

LeihSy -

Das digitale Verleihsystem der Hochschule Esslingen

Dokumentation der Projektarbeit
im Studiengang
Softwaretechnik und Medieninformatik

vorgelegt von

Asinas Esber

Matr.-Nr.: 768274

Ceyda Yoldas

Matr.-Nr.: 768881

Dennis Mika

Matr.-Nr.: 777972

Yannick Jacobs

Matr.-Nr.: 771697

Malte Stein

Matr.-Nr.: 771640

Betreuer: Andreas Heinrich
Abgabedatum: 18. November 2025

Kurzfassung

Das Modul **Projekt Softwaretechnik und Medieninformatik** aus dem vierten Semester beinhaltet die Entwicklung einer **Fullstack-Applikation**. Im Rahmen des Projektes sollen die Studierenden die Kenntnisse, welche sie in den vorherigen Semestern erworben haben, in einem praxisnahen Umfeld einsetzen und vertiefen. Dabei soll das **Projekt- und Zeitmanagement** geübt werden sowie soziale Kompetenzen wie **Teamfähigkeit** und **Konfliktfähigkeit** weiterentwickelt werden. Die folgenden Seiten der Dokumentation befassen sich mit der **Planung**, **Umsetzung** und **Bewertung** des Projektes und beleuchten alle relevanten Aspekte der Projektarbeit.

Schlagwörter: Projektmanagement, Zeitmanagement, Fullstack, Backend, Frontend, UI, Datenbanken, UML

Inhaltsverzeichnis

1	Erklärung der Aufgabe	6
1.1	Ist-Zustand	6
1.2	Soll-Zustand	6
1.3	Zielgruppe	7
2	Anforderungsanalyse	7
2.1	Must-Have Features	7
2.1.1	Authentifizierung & Benutzerverwaltung	7
2.1.2	Gegenstandsverwaltung	8
2.1.3	Ausleihprozess	8
2.1.4	Email-System	9
2.1.5	Administration & Übersicht	9
2.1.6	Protokollierung & Datenschutz	9
2.1.7	Technische Anforderungen	9
2.2	Nice-To-Have Features	9
2.2.1	Hohe Priorität (++)	9
2.2.2	Mittlere Priorität (+)	10
2.2.3	Niedrige Priorität (-)	10
2.3	Funktionale Anforderungen	10
2.4	Nicht-funktionale Anforderungen	11
2.4.1	Performance und Skalierbarkeit	11
2.4.2	Gebrauchstauglichkeit (Usability)	12
2.4.3	Sicherheit	12
2.4.4	Zuverlässigkeit und Verfügbarkeit	13
2.4.5	Wartbarkeit und Erweiterbarkeit	13
2.4.6	Datenschutz und Compliance	13
2.4.7	Kompatibilität	14
2.4.8	Dokumentation	14
2.5	User-Stories	14
2.5.1	Epic 1: Authentifizierung & Benutzerverwaltung	14
2.5.2	Epic 2: Gegenstandsverwaltung	15
2.5.3	Epic 3: Ausleihprozess – Warenkorb	16
2.5.4	Epic 4: Ausleihprozess – Terminverwaltung	17
2.5.5	Epic 5: Ausgabe/Rückgabe vor Ort	19
2.5.6	Epic 6: E-Mail-Benachrichtigungen	20
2.5.7	Epic 7: Protokollierung & Datenschutz	20
2.5.8	Epic 8: Administration & Reporting	21
2.5.9	Epic 9: Integration & Schnittstellen	21
2.5.10	Epic 10: Technische Anforderungen	22
2.5.11	Epic 11: Nice-to-Have Features	22
3	Zeitmanagement	25
3.1	Zeiterfassung	25

4	Projektmanagement	26
4.1	Methode	26
4.2	Organisation	26
4.3	Eingesetzte Tools	26
5	Entwicklungsumgebung	27
5.1	Eingesetzte IDE	27
5.2	Eingesetzte Frameworks	27
5.3	Server	27
5.4	Datenbank	28
5.5	Development-Server	28
5.6	Version Control management System	29
5.6.1	CI/CD Pipeline	29
6	UI-Prototyp	29
6.1	Farbpalette	30
6.1.1	Version 1 (Erstkonzept)	30
6.1.2	Version 2 (Wireframe)	30
6.1.3	Finale, farbige Version	30
6.2	Erste Prototypen	30
6.2.1	Version 1 – Grundidee	30
6.2.2	Version 2 – Änderungen gegenüber Version 1	35
6.3	Visuelle Umsetzung der Funktionen	41
6.3.1	Login-Seite	41
6.3.2	Geräte-Details	41
6.3.3	Warenkorb	42
7	User Research	44
7.1	Proto-Personas	44
7.1.1	Persona A – Lena Schmid (Studierende)	44
7.1.2	Persona B – Max Schmidt (IT-Admin / Systemverantwortlicher)	44
7.1.3	Persona C – Prof. Tom Fischer (Lehrender/Verleiher)	45
7.2	Interview Auswertung	46
7.3	Auswertung (Online-Umfrage, n = 28)	47
7.3.1	Stichprobe & Nutzung	47
7.3.2	Zufriedenheit (Ist-Prozess)	47
7.3.3	Größte Pain Points (Top-Nennungen)	47
7.3.4	Adoptionsbereitschaft für ein digitales System	48
7.3.5	Top-Features (max. 3 Stimmen)	48
7.3.6	Unverzichtbare Filter	49
7.3.7	Verfügbarkeit je Campus	50
7.3.8	Storno-Fenster (Abhol-/Rückgabe-Slots)	50
8	Softwarearchitektur	51
8.1	Entity-Relationship-Diagramme	51
8.2	Logische Sichten	53
8.2.1	Verteilungssicht	53
8.2.2	Struktursicht	54
8.2.3	Verhaltenssicht	57

