Helvetia Versicherungen

Dokumentation Modul 223

Von Simon Fäs

IN22 Fäs Simon
13.5.2024

Inhaltsverzeichnis

Einleitung

In dieser Dokumentation geht es ums Planen, Design, Implementieren und Testen von den Erweiterungen an den Applikationen aus den Modulen 295, 335 und 294.

Analyse

Zuerst musste der Auftrag analysiert werden. Das wurde in einem UML-Use-Case Diagramm festgehalten.
Im folgenden Bild sind die Use Cases für die Benutzergruppen Mitarbeiter aufgezeigt.

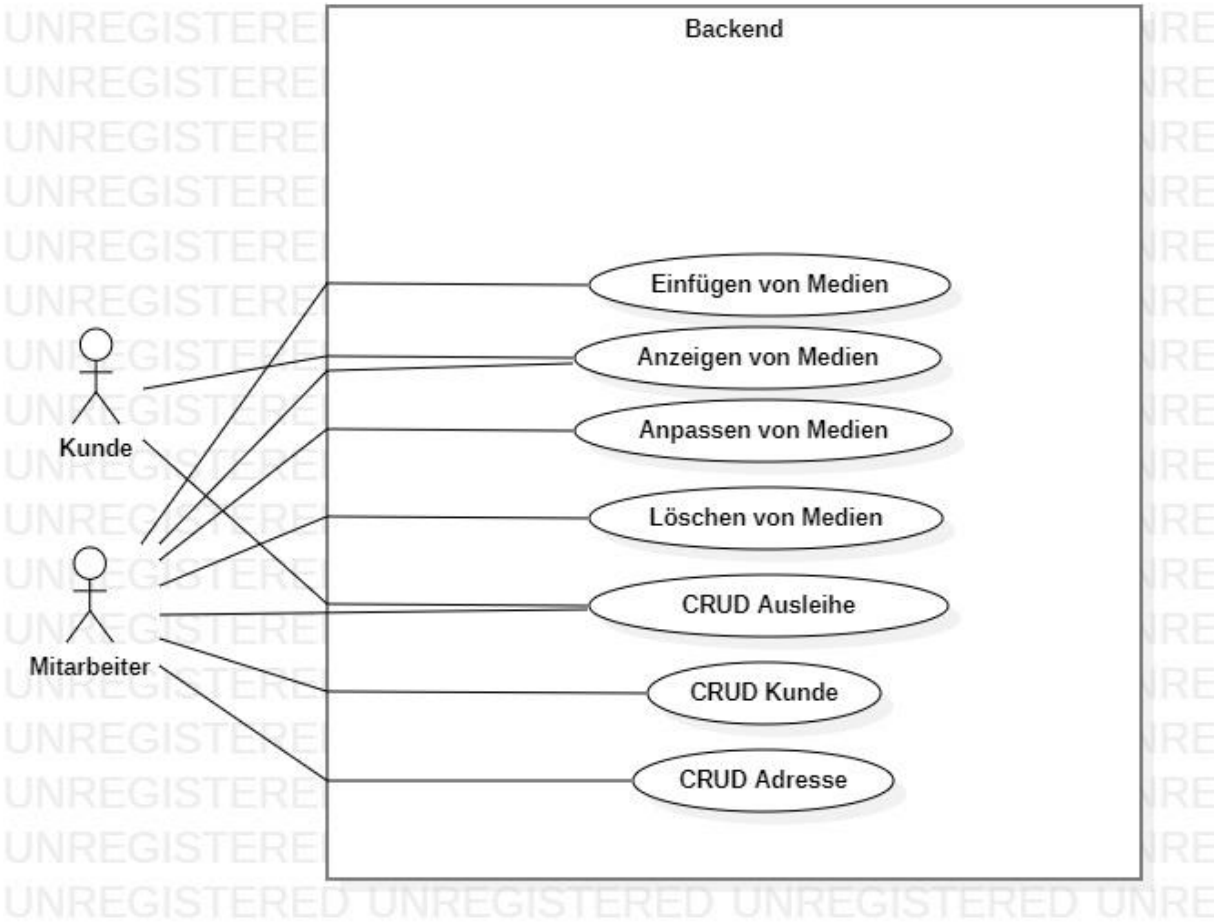

Abbildung 1: Use Case Diagramm

Design

Im folgenden Kapitel ist dokumentiert, wie die Schnittstellen designt sind, bzw. wie sie dann angesprochen werden können. Die Schnittstellen sind wieder nach Ressourcen unterteilt. Ausserdem sind die einzelnen Klassen mit Abhängigkeiten und Verantwortlichkeiten beschrieben.

