Technische Universität Braunschweig

ISF

Mediaverse

Software-Entwicklungspraktikum (SEP)
Sommersemester 2023

Testspezifikation

Auftraggeber
Technische Universität Braunschweig
Institut für Robotik und Prozessinformatik - IRP
Mühlenpfordtstraße 23
38106 Braunschweig

Betreuer: Dr. Bertold Bongardt, Heiko Donat, Sven Tittel, Christopher Lösch

Auftragnehmer:

Name	E-Mail-Adresse
Michèle Eger	m.eger@tu-braunschweig.de
Fahd Ferjani	f.ferjani@tu-braunschweig.de
Yassine Kechiche	y.kechiche@tu-braunschweig.de
Yiğit Kemal Çağlar	y.caglar@tu-braunschweig.de
Oussema Ben Smida	o.ben-smida@tu-braunschweig.de
Subing Shen	subing.shen@tu-braunschweig.de
Quynh Tran	quynh.tran@tu-braunschweig.de
Theodore Zebua	t.zebua@tu-braunschweig.de

Braunschweig, 12. Juli 2023

Bearbeiterübersicht

Kapitel	Autoren	Kommentare
1	Yigit,Oussema, Michéle,Theodore	...
2	Yassine, Fahd	...
2.1	Yassine, Fahd	...
2.2	Quynh, Subing	...
2.3	Quynh, Subing	...
2.4	Subing, Quynh	...
2.5	Subing, Quynh	...
3	Yigit,Oussema, Michéle,Theodore	...
3.1	Yigit,Oussema, Michéle,Theodore	...
3.2	Yigit,Oussema, Michéle,Theodore	...
3.3	Yigit,Oussema, Michéle,Theodore	zwei neue Tests wurden aufgrund des Feedbacks hinzugefügt.
4	Yigit,Oussema, Michéle,Theodore	...
4.1	Yigit,Oussema, Michéle,Theodore	...
4.2	Yigit,Oussema, Michéle,Theodore	...
4.3	Yigit,Oussema, Michéle,Theodore	...
5	Subing,Quynh, Fahd,Yassine	...
5.1	Subing,Quynh, Fahd,Yassine	...
5.2	Subing,Quynh, Fahd,Yassine	...
5.3	Subing,Quynh, Fahd,Yassine	...
6	Fahd,Yassine	...

Inhaltsverzeichnis

1 Einleitung **7**

2 Testplan **8**
2.1 Zu testende Komponenten 8
2.2 Zu testende Funktionen/Merkmale 9
2.3 Nicht zu testende Funktionen 10
2.4 Vorgehen 10
2.4.1 Komponententests 10
2.4.2 Integrationstest 10
2.4.3 Systemtest 10
2.4.4 Abnahme- und Funktionstests 10
2.5 Testumgebung 11

3 Abnahmetest **12**
3.1 Zu testende Anforderungen 12
3.2 Testverfahren 13
3.2.1 Testskripte 14
3.3 Testfälle 14
3.3.1 Testfall $\langle T110 \rangle$ - Konto erstellen 15
3.3.2 Testfall $\langle T120 \rangle$ - Log in 16
3.3.3 Testfall $\langle T210 \rangle$ - Suchfunktion nach beliebigen Ressourcen (mit bestimmten Suchfilter) 17
3.3.4 Testfall $\langle T220 \rangle$ - Suchfunktion nach beliebigen Ressourcen (durch Einscannen des QR-Codes) 18
3.3.5 Testfall $\langle T300 \rangle$ - Verfügbarkeit, Ausleihen und Ausleih-Verlängerung . . . 19
3.3.6 Testfall $\langle T400 \rangle$ - Herunterladen von Ressourcen 20
3.3.7 Testfall $\langle T500 \rangle$ - Hinzufügen von Ressourcen 21
3.3.8 Testfall $\langle T600 \rangle$ - Bearbeiten von Ressourcen 22
3.3.9 Testfall $\langle T700 \rangle$ - Löschen von Ressourcen 23
3.3.10 Testfall $\langle T800 \rangle$ - Favoritenlisten 24
3.3.11 Testfall $\langle T900 \rangle$ - Spracheinstellung 25
3.3.12 Testfall $\langle T1000 \rangle$ - Verlauf der (eigenen) Aktionen 26
3.3.13 Testfall $\langle T1110 \rangle$ - Anlegen von Benutzern und Mediathekaren 27

3.3.14 Testfall ⟨T1120⟩ - Editieren von Benutzern und Mediathekaren 28
3.3.15 Testfall ⟨T1130⟩ - Löschen von Benutzern und Mediathekaren 29
3.3.16 Testfall ⟨T1200⟩ - QR-Code 30
3.3.17 Testfall ⟨T1300⟩ - <Q30> Antwortzeit zum Anzeigen der Verfügbarkeit einer Ressource 31

4 Integrationstest **32**
4.1 Zu testende Komponenten 32
4.2 Testverfahren 33
4.2.1 Testskripte 33
4.3 Testfälle 33
4.3.1 Testfall ⟨T1100⟩ - Komponente C10+C70 34
4.3.2 Testfall ⟨T1200⟩ - Komponente C10 + C20 35
4.3.3 Testfall ⟨T1210⟩ - Komponente C10 + C30 36
4.3.4 Testfall ⟨T1300⟩ - Komponente C20+C30 37
4.3.5 Testfall ⟨T1400⟩ - Komponente C30+C50 38
4.3.6 Testfall ⟨T1410⟩ - Komponente C30+C50 39
4.3.7 Testfall ⟨T1420⟩ - Komponente C30+C50 40
4.3.8 Testfall ⟨T1500⟩ - Komponente C10+C20 41
4.3.9 Testfall ⟨T1600⟩ - Komponente C30+C40 42
4.3.10 Testfall ⟨T1700⟩ - Komponente c40+c60 43

5 Unit-Tests **44**
5.1 Zu testende Komponenten 44
5.2 Testverfahren 45
5.2.1 Testskripte 45
5.3 Testfälle 45
5.3.1 Testfall ⟨T1700⟩ - test_create_user_success 46
5.3.2 Testfall ⟨T1800⟩ - test_create_token_for_activated_user 47
5.3.3 Testfall ⟨T1810⟩ - test_create_token_for_non_activated_user 48
5.3.4 Testfall ⟨T1900⟩ - test_update_user_profile 49
5.3.5 Testfall ⟨T2000⟩ - test_retrieve_profile_success 50
5.3.6 Testfall ⟨T2100⟩ - test_create_resource 51
5.3.7 Testfall ⟨T2200⟩ - test_delete_resource 52
5.3.8 Testfall ⟨T2300⟩ - - test_update_resource 53
5.3.9 Testfall ⟨T2400⟩ - test_create_resource_only_mediathekar 54
5.3.10 Testfall ⟨T2500⟩ - test_search_resource 56
5.3.11 Testfall ⟨T2600⟩ - test_search_resource_with_filter 57
5.3.12 Testfall ⟨T2700⟩ - test_search_resource_with_qrcode 58
5.3.13 Testfall ⟨T2800⟩ - test_borrow_resource 59

5.3.14 Testfall ⟨T2900⟩ - test_borrow_resource_not_available 60
5.3.15 Testfall ⟨T3000⟩ - test_borrow_resource_invalid_resource 61
5.3.16 Testfall ⟨T3100⟩ - test_borrow_resource_due_date 62
5.3.17 Testfall ⟨T3200⟩ - test_add_resource_to_favouritelist 63
5.3.18 Testfall ⟨T3300⟩ - test_remove_resource_from_favouritelist 64
5.3.19 Testfall ⟨T3400⟩ - test_share_items_from_favouritelist 65

6 Glossar **66**

Abbildungsverzeichnis

1 Einleitung

Das Testen von Software ist ein unverzichtbarer Bestandteil eines jeden Softwareentwicklungsprozesses. Diese Testdokumentation beschäftigt sich mit der Testspezifikation für das Mediaverse-Projekt. Das Ziel des Mediaverse-Projekts ist die Entwicklung einer webbasierten Plattform zur Verwaltung und Bereitstellung von Multimedia-Inhalten. Die Plattform umfasst eine Vielzahl von Funktionen, darunter die Verwaltung von Benutzerkonten, das Hochladen und Verwalten von Multimedia-Inhalten sowie die Suche und Wiedergabe von Inhalten.

Um sicherzustellen, dass die Plattform den Anforderungen entspricht, werden wir uns in dieser Abnahmetestspezifikation auf eine detaillierte und umfassende Teststrategie konzentrieren. Dabei orientieren wir uns an den Vorgaben des IEEE 829-Standards. Die Tests werden alle Funktionen der Plattform abdecken und Fehler und Mängel aufdecken, die anschließend behoben werden müssen, um eine reibungslose Nutzung der Plattform zu gewährleisten.

In diesem Dokument werden wir uns mit der Beschreibung der zu testenden Funktionen und der Testumgebung befassen, sowie die Durchführung der Tests und die Auswertung der Testergebnisse behandeln. Unsere Testabläufe sollen sicherstellen, dass die Plattform den spezifizierten Anforderungen entspricht und die Qualitätsstandards erfüllt, um eine erfolgreiche Implementierung und Inbetriebnahme des Mediaverse-Projekts zu gewährleisten.

2 Testplan

Das nachfolgende Kapitel beschreibt den Testplan für das Mediaverse-Projekt. Es werden die zu testenden Komponenten und Funktionen erläutert sowie diejenigen, die nicht getestet werden. Zusätzlich wird die allgemeine Vorgehensweise zum Testen der Funktionen beschrieben und erläutert, um sicherzustellen, dass das Mediaverse-Projekt zuverlässig und fehlerfrei funktioniert. Dabei werden alle relevanten Aspekte des Mediaverse-Projekts berücksichtigt, um eine optimale Testabdeckung zu gewährleisten.