9	Implementierung	59
9.1	Keycloak	59
9.2	Frontend	60
9.2.1	User Stories	60
9.2.2	Verwendete Technologien & Frameworks	61
9.2.3	Struktursicht	61

Abbildungsverzeichnis

1	Netzwerkübersicht Development-Server	28
2	Entwurf der CI/CD Pipeline Architektur in GitHub	29
3	Login screen	31
4	Inventarkatalog	32
5	Geräte-Details Seite	33
6	Persönlicher Bereich	34
7	Aktualisierter Inventarkatalog	35
8	Kalender in Produktdetails	36
9	Abholungszeitfenster	37
10	Warenkorb	38
11	Standort des Geräts	39
12	Aktualisierter persönlicher Bereich	40
13	Aktualisierter Login screen	41
14	Geräte-Details Seite mit aktualisierten Design	42
15	Verbesserter Warenkorb	43
16	Persona Lena Schmid	44
17	Persona Max Schmidt	45
18	Persona Tom Fischer	46
19	Anteil Studenten mit kürzlichen Ausleihen	47
20	Zufriedenheit mit aktuellem Prozess	47
21	aktuelle Pain Points	48
22	Adoptionsbereitschaft für digitales System	48
23	Wichtigste Features	49
24	Unverzichtbare Filter	49
25	Wichtigkeit Verfügbarkeit nach Campus	50
26	Länge Stornierbarkeit	50
27	Entity-Relationship-Diagramm	52
28	Diagramm zur Verteilungssicht	53
29	Komponentendiagramm	54
30	Sequenzdiagramm zum Ausleihprozess	57
31	Sequenzdiagramm zum Rückgabeprozess	58
32	Sequenzdiagramm zum Login	59

1 Erklärung der Aufgabe

1.1 Ist-Zustand

Aktuell erfolgt die Geräteausleihe an der Hochschule in Papierform. Um ein Gerät auszuliehen, müssen sich die Studierenden einen analogen Leihschein ausdrucken und sich bei einer betreuenden Person melden. Eine digitale Übersicht über vorhandene oder ausgeliehene Geräte existiert derzeit nicht und die Rückgaben werden händisch notiert, wodurch häufig Unklarheiten entstehen.

Typische Probleme:

- Geräte werden verspätet zurückgebracht.
- Keine automatische Erinnerung an Rückgabefristen.
- Keine genaue Übersicht über verfügbare Geräte.
- Papierverbrauch durch gedruckte Leihscheine führt zu unnötiger Ressourcenverschwendung.
- Der gesamte Prozess ist zeitaufwendig und fehleranfällig.

Durch die fehlende Digitalisierung ist der Ausleih-Prozess unübersichtlich, ineffizient und nicht nachhaltig. Und genau das wollen wir ändern.

1.2 Soll-Zustand

Unser Ziel ist es, den bisherigen papierbasierten Ablauf mit einer gebrauchstauglichen und nachhaltigen Website zu ersetzen und den Ausleih-Prozess von Anfang bis Ende digital im Überblick zu behalten.

Studierende, Dozenten und Mitarbeitende können sich dabei mit ihrem Hochschul-Account anmelden, verfügbare Geräte einsehen sowie Reservierungen und Verlängerungen tätigen.

Unsere wichtigen Punkte dabei sind:

- Ein klarer, schneller, papierloser Ablauf und einfache Bedienung.
- Webbasierte Plattform.
- Anmeldung über Hochschul-Account.
- Unterschiedliche Rollen: Studierende/Dozenten und Administratoren.
- Einfache Verwaltung für Studierende, Dozenten und Mitarbeitende.
- Integration von Benachrichtigungsfunktionen.

- Eine Datenbank zur Speicherung von Geräten, Nutzern und Ausleihen.

Bei der Software-Implementierung achten wir darauf, dass der Code strukturiert, effizient und funktionsfähig ist.

1.3 Zielgruppe

Die Hauptzielgruppe unseres digitalen Verleihsystems sind Studierende, Dozenten und Mitarbeitende der Hochschule Esslingen.

Die Dozenten möchten Geräte für Lehrveranstaltungen bereitstellen, zum Beispiel für das Wahlfachmodul “Digitale Fotografie”. Auch für die Privatnutzung soll es möglich sein, dass einige Dozenten ihr eigenes Equipment hochladen und verleihen.

Studierende benötigen daher eine schnelle und digitale Möglichkeit, Geräte auszuleihen. Die Mitarbeitenden, die den Ausleih-Prozess betreuen, brauchen eine übersichtliche Verwaltung, um alle Ausleihen gründlich und vertraulich dokumentieren zu können.

Das digitale Leihsystem sorgt dafür, dass die Zielgruppen Zeit sparen, den Überblick behalten und einen reibungslosen Prozess haben.

2 Anforderungsanalyse

Die Anforderungsanalyse für das Projekt LeihSy bildet die Grundlage für die Entwicklung des digitalen Verleihsystems an der Hochschule Esslingen. Sie wurde in Zusammenarbeit mit dem Betreuer Andreas Heinrich und dem Kunden Christian Haas erarbeitet und beschreibt sowohl funktionale als auch nicht-funktionale Anforderungen. Die Anforderungen sind priorisiert in Must-Have Features und Nice-to-Have Features (erweiterte Funktionalität) unterteilt, um eine schrittweise und agile Umsetzung innerhalb des Projektzeitraums zu ermöglichen.