Ressource «Medium»

In der folgenden Tabelle sind alle Schnittstellen der Ressource «Medium» aufgelistet.

HTTP-Methode	URI	Anforderung	Rollen
GET	/bibliothek/medium/9	Liefert das Medium mit ID 9 zurück	ROLE_USER, ROLE_ADMIN
GET	/bibliothek/medium/titel/Beispiel%20Buch	Liefert Medien mit dem Titel «Beispiel Buch» zurück	ROLE_USER, ROLE_ADMIN
GET	/bibliothek/medium	Können alle ausgegeben werden	ROLE_USER, ROLE_ADMIN
POST	/bibliothek/medium	Fügt das im Request-Body mitgegebene JSON-Objekt in die DB ein. Beispiel JSON-Objekt: { "titel" : "Beispiel", "autor" : "Max Muster" }	ROLE_ADMIN
PUT	/bibliothek/medium/9	Ändert in diesem Fall Medium mit der ID 9 entsprechend dem JSON-Objekt, welches im Request-Body mitgegeben wird. Ist das Medium mit angegebener ID nicht vorhanden, wird ein neues erstellt. Beispiel JSON-Objekt: { «titel» : «Beispiel», „autor“ : „Max Muster“, „genre“ : „Action“, „altersFreigabe“ : 0, „ISBN/EAN“ : „978-3-8668-0192-9“, „standortCode“ : „A23“ }	ROLE_ADMIN
DELETE	/bibliothek/medium/9	Löscht Medium mit der ID=9	ROLE_ADMIN

Ressource «Kunde» inklusive «Adresse»

In der folgenden Tabelle sind alle Schnittstellen der Ressource «Kunde» und der Ressource «Adresse» aufgelistet.
Für die Ressource «Adresse» ist nicht vorgesehen diese ändern zu können, da eine Adresse von mehreren Kunden unabhängig voneinander bewohnt werden kann. Im Falle eines Umzugs wird automatisch die Adressänderung vorgenommen.

HTTP-Methode	URI	Anforderung	Rollen
GET	/bibliothek/kunde/9	Liefert Kunde mit der ID=9	ROLE_ADMIN
GET	/bibliothek/kunde/nachname/Muster	Liefert alle Kunden mit dem Namen Muster	ROLE_ADMIN
GET	/bibliothek/kunde/adresse/Baslerstrasse	Liefert alle Kunden als JSON-Array, die in der Adresse den String «Baslerstrasse» enthalten	ROLE_ADMIN
GET	/bibliothek/adresse/postleitzahl/4313	Liefert alle Adressen mit der Postleitzahl=4313	ROLE_ADMIN
GET	/bibliothek/adresse/strasse/Baslerstrasse	Liefert alle Adressen mit der Strasse=«Baslerstrasse»	ROLE_ADMIN
GET	/bibliothek/adresse	Liefert alle Adressen	ROLE_ADMIN
POST	/bibliothek/kunde	Fügt das im Request-Body mitgegebene JSON-Objekt in die DB ein. Beispiel JSON-Objekt: { "strasse" : "Baslerstrasse", "ort" : "Möhlin", "postleitzahl" : 4313, "vorname" : "Max", "nachname" : "Muster", "geburtsdatum" : "2003-	ROLE_ADMIN