2.1 Zu testende Komponenten

Die folgenden Komponenten müssen in Mediaverse getestet werden, einschließlich ihrer Versionsnummern, der Mediumsform, auf der die Software vorliegt, und ob die Software vor dem Testen transformiert werden muss:

C10: Benutzeraktivitätenverfolgung:
Testen, ob die Funktion zur Verfolgung von Benutzeraktivitäten ordnungsgemäß funktioniert und alle relevanten Informationen erfasst werden. Überprüfen, ob die erfassten Daten korrekt gespeichert und verwaltet werden.

C20: Ressourcenverwaltung :
Es muss getestet werden, ob die Ressourcenverwaltungsfunktionen fehlerfrei ausgeführt werden. Dies umfasst das Hochladen, Bearbeiten, Löschen und Anzeigen von Ressourcen. Überprüfen, ob die Ressourcen korrekt gespeichert, verwaltet und angezeigt werden.

C30:Suchfunktion :
Testen, ob die Suchfunktion ordnungsgemäß funktioniert und die gewünschten Ergebnisse liefert. Überprüfen, ob die Suchergebnisse relevant und korrekt sind.

C40: Ausleihverwaltung :
Es muss getestet werden, ob die Ausleihverwaltungsfunktionen ordnungsgemäß funktionieren. Dies umfasst das Ausleihen und Rückgeben von Ressourcen sowie die Überprüfung der Verfügbarkeit und des Status von Ressourcen.verursachen.

C50: Favoritenverwaltung :
Testen, ob die Funktionen zur Verwaltung von Favoriten fehlerfrei ausgeführt werden. Überprüfen, ob Benutzer Favoriten hinzufügen, entfernen und anzeigen können, und ob die Favoritenliste korrekt gespeichert und verwaltet wird. .

C60: Sprachverwaltung :
Es muss getestet werden, ob die Funktionen zur Sprachverwaltung ordnungsgemäß funktionieren. Überprüfen, ob Benutzer die Spracheinstellungen ändern können und ob die Anwendung die entsprechenden Sprachdateien korrekt verwendet.

2.2 Zu testende Funktionen/Merkmale

In diesem Abschnitt werden die zu testenden Funktionen und Merkmale aufgelistet.

- <F10> Registrierung
- <F20> Login
- <F30> Suchfunktion nach beliebigen Ressourcen
- <F40> Verfügbarkeit, Ausleihen und Ausleih-Verlängerung von Ressourcen
- <F50> Download von Ressourcen
- <F60> Hinzufügen / Bearbeiten / Löschen von Ressourcen
- <F70> Erstellen von Favoritenlisten
- <F80> Spracheinstellung
- <F90> Verlauf der (eigenen) Aktionen
- <F100> Anlegen von Benutzern und Mediathekaren
- <F110> Editieren und Löschen von Benutzern und Mediathekaren
- <F120> QR-Code
- <Q30> Die Funktionen ⟨F30⟩ und ⟨F40⟩ sollte nicht länger als 5 Sekunden Antwortzeit benötigen.

2.3 Nicht zu testende Funktionen

Die webbasierte Applikation wird auf einem Server intstalliert und dem entsprechend muss keine Hardware geprüft werden. Darüber hinaus wird das Linux-Betriebssystem in seiner Funktionalität nicht getestet, da dies nicht Teil der Web-Applikation ist.

2.4 Vorgehen

2.4.1 Komponententests

Die Komponenten von 'Mediaverse', die in Kapitel 2.1 erwähnt werden, werden separat getestet. Der Testprozess folgt den Testfällen im folgenden Abschnitt und wird dokumentiert. Dadurch wird nachvollziehbar, an welcher Stelle Fehler oder Probleme auftreten. Zusätzlich wird die Isolation jeder Komponente sichergestellt.

2.4.2 Integrationstest

Aus einer übergeordneten Perspektive wird im Rahmen der zweiten Testphase der Bottom-Up-Ansatz angewendet, um die einzelnen Komponenten schrittweise zu größeren Gruppen zu integrieren. Dadurch lässt sich überprüfen, ob es Fehler bei der Interaktion der Komponenten gibt, insbesondere während der Kommunikation zwischen ihnen.

2.4.3 Systemtest

Nach Abschluss des Integrationstests wird das System als Ganzes getestet. Zu diesem Zweck sollten alle Programmfunktionen und eine realistische Produktionsumgebung für den Test bereitstehen, um mögliche Fehler zu überprüfen. Während dieses Prozesses kann die Zuverlässigkeit und Stabilität von 'Mediaverse' verbessert werden.

2.4.4 Abnahme- und Funktionstests

Nach Abschluss der oben genannten Schritte und der Behebung aufgetretener Fehler kann der Abnahmetest durchgeführt werden. In diesem Prozess bewertet der Kunde das Produkt, indem er überprüft, ob alle Funktionen ordnungsgemäß auf der Kundenseite funktionieren. Basierend auf dem Feedback des Kunden wird 'Mediaverse' weiter optimiert oder endgültig übergeben.

2.5 Testumgebung

Die Applikation wird auf einem lokalen Server auf einem PC installiert, der mit dem Linux-Ubuntu-System läuft. Zunächst wird getestet, ob die Anwendung korrekt bereitgestellt werden kann, und es besteht die Möglichkeit, die 'Client'-Website auf verschiedenen Geräten zu testen. Die unterstützten Geräte/Systeme umfassen MacBook mit macOS, PC/Laptop mit Windows sowie Mobiltelefone/Tablets mit iOS und Android. Darüber hinaus kann auch die Benutzerfreundlichkeit von 'Mediaverse' auf Systemen mit unterschiedlichen Sprachen getestet werden.

3 Abnahmetest

Der Zweck des Tests besteht darin, sicherzustellen, dass das Produkt alle Anforderungen erfüllt und vollständig ist, bevor es an den Kunden übergeben wird. Die zu testenden Anforderungen werden beschrieben und daraus resultierende Testfälle werden erstellt, um das ordnungsgemäße Funktionieren des Programms und das Erfüllen der Anforderungen zu überprüfen.

Der Abnahmetest dient als Nachweis für die erfolgreiche Erbringung der vereinbarten Leistungen zwischen dem Kunden und dem Auftragnehmer. Im nächsten Abschnitt wird detailliert beschrieben, welche Testverfahren verwendet werden, um die gewünschten Resultate zu erreichen.

3.1 Zu testende Anforderungen

In diesem Abschnitt werden alle zu testenden Anforderungen aufgeführt

Nr	**Anforderung**	**Testfälle**	**Kommentar**
1.1	<F10> Konto erstellen	<T110>	Kann ein Benutzer ein Konto erfolgreich registrieren?
1.2	<F20> Konto anmelden (Log in)	<T120>	Kann sich ein Nutzer mit seinem registrierten Konto anmelden und auf Mediaverse zugreifen?>
2.1	<F30> Suchfunktion nach beliebigen Ressourcen (mit bestimmten Suchfilter)	<T210>	Kann ein Nutzer die gesuchte Ressource mit Hilfe der Suchmaschine und deren Filtermöglichkeiten finden?
2.2	<F30> Suchfunktion nach beliebigen Ressourcen (durch Einscannen des QR-Codes)	<T220>	Kann ein Benutzer die gesuchte Ressource durch Scannen des QR-Codes einer in der Datenbank gespeicherten Ressource finden?

3	<F40> Verfügbarkeit, Ausleihen und Ausleih-Verlängerung	<T300>	<RM4>,<RM5>,<RM6>, <RM7>, <RS2>, <RC3>
4	<F50> Herunterladen von Ressourcen	<T400>	<RM10>
5	<F60> Hinzufügen / Bearbeiten / Löschen von Ressourcen	<T500>, <T600>, <T700>	<RM13>,<RM14>, <RM15>
6	<F70> Favoritenlisten	<T800>	<RM8>
7	<F80> Spracheinstellung	<T900>	<RS3>, <RC4>
8	<F90> Verlauf der (eigenen) Aktionen	<T1000>	<RM19>
9	<F100> Anlegen, Editieren und Löschen von Benutzern und Mediathekaren	<T1110>, <T1120>, <T1130>	<RM17>,<RM18>
11	<F120> QR-code	<T1200>	<RM6>
12	<Q30> Antwortzeit der Funktionen <F30> und <F40>	<T1300>	Die Funktionen ⟨F30⟩ und ⟨F40⟩ sollte nicht länger als 5 Sekunden Antwortzeit benötigen.

3.2 Testverfahren

Die Software wird mittels manuellem Testen überprüft. Beim manuellen Testen führen Tester die Testfälle und Szenarien manuell aus, ohne den Einsatz von automatisierten Tools. Die Tester interagieren direkt mit der Software, geben Eingabedaten ein und überprüfen die Ausgaben anhand der erwarteten Ergebnisse. Durch manuelles Testen können verschiedene Aspekte der Software, wie Benutzerfreundlichkeit, Funktionalität und Zuverlässigkeit, bewertet werden. Die Methode des manuellen Testens ermöglicht eine flexible und kontrollierte Durchführung der Tests, bei der Tester ihre Erfahrung und Intuition einsetzen können, um potenzielle Fehler und Probleme zu entdecken.

3.2.1 Testskripte

Es ist nicht geplant, Testskripte zu verwenden; falls deren Implementierung erforderlich sein sollte, werden die Einzelheiten und Informationen dazu in den nachfolgenden Dokumenten erläutert werden.

3.3 Testfälle

Im Folgenden werden detaillierte Informationen zu den einzelnen Testfällen, deren Abhängigkeiten, Voraussetzungen und den einzelnen Test-schritten dargestellt.