2.1 Must-Have Features

2.1.1 Authentifizierung & Benutzerverwaltung

- **Single Sign-On (SSO) via Keycloak:** Integration von OpenID Connect zur Authentifizierung mit dem Hochschul-RZ-Account
- **Rollenbasierte Zugriffskontrolle:** Drei Benutzerrollen (Administrator, Verleiher, Student/Entleiher) mit spezifischen Berechtigungen
- **Rollenvergabe über Keycloak-Benutzergruppen:** Automatische Zuweisung von Rollen basierend auf Gruppenzugehörigkeit

2.1.2 Gegenstandsverwaltung

- **CRUD-Operationen für Verleihgegenstände:** Erstellen, Anzeigen, Bearbeiten und Löschen von Gegenständen durch Administratoren und Verleiher
- **Pflichtattribute:** Inventarnummer, Name, Beschreibung, Kategorie, Foto, Zubehör, Lagerort, Status
- **Sets von gleichartigen Gegenständen:** Möglichkeit, mehrere identische Gegenstände mit eindeutigen Nummern als Set anzulegen (z. B. 10x Meta Quest)
- **Kategorien und Filter:** Kategorisierung von Gegenständen und Filtermöglichkeiten nach Kategorie, Verfügbarkeit, Lagerort und Volltext-Suche
- **Verleiher-Zuordnung:** Verleiher sehen und verwalten nur ihre zugeordneten Gegenstände
- **Benötigte Zusatzgegenstände:** Automatische Anzeige von empfohlenem Zubehör zu Hauptgegenständen (z. B. Speicherkarte zu Kamera)

2.1.3 Ausleihprozess

- **Warenkorb-Funktionalität:** Studierende können mehrere Gegenstände in einen Warenkorb legen
- **Terminvorschlag über Kalender:** Auswahl des gewünschten Ausleihzeitraums mit Anzeige der Verfügbarkeit
- **Verfügbarkeitskalender auf Detailseite:** Anzeige der Verfügbarkeit eines Gegenstandes in einer Kalenderansicht
- **Bestätigung/Ablehnung durch Verleiher:** Verleiher können Ausleiheanfragen bestätigen oder ablehnen
- **Automatische Stornierung:** Nicht bestätigte Anfragen werden nach 24 Stunden automatisch storniert
- **Digitale Ausgabe und Rückgabe:** Email-Bestätigung bei Ausgabe und Rückgabe vor Ort mit Login-basierter Bestätigung durch Studierende
- **Ausleihverlauf:** Studierende können ihren persönlichen Ausleihverlauf einsehen (aktuelle, offene und vergangene Ausleihen)

2.1.4 Email-System

- **Automatische Benachrichtigungen:** Email-Versand bei verschiedenen Ereignissen (Anfrage erstellt, bestätigt, abgelehnt, Ausgabe, Rückgabe)
- **Erinnerungsmails:** Automatische Erinnerung 2 Tage vor Fälligkeit
- **Überfälligkeits-Benachrichtigungen:** Benachrichtigung an Student und Verleiher bei nicht fristgerechter Rückgabe

2.1.5 Administration & Übersicht

- **Admin-Dashboard:** Zentrale Übersicht über offene Anfragen, bestätigte aber nicht abgeholte Ausleihen, aktuelle und zukünftige Ausleihen sowie Statistiken
- **Systemkonfiguration:** Administratoren können Systemeinstellungen verwalten

2.1.6 Protokollierung & Datenschutz

- **Audit-Log:** Protokollierung aller relevanten Aktionen (Wer hat wann was ausgeliehen/zurückgegeben, Änderungen an Gegenständen, Login-Versuche)
- **DSGVO-Konformität:** Datensparsamkeit, Zweckbindung, definierte Aufbewahrungsfristen, Recht auf Datenexport und Löschung
- **Datenschutzkonforme Speicherung:** Verschlüsselte Session-Daten, automatische Löschung temporärer Daten

2.1.7 Technische Anforderungen

- **Containerisierung mit Docker:** Alle Komponenten (Frontend, Backend, Datenbank, Keycloak) sind containerisiert und über Docker-Compose startbar
- **PostgreSQL-Datenbank:** Persistente Datenspeicherung
- **Responsive Design:** Optimierung für Desktop und Mobile

2.2 Nice-To-Have Features

2.2.1 Hohe Priorität (++)

- **Budgetplanung mit Tagessätzen:** Verwaltung von Budgets mit definierten Tagessätzen für Gegenstände
- **Private Verleihgegenstände:** Möglichkeit für Nutzer, private Gegenstände im System zu verwalten und zu verleihen

- **Batch-Erstellung von Gegenständen:** Vereinfachte Massenerstellung von Gegenständen
- **CI/CD-Pipeline:** Automatisierte Builds, Tests und Deployments über GitHub Actions
- **QR/Barcode-Scanner:** Scannen von QR-Codes zur schnellen Identifikation von Gegenständen bei Ausgabe und Rückgabe
- **Code-Qualitätsprüfung:** Integration von SonarQube zur statischen Code-Analyse

2.2.2 Mittlere Priorität (+)

- **Verleihgruppen:** Mehrere Verleiher können als Gruppe zusammenarbeiten und gemeinsam Bestände verwalten
- **InSy-Integration:** POST-API-Endpoint für Datenimport aus dem Inventarisierungssystem InSy (Inventarnummer, Name, Beschreibung, Kategorie, Lagerort)
- **Verschiedene Email-Templates:** Individuelle Templates für unterschiedliche Anlässe und Zielgruppen
- **Verlängerungsfunktion:** Studierende können laufende Ausleihen verlängern (sofern keine Folgebuchungen existieren)

2.2.3 Niedrige Priorität (-)

- **Studentengruppen mit gemeinsamem Budget:** Studierende können sich zu Gruppen zusammenschließen und ein gemeinsames Budget nutzen
- **Dozenten-Freigabe-Workflow:** Dozenten können Ausleiheanfragen für Projektarbeiten freigeben
- **Detaillierte Budget-Reports:** Umfassende Auswertungen und Reports über Budgetauslastung und Kosten

2.3 Funktionale Anforderungen

Die funktionalen Anforderungen beschreiben das grundlegende Verhalten und die Kernfunktionalitäten des Systems LeihSy aus fachlicher Sicht. Sie definieren, was das System leisten muss, ohne dabei auf technische Implementierungsdetails einzugehen.