		03-09", "email" : "max.muster@gmail.com" }	
PUT	/bibliothek/kunde/9	Ändert in diesem Fall Medium mit der ID 9 entsprechend dem JSON-Objekt, welches im Request-Body mitgegeben wird. Ist das Medium mit angegebener ID nicht vorhanden, wird ein neues erstellt. Beispiel JSON-Objekt: { "strasse" : "Baslerstrasse", "ort" : "Möhlin", "postleitzahl" : 4313, "email" : "max.muster@gmail.com" }	ROLE_ADMIN
DELETE	/bibliothek/kunde/9	Löscht Objekt mit der ID=9	ROLE_ADMIN
DELETE	/bibliothek/adresse/9	Löscht Adresse mit der ID=9, insofern diese nicht mehr referenziert wird	ROLE_ADMIN

Ressource «Ausleihe»

In der folgenden Tabelle sind alle Schnittstellen der Ressource «Ausleihe» aufgelistet.

HTTP-Methode	URI	Anforderung	Rollen
GET	/bibliothek/ausleihe/9	Liefert die Ausleihe mit der Medien-ID 9 zurück	ROLE_USER, ROLE_ADMIN
POST	/bibliothek/ausleihe	Fügt das im Request-Body mitgegebene JSON-Objekt in die DB ein. Beispiel JSON-Objekt:	ROLE_USER, ROLE_ADMIN

		{"kunde" : {"id" : 9}, "medium" : {"id": 9}}	
PUT	/bibliothek/ausleihe/9	Ändert in diesem Fall das Rückgabedatum der Ausleihe mit der Medien-ID=9. Wenn das Objekt mit dieser ID nicht existiert, dann wird dieses dem mitgelieferten JSON-Objekt entsprechend erstellt Beispiel JSON-Objekt: {"ausleihedauer " : "28" }	ROLE_USER, ROLE_ADMIN
DELETE	/bibliothek/ausleihe/9	Löscht die Ausleihe mit der Medien-ID=9	ROLE_USER, ROLE_ADMIN

Hinzugefügte Klassen (Backend)

Im folgenden Kapitel sind alle Klassen mit Verantwortlichkeiten, Abhängigkeiten (Kollaborateure) aufgeführt. Die aufgeführten Klassen sind ausschliesslich Klassen, welche im Backend implementiert wurden.

Klasse	Verantwortlichkeiten	Kollaborateure
User	• Entity für Berechtigungen	
UserRepository	• Interface, um Entities von Server zu holen	User resp. Kunde JpaRepository
BibliothekUserDetails	• Wird von Spring Security benötigt • Lädt den User mit dem gegebenen Namen aus der Datenbank	User resp. Kunde
BibliothekUserDetailsService	• Wird von Spring Security benötigt • Lädt den User mit dem gegebenen Namen aus der Datenbank	User resp. Kunde BibliothekUserDetails UserRepository
PasswordEncoderDummy	• Wird von Spring Security benötigt	PasswordEncoder

	• Wandelt Passwort in einen Hash um (Nicht in unserem Fall) • Überprüft Passwörter	
SecurityConfig	• Wird von Spring Security benötigt • Bildet den Spring-Kontext für Authentifizierung • Bildet den Spring-Kontext für Autorisierung	BibliothekUserDetailsService PasswordEncoder AuthenticationManager SecurityFilterChain

Hinzugefügte Klassen (Android App)

Im folgenden Kapitel sind alle Klassen mit Verantwortlichkeiten, Abhängigkeiten (Kollaborateure) aufgeführt. Die aufgeführten Klassen sind ausschliesslich Klassen, welche im Backend implementiert wurden.

Klasse/Interface	Verantwortlichkeiten	Kollaborateure
AsyncBaseListener (Interface)	• Beschreibt Callback Methoden, um Fehler asynchron zu behandeln.	
AsyncLoginListener (Interface)	• Beschreibt eine Callback Methode, um erfolgreiches Einloggen asynchron zu behandeln. • Erweitertet AsyncBaseListener	AsyncBaseListener
LoginActivity	• Verantwortlich, um Login-Oberfläche darzustellen • Wird gebraucht, um Credentials vom User abzufragen. • Benötigt RestCredentialTester • Implementiert AsyncLoginListener	AsyncLoginListener RestCredentialTester
RestCredentialTester	• Klasse, um die Logindaten zu testen. Braucht Listener mit implementierten Methoden von AsyncLoginListener	AsyncLoginListener

SingletonAppData	• Singleton-Klasse, um Infos in der App viewübergreifend zu speichern.	