3.3.1 Testfall $\langle T110 \rangle$ - Konto erstellen

Ziel

Der Nutzer konnte erfolgreich ein Konto erstellen und seine Daten , E-Mail-Adresse, Benutzername und Passwort, wurden erfolgreich hinterlegt.

Objekte/Methoden/Funktionen

Objekte: Benutzername, E-Mail-Adresse, Passwort

Funktionen: <F10>

Pass/Fail Kriterien

Überprüfung der Kriterien durch Anmeldung einer Testperson. Der Test ist erfolgreich, wenn sich die Person in das System einloggen konnte.

Vorbedingung

Der Nutzer benötigt eine E-Mail-Adresse und sollte über eine stabile Internetverbindung und ein Gerät verfügen, das mit einem Browser auf die Mediaverse-Website zugreifen kann und navigieren kann.

Einzelschritte

Eingabe:

1. Die Webseite von Mediaverse öffnen
2. Beim Registrierungsfeld 'here' klicken
3. E-Mail-Adresse, Benutzername und zweimal das Passwort eingeben sowie auf 'Create account' klicken
4. Anzeige des Textes "Registrierung erfolgreich abgeschlossen", dann weiter zum Login-Display
5. Warten, bis der Administrator das Konto mit den entsprechenden Rechten aktiviert hat. (Der Nutzer wird durch eine direkte E-Mail benachrichtigt.)
5. Eingabe der Anmeldedaten in das Login-Feld
6. Auf 'Login' klicken

Ausgabe: Homepage von Mediaverse.

Beobachtungen / Log / Umgebung

Die Startseite der Website wird angezeigt. Je nach der Rolle des Benutzers werden bestimmte Rechte zugewiesen und der Account wird aktiviert.

Abhängigkeiten

Für diesen Test besteht es keine Abhängigkeiten.

3.3.2 Testfall $\langle T120 \rangle$ - Log in

Ziel

Der Nutzer kann sich mit seinem registrierten Konto bei Mediaverse anmelden, der Login ist erfolgreich.

Objekte/Methoden/Funktionen

Objekte: Benutzername, E-Mail-Adresse, Passwort

Funktionen: <F20>

Pass/Fail Kriterien

Überprüfung der Kriterien durch Anmeldung einer Testperson. Der Test ist erfolgreich, wenn sich die Person in das System einloggen konnte.

Der Test ist fehlgeschlagen, wen sich die Person in das System nicht einloggen konnte.

Vorbedingung

Der Nutzer benötigt ein erfolgreich registriertes Konto.

Einzelschritte

Eingabe:

1. Die Webseite von Mediaverse öffnen.
2. Die Login-Seite erscheint.
3. Eingabe der Anmeldedaten (Benutzername oder E-Mail Adresse und Passwort) in das Login-Feld.
4. Auf 'Login' klicken.

Erfolgreiche Ausgabe: Homepage von Mediaverse.

Beobachtungen / Log / Umgebung

Die Startseite der Mediaverse wird angezeigt.

Abhängigkeiten

<T100>

3.3.3 Testfall $\langle T210 \rangle$ - Suchfunktion nach beliebigen Ressourcen (mit bestimmten Suchfilter)

Ziel

Es wird überprüft, ob bei der Suche nach einer gewünschten Ressource mit Hilfe der Filtereinstellungen passende Ergebnisse angezeigt werden.

Objekte/Methoden/Funktionen

Funktion: <F30>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Benutzer den gesuchten Inhalt durch Eingabe der Referenznummer (id) oder der Referenzwörter finden kann (beide Optionen sollten getestet werden).

Der Test ist nicht erfolgreich, wenn mit den richtigen Angaben der gesuchte Inhalt nicht zu sehen ist.

Vorbedingung

Der Benutzer hat ein registriertes Konto und ein funktionierendes Gerät, mit dem er auf Mediaverse zugreifen kann.

Einzelschritte

Eingabe:

1. Zur Homepage gehen
2. In Suchfeld klicken
3. Suchbegriff eingeben
4. Suchbegriff bestätigen

Ausgabe: Es wird eine Liste mit gesuchtem Objekt sowie dazu ähnliche Ressourcen auf der Seite angezeigt bzw. vorgeschlagen.

Beobachtungen / Log / Umgebung

Die passende Ressource sollte auftauchen.

Abhängigkeiten

<T110>, <T120>

3.3.4 Testfall $\langle T220 \rangle$ - Suchfunktion nach beliebigen Ressourcen (durch Einscannen des QR-Codes)

Ziel

Es wird überprüft, ob bei der Suche nach einer gewünschten Ressource durch Scannen des QR-Codes eines Inhalts passende Ergebnisse angezeigt werden.

Objekte/Methoden/Funktionen

Funktion: <F30>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Benutzer den gesuchten Inhalt durch Scannen des QR-Codes finden konnte. Der Test ist erfolglos, wenn keine relevanten Treffer zu sehen sind.

Vorbedingung

Der Benutzer hat ein registriertes Konto und ein funktionierendes Gerät, mit dem er auf Mediaverse zugreifen kann.

Einzelschritte

Eingabe:

1. Zur Homepage gehen

2. Auf das Kamerasymbol neben der Suchmaschine tippen. (der Nutzer wird zu einer Kamera-App weitergeleitet)

4. ein Foto des QR-Codes eines Inhalts machen.

Ausgabe: die Inhaltsinformationsseite der gesuchten Ressource/des gesuchten Inhalts.

Beobachtungen / Log / Umgebung

Die passende Ressource sollte auftauchen.

Abhängigkeiten

<T110>, <T120>, <T210>

3.3.5 Testfall $\langle T300 \rangle$ - Verfügbarkeit, Ausleihen und Ausleih-Verlängerung

Ziel

Es wird hier getestet, ob die Ausleih-verfügbarkeit von Ressourcen korrekt angezeigt wird, wenn eine Ressource verfügbar ist und ob der Nutzer eine ausgeliehene Ressource erfolgreich verlängern können, wenn dies erlaubt ist.

Objekte/Methoden/Funktionen

Funktion: <F40>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Benutzer die gewählte Ressource ausleihen kann und dann die Ausleihe auch verlängern kann.
Der Test gilt als nicht bestanden, wenn die Ausleih-Verfügbarkeit falsch angezeigt wird oder der Nutzer die Ausleihe nicht verlängern kann.

Vorbedingung

Benutzer muss erstmals eingeloggt sein (<T100>) und eine Ressource auswählen (<T200>).

Einzelschritte

Eingabe:
1. Auf eine Ressource klicken.
2. Auf neue Seite zur Ausleihe der Ressource weitergeleitet
3. Checkt, ob die Ressource verfügbar ist
4. Auf 'ausleihen' klicken um die Ressource auszuleihen
5. Auf 'verlängern' klicken um die Ausleihe zu verlängern

Ausgabe: 'Ressource erfolgreich ausgeliehen' und nach Verlängerung 'Ausleihe verlängert'

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob der Benutzer eine oder mehrere Ressourcen erfolgreich ausleihen und ihre Ausleihe verlängern kann.

Abhängigkeiten

<T100>,<T200>

3.3.6 Testfall $\langle T400 \rangle$ - Herunterladen von Ressourcen

Ziel

Herunterladen von online verfügbaren Ressourcen zur ausleihe.

Objekte/Methoden/Funktionen

Objekte: verfügbares Material zum Herunterladen in digitalem Format
Funktion: <F50>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die ausgewählte digitale Ressource erfolgreich heruntergeladen wurde und sich in den Dateien des Rechners findet.
Der Test ist gescheitert, wenn die Ressource nicht heruntergeladen werden kann.

Vorbedingung

Der Verwalter ist eingeloggt (<T100>), hat ein passendes Dokument, das zum Download angeboten wird, ausgesucht und es ausgewählt.

Einzelschritte

Eingabe: Auf den 'Download' Button klicken.
Ausgabe: Die Seite mit der heruntergeladenen Ressource erscheint.

Beobachtungen / Log / Umgebung

Die heruntergeladene Ressource soll auf dem Gerät des Nutzers verfügbar sein.

Abhängigkeiten

<T100>, <T200>

3.3.7 Testfall $\langle T500 \rangle$ - Hinzufügen von Ressourcen

Ziel

Der Verwalter kann eine neue Ressource hinzufügen.

Objekte/Methoden/Funktionen

Objekte: Ressourcen wie Bücher, Ebooks, PDF-Dateien, Hardware

Methode: hinzufügen

Funktion: <F60>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Ressource erfolgreich hinzugefügt wurde und sich in der Datenbank befindet.

Der Test ist nicht erfolgreich, wenn die Ressource nicht hinzugefügt werden könnte.

Vorbedingung

Der Verwalter ist eingeloggt. Die Ressource befindet sich noch nicht im System.

Einzelschritte

Eingabe:

1. Der Verwalter loggt sich auf 'Mediaverse' ein.
2. Es wird nach gewünschte Ressourcen gesucht.
3. Auf 'Löschen' Button klicken.

Ausgabe: Text erscheint: 'Die Ressource wurde erfolgreich zu Datenbank hinzugefügt.'

Beobachtungen / Log / Umgebung

Das hinzugefügte Dokument erscheint in der Liste der Ressourcen.

Abhängigkeiten

<T100>, <T200>

3.3.8 Testfall $\langle T600 \rangle$ - Bearbeiten von Ressourcen

Ziel

Der Verwalter kann eine eine bestehende Ressource bearbeiten.

Objekte/Methoden/Funktionen

Objekte: Ressourcen wie Bücher, Ebooks, PDF-Dateien, Hardware

Methode: bearbeiten

Funktion: <F60>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn sich die Ressource mit den geänderten Informationen in der Datenbank befindet.