Benutzerverwaltung Das System muss eine sichere Authentifizierung über Keycloak (SSO) mit dem Hochschul-RZ-Account ermöglichen und drei Benutzerrollen (Administrator, Verleiher, Student/Entleiher) mit unterschiedlichen Berechtigungen unterstützen.

Gegenstandsverwaltung Das System muss die vollständige Verwaltung von Verleihgegenständen ermöglichen, einschließlich Anlegen, Bearbeiten, Löschen, Kategorisieren und Filtern. Gegenstände müssen eindeutig identifizierbar sein und können als Sets mit mehreren Einzelexemplaren angelegt werden.

Ausleihprozess Das System muss den gesamten Ausleihprozess digital abbilden: von der Anfrage über die Bestätigung/Ablehnung durch den Verleiher bis hin zur dokumentierten Ausgabe und Rückgabe vor Ort. Verfügbarkeiten müssen automatisch geprüft und nicht bestätigte Anfragen nach 24 Stunden automatisch storniert werden.

Benachrichtigungssystem Das System muss automatisch Email-Benachrichtigungen zu allen relevanten Ereignissen im Ausleihprozess versenden, einschließlich Erinnerungen vor Fälligkeit und Mahnungen bei Überfälligkeit.

Übersichten und Verlauf Das System muss allen Benutzergruppen rollenbezogene Übersichten bereitstellen: Studierende sehen ihren persönlichen Ausleihverlauf, Verleiher sehen Anfragen und Ausleihen ihrer zugeordneten Gegenstände, Administratoren erhalten ein zentrales Dashboard mit Statistiken und Gesamtübersicht.

Integration Das System muss eine REST-API-Schnittstelle zum Import von Gegenstandsdaten aus dem Inventarisierungssystem InSy bereitstellen.

Protokollierung Das System muss alle sicherheitsrelevanten und geschäftsrelevanten Ereignisse lückenlos und unveränderbar protokollieren (Audit-Log).

Datenschutz Das System muss DSGVO-konform sein und die Prinzipien der Datensparsamkeit, Zweckbindung und Speicherbegrenzung einhalten. Benutzer müssen ihre Daten exportieren und die Löschung ihres Accounts beantragen können.

2.4 Nicht-funktionale Anforderungen

Die nicht-funktionalen Anforderungen beschreiben qualitative Eigenschaften des Systems LeihSy, die über die reine Funktionalität hinausgehen. Sie definieren, wie gut das System seine Funktionen erfüllt und welche Rahmenbedingungen eingehalten werden müssen.

2.4.1 Performance und Skalierbarkeit

- **Antwortzeiten:** Das System muss eine hohe Reaktionsgeschwindigkeit gewährleisten. Seiten müssen innerhalb von maximal 2 Sekunden laden, API-Anfragen innerhalb von 500 Millisekunden beantwortet werden.

- **Gleichzeitige Benutzer:** Das System muss mindestens 20 gleichzeitige Benutzer ohne Performanceeinbußen unterstützen können. Bei höherer Last muss eine horizontale Skalierung möglich sein.
- **Datenbankperformance:** Datenbankabfragen müssen optimiert sein (Indizes, effiziente Queries). Die Datenbank muss mindestens 500 Gegenstände und 100 Ausleihvorgänge verwalten können.

2.4.2 Gebrauchstauglichkeit (Usability)

- **Intuitive Bedienung:** Die Benutzeroberfläche muss intuitiv und selbsterklärend sein. Neue Benutzer sollen ohne Schulung grundlegende Funktionen (Suchen, Anfragen, Warenkorb) nutzen können.
- **Responsive Design:** Das System muss auf verschiedenen Endgeräten (Desktop, Tablet, Smartphone) einwandfrei funktionieren und eine optimale Benutzererfahrung bieten.
- **Barrierefreiheit:** Die Anwendung soll grundlegende Accessibility-Standards einhalten (kontrastreiche Darstellung, semantisches HTML, Tastaturnavigation).
- **Mehrsprachigkeit (Optional):** Das System sollte mehrsprachig erweiterbar sein. Initial wird Deutsch unterstützt, eine spätere Erweiterung um Englisch soll möglich sein.

2.4.3 Sicherheit

- **Authentifizierung und Autorisierung:** Alle Zugriffe müssen über Keycloak authentifiziert sein. Nicht authentifizierte Anfragen werden abgelehnt. Autorisierung erfolgt rollenbasiert auf allen Endpunkten.
- **Datenverschlüsselung:** Die Kommunikation zwischen Client und Server muss über HTTPS/TLS verschlüsselt sein. Passwörter und sensitive Daten müssen verschlüsselt gespeichert werden.
- **API-Sicherheit:** API-Endpunkte (z. B. InSy-Import) müssen über API-Keys oder OAuth authentifiziert sein. Rate-Limiting verhindert Missbrauch.
- **Session-Management:** Sessions müssen nach Inaktivität automatisch ablaufen (Standard: 8 Stunden). Logout muss sowohl lokale als auch Keycloak-Session beenden.
- **Input-Validierung:** Alle Benutzereingaben müssen auf Client- und Server-Seite validiert werden, um Injection-Angriffe (SQL, XSS) zu verhindern.

