Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 22/10/2025 | 0.9 | Initial Document | M. |
| 24/10/2025 | 1.0 | Added content and diagram pictures | A., J., M. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, Acronyms, and Abbreviations 4

1.4 References 4

1.5 Overview 4

2. Flow of Events—Design 4

2.1 Sequenzdiagramm-basierter Ablauf 4

2.2 Aktivitätsdiagramm-basierter Ablauf 5

2.2.1 Hauptablauf 5

2.2.2 Alternative Abläufe 5

2.2.3 Komponenteninteraktion 5

3. Derived Requirements 5

3.1 Functional Requirements 5

3.2 Non-Functional Requirements 5

3.3 Implementation Requirements 5

# Introduction

## Purpose

Dieses Dokument beschreibt die Realisierung des "Login" Use-Cases für *das Spiel* ***Hexfields: Dominion***.

## Scope

Dieses UCRS deckt den Login-Prozess ab und ist Teil der Überkategorie "Account Management". Verwandte Use-Cases umfassen "Login as Guest", "Logout", "Password-Reset" und "Registration".

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Begriff** | **Erklärung** |
| User/Spieler | Der Benutzer des Systems |
| Client | Die clientseitige Anwendung (Web-Browser) |
| Server | Der Anwendungsserver |
| Datenbank | Die persistente Datenspeicherung auf externer Datenbank |

## References

* **Use-Case Spezifikation**: Login (/srs/account\_management/login/login.md)
* **Sequenzdiagramm**: Login (/srs/account\_management/login/login\_sequence.txt)
* **Aktivitätsdiagramm**: Login (eingebettet in login.md, s.o.)

## Overview

Dieses Dokument beschreibt im Abschnitt 2 den Entwurfsablauf des Login-Prozesses anhand von Sequenz- und Aktivitätsdiagrammen. Abschnitt 3 listet abgeleitete Anforderungen auf. In Abschnitt 4 befinden sich Kopien der referenzierten Diagramme.

# Flow of Events—Design

Der Login-Use-Case wird durch die Zusammenarbeit mehrerer Komponenten realisiert: Spieler (Benutzer), Client, Server und Datenbank.

## Sequenzdiagramm-basierter Ablauf

Das Sequenzdiagramm zeigt die zeitliche Abfolge der Interaktionen zwischen den Systemkomponenten:

1. Der Spieler gibt seine Anmeldedaten im Client ein und betätigt die "Log in" Schaltfläche
2. Der Client leitet die Anmeldedaten an den Server weiter
3. Der Server übermittelt die Anmeldedaten zur Überprüfung an die Datenbank
4. Die Datenbank prüft die Anmeldedaten und bestätigt die Anmeldung
5. Der Server bestätigt die erfolgreiche Anmeldung an den Client
6. Der Client speichert die Anmeldedaten lokal und zeigt die Bestätigung an

## Aktivitätsdiagramm-basierter Ablauf

Das Aktivitätsdiagramm beschreibt den detaillierten Workflow mit Haupt- und Alternativpfaden:

### Hauptablauf

* Start 🡪 Anmeldeseite öffnen 🡪 Anmeldedaten eingeben 🡪 Login-Button klicken
* Anmeldedaten validieren 🡪 Server erreichbar? 🡪 Anmeldedaten an Server senden
* Anmeldedaten korrekt? 🡪 Anmeldung bestätigen 🡪 Anmeldedaten lokal speichern
* Zur Startseite navigieren 🡪 Erfolgreich angemeldet

### Alternative Abläufe

* **Server nicht erreichbar**: Fehlermeldung anzeigen, Rückkehr zur Dateneingabe
* **Falsche Anmeldedaten**: Fehlermeldung anzeigen, Rückkehr zur Dateneingabe
* **Konto gesperrt**: Spezifische Fehlermeldung anzeigen, Rückkehr zur Dateneingabe

### Komponenteninteraktion

Die Realisierung nutzt eine klare Trennung der Verantwortlichkeiten:

* **Client**: Benutzerinteraktion, lokale Datenspeicherung, UI-Feedback
* **Server**: Authentifizierungslogik, Kommunikation mit Datenbank
* **Datenbank**: Persistente Speicherung und Validierung der Benutzerdaten

# Derived Requirements

## Functional Requirements

* **Sicherheit**: Anmeldedaten dürfen nicht unverschlüsselt übertragen werden
* **Performance**: Die Authentifizierung soll innerhalb von 5 Sekunden abgeschlossen sein
* **Verfügbarkeit**: Der Login-Service muss eine Verfügbarkeit von 99% während Stoßzeiten (8-22 Uhr) gewährleisten
* **Benutzerfreundlichkeit**: Fehlermeldungen müssen klar und hilfreich formuliert sein