Der Test schlägt fehl, wenn die Änderung nicht gespeichert werden könnte

Vorbedingung

Der Verwalter ist eingeloggt und hat das zu ändernde Dokument ausgewählt. Die Ressource muss sich bereits in der Datenbank befinden.

Einzelschritte

Eingabe:

1. In Suchfeld klicken
2. Ressource suchen
3. Auf 'Bearbeiten' klicken
4. Ressource bearbeiten
5. Auf 'Fertig' klicken

Ausgabe: Text erscheint: 'Die Ressource wurde erfolgreich bearbeitet.

'Die Ressource befindet sich nun mit den geänderten Informationen in der Datenbank.

Beobachtungen / Log / Umgebung

Das hochgeladene Dokument ist für andere Benutzer zur weiteren Verwendung/Bearbeitung verfügbar.

Abhängigkeiten

<T100>, <T200>

3.3.9 Testfall $\langle T700 \rangle$ - Löschen von Ressourcen

Ziel

Der Verwalter kann eine bestehende Ressource löschen.

Objekte/Methoden/Funktionen

Objekte: Ressourcen wie Bücher, Ebooks, PDF-Dateien, Hardware

Methode: löschen

Funktion: <F60>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Ressource aus der Datenbank entfernt wurde.

Der Test ist nicht erfolgreich, wenn die Ressource nicht gelöscht werden kann.

Vorbedingung

Der Verwalter hat sich bei Mediaverse angemeldet (<T100>) und die zu löschende Ressource gesucht und ausgewählt (<T200>).

Die Ressource muss bereits in der Datenbank bestehen.

Einzelschritte

Eingabe:

1. Auf die Ressource gehen
2. Auf 'Löschen' klicken

Ausgabe: Text erscheint: 'Die Ressource wurde erfolgreich gelöscht.'

Die Ressource befindet sich nicht mehr in der Datenbank.

Beobachtungen / Log / Umgebung

Die Ressource ist nicht mehr auffindbar.

Abhängigkeiten

<T100>, <T200>

3.3.10 Testfall ⟨$T800$⟩ - Favoritenlisten

Ziel

Es wird überprüft, ob die Ressourcen erfolgreich zur Favoritenliste hinzugefügt werden.

Objekte/Methoden/Funktionen

Methode: hinzufügen

Funktion: <F70>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die gewählte Ressource in die Favoritenliste hinzugefügt werden könnte.

Der Test wird als gescheitert angesehen, wenn es nicht möglich ist, die ausgewählten Ressourcen in der Favoritenliste zu speichern.

Vorbedingung

Der Benutzer hat sich erfolgreich bei seinem Konto angemeldet und kann die Startseite von Mediaverse besuchen. (<T100>)

Einzelschritte

Eingabe:

1. App öffnen
2. Auf den Stern (Zur Liste hinzufügen) Button klicken

Ausgabe: 'Ressource zur Favoritenliste hinzugefügt'

Beobachtungen / Log / Umgebung

Es wird beobachtet, wie die ausgewählte Elemente zur Favoritenliste hinzugefügt wird.

Abhängigkeiten

<T100>

3.3.11 Testfall ⟨$T900$⟩ - Spracheinstellung

Ziel

Es wird überprüft, dass die Sprache auf eine andere geändert wird.

Objekte/Methoden/Funktionen

Funktion:<F80>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Seite auf die gewünschte Sprache eingestellt ist.

Der Test ist gescheitert, wenn die Sprache nicht geändert wird.

Vorbedingung

Web App ist geöffnet.

Nutzer ist registriert.

Nutzer ist angemeldet.

Einzelschritte

Eingabe:

1.App öffnen

2.Auf 'Help' Schaltfläche klicken

3.Auf 'Change language' Schaltfläche klicken

4.Die gewünschte Sprache wählen

Ausgabe: Die Sprache wird erfolgreich verändert.

Beobachtungen / Log / Umgebung

Es wird beobachtet, wie der Nutzer die Sprache ändert, wie er sie ausgewählt hat.

Abhängigkeiten

<T100>

3.3.12 Testfall $\langle T1000 \rangle$ - Verlauf der (eigenen) Aktionen

Ziel

Es wird überprüft, ob der Verlauf des Nutzers korrekt angezeigt wird und welche Ressourcen er ausgeliehen oder hochgeladen hat.

Objekte/Methoden/Funktionen

Funktion: <F90>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn der Verlauf korrekt angezeigt wird.

Der Test wird als nicht bestanden betrachtet, wenn entweder ein falscher Verlauf angezeigt wird oder gar kein Verlauf angezeigt wird.

Vorbedingung

Nutzer muss erfolgreicher eingeloggt sein und danach auf 'Verlauf'-Button klicken.

Einzelschritte

Eingabe:

1. App öffnen
2. Auf 'Verlauf' Button klicken

Ausgabe: Der Verlauf wird angezeigt.

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob alle ausgeliehenen und hochgeladenen Ressourcen auf die Verlaufsseite aufgeführt sind.

Abhängigkeiten

<T100>

3.3.13 Testfall ⟨$T1110$⟩ - Anlegen von Benutzern und Mediathekaren

Ziel

Verwalter sind in der Lage, Benutzer- und Mediathekarkonten zur Mediathek hinzuzufügen.

Objekte/Methoden/Funktionen

Methode: anlegen

Funktion: <F100>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Liste der Konten angezeigt werden kann, damit der Administrator weitere Verwaltungsaufgaben durchführen kann.

Der Test ist fehlgeschlagen, wenn der hinzugefügte Nutzer nicht in der Liste auftaucht.

Vorbedingung

Verwalter muss sich erstmals als Administrator einloggen und auf den 'Kontenverwaltung'-Button klicken.

Einzelschritte

Eingabe:

1. Verwalter loggt sich auf 'Mediaverse' ein.
2. Auf 'Kontenverwaltung' Button klicken.
3. Verwalter geht auf Konto anlegen.
4. Die Daten werden eingetragen.
5. Der Verwalter klickt auf den Button 'speichern'. Ausgabe: 'Das Konto wurden angelegt.'

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob die Anlegung von Benutzern oder Mediathekaren korrekt funktioniert.

Abhängigkeiten

<T100>

3.3.14 Testfall ⟨T1120⟩ - Editieren von Benutzern und Mediathekaren

Ziel

Verwalter sind in der Lage, Benutzer- und Mediathekarkonten zu verwalten.

Objekte/Methoden/Funktionen

Methode: bearbeiten

Funktion: <F110>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn die Liste der Konten mit den Änderungen angezeigt werden kann, damit der Administrator weitere Verwaltungsaufgaben durchführen kann.

Der Test ist fehlgeschlagen, wenn der Verwaltungsversuch aufgrund von Netzwerkfehler fehlgeschlagen ist.

Vorbedingung

Verwalter muss sich erstmals als Administrator einloggen und auf den 'Kontenverwaltung'-Button klicken.

Einzelschritte

Eingabe:

1. Verwalter loggt sich auf 'Mediaverse' ein.
2. Auf 'Kontenverwaltung' Button klicken.
3. Verwalter wählen ein Konto aus.
4. Verwalter nimmt Äderungen vor und klickt auf den Button 'bestätigen'.

Ausgabe: 'Das Konto wurde geändert.'

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob die Bearbeitung von Benutzern oder Mediathekaren korrekt funktioniert.

Abhängigkeiten

<T100>

3.3.15 Testfall ⟨$T1130$⟩ - Löschen von Benutzern und Mediathekaren

Ziel

Verwalter sind in der Lage, Benutzer- und Mediathekarkonten zu löschen.

Objekte/Methoden/Funktionen

Methode: löschen

Funktion: <F110>

Pass/Fail Kriterien

Der Test ist erfolgreich, wenn in der Liste der Nutzer, der gelöschte Account nicht mehr auffindbar ist.

Der Test ist fehlgeschlagen, wenn der gelöschte Benutzer- oder Mediathekar sich noch in der Liste der Konten befindet.

Vorbedingung

Verwalter muss sich erstmals als Administrator einloggen und auf den 'Kontenverwaltung'-Button klicken.

Einzelschritte

Eingabe:

1. Verwalter loggt sich auf 'Mediaverse' ein.
2. Auf 'Kontenverwaltung' Button klicken.
3. Verwalter wählen ein Konto aus.
4. Auf den Button 'löschen' drücken.

Ausgabe: 'Das Konto wurde gelöscht.'

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob die Löschung von Benutzern oder Mediathekaren korrekt funktioniert.

Abhängigkeiten

<T100>

3.3.16 Testfall ⟨$T1200$⟩ - QR-Code

Ziel

Es wird getestet, ob der QR-Reader die Verfügbarkeit der Ressource korrekt ermittelt und dem Benutzer entsprechende Informationen liefert.

Objekte/Methoden/Funktionen

Funktion: <F120>

Pass/Fail Kriterien

Der Test gilt als bestanden, wenn nach dem Scannen des QR-Codes die entsprechende Ressource erfolgreich angezeigt wird.

Der Test wird als gescheitert betrachtet, wenn nach dem Scannen des QR-Codes keine Ressource angezeigt wird.

Vorbedingung

Der Nutzer ist in der App eingeloggt.

Einzelschritte

Eingabe:

1. App öffnen
2. Auf dem QR-Symbol klicken
3. Ein QR-Code anzeigen

Ausgabe: Die gewünschte Ressource wird dem Benutzer angezeigt.

Beobachtungen / Log / Umgebung

Es wird beobachtet, was nach dem Scannen des QR-Codes angezeigt wird.