2.4.4 Zuverlässigkeit und Verfügbarkeit

- **Verfügbarkeit:** Das System soll eine Verfügbarkeit von mindestens 95 % während der Hochschul-Öffnungszeiten (Mo–Fr, 8–18 Uhr) gewährleisten.
- **Fehlerbehandlung:** Das System muss Fehler graceful behandeln und dem Benutzer verständliche Fehlermeldungen anzeigen. Technische Details werden nur an Administratoren ausgegeben.
- **Datensicherung:** Datenbank-Backups müssen täglich automatisch erstellt und mindestens 30 Tage aufbewahrt werden. Recovery-Tests müssen regelmäßig durchgeführt werden.

2.4.5 Wartbarkeit und Erweiterbarkeit

- **Code-Qualität:** Der Code muss Clean-Code-Prinzipien folgen, gut dokumentiert und testbar sein. Komplexe Logik muss durch Kommentare erläutert werden.
- **Modularität:** Das System muss modular aufgebaut sein (Frontend/Backend-Trennung, Service-Architektur). Einzelne Module können unabhängig voneinander weiterentwickelt werden.
- **Versionierung:** Der Code muss über Git versioniert werden. Commit-Messages folgen einem einheitlichen Standard (Conventional Commits).
- **Testabdeckung:** Kritische Funktionen müssen durch Unit-Tests abgedeckt sein (angestrebte Testabdeckung: >60 %). Integration-Tests decken die wichtigsten User-Flows ab.
- **Deployment:** Das System muss über Docker-Container deployt werden können. Ein `docker-compose.yml` ermöglicht das Starten der gesamten Anwendung mit einem Befehl.

2.4.6 Datenschutz und Compliance

- **DSGVO-Konformität:** Das System muss alle Anforderungen der DSGVO erfüllen: Datensparsamkeit, Zweckbindung, Speicherbegrenzung, Auskunftsrecht, Recht auf Löschung.
- **Datenminimierung:** Es dürfen nur die für den Betrieb notwendigen personenbezogenen Daten gespeichert werden. Keine Sammlung von Daten „auf Vorrat“.
- **Audit-Trail:** Alle Zugriffe auf personenbezogene Daten und sicherheitsrelevante Aktionen müssen nachvollziehbar protokolliert werden.

- **Datenschutzerklärung:** Das System muss eine Datenschutzerklärung bereitstellen, die Benutzer über die Datenverarbeitung informiert.

2.4.7 Kompatibilität

- **Browser-Kompatibilität:** Das System muss in den gängigen modernen Browsern funktionieren (Chrome, Firefox, Safari, Edge – jeweils aktuelle Version und eine Version zurück).
- **Keycloak-Kompatibilität:** Das System muss mit der an der Hochschule eingesetzten Keycloak-Version kompatibel sein und OpenID Connect-Standards einhalten.
- **API-Standards:** REST-APIs müssen RESTful-Prinzipien folgen und Standard-HTTP-Methoden (GET, POST, PUT, DELETE) verwenden.

2.4.8 Dokumentation

Technische Dokumentation: Es muss eine technische Dokumentation existieren, die Architektur, Datenmodell, API-Schnittstellen und Deployment-Prozess beschreibt.

2.5 User-Stories

Die User Stories beschreiben die Anforderungen aus Sicht der verschiedenen Benutzergruppen und des Systems.

2.5.1 Epic 1: Authentifizierung & Benutzerverwaltung

US-001: Als Student möchte ich mich mit meinem RZ-Account anmelden können

- Akzeptanzkriterien:
 - Login über Keycloak
 - Weiterleitung nach erfolgreichem Login
 - Session-Management

US-002: Als Administrator möchte ich Rollen verwalten können

- Akzeptanzkriterien:
 - Rollen aus Keycloak-Gruppen übernehmen
 - Berechtigungen pro Rolle definiert

US-003: Als Verleiher möchte ich nur meine zugeordneten Gegenstände sehen

- Akzeptanzkriterien:
 - Zuordnung Verleiher \leftrightarrow Gegenstände
 - Filterung nach Berechtigung

2.5.2 Epic 2: Gegenstandsverwaltung

US-004: Als Admin möchte ich einzelne Gegenstände anlegen können

- Akzeptanzkriterien:
 - Stammdaten (Name, Inventarnr, Beschreibung, Kategorie, Lagerort)
 - Formular mit allen Pflichtfeldern
 - Bildupload
 - Speicherung in DB

US-005: Als Admin möchte ich Sets von gleichen Gegenständen anlegen

- Akzeptanzkriterien:
 - Anzahl eingeben
 - Automatische Nummerierung
 - Jedes Item eindeutig identifizierbar

US-006: Als Admin möchte ich Gegenstände bearbeiten/löschen können

- Akzeptanzkriterien:
 - Edit-Formular
 - Soft-Delete (für Protokolle)
 - Nur löschen, wenn keine aktiven Ausleihen

US-007: Als Student möchte ich verfügbare Gegenstände durchsuchen können

- Akzeptanzkriterien:
 - Übersichtsseite mit allen Gegenständen
 - Filter nach Kategorie, Verfügbarkeit, Lagerort
 - Suchfunktion (Volltextsuche)
 - Anzeige Verfügbarkeit

US-008: Als Student möchte ich Detailinfos zu Gegenständen sehen

- Akzeptanzkriterien:
 - Detailseite mit Bild, Beschreibung
 - Anzeige benötigtes Zubehör
 - Verfügbarkeitskalender

US-009: Als System möchte ich Bilder speichern und ausliefern können

- Akzeptanzkriterien:
 - Bildupload (max. Größe 5 MB, Format JPG/PNG/WebP)
 - Speicherung im Filesystem
 - Optimierung/Thumbnails

