spielzug weiterleiten

Name des Use-Case	Spielzug übertragen
ID	Server 1.4
Beschreibung	Der Spielzug eines Spieler wird an seinen Gegenspieler weitergeleitet
Beteiligte Akteure	Server, Client
Vorbedingungen	Der Spielzug eines Clients, der sich in einem Spiel befindet, wurde empfangen
Nachbedingungen	
Detaillierter Ablauf	<p>Der Server empfängt den durchgeführten Spielzug eines Spielers und leitet diesen an seinen Gegenspieler weiter.</p> <p>Bevor der Spielzug weitergeleitet wird, wird er validiert (Anwendungsfall Server 1.4)</p>
Priorität	Hoch

### 3.3.5 - Spielzug validieren

Name des Use-Case	Spielzug validieren
ID	Server 1.5
Kurzbeschreibung	Server überprüft den empfangenen Spielzug eines Spielers auf Regelkonformität
Beteiligte Akteure	Server
Vorbedingungen	Der Spielzug eines Clients, der sich in einem Spiel befindet, wurde empfangen
Nachbedingungen	
Detaillierter Ablauf	<p>Der empfangene Spielzug wird auf Regelkonformität geprüft.</p> <p>Sollte der Spielzug nicht Regelkonform sein, wird das Spiel aufgrund von verdacht auf Modifikation des Clients (der App) vorzeitig beendet und das Spiel wird nicht gewertet.</p> <p>Dazu wird über Firebase ein Json String gesendet. Dieser enthält folgende attribute:</p> <ul style="list-style-type: none"><li>• kind - enthält den string "game_over"</li><li>• reason - enthält den string "invalid_move"</li></ul>
Priorität	Niedrig

### 3.3.6 - Spieler registrieren

Name des Use-Case	Spieler registrieren
ID	Server 1.6
Kurzbeschreibung	Spieler teilt dem Server seinen gewünschten Nutzernamen mit und bekommt, falls dieser Nicht vergeben ist, ein Identifikationstoken (passwordtoken) vom Server zugewiesen
Beteiligte Akteure	Server
Detaillierter Ablauf	<p>Zum anmelden wird von Client ein HTTP-POST request an *serverurl*/register gesendet. Der Request hat folgende Parameter (URL-Encoded):</p> <ul style="list-style-type: none"><li>• Name: "name"</li><li>• Wert: den gewünschten Benutzernamen.</li> <li>• Name: "firebase_instance_id"</li><li>• Wert: die Firebase Instance Id der installierten App-Instanz</li></ul> <p>Der Server antwortet darauf mit einem einer Firebase Json Nachricht mit dem Parameter namens "password_token", was das generierte Passwort des Benutzernamens enthält. Wenn der name bereits vergeben ist wird kein passwordtoken mitgeschickt.</p>
Priorität	Niedrig



### 3.3.7 - Remis beantragen

Name des Use-Case	Remis beantragen
ID	Server 1.7
Kurzbeschreibung	Spieler beantragt ein unentschieden
Beteiligte Akteure	Server
Vorbedingungen	Spieler hat mindestens ein laufendes Spiel, für das er Remis beantragen kann.
Nachbedingungen	
Detaillierter Ablauf	<p>Wenn beide Spieler remis beantragt haben, oder eine besondere bedingung eintritt, wird das Spiel vorzeitig beendet.</p> <p>Ein Remis wird vom Client als Post-Request an <code>*serverurl*/request_remis</code> beantragt.</p> <p>Die Parameter sind:</p> <ul style="list-style-type: none"><li>• <code>"password_token"</code> - Bei Registrierung erhalten</li><li>• <code>"game_id"</code> - Bei Spielstart erhalten</li></ul> <p>Wenn der gegenspieler Remis beantragt, man selbst jedoch noch kein Remis beantragt hat wird folgendes Json Objekt empfangen:</p> <ul style="list-style-type: none"><li>• <code>"kind"</code> - Enthält den String <code>"remis_request"</code></li></ul> <p>Wenn das Spiel durch ein Remis beendet wird, wird folgende Nachricht als Json Objekt empfangen</p> <ul style="list-style-type: none"><li>• <code>"kind"</code> - Enthält den String <code>"game_over"</code></li><li>• <code>"reason"</code> - Enthält einen der folgenden Strings<ul style="list-style-type: none"><li>○ <code>"remis_consent"</code> Beide spieler haben Remis beantragt</li><li>○ <code>"remis_stalemate"</code> Spielende durch Patt</li><li>○ <code>"50turns"</code> 50 Züge keine Figur geschlagen &amp; kein Bauer bewegt &amp; gegenspieler hat Remis beantragt</li><li>○ <code>"repetition"</code> Die selbe Stellung des Spielfelds zum dritten mal und ein Spieler hat Remis beantragt</li><li>○ <code>"invalid_move"</code></li></ul></li></ul>
Priorität	Niedrig

### **3.4 Zusätzliche Anforderungen**

#### **3.4.1 - Sicherheit**

Alle Nutzerdaten müssen vor unbefugtem Zugriff geschützt sein. Die von den Clients empfangenen Daten müssen desinfiziert werden.

#### **3.4.2 - Fehleraufzeichnung**

Sollte beim Ausführen des ChessTime Servers ein Fehler auftreten, wird die Fehlermeldung lokal aufgezeichnet und der Server wird neu gestartet.

#### **3.4.3 - Performance**

Der Server muss bis zu 1000 gleichzeitige Nutzer unterstützen.

#### **3.4.4 - Verfügbarkeit**

Das System muss 24 Stunden pro Tag und 7 Tage die Woche, mit einer maximalen downtime von 5%, verfügbar sein.

#### **3.4.5 - Datenpersistenz**

Das System speichert alle notwendigen Daten in eine relationale Datenbank.

#### **3.4.6 - Verwendung von Technologien**

Der Server wird mit Java 1.8 umgesetzt.

#### **3.4.7 - Plattform**

Das System wird als Systemunabhängige Java-Konsolenanwendung programmiert.

#### **3.4.8 - Abgabedatum**

Das System muss bis zum 8. Oktober 2018 fertiggestellt sein.

#### **3.4.9 - Hardwareanforderungen**

- Netzwerkverbindung möglich
- mind. 2 GB freier festplattenspeicher
- mind. 2 GB Arbeitsspeicher