Abhängigkeiten

<T100>

3.3.17 Testfall ⟨$T1300$⟩ - <Q30> Antwortzeit zum Anzeigen der Verfügbarkeit einer Ressource

Ziel

Es wird getestet, ob Mediaverse in den fünf Browsern Browsers funktioniert: Chrome, Firefox, Safari und Edge. Auf diese Weise wird sichergestellt, dass die Funktionen und Komponenten in verschiedenen Browsern einheitlich ausgeführt werden können.

Objekte/Methoden/Funktionen

Funktionen: Alle

Pass/Fail Kriterien

Der Test gilt als erfolgreich, wenn die wichtigsten Funktionen wie Suchmaschine, Anmeldung, Registrierung, Nutzung von Ressourcen, Hinzufügen/Löschen/Entfernen von Ressourcen ohne Verzögerungen oder Probleme wie nicht angezeigte Bilder oder nicht funktionierende Links ausgeführt werden können.

Der Test gilt als nicht erfolgreich, wenn einer der oben genannten Schritte nicht ausgeführt werden kann.

Vorbedingung

Ein Rechner mit Internetanschluss und der Fähigkeit, die oben genannten 5 verschiedenen Browser zu verwenden.

Einzelschritte

Eingabe:

1. Ressourcensuche mittels QR-Code Scan, Textsuche oder Eingabe der ID-Nummer
2. Klicken auf eine Ressource.
3. Weiterleitung auf neue Seite zur ausgewählten Ressource.

Ausgabe: Die Verfügbarkeit von der gewünschten Ressource wird innerhalb von 5 Sekunden Benutzer angezeigt.

Beobachtungen / Log / Umgebung

Es wird beobachtet, wie lange es nach erfolgreichem Laden der Infoseite der Ressource dauern wird, bis die Verfügbarkeit angezeigt wird.

Abhängigkeiten

<T210>, <T220>, <T300>

4 Integrationstest

Dieses Kapitel enthält die Beschreibung der Integrationstests. Hiermit wird die Kommunikation und Zusammenarbeit zwischen den unterschiedlichen Komponenten getestet.Die Tests sollen dazu dienen, etwaige Probleme zu erkennen und die allgemeine Benutzerfreundlichkeit der Website zu verbessern.

4.1 Zu testende Komponenten

Nr	**Komponenten**	**Testfälle**	**Kommentar**
1	<C10> Benutzerverwaltung	<T1200>, <T1210>, <T1500>,	
2	<C20> Ressourcenverwaltung	<T1200>, <T1300>, <T1500>,	
3	<C30> Suchfunktion	<T1210>, <T1300>, <T1400>, <T1600>	
4	<C40> Ausleihverwaltung	<T1600>	
5	<C50> Favoritenverwaltung	<T1400>, <T1410>, <T1420>	
6	<C60>Benutzeraktivitätenverfolgung	<1700>	
7	<C70> Sprachverwaltung	<T1100>	

4.2 Testverfahren

Es werden Black-Box-Tests durchgeführt, um die Integration der verschiedenen Komponenten in die Mediathek-Website zu prüfen. Diese Tests konzentrieren sich auf die Interaktion zwischen der verschiedenen Komponenten.

4.2.1 Testskripte

Es ist nicht geplant, Testskripte zu verwenden; falls deren Implementierung erforderlich sein sollte, werden die Einzelheiten und Informationen dazu in den nachfolgenden Dokumenten erläutert werden. Ziel ist es, eine reibungslose Integration und Funktionalität zu gewährleisten, ohne auf die interne Funktionsweise der Komponenten zuzugreifen.

4.3 Testfälle

4.3.1 Testfall $\langle T1100 \rangle$ - Komponente C10+C70

Ziel

Testen, ob die Verwaltungssprache nach den Wünschen des Benutzers geändert werden kann.

Objekte/Methoden/Funktionen

<F20>, <F80>

Pass/Fail Kriterien

Pass: Benutzer ist eingeloggt und die Sprache könnte nach seinem Auswahl geändert werden.

Fail: Benutzer könnte nicht einloggen/Nach erfolgreichem Einloggen könnte die Sprache nicht geändert werden.

Vorbedingung

Benutzer ist registriert

Einzelschritte

1. Zur Sprachverwaltung gehen
2. Eine von 7 angebotenen Sprachen auswählen
3. Sprache ändern

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob die Sprachänderung für alle Seiten gilt

Besonderheiten

keine Besonderheiten

Abhängigkeiten

keine Abhängigkeiten

4.3.2 Testfall $\langle T1200 \rangle$ - Komponente C10 + C20

Ziel

Registrierung und Anmeldung als Mediathekar in Mediaverse und Überprüfung, ob die dieser Ressourcen verwalten kann.

Objekte/Methoden/Funktionen

<F10>, <F20>, <F30>

Pass/Fail Kriterien

Pass: Die Registrierung und Anmeldung war erfolgreich und der Mediathekar wir auf die Homepage weitergeleitet, wo er eine weitere Seite zur Ressourcenerstellung vorfindet und neue Ressourcen hinzufügen kann.

Fail: Fehler bei der Registrierung oder bei der Anmeldung, sowie keine weitere Seite für die Ressourcenerstellung.

Vorbedingung

Es sind keine Vorbedingungen nötig.

Einzelschritte

1. Aufrufen der Registrierungsseite.
2. Eintragung der Nutzerdaten wie Benutzername, Passwort, Passwort Wiederholen und der E-Mail.
3.Aktivierung des Accounts sowie vergeben weiterer Rechte vom Administrator.
4. Anmeldung mit dem Benutzernamen und Passwort.
5. Nach der Weiterleitung auf die Homepage, aufrufen der Seite "Create Resource".
6. Eintragung der Felder und auf Create klicken.

Beobachtungen / Log / Umgebung

Beobachten, ob die Registrierung und Anmeldung erfolgreich verläuft und ob der Mediathekar auf die Seite "Create ResourceSZugriff hat und weitere Ressourcen hinzufügen kann.

Besonderheiten

Es gibt keine Besonderheiten.

Abhängigkeiten

<1210>

4.3.3 Testfall $\langle T1210 \rangle$ - Komponente C10 + C30

Ziel

Registrierung und Anmeldung als Benutzer in Mediaverse und Überprüfung, ob die Suchmaschine dem Benutzer die relevanten Ergebnisse liefert.

Objekte/Methoden/Funktionen

<F10>, <F20>, <F60>

Pass/Fail Kriterien

Pass: Die Registrierung und Anmeldung war erfolgreich und der Benutzer wir auf die Homepage weitergeleitet, wo er nach beliebigen Ressourcen suchen kann und geeignete Ergebnisse von der Suchmaschine geliefert bekommt.

Fail: Fehler bei der Registrierung oder bei der Anmeldung, sowie Ausgabe der Falschen Suchbergebnisse.

Vorbedingung

Es sind keine Vorbedingungen nötig.

Einzelschritte

1. Aufrufen der Registrierungsseite.
2. Eintragung der Nutzerdaten wie Benutzername, Passwort, Passwort Wiederholen und der E-Mail.
3.Aktivierung des Accounts vom Administrator.
4. Anmeldung mit dem Benutzernamen und Passwort.
5. Nach der Weiterleitung auf die Homepage, Eintragung des Suchbegriffs in das Suchfeld.
6. Auf Suche klicken.

Beobachtungen / Log / Umgebung

Beobachten, ob die Registrierung und Anmeldung erfolgreich verläuft und ob die Suche die richtigen Ergebnisse liefert.

Besonderheiten

Es gibt keine Besonderheiten.

Abhängigkeiten

<1200>

4.3.4 Testfall $\langle T1300 \rangle$ - Komponente C20+C30

Ziel

Suche nach einer Ressource und Überprüfung, ob die Suchmaschine relevante Ergebnisse liefert.

Objekte/Methoden/Funktionen

<F30>, <F70>

Pass/Fail Kriterien

Pass: Die Suchergebnisse werden angezeigt, die das gesuchte Schlüsselwort enthalten

Fail: Die Suchergebnisse werden angezeigt, die nicht das gesuchte Schlüsselwort enthalten.

Vorbedingung

Der Benutzer ist eingeloggt und hat Zugriff auf die Suchmaschine.

Einzelschritte

1.Aufrufen der Mediaverse.

2.Eingabe eines Schlüsselworts (z. B. Python) in die Suchzeile.

3.Anklicken der Schaltfläche Suche.

Beobachtungen / Log / Umgebung

Beobachten, ob die Suchergebnisseite angezeigt wird und ob die Suchergebnisse das Schlüsselwort enthalten.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T1400>, <1200>, <1210>

4.3.5 Testfall $\langle T1400 \rangle$ - Komponente C30+C50

Ziel

Testen, ob eine Ressource aus den Suchergebnissen in die Favoritenliste aufgenommen werden kann.

Objekte/Methoden/Funktionen

<F30>, <F70>

Pass/Fail Kriterien

Pass: Die gewählte Ressource erscheint in der Favoritenliste des Benutzers.

Fail: Die ausgewählte Ressource erscheint nicht in der Favoritenliste des Benutzers .

Vorbedingung

Der Benutzer ist eingeloggt und hat Zugriff auf die Suchmaschine und die Favoritenfunktion, Testfall <T1300> wurde durchgeführt.

Einzelschritte

1.Anklicken einer Ressource in den Suchergebnissen.

2.Überprüfen, ob der Benutzer auf die Ressourcenseite umgeleitet wird.

3.Klicken auf das Sternsymbol, um die Ressource zur Favoritenliste hinzuzufügen.

Beobachtungen / Log / Umgebung

Beobachten, ob die Ressource erfolgreich zur Favoritenliste hinzugefügt wurde.(Das Sternsymbol sollte sich ändern, um anzuzeigen, dass die Ressource hinzugefügt wurde.)