Transaktionen und Sperren von Daten

Transaktionen und das Sperren von Daten ist ein wichtiger Teil, um die Integrität der Daten sicherzustellen.

Transaktion

Die richtige Isolation von Transaktionen ist wichtig, weil es drei Phänomene gibt, die auftreten können, wenn das Isolierungslevel nicht genug ist. Voraussetzung für diese Phänomene sind, dass Autocommit ausgeschaltet ist.

Die Phänomene sind:

- Dirty Read (Uncommited Daten einer anderen Transaktion können bereits gelesen werden)
- Non-Repeatable-Read (Non-Repeatable Reads liegen vor, wenn Ihre Transaktion bestätigte UPDATES einer anderen Transaktion liest. Dieselbe Zeile hat jetzt andere Werte als zu Beginn Ihrer Transaktion)
- Phantom Read (Phantom-Reads sind Non-Repeatable Reads ähnlich, aber beim Lesen von bestätigten INSERTS und/oder DELETES aus einer anderen Transaktion. Es gibt neue Zeilen oder Zeilen, die verschwunden sind, seit Sie die Transaktion begonnen haben)

Dirty Read

Hier ist dokumentiert, wie sich Dirty Read reproduzieren lässt.

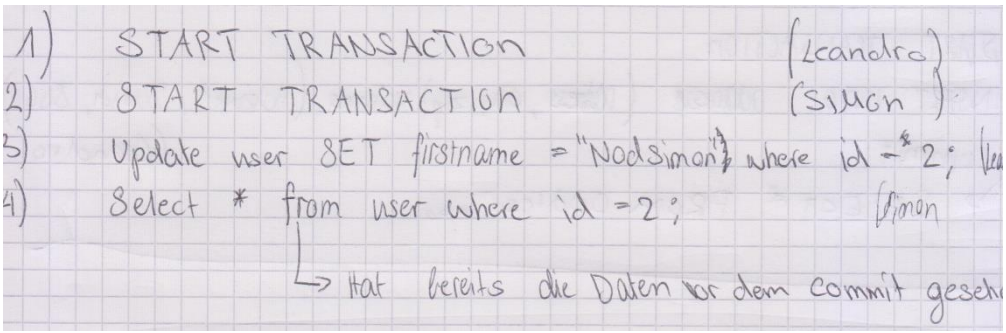

Abbildung 2: Anleitung zum Reproduzieren Dirty Read

Non-Repeatable-Read

Hier ist dokumentiert, wie sich Non-Repeatable Read reproduzieren lässt.