US-026: Als Admin möchte ich Zusatzgegenstände zu Hauptgegenständen verknüpfen

- Akzeptanzkriterien:
 - Auswahl von Zusatzgegenständen bei Gegenstandsbearbeitung
 - Markierung als „empfohlen“ oder „erforderlich“
 - Anzeige beim Hinzufügen zum Warenkorb

US-027: Als Admin möchte ich Kategorien verwalten können

- Akzeptanzkriterien:
 - Kategorien hinzufügen, bearbeiten
 - Vordefinierte Kategorien (VR-Equipment, Foto-Equipment, IT-Geräte, Audio, Video, Sonstiges)
 - Kategorien können nicht gelöscht werden, wenn Gegenstände zugeordnet sind

2.5.3 Epic 3: Ausleihprozess – Warenkorb

US-010: Als Student möchte ich Gegenstände zum Warenkorb hinzufügen

- Akzeptanzkriterien:
 - Button „In den Warenkorb“
 - Warenkorb-Icon mit Counter
 - Warenkorbansicht

US-011: Als Student möchte ich im Warenkorb einen Zeitraum wählen

- Akzeptanzkriterien:
 - Kalender-Widget
 - Validierung Verfügbarkeit
 - Anzeige Konflikte
 - Blockierung nicht verfügbarer Zeiträume

US-012: Als Student möchte ich eine Ausleihanfrage abschicken

- Akzeptanzkriterien:
 - Alle Warenkorb-Items auf einmal
 - Status „Wartend auf Bestätigung“
 - Email an Verleiher
 - Optional: Kommentarfeld

US-028: Als Student möchte ich empfohlene Zusatzgegenstände sehen

- Akzeptanzkriterien:
 - Beim Hinzufügen zum Warenkorb werden Zusatzgegenstände angezeigt
 - Zusatzgegenstände können mit einem Klick hinzugefügt werden
 - Warnung bei nicht verfügbaren erforderlichen Zusatzgegenständen

2.5.4 Epic 4: Ausleihprozess – Terminverwaltung

US-013: Als Verleiher möchte ich offene Anfragen sehen

- Akzeptanzkriterien:
 - Dashboard mit Pending-Anfragen
 - Sortierung nach Datum
 - Filter nach Gegenstand
 - Hervorhebung von Anfragen nahe der 24h-Frist

US-014: Als Verleiher möchte ich Anfragen bestätigen/ablehnen

- Akzeptanzkriterien:
 - Buttons Annehmen/Ablehnen
 - Pflichtfeld für Begründung bei Ablehnung
 - Optional: Kommentarfeld bei Bestätigung

- Status-Update
- Email an Student

US-015: Als System möchte ich unbestätigte Anfragen nach 24h stornieren

- Akzeptanzkriterien:
 - Cronjob/Scheduled Task
 - Automatische Stornierung
 - Email-Benachrichtigung an Student
 - Gegenstände werden wieder als verfügbar markiert

US-016: Als Student möchte ich eine Ausleihe verlängern können

- Akzeptanzkriterien:
 - Verlängerung anfragen
 - Nur wenn verfügbar (keine Folgebuchungen)
 - Verleiher muss bestätigen
 - Maximal 2 Verlängerungen pro Ausleihe

US-017: Als Student möchte ich meinen Ausleihverlauf sehen

- Akzeptanzkriterien:
 - Liste aller Ausleihen (vergangene + aktuelle + offene Anfragen)
 - Status angezeigt (Ausgeliehen, Ausstehend, Bestätigt, Zurückgegeben, Abgelehnt)
 - Details einsehbar
 - Chronologische Sortierung

US-029: Als Verleiher möchte ich aktuelle und zukünftige Ausleihen sehen

- Akzeptanzkriterien:
 - Übersicht über alle aktuell ausgeliehenen Gegenstände
 - Übersicht über bestätigte, aber noch nicht abgeholte Ausleihen
 - Sortierung nach Rückgabe-/Abholtermin
 - Hervorhebung überfälliger Rückgaben

2.5.5 Epic 5: Ausgabe/Rückgabe vor Ort

US-018: Als System möchte ich bei Abholung eine Bestätigungsmail senden

- Akzeptanzkriterien:
 - Verleiher triggert Email
 - Link mit Token
 - Gültig für 15 Minuten

US-019: Als Student möchte ich die Ausgabe per Email bestätigen

- Akzeptanzkriterien:
 - Login via Keycloak beim Klick
 - Bestätigung speichern
 - Status → „Ausgeliehen“
 - Zeitstempel erfassen

US-020: Als System möchte ich die Rückgabe dokumentieren

- Akzeptanzkriterien:
 - Analog zu Ausgabe
 - Status → „Verfügbar“
 - Protokolleintrag
 - Zeitstempel erfassen

US-030: Als Verleiher möchte ich Gegenstände per QR-Code scannen können

- Akzeptanzkriterien:
 - QR-Code wird für jeden Gegenstand generiert
 - QR-Code als PDF downloadbar
 - Scanner-Funktion in der Webanwendung (Kamera-Zugriff)
 - Nach Scan wird Gegenstand automatisch geladen
 - Alternative: Manuelle Eingabe der Inventarnummer

2.5.6 Epic 6: E-Mail-Benachrichtigungen

US-021: Als System möchte ich Erinnerungsmails versenden

- Akzeptanzkriterien:
 - 2 Tage vor Rückgabe
 - Bei Überfälligkeit (1 Tag nach Fälligkeit, dann wöchentlich)
 - Cronjob täglich
 - Überfälligkeits-Mail geht an Student und Verleiher