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T1300>, <T2500>, <T3300>

4.3.6 Testfall $\langle T1410 \rangle$ - Komponente C30+C50

Ziel

Testen, ob die personalisierte Favoritenliste angezeigt werden kann.

Objekte/Methoden/Funktionen

<F30>, <F70>

Pass/Fail Kriterien

Pass: Die personalisierte Favoritenliste des Benutzers wird angezeigt.

Fail: Die personalisierte Favoritenliste des Benutzers wird nicht angezeigt.

Vorbedingung

Der Benutzer ist eingeloggt und hat Zugriff auf die Suchmaschine, Testfall <T1400> wurde durchgeführt.

Einzelschritte

1.Aufrufen der Seite der Favoritenliste.

Beobachtungen / Log / Umgebung

Beobachten, ob :

1. Die Seite mit der Favoritenliste angezeigt wird.

2. Alle hinzugefügten Ressourcen in der Favoritenliste angezeigt werden.

3 Die Seite der Favoritenliste eine Option zum Löschen von Ressourcen bietet.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T1400>

4.3.7 Testfall $\langle T1420 \rangle$ - Komponente C30+C50

Ziel

Testen, ob Ressourcen aus der Favoritenliste gelöscht werden können.

Objekte/Methoden/Funktionen

<F30>, <F70>

Pass/Fail Kriterien

Pass: Die aus der Favoritenliste ausgewählte Ressource wurde aus der Favoritenliste entfernt.

Fail: Die aus der Favoritenliste ausgewählte Ressource wurde aus der Favoritenliste nicht entfernt.

Vorbedingung

Testfall <T1410> sollte bereits für die Anzeige der personalisierten Favoritenliste durchgeführt werden.

Einzelschritte

1.Eine Ressource aus der Favoritenliste auswählen, die gelöscht werden soll.

2.Auf das Symbol der Option "Löschen"klicken.

Beobachtungen / Log / Umgebung

Es ist zu beachten, dass die ausgewählte Ressource aus der Favoritenliste entfernt wird, und überprüft werden, ob die Favoritenliste aktualisiert wurde und die gelöschte Ressource nicht mehr angezeigt wird.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T1400>, <T1410>, <T3400>

4.3.8 Testfall $\langle T1500 \rangle$ - Komponente C10+C20

Ziel

Es wird überprüft, ob die Benutzerverwaltung ordnungsgemäß mit der Ressourcenverwaltung integriert ist, indem überprüft wird, ob der erstellte Benutzer die erforderlichen Berechtigungen hat, um Ressourcen hinzuzufügen, zu editieren oder zu löschen.

Objekte/Methoden/Funktionen

<F10>,<F20>,<F60>

Pass/Fail Kriterien

Pass: Der Verwalter (Mediathekar) hat die erforderlichen Berechtigungen, um Ressourcen hinzuzufügen, zu editieren und zu löschen.

Fail: Mediathekar hat nicht die erforderlichen Berechtigungen, um Ressourcen hinzuzufügen, zu editieren und zu löschen oder ein normaler Benutzer kann solche Aktionen durchführen.

Vorbedingung

Der Verwalter ist registriert und eingeloggt

Einzelschritte

1. Der Verwalter loggt sich auf ‘Mediaverse’ als Mediathekar ein.
2. Der Verwalter versucht, mithilfe der Button"create Resources"neue Ressourcen hinzuzufügen.

Beobachtungen / Log / Umgebung

Es wird beobachtet, ob die entsprechende Berechtigung je nach Benutzertyp gewährt wird.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T2000>, <T2100>

4.3.9 Testfall $\langle T1600 \rangle$ - Komponente C30+C40

Ziel

Es wird Überprüft, ob die Suchfunktion korrekt mit der Ausleihverwaltung integriert ist, indem eine Ressource gesucht, ausgeliehen und die Verfügbarkeit korrekt verwaltet wird.

Objekte/Methoden/Funktionen

<F30>, <F40>

Pass/Fail Kriterien

Pass: Die Ressource wird erfolgreich gesucht, ausgeliehen und die Verfügbarkeit korrekt verwaltet.

Fail: Die Ressource kann nicht gefunden oder ausgeliehen werden, oder die Verfügbarkeit wird nicht korrekt verwaltet.

Vorbedingung

Der Verwalter ist eingeloggt und es sind Ressourcen in der Anwendung vorhanden.

Einzelschritte

1.Suche nach einer Ressource mit der Funktion search() basierend auf bestimmten Kriterien (ID,Keyword,Hardware,QR)

2.Wähle eine Ressource aus den Suchergebnissen.

3.Versuche, die gewählte Ressource mit der Ausleihfunktion auszuleihen.

Beobachtungen / Log / Umgebung

Es wurde beobachtet, dass die Ressource nach der Ausleihe als 'ausgeliehen' markiert wird und nicht mehr in den Suchergebnissen für Ausleihen angezeigt wird.

Besonderheiten

Keine Besonderheiten

Abhängigkeiten

<T1500>, <T1210>

4.3.10 Testfall $\langle T1700 \rangle$ - Komponente c40+c60

Ziel

Es wird getestet, ob durch die Ausleihfunktion und die Benutzeraktivitätenverfolgung richtig angezeigt wird.

Objekte/Methoden/Funktionen

<F40>, <F90>

Pass/Fail Kriterien

Pass: Die Ausleihaktivitäten werden nicht angezeigt, obwohl etwas ausgeliehen wurde.
Fail: Alle Ausleihaktivitäten werden mit den richtigen Informationen, wie Datum der Ausleihe angezeigt.

Vorbedingung

Der Benutzer muss angemeldet sein und sich eine ausleihbare Ressource aussuchen.

Einzelschritte

1. Wähle eine Ressource aus.
2. Klicke auf den Button ausleihen.
3. Navigiere zu der Ressourcen Liste.

Beobachtungen / Log / Umgebung

Die ausgeliehene Ressource sollte in der Ressourcen Liste zu finden sein mit dem vermerk, wann es wieder zurückgegeben werden muss.

Besonderheiten

Keine Besonderheiten

Abhängigkeiten

Es gibt keine Abhängigkeiten

5 Unit-Tests

In diesem Kapitel werden die Unit-Tests für die Komponenten in Mediaverse beschrieben. Das Ziel dieser Tests ist es, sicherzustellen, dass jede einzelne Komponente ordnungsgemäß funktioniert und die erwarteten Ergebnisse liefert.

Die zu testenden Komponenten umfassen die Benutzerverwaltung, Ressourcenverwaltung, Suchfunktion, Favoritenverwaltung und Ausleihverwaltung. In der folgenden Liste werden die einzelnen Komponenten und die zugehörigen zu testenden Klassen beschrieben. Anschließend wird auf die Testverfahren und Testskripte eingegangen. Jeder Testfall wird detailliert erläutert, um sicherzustellen, dass alle getesteten Methoden korrekt funktionieren.

Das übergeordnete Ziel der Unit-Tests ist es, die Funktionalität jeder Komponente in Mediaverse auf ihre korrekte Arbeitsweise zu überprüfen. Dadurch wird gewährleistet, dass die einzelnen Funktionen einwandfrei arbeiten und das Gesamtsystem stabil bleibt.

5.1 Zu testende Komponenten

Nr	Komponenten	Testfälle	Kommentar
1	<10> Benutzerverwaltung	<T1700>, <T1800>, <T1810>, <T1900>, <T2000>	Es wird getestet, ob die Authentifizierungsfunktion ordnungsgemäß funktioniert.
2	<20> Ressourcenverwaltung	<T2100>, <T2200>, <T2300>, <T2400>	Es wird getestet, ob der Backend-Server ordnungsgemäß funktioniert.
3	<30> Suchfunktion	<T2500>, <T2600>, <T2700>	Es wird getestet, ob die Frontend-Benutzeroberfläche ordnungsgemäß funktioniert.

4	<40> Ausleihverwaltung	<T2800>, <T2900>, <T3000>, <T3100>	Es wird getestet, ob die Nutzung von verschiedenen Ressourcentypen ordnungsgemäß funktioniert.
5	<50> Favoritenverwaltung	<T3200>, <T3300>, <T3400>	Es wird getestet, ob die Ressourcen Liste ordnungsgemäß funktioniert.

5.2 Testverfahren

Um die Unit-Tests durchzuführen, verwenden wir das entsprechende Python unittest-Modul sowie die integrierte Python HTTP-Bibliothek. Darüber hinaus wird Swagger als Hilfsmittel verwendet, um API-Tests zu unterstützen. Zum Beispiel kann Swagger verwendet werden, um bei der Erstellung von Testfällen zu helfen.

5.2.1 Testskripte

Für die Durchführung von Tests in Python werden Testskripte mithilfe des Python-Moduls “unittest” erstellt.Die Unittests werden im “tests”-Ordner in jedem Modulordner des Projekts bereitgestellt und von dort mit dem allgemeinen Djangotestbefehl python manage.py test ausgeführt.

5.3 Testfälle

5.3.1 Testfall $\langle T1700 \rangle$ - test_create_user_success

Ziel

Testen, ob das Erstellen eines Benutzers erfolgreich ist.

Objekte/Methoden/Funktionen

- create_groups.py

Pass/Fail Kriterien

Pass:

- Der Benutzer wird erfolgreich erstellt.
- Der HTTP-Statuscode ist 201 CREATED.
- Das Passwort des Benutzers wird erfolgreich überprüft.
- Das Antwortdatenobjekt enthält nicht das Feld "password".

Fail:

- Das Erstellen des Benutzers oder die Überprüfung des Passworts schlägt fehl.