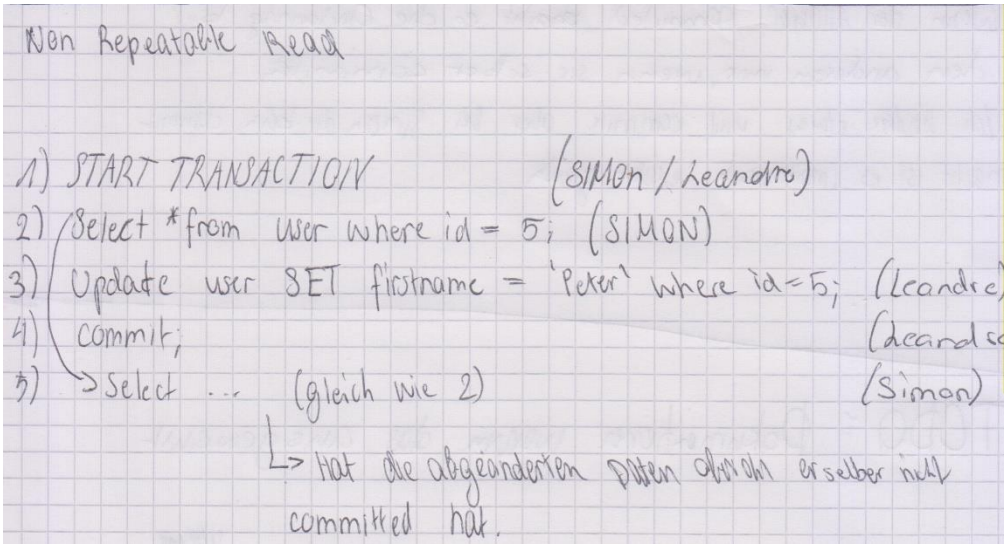

Abbildung 3: Anleitung zum Reproduzieren Non-Repeatable Read

Phantom Read

Hier ist dokumentiert, wie sich Phantom Read reproduzieren lässt.

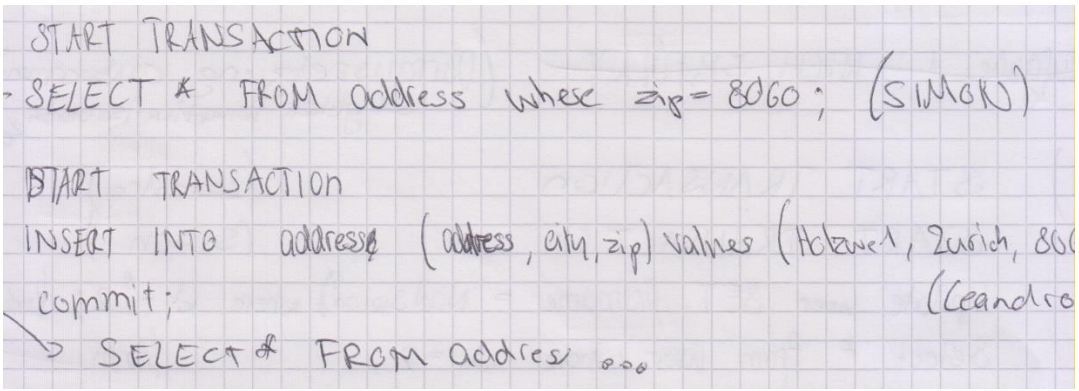

Abbildung 4: Anleitung zum Reproduzieren Phantom Read

Dies kann durch verschiedene Isolationslevel verhindert werden. Unten in der Grafik ist zu sehen, welches Isolationslevel was verhindert.

Isolation Level	Dirty Read	Non-Repeatable read	Phantom Read
Read Uncommitted	Possible	Possible	Possible
Read Committed	Not possible	Possible	Possible
Repeatable Read	Not possible	Not possible	Possible*
Serializable	Not possible	Not possible	Not possible

* Not possible for InnoDB, which uses snapshots for Repeatable Read.

Abbildung 5: Screenshot Fälle bei verschiedenen Isolationslevel. Quelle

Da wir als DB-Engine InnoDB verwenden, wird «Repeatable Read» verwendet, da mit diesem Isolationslevel bereits alle Fälle abgedeckt werden können.

Sperren von Daten/Tabellen

Implementierung

Testen (Backend)

Da fast alle Tests des Backends automatisiert sind, stehen hier nur die Tests, um die Isolationslevel zu überprüfen. Wichtig ist für alle Tests, dass Auto-Commit nicht aktiv ist.

Dirty Read

Mit diesem Test wird getestet, ob Dirty Read geht. Dazu müssen die folgenden Schritte befolgt werden.

1. Manuell eine Transaktion starten
2. Einen Eintrag erstellen, aber noch nicht commiten
3. Versuchen die neu erstellte Ressource über die API des Backends aufzurufen

Erwartet ist, dass kein Eintrag nach dem Befolgen der Schritte gefunden wird.

Non-repeatable Read

Mit diesem Test wird getestet, ob Non-Repeatable Read geht. Dazu müssen die folgenden Schritte befolgt werden.

1. Manuell eine Transaktion starten
2. Nach einem Eintrag suchen
3. Diesen Eintrag über die API des Backends verändern
4. Nach diesem Eintrag suchen

Erwartet ist, dass der Eintrag zwischen Schritt 2 und 4 sich nicht verändert.