US-022: Als System möchte ich Statusänderungs-Mails versenden

- Akzeptanzkriterien:
 - Bei Bestätigung/Ablehnung von Anfragen
 - Bei Stornierung
 - Bei Ausgabe/Rückgabe
 - Template-basiert mit Corporate Design
 - Personalisierung mit Benutzerdaten

2.5.7 Epic 7: Protokollierung & Datenschutz

US-023: Als System möchte ich alle Ausleihe-Ereignisse protokollieren

- Akzeptanzkriterien:
 - Ausgeliehen, Zurückgegeben, Verlängert, Erstellt, Geändert, Gelöscht
 - Timestamp, User, Aktion, betroffenes Objekt
 - Unveränderbar (Write-Once)
 - Login-Versuche protokollieren

US-024: Als System möchte ich Studentendaten sparsam speichern

- Akzeptanzkriterien:
 - Nur Name, Email, Matrikelnummer aus Keycloak
 - Keine Duplikation
 - Automatisches Löschen nach definierten Fristen (abgeschlossene Ausleihen nach 2 Jahren, Logs nach 1 Jahr)
 - Anonymisierung inaktiver Accounts nach 3 Jahren

US-031: Als Admin möchte ich Audit-Logs einsehen können

- Akzeptanzkriterien:
 - Chronologische Liste aller Ereignisse
 - Filtermöglichkeiten (Zeitraum, Benutzer, Aktionstyp)
 - Detailansicht mit allen Log-Informationen

2.5.8 Epic 8: Administration & Reporting

US-025: Als Admin möchte ich einen Überblick über alle Ausleihen haben

- Akzeptanzkriterien:
 - Dashboard mit Statistiken
 - Aktuelle Ausleihen
 - Überfällige Items
 - Offene Anfragen
 - Bestätigte, aber nicht abgeholte Ausleihen
 - Vorplanung (Vorschau zukünftiger Ausleihen, Filterung)
 - Top 10 meistgeliehene Gegenstände
 - Auslastung nach Kategorie

US-034: Als Admin möchte ich Verleiher zu Gegenständen zuordnen können

- Akzeptanzkriterien:
 - Bei Gegenstandsbearbeitung Verleiher auswählbar
 - Übersicht aller Zuordnungen
 - Filter nach Verleiher
 - Ein Gegenstand kann nur einem Verleiher zugeordnet sein

2.5.9 Epic 9: Integration & Schnittstellen

US-037: Als System möchte ich Daten aus InSy importieren können

- Akzeptanzkriterien:
 - REST-API Endpoint: POST `/api/insy/import`
 - Authentifizierung über API-Key

- Import von Inventarnummer, Name, Beschreibung, Kategorie, Lagerort
- Validierung der Daten
- Bei bestehender Inventarnummer: Update statt Duplikat
- Import-Vorgänge werden protokolliert

2.5.10 Epic 10: Technische Anforderungen

US-039: Als Entwickler möchte ich das System über Docker deployen können

- Akzeptanzkriterien:
 - Dockerfile für Frontend, Backend, Datenbank
 - `docker-compose.yml` für Orchestrierung
 - Environment-Variablen für Konfiguration
 - Persistent Volumes für Datenbank
 - Health-Checks für alle Services
 - Start mit `docker-compose up`

US-040: Als System möchte ich auf verschiedenen Geräten funktionieren

- Akzeptanzkriterien:
 - Responsive Design für Desktop, Tablet, Smartphone
 - Optimierte Layouts für verschiedene Bildschirmgrößen
 - Touch-Gesten auf Mobile unterstützt

2.5.11 Epic 11: Nice-to-Have Features

US-032: Als Student möchte ich meine Daten exportieren können

- Akzeptanzkriterien:
 - Button „Meine Daten exportieren“
 - Export als JSON oder PDF
 - Enthält Benutzerprofil und Ausleihhistorie
 - Download-Link 24 Stunden gültig

US-033: Als Student möchte ich die Löschung meines Accounts beantragen können

- Akzeptanzkriterien:

- Löschungsantrag über Formular
- Keine aktiven Ausleihen dürfen bestehen
- Anonymisierung statt vollständiger Löschung (Audit-Integrität)
- Bestätigung per Email

US-035: Als Admin möchte ich Systemeinstellungen konfigurieren können

- Akzeptanzkriterien:
 - Session-Timeout konfigurierbar
 - Stornierungsfrist (Standard: 24h) konfigurierbar
 - Erinnerungs-Email Vorlaufzeit konfigurierbar
 - Min/Max Ausleihdauer konfigurierbar
 - SMTP-Server-Einstellungen
 - Änderungen werden im Audit-Log protokolliert

US-036: Als Admin möchte ich Statistiken exportieren können

- Akzeptanzkriterien:
 - Export als CSV oder PDF
 - Ausleihen pro Zeitraum
 - Auslastung nach Kategorie
 - Top-Gegenstände nach Ausleihhäufigkeit
 - Frei wählbarer Zeitraum

US-038: Als Admin möchte ich Import-Logs einsehen können

- Akzeptanzkriterien:
 - Übersicht aller Import-Vorgänge
 - Zeitstempel, Anzahl importierter Gegenstände, Fehler
 - Detailansicht mit Fehlermeldungen

US-041: Als Admin möchte ich Budgets mit Tagessätzen verwalten können

- Akzeptanzkriterien:
 - Tagessätze pro Gegenstand definierbar
 - Studenten haben ein Budget-Konto

- Budget wird bei Ausleihe berechnet und abgezogen
- Übersicht über Budget-Auslastung

US-042: Als Benutzer möchte ich private Gegenstände verwalten können

- Akzeptanzkriterien:
 - Checkbox „Privater Gegenstand“ beim Anlegen
 - Private Gegenstände sind nur für den Besitzer sichtbar
 - Separate Auflistung privater Gegenstände