Vorbedingung

Eine Testdatenbank wurde initialisiert.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test user.tests.PublicUserApiTests.test_create_user_success”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

- Es werden die Standard-Django-Testtools verwendet.
- Der Test basiert auf einem Unit-Test-Framework.

Abhängigkeiten

<T110>, <T120>

5.3.2 Testfall ⟨$T1800$⟩ - test_create_token_for_activated_user

Ziel

Testen, ob bei gültigen Anmeldeinformationen, die vom Administrator aktiviert wurden, ein Token zurückgegeben wird.

Objekte/Methoden/Funktionen

• create_user

Pass/Fail Kriterien

Pass:

• Für gültige Anmeldeinformationen, die vom Administrator aktiviert wurden, wird ein Token zurückgegeben. • Der HTTP-Statuscode ist 200 OK.

Fail:

• Bei gültigen Anmeldeinformationen, die vom Administrator aktiviert wurden, wird kein Token zurückgegeben.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden und wurde vom Administrator aktiviert.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test user.tests.AuthenticationApiTests.test_create_token_for_activated_user”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

• Der Test überprüft, ob ein Token für einen vom Administrator aktivierten Benutzer erstellt wird.

Abhängigkeiten

<T1700> test_create_user_success

5.3.3 Testfall ⟨$T1810$⟩ - test_create_token_for_non_activated_user

Ziel

Testen, ob bei gültigen Anmeldedaten, die nicht vom Administrator aktiviert wurden, ein Fehler zurückgegeben wird.

Objekte/Methoden/Funktionen

- API-Endpunkt: `api/token/`

Pass/Fail Kriterien Pass:

- Die Anfrage mit ungültigen Anmeldedaten wurde abgelehnt.
- Das Antwortdatenobjekt enthält kein Token.
- Der HTTP-Statuscode ist 400 BAD REQUEST.

Fail:

- Die Anfrage mit ungültigen Anmeldedaten wurde nicht abgelehnt oder ein Token wurde zurückgegeben.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden, wurde jedoch nicht vom Administrator aktiviert.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test user.tests.PublicUserApiTests.test_create_token_for_non_activated_user”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

keine

5.3.4 Testfall ⟨T1900⟩ - test_update_user_profile

Ziel

Testen, ob das Aktualisieren des Benutzerprofils für den authentifizierten Benutzer erfolgreich ist.

Objekte/Methoden/Funktionen

- Modelle: Benutzermodell
- API-Endpunkt: `api/me/`

Pass/Fail Kriterien

Pass:

- Das Benutzerprofil wurde erfolgreich aktualisiert und alle Änderungen wurden übernommen.
- Der HTTP-Statuscode ist 200 OK.

Fail:

- Das Aktualisieren des Benutzerprofils ist fehlgeschlagen oder die Änderungen wurden nicht korrekt übernommen.

Vorbedingung

Ein Benutzer ist authentifiziert und eingeloggt.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test user.tests.PrivateUserApiTests.test_update_user_profile”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

<T1700>

5.3.5 Testfall $\langle T2000 \rangle$ - test_retrieve_profile_success

Ziel

Testen, ob das Abrufen des Profils für den eingeloggten Benutzer erfolgreich ist.

Objekte/Methoden/Funktionen

- Keine

Pass/Fail Kriterien

Pass:

- Das Profil des Benutzers wird erfolgreich abgerufen.
- Der HTTP-Statuscode ist 200 OK.
- Die zurückgegebenen Daten entsprechen den erwarteten Werten.

Fail:

- Das Abrufen des Profils schlägt fehl oder die zurückgegebenen Daten entsprechen nicht den erwarteten Werten.

Vorbedingung

Ein Benutzer ist eingeloggt und verfügt über ein Profil.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test user.tests.UserProfileApiTests.test_retrieve_profile_success”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

keine

5.3.6 Testfall $\langle T2100 \rangle$ - test_create_resource

Ziel

Testen, ob das Erstellen einer Ressource erfolgreich ist.

Objekte/Methoden/Funktionen

create_resource(user, **params)

Pass/Fail Kriterien

Pass:

- Die Ressource wird erfolgreich erstellt.
- Der HTTP-Statuscode ist 201 CREATED.

Fail:

- Das Erstellen der Ressource schlägt fehl.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:
"python manage.py test resource.tests.test_resource_api.PrivateResourceeApiTests.test _create_resource"

Beobachtungen / Log / Umgebung

In der Konsole wird "OK" ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

- Es werden die Standard-Django-Testtools verwendet.
- Der Test basiert auf einem Unit-Test-Framework.

Abhängigkeiten

- Die Methode create_resource muss korrekt implementiert sein.
- Der Benutzer muss in der Datenbank vorhanden sein.

5.3.7 Testfall ⟨$T2200$⟩ - test_delete_resource

Ziel

Testen, ob das Löschen einer Ressource erfolgreich ist.

Objekte/Methoden/Funktionen

- creat_resource(user, **params).
- detail_url(resource_id).

Pass/Fail Kriterien

Pass:

- Die Ressource wird erfolgreich gelöscht.
- Der HTTP-Statuscode ist 204 NO CONTENT.
- Die Ressource existiert nicht mehr in der Datenbank.

Fail:

- Das Löschen der Ressource schlägt fehl.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden und eine Ressource wurde erstellt.

Einzelschritte

1.Das Terminal öffnen.

2.In das Verzeichnis des Projekts wechseln.

3.Den entsprechenden Django-Unittest mit dem Befehl:

”python manage.py test resource.tests.test_resource_api.PrivateResourceeApiTests.test _delete_resource”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

- Es werden die Standard-Django-Testtools verwendet.
- Der Test basiert auf einem Unit-Test-Framework.

Abhängigkeiten

- Die Methode create_resource muss korrekt implementiert sein.
- Die Methode detail_url muss korrekt implementiert sein.
- Der Benutzer muss in der Datenbank vorhanden sein.
- Es muss mindestens eine Ressource vorhanden sein, die dem Benutzer gehört.

5.3.8 Testfall $\langle T2300 \rangle$ - - test_update_resource

Ziel

Testen, ob das Aktualisieren einer Ressource erfolgreich ist.

Objekte/Methoden/Funktionen

- create_resource(user, **params)
- detail_url(resource_id)

Pass/Fail Kriterien

Pass:

- Die Ressource wird erfolgreich aktualisiert.
- Der HTTP-Statuscode ist 200 OK.
- Die Ressource enthält die aktualisierten Daten gemäß dem Payload.

Fail:

- Das Aktualisieren der Ressource schlägt fehl.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden und eine Ressource wurde erstellt.

Einzelschritte

1.Das Terminal öffnen.

2.In das Verzeichnis des Projekts wechseln.

3.Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test resource.tests.test_resource_api.PrivateResourceeApiTests.test _update_resource”

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

- Es werden die Standard-Django-Testtools verwendet.
- Der Test basiert auf einem Unit-Test-Framework.

Abhängigkeiten

- Die Methode create_resource muss korrekt implementiert sein.
- Die Methode detail_url muss korrekt implementiert sein.
- Der Benutzer muss in der Datenbank vorhanden sein.
- Es muss mindestens eine Ressource vorhanden sein, die dem Benutzer gehört.

5.3.9 Testfall $\langle T2400 \rangle$ - test_create_resource_only_mediathekar

Ziel

Testen, ob nur Mediathekar-Benutzer Ressourcen erstellen können.

Objekte/Methoden/Funktionen

- create_user(**params).
- Resources_URL.

Pass/Fail Kriterien

Pass:

- Der nicht als Mediathekar markierte Benutzer kann keine Ressourcen erstellen.
  - Der HTTP-Statuscode ist 403 FORBIDDEN.
  - Die Ressource wurde nicht erstellt.

Fail:

- Der nicht als Mediathekar markierte Benutzer kann Ressourcen erstellen.

Vorbedingung

Ein Benutzer ist in der Datenbank vorhanden, ist nicht als Mediathekar markiert und ist authentifiziert.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test resource.tests.test_resource_api.PrivateResourceeApiTests.test _create_resource_only_mediathekar”.

Beobachtungen / Log / Umgebung

In der Konsole wird ”OK” ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

- Es werden die Standard-Django-Testtools verwendet.
- Der Test basiert auf einem Unit-Test-Framework.

Abhängigkeiten

- Die Methode create_user muss korrekt implementiert sein.
- Die Variable Resources_URL muss korrekt gesetzt sein.
- Der Benutzer muss in der Datenbank vorhanden sein und nicht als Mediathekar markiert

sein.

5.3.10 Testfall $\langle T2500 \rangle$ - test_search_resource

Ziel

Testen, ob die gesuchte Ressource ausgegeben wird.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt

Pass/Fail Kriterien

Pass:

- Die gesuchte Ressource wurde gefunden und erfolgreich angezeigt.

Fail:

- Die gesuchte Ressource wurde nicht gefunden. Die Suche ist fehlgeschlagen.

Vorbedingung

Es existieren Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test resource.tests.SearchResourceTestCase.test_search_resource eingeben.

Beobachtungen / Log / Umgebung

In der Konsole wird ’OK’ ausgegeben, wenn der Test erfolgreich ist, oder ’Failure’, wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

keine

5.3.11 Testfall ⟨$T2600$⟩ - test_search_resource_with_filter

Ziel

Testen, ob die Suche nach der Ressource mit Anwendung der Filter-Funktion erfolgreich ausgegeben wird.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt

Pass/Fail Kriterien

Pass:

- Es werden Ressourcen mit der eingegebenen Beschränkung durch den Filter erfolgreich gefunden und angezeigt.

Fail:

- Die gesuchte Ressource mit Beschränkung durch die Filterung wurde nicht gefunden. Die Filterung ist fehlgeschlagen.