Phantom Read

Mit diesem Test wird getestet, ob Phantom Read geht. Dazu müssen die folgenden Schritte befolgt werden.

1. Manuell eine Transaktion starten
2. Nach Einträgen suchen

3. Einen neuen Eintrag über die API des Backends erstellen
4. Nach Einträgen suchen

Erwartet ist, dass sich die Liste der Einträge zwischen Schritt 2 und 4 sich nicht verändert.

Testen (Android App)

Im folgenden Abschnitt wird das Thema Testen behandelt. Pro Ressource für jede CRUD-Operation gibt es mindestens einen Test. Zusätzlich

Funktionstest

Im folgenden Abschnitt werden alle Tests für grundlegende Funktionen der Applikation beschrieben.

1. Test: Einloggen

Aktion: Einloggen mit den Logindaten User: «Hans» und Passwort «wurst»

Erwartet: Navigation zur leeren Startseite.

2. Test: Navigationsmenü

Um diesem Test zu bestehen, muss mit dem Menü erfolgreich von einer View zu einer anderen View navigiert werden.

3. Test: «Erstellen»-Knopf

Aktion: «Erstellen»-Knopf in einer Tabellen-View drücken.

Erwartet: Navigation zu der entsprechenden View zum Erstellen eines Eintrags («Ausleihetabelle»-View => «Ausleihe erstellen»-View, «Mediumtabelle»-View => «Medium erstellen»-View).

4. Test: «Abbrechen»-Knopf

Aktion: Klicken des «Abbrechen»-Knopf auf einer View zum Erstellen von Einträgen.

Erwartet: Navigation zur entsprechenden Tabellen-View des jeweiligen Objekts («Ausleihe erstellen»-View => «Ausleihetabelle»-View, «Medium erstellen»-View => «Mediumtabelle»-View).

Ressource «Medium»

Im folgenden Abschnitt sind alle Tests der Ressource «Medium» beschrieben.

Voraussetzung

Voraussetzungen für die folgenden Tests sind:

- Eingeloggt sein mit den Logindaten User: «Hans» und Passwort «wurst»

1. Test: Medium erstellen

Aktion: Zuerst muss zur Ansicht «Medium erstellen» navigiert werden. Anschliessend müssen die Eingabefelder Titel und Autor befüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Medientabelle» und der Eintrag mit den richtigen Werten muss sichtbar sein.

2. Test: Medien anzeigen

Um diesen Test zu bestehen, muss das vorher erstellte Medium in der View «Medientabelle» angezeigt werden.

3. Test: Medium anpassen

Aktion: Zuerst muss das erstellte Element mit langem Klick ausgewählt werden. Dann müssen alle noch leeren Inputs mit Testdaten gefüllt werden.

Erwartet: Automatische Navigation zur View «Medientabelle» und alle Daten des Eintrags müssen mit dem übereinstimmen, was eingegeben wurden.

4. Test: Medium löschen

Aktion: Zuerst muss das geänderte Element mit langem Klick ausgewählt werden. Um das Element zu löschen, muss «Löschen» geklickt werden und den Dialog mit «OK» bestätigen.

Erwartet: Automatische Navigation zur View «Medientabelle» und der Eintrag darf nicht mehr in der Tabelle sichtbar sein.

Ressource «Ausleihe»

Im folgenden Abschnitt sind alle Tests der Ressource «Ausleihe» beschrieben.

Voraussetzungen

Voraussetzungen für die folgenden Tests sind:

- Mindestens ein Kunde
- Mindestens ein Medium
- Eingeloggt sein mit den Logindaten User: «Hans» und Passwort «wurst»

1. Test: Ausleihe erstellen

Aktion: Zuerst muss zur Ansicht «Ausleihe erstellen» navigiert werden. Anschliessend müssen die Eingabefelder mit der Medien-ID und der Kunden-ID befüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Ausleihetabelle» und der Eintrag mit den richtigen Werten muss sichtbar sein.

2. Test: Ausleihen anzeigen

Um diesen Test zu bestehen, muss die vorher erstellte Ausleihe in der View «Ausleihetabelle» angezeigt werden.