## Non-Functional Requirements

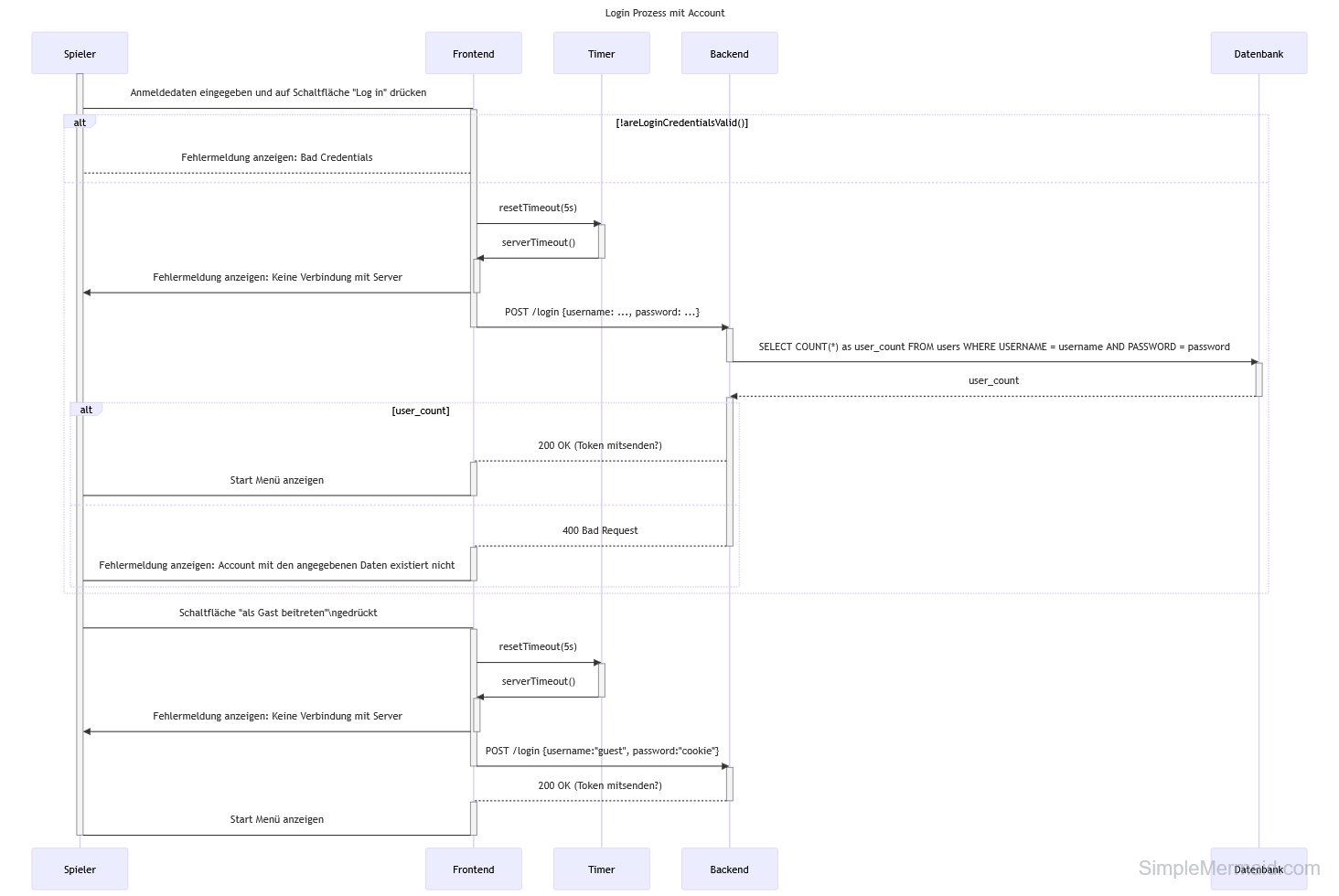
* **Sichere Datenspeicherung**: Anmeldedaten müssen sicher lokal gespeichert werden
* **Session-Management**: Nach erfolgreichem Login muss eine Benutzersession etabliert werden
* **Fehlerbehandlung**: Robustes Fehlerhandling für Netzwerk- und Serverprobleme
* **Validierung**: Serverseitige Validierung der Eingabedaten (zur Vermeidung von SQLi o.Ä.)

## Implementation Requirements

* **Kompatibilität**: Unterstützung verschiedener Browser und Endgeräte
* **Einfache Bedienung**: Passwort-Manager sollten mit den Eingabefeldern kompatibel sein

# Appendix

## Sequenzdiagramm Login



## Aktivitätsdiagramm Login