Vorbedingung

Es existieren Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

"python manage.py test resource.tests.SearchResourceTestCase.test_search_resource_filter" eingeben

Beobachtungen / Log / Umgebung

In der Konsole wird 'OK' ausgegeben, wenn der Test erfolgreich ist, oder 'Failure', wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T2500>

5.3.12 Testfall ⟨$T2700$⟩ - test_search_resource_with_qrcode

Ziel

Testen, ob die Suche nach der Ressource mit Anwendung der QR-Code Funktion erfolgreich ausgegeben wird.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt

Pass/Fail Kriterien

Pass:

- Es werden Ressourcen mit dem eingegebenen QR-Code erfolgreich gefunden und angezeigt.

Fail:

- Die gesuchte Ressource mit dem eingegebenen QR-Code wurde nicht gefunden. Die Suche ist fehlgeschlagen.

Vorbedingung

Es existieren Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test resource.tests.SearchResourceTestCase.test_search_resource_qrcode” eingeben

Beobachtungen / Log / Umgebung

In der Konsole wird ’OK’ ausgegeben, wenn der Test erfolgreich ist, oder ’Failure’, wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

<T2500>

5.3.13 Testfall ⟨$T2800$⟩ - test_borrow_resource

Ziel

Testen, ob die Ausleihe erfolgreich verarbeitet werden kann.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt

Pass/Fail Kriterien

Pass:

- Die HTTP-Antwort hat den Statuscode 200 (HTTP_200_OK).
- Es existiert genau eine Ausleihtransaktion in der Datenbank.
- Die Ausleihtransaktion ist dem richtigen Benutzer (self.user) zugeordnet.
- Das Attribut "returned"der Ausleihtransaktion ist auf "False"gesetzt.

Fail:

- Die HTTP-Antwort hat einen anderen als den erwarteten Statuscode, z.B. 400 (HTTP_400_BAD_REQUEST).
- Es existiert keine Ausleihtransaktion in der Datenbank.

Vorbedingung

Es existiert ein Ressourcenobjekt und ein Benutzerobjekt. Der Benutzer ist authentifiziert und eingeloggt.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test borrow.tests.BorrowResourceTestCase.test_borrow_resource”

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

5.3.14 Testfall $\langle T2900 \rangle$ - test_borrow_resource_not_available

Ziel

Überprüfen, ob die Funktion zur Ausleihe einer Ressource korrekt auf die Nichtverfügbarkeit der Ressource reagiert.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt

Pass/Fail Kriterien

Pass:

- Wenn die Funktion korrekt auf die Nichtverfügbarkeit der Ressource reagiert und eine entsprechende Fehlermeldung zurückgibt, gilt der Test als bestanden.

Fail:

- Wenn die Funktion nicht korrekt auf die Nichtverfügbarkeit der Ressource reagiert oder keine Fehlermeldung zurückgibt, gilt der Test als fehlgeschlagen.

Vorbedingung

Es existiert ein Ressourcenobjekt und ein Benutzerobjekt. Der Benutzer ist authentifiziert und eingeloggt. Die Ressource ist als nicht verfügbar markiert.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:
"python manage.py test borrow.tests.BorrowResourceTestCase.test_borrow_resource_not_available"

Beobachtungen / Log / Umgebung

In der Konsole wird "OK" ausgegeben, wenn der Test erfolgreich ist, oder "Failure", wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

keine Abhängigkeiten

5.3.15 Testfall $\langle T3000 \rangle$ - test_borrow_resource_invalid_resource

Ziel

Überprüfen, ob die Funktion zur Ausleihe einer Ressource korrekt auf eine ungültige Ressourcen-ID reagiert und die entsprechende Fehlermeldung zurückgibt.

Objekte/Methoden/Funktionen

- Ungültige Ressourcen-ID

Pass/Fail Kriterien

Pass:

- Wenn die Funktion korrekt auf eine ungültige Ressourcen-ID reagiert und den erwarteten HTTP-Statuscode "404 Not Found"liefert, gilt der Test als bestanden.

Fail:

- Wenn die Funktion nicht korrekt auf eine ungültige Ressourcen-ID reagiert oder einen anderen Statuscode zurückgibt, gilt der Test als fehlgeschlagen.

Vorbedingung

Keine besonderen Vorbedingungen erforderlich.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test borrow.tests.BorrowResourceTestCase.test_borrow_resource”

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

keine Abhängigkeiten

5.3.16 Testfall ⟨$T3100$⟩ - test_borrow_resource_due_date

Ziel

Überprüfen, ob das Ausleihdatum einer Ressource korrekt auf das aktuelle Datum plus 14 Tage festgelegt wird.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Ausleihdatum

Pass/Fail Kriterien

Pass:

- Wenn das Ausleihdatum korrekt auf das erwartete Datum plus 14 Tage festgelegt wird und der HTTP-Statuscode "200 OKßurückgegeben wird, gilt der Test als bestanden.

Fail:

- Wenn das Ausleihdatum nicht korrekt festgelegt wird oder ein anderer Statuscode zurückgegeben wird, gilt der Test als fehlgeschlagen.

Vorbedingung

Stellen Sie sicher, dass eine Ressource mit der angegebenen Ressourcen-ID oder Name vorhanden ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test borrow.tests.BorrowResourceTestCase.test_borrow_resource_due_date”

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine Besonderheiten

Abhängigkeiten

keine Abhängigkeiten

5.3.17 Testfall ⟨$T3200$⟩ - test_add_resource_to_favouritelist

Ziel

Testen, ob das Hinzufügen einer Ressource zu der Favoritenliste erfolgreich funktioniert.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt
- Favoritenlistenobjekt

Pass/Fail Kriterien

Pass:

- Die ausgewählte Ressource wurde erfolgreich in die Favoritenliste gelegt und dem Benutzer angezeigt.

Fail:

- Die auserwählte Ressource wurde nicht erfolgreich in die Favoritenliste gelegt. Dem Benutzer wird seine ausgewählte Ressource nicht in der Favoritenliste angezeigt.

Vorbedingung

Es existieren ein Favoritenlistenobjekt, Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:

“python manage.py test favourite.tests.FavouriteResourceTestCase.test_add_resource’ eingeben.

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

keine

5.3.18 Testfall ⟨$T3300$⟩ - test_remove_resource_from_favouritelist

Ziel

Testen, ob das Entfernen einer Ressource zu der Favoritenliste erfolgreich funktioniert.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt
- Favoritenlistenobjekt

Pass/Fail Kriterien

Pass:

- Die ausgewählte Ressource wurde erfolgreich aus der Favoritenliste entfernt. Dem Benutzer wird die ausgewählte Ressource nicht mehr in seiner Favoritenliste angezeigt.

Fail:

- Die ausgewählte Ressource wurde nicht erfolgreich aus der Favoritenliste entfernt. Dem Benutzer wird die ausgewählte Ressource immer noch in seiner Favoritenliste angezeigt.

Vorbedingung

Es existieren ein Favoritenlistenobjekt, Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:
“python manage.py test favourite.tests.FavouriteResourceTestCase.test_remove_resource_favouritelist’
eingeben.

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

<T3300>

5.3.19 Testfall ⟨T3400⟩ - test_share_items_from_favouritelist

Ziel

Testen, ob das Teilen von Ressourcen aus der Favoritenliste erfolgreich funktioniert.

Objekte/Methoden/Funktionen

- Ressourcenobjekt
- Userobjekt
- Favoritenlistenobjekt

Pass/Fail Kriterien

Pass:

- Der Benutzer konnte eine Ressource aus seiner Favoritenliste erfolgreich mit anderen Benutzern teilen.

Fail:

- Der Benutzer konnte eine Ressource aus seiner Favoritenliste nicht erfolgreich teilen.

Vorbedingung

Es existieren ein Favoritenlistenobjekt, Ressourcenobjekte sowie ein Benutzerobjekt, das erfolgreich authentifiziert und eingeloggt ist.

Einzelschritte

1. Das Terminal öffnen.
2. In das Verzeichnis des Projekts wechseln.
3. Den entsprechenden Django-Unittest mit dem Befehl:
“python manage.py test favourite.tests.FavouriteResourceTestCase.test_share_resource_favouritelist’
eingeben

Beobachtungen / Log / Umgebung

In der Konsole wird “OK” ausgegeben, wenn der Test erfolgreich ist, oder “Failure”, wenn der Test fehlschlägt.

Besonderheiten

keine

Abhängigkeiten

<T3300>

6 Glossar

Android: Betriebssystem für mobile Geräte wie Smartphones und Tablets.

Client: Software oder Gerät, das mit einem Server kommuniziert. Linux-Ubuntu System: Betriebssystem basierend auf Linux.

EDV-Kenntnisse: Die Abkürzung EDV steht für „elektronische Datenverarbeitung“. Damit sind alle Vorgänge gemeint, die wir am PC und anderen digitalen Medien vollziehen. EDV Kenntnisse sind also ein Synonym für Computerkenntnisse oder IT-Kenntnisse.

Homepage(Startseite): Erste Seite einer Website mit Informationen und Links.

IEEE 829-Standards:Ein Dokumentationsstandard für Software- und Systemtests. Er definiert die Anforderungen an Testdokumentationen, einschließlich Testplänen, Testfällen, Testberichten und Testprotokollen.

iOS: Betriebssystem für iPhones, iPads und iPod Touch.

Linux: Ein Open-Source-Betriebssystem, das auf der Unix-Philosophie basiert und weit verbreitet in der Software-Entwicklung eingesetzt wird.

Macbook: Laptop von Apple.

MacOS: Betriebssystem für Mac-Computer von Apple.

Windows 11: Das neueste Betriebssystem von Microsoft, das im Jahr 2021 veröffentlicht wurde.

Webbasierte Applikation: Software, die über einen Webbrowser läuft.