3. Test: Ausleihe anpassen

Aktion: Zuerst muss das erstellte Element mit langem Klick ausgewählt werden. Um die Ausleihdauer anzupassen, muss «Verlängern» geklickt werden.

Erwartet: Automatische Navigation zur View «Ausleihetabelle» und das Rückgabedatum des Eintrags um 14 Tage nach vorne verschoben.

4. Test: Ausleihe löschen

Aktion: Zuerst muss das geänderte Element mit langem Klick ausgewählt werden. Um das Element zu löschen, muss «Löschen» geklickt werden und den Dialog mit «OK» bestätigen.

Erwartet: Automatisch Navigation zur View «Ausleihetabelle» und der Eintrag darf nicht mehr in der Tabelle sichtbar sein.

Testen (Frontend)

Im folgenden Abschnitt wird das Thema Testen behandelt. Pro Ressource für jede CRUD-Operation gibt es mindestens einen Test.

Funktionstest

Im folgenden Abschnitt werden alle Tests für grundlegende Funktionen der Applikation beschrieben.

1. Test: Einloggen

2. Test: Navigationsmenü

2Um diesem Test zu bestehen, muss mit dem Menü erfolgreich von einer View zu einer anderen View navigiert werden.

3. Test: «Erstellen»-Knopf

Aktion: «Erstellen»-Knopf in einer Tabellen-View drücken.

Erwartet: Navigation zu der entsprechenden View zum Erstellen eines Eintrags («Ausleihetabelle»-View => «Ausleihe erstellen»-View, «Medientabelle»-View => «Medium erstellen»-View).

4. Test: «Abbrechen»-Knopf

Aktion: Klicken des «Abbrechen»-Knopf auf einer View zum Erstellen von Einträgen.

Erwartet: Navigation zur entsprechenden Tabellen-View des jeweiligen Objekts («Ausleihe erstellen»-View => «Ausleihetabelle»-View, «Medium erstellen»-View => «Medientabelle»-View, «Kunde erstellen»-View => «Kundentabelle»-View).

Ressource «Medium»

Im folgenden Abschnitt sind alle Tests der Ressource «Medium» beschrieben.

Voraussetzung

1. Test: Medium erstellen

Aktion: Zuerst muss zur Ansicht «Medium erstellen» navigiert werden. Anschliessend müssen die Eingabefelder Titel und Autor befüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Medientabelle» und der Eintrag mit den richtigen Werten muss sichtbar sein.

2. Test: Medien anzeigen

Um diesen Test zu bestehen, muss das vorher erstellte Medium in der View «Medientabelle» angezeigt werden.

3. Test: Medium anpassen

Aktion: Zuerst muss das erstellte Element mit Klick auf das Stift-Symbol ausgewählt werden. Dann müssen alle noch leeren Inputs mit Testdaten gefüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Medientabelle» und alle Daten des Eintrags müssen mit dem übereinstimmen, was eingegeben wurden. $

4. Test: Medium löschen

Aktion: Um das Element zu löschen, muss das Löschen-Symbol des Eintrags geklickt werden und der Dialog mit «OK» bestätigt werden.

Erwartet: Eintrag darf nicht mehr in der Tabelle sichtbar sein.

Ressource «Ausleihe»

Im folgenden Abschnitt sind alle Tests der Ressource «Ausleihe» beschrieben.

Voraussetzungen

Voraussetzungen für die folgenden Tests sind:

- Mindestens ein Kunde
- Mindestens ein Medium
- Eingeloggt sein mit den Logindaten User: «Hans» und Passwort «wurst»

1. Test: Ausleihe erstellen

Aktion: Zuerst muss zur Ansicht «Ausleihe erstellen» navigiert werden. Anschliessend müssen die Eingabefelder mit der Medien-ID und der Kunden-ID befüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Ausleihetabelle» und der Eintrag mit den richtigen Werten muss sichtbar sein.

2. Test: Ausleihen anzeigen

Um diesen Test zu bestehen, muss die vorher erstellte Ausleihe in der View «Ausleihetabelle» angezeigt werden.

3. Test: Ausleihe anpassen

Aktion: Zuerst muss das erstellte Element mit Klick auf das Stift-Symbol ausgewählt werden. Um die Ausleihdauer anzupassen, muss der grüne Knopf «Verlängern» geklickt werden.

Erwartet: Automatische Navigation zur View «Ausleihetabelle» und das Rückgabedatum des Eintrags um 14 Tage nach vorne verschoben.

4. Test: Ausleihe löschen

Aktion: Um das Element zu löschen, muss das Löschen-Symbol des Eintrags geklickt werden und der Dialog mit «OK» bestätigt werden.

Erwartet: Eintrag darf nicht mehr in der Tabelle sichtbar sein.

Ressource «Kunde»

Im folgenden Abschnitt sind alle Tests der Ressource «Kunde» beschrieben.

Voraussetzung

Voraussetzungen für die folgenden Tests sind:

- Eingeloggt sein mit den Logindaten User: «Hans» und Passwort «wurst»

1. Test: Kunde erstellen

Aktion: Zuerst muss zur Ansicht «Kunde erstellen» navigiert werden. Anschliessend müssen die Eingabefelder mit Testdaten befüllt werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Kundentabelle» und der Eintrag muss nach dem Suchen nach Nachnamen in der Suchleiste mit den richtigen Werten sichtbar sein.

2. Test: Kunde anzeigen

Um diesen Test zu bestehen, muss der vorher erstellte Kunde Eintrag nach dem Suchen nach Nachnamen in der Suchleiste in der View «Kundentabelle» angezeigt werden.

3. Test: Kunde anpassen

Aktion: Zuerst muss das erstellte Element mit Klick auf das Stift-Symbol ausgewählt werden. Dann müssen Adresse, Ort, Postleitzahl und Email angepasst werden. Abschliessend muss der grüne Knopf «Speichern» gedrückt werden.

Erwartet: Automatische Navigation zur View «Kundentabelle» und alle Daten des Eintrags müssen mit dem übereinstimmen, was eingegeben wurden.

4. Test: Kunde löschen

Aktion: Um das Element zu löschen, muss das Löschen-Symbol des Eintrags geklickt werden und der Dialog mit «OK» bestätigt werden.

Erwartet: Eintrag darf nicht mehr in der Tabelle sichtbar sein.

Fazit

Mein Fazit zu dem ganzen Projekt ist, dass alles relativ gut eingehalten werden konnte. Es gab allerdings, wie in jedem Projekt, Entscheidungen zu treffen. Zum Beispiel, wie die Anmelde-View aussehen soll oder wie getestet werden soll.

Mein persönliches Fazit ist, dass ich viel über Authentisierung und Autorisierung lernen konnte.

Glossar

Begriffe	Bedeutung
Anwender-Software	Software, die für ein Anwender (Mensch) bestimmt ist => keine API. Beispiel: Word
Anwendung	Ein Programm, welches eine bestimmte Aufgabe erfüllt, z.B. Word, Minecraft, aber auch REST-API
Applikation	
Basis-Software	Betriebssystem
Dateisystem	System zur Verwaltung von Dateien, z.B. FAT32, NTFS oder EXT4
Datenbank	Software zur Speicherung von Daten in Form von Einträgen
FAT32	Altes Dateisystem von Windows, welches kompatibel mit Windows, Linux und Mac ist. Wird heute hauptsächlich noch für USB-Sticks verwendet.
Multiuser Betriebssystem	Betriebssystem, welches mehrere Benutzer verwalten kann. Jeder Benutzer hat seine eigene Rolle mit Rechten.
NTFS	Aktuelles Dateisystem von Windows
Prozess	Ein Prozess ist ein aktives Programm, welches Ressourcen braucht. Ein Prozess hat dabei immer einen Zustand wird
Task	
Thread	Ein Prozess kann einen oder mehrere Threads gleichzeitig ausführen
Ressourcen	Mittel, die ein Prozess benötigt, um seine Aufgabe auszuführen können

Bildverzeichnis