US-043: Als Verleiher möchte ich mit anderen Verleihern in Gruppen zusammenarbeiten

- Akzeptanzkriterien:
 - Verleihgruppen erstellen
 - Gegenstände der Gruppe zuordnen
 - Alle Gruppenmitglieder können Anfragen bearbeiten

US-044: Als Student möchte ich mit anderen Studenten eine Gruppe bilden

- Akzeptanzkriterien:
 - Studentengruppen erstellen
 - Gemeinsames Budget für die Gruppe
 - Alle Gruppenmitglieder sehen Gruppenausleihen

US-045: Als Dozent möchte ich Ausleiheanfragen für Projekte freigeben

- Akzeptanzkriterien:
 - Studenten können Projekt angeben
 - Anfrage geht an Dozenten zur Freigabe
 - Nach Freigabe geht Anfrage an Verleiher
 - Workflow: Student → Dozent → Verleiher

US-046: Als Entwickler möchte ich eine CI/CD-Pipeline einrichten

- Akzeptanzkriterien:
 - Automatische Tests bei jedem Push
 - Automatisches Deployment bei Merge in Main-Branch
 - Code-Qualitätsprüfung mit SonarQube
 - GitHub Actions Integration

3 Zeitmanagement

In diesem Kapitel wird die zeitliche Umgebung beschrieben, in deren Rahmen sich das Projekt bewegt. Dazu zählen zuerst die Aufwandsschätzung in Stunden und außerdem die tatsächliche Zeiterfassung. Sowohl die Zeitplanung als auch die Zeiterfassung läuft über die Tabelle, die über diesen [Link](#) abrufbar ist. Das Vorgehen wird nachfolgend näher betrachtet.

Zur besseren Einordnung und Abwägung, welche Features implementiert werden sollen, wurde eine Aufwandsschätzung erstellt. Nachdem zuvor jedes Feature in Form User Stories in seine Unteraufgaben aufgeteilt wurde, orientierte sich die Aufwandsschätzung an den User Stories. Es wurde trotz eingeschränkten Wissens über die zu verwendenden Technologien so gut wie möglich ein Zeitrahmen in Arbeitsstunden festgelegt, in dem mit der fertigen Implementierung des Features gerechnet wird. Darüber hinaus wurde eine pauschale Zeit festgelegt, um den Arbeitsaufwand, der zusätzlich zur Implementierung in Form beispielsweise der Dokumentation des Arbeitsschrittes anfällt, zu beschreiben. Die Aufwandsschätzung wurde außerdem von der reinen Textform in eine Tabelle überführt, um durch Verknüpfungen der Werte eine automatische Anpassung der Gesamtzeit bei der Änderung einzelner Werte zu ermöglichen. Die User Stories wurden nach der Sinnhaftigkeit ihrer Reihenfolge chronologisch sortiert und in einzelne Sprints unterteilt. Für jeden Sprint wurde in der Tabelle für eine bessere Übersichtlichkeit ein eigenes Tabellenblatt erstellt und die geschätzte Gesamtzeit für den Sprint anhand der in der Tabelle eingetragenen User Stories automatisch berechnet. Darüber hinaus wurde ein Tabellenblatt für einmalige Zeitaufwendungen angelegt. Dazu zählen vor allem die Vorbereitungen für Präsentationen und Meilenstein-Abgaben sowie Seminare. Im gleichen Stil wurde ein Tabellenblatt für wöchentlich wiederkehrende Zeitaufwendungen angelegt, in dem die entsprechende Zeit für beispielsweise Regelmeetings zur besseren Übersichtlichkeit einzeln geschätzt wird. Die Gesamtzeiten der einzelnen Tabellenblätter wurde dann in einem weiteren Tabellenblatt zu einer gesamten Zeitschätzung für das ganze Projekt zusammengerechnet.

3.1 Zeiterfassung

Für die Arbeitszeiterfassung wurde von Google Sheets auf Clockify umgestellt, da Clockify eine deutlich effizientere und integrierte Lösung bietet. Besonders wichtig war dabei die nahtlose Anbindung an Jira, über die Arbeitszeiten direkt auf Aufgaben und Tickets gebucht werden können, ohne manuelle Übertragungsfehler oder doppelten Aufwand. Darüber hinaus lässt sich Clockify nicht nur im Browser, sondern auch als Desktop- und Mobile-App nutzen, was die Erfassung unterwegs oder während Meetings erheblich vereinfacht und den gesamten Prozess flexibler und gebrauchstauglicher macht.

4 Projektmanagement

In diesem Kapitel wird beschrieben, wie das Management des Projekts aufgebaut ist. Ein besonderes Augenmerk liegt hierbei auf der Projektmanagement-Methode sowie auf der Organisation. Außerdem werden die in diesem Rahmen verwendeten Tools behandelt.

4.1 Methode

Es wird eine agile Projektmethode verwendet. Es wurde entschieden, ein Kanban Board zu verwenden, um die Aufgaben übersichtlich und variabel einzuteilen. Zusätzlich werden Elemente von Scrum übernommen. Hauptsächlich ist hierbei die Aufteilung in jeweils 2 Wochen langen Sprints zu nennen. Die Daily Meetings wurden durch ein wöchentliches Regelmeeting mit dem Betreuer und ein wöchentliches Meeting mit dem Team ersetzt. Um das Projekt schlank zu halten, wurde auf manche Scrum-Elemente wie die Sprint Reviews verzichtet.

4.2 Organisation

Die Kommunikation innerhalb des Teams läuft vor allem über den Teamchat und die wöchentlichen Team-Meetings. In den Team-Meetings werden die To-Dos festgelegt und verteilt. Die To-Dos werden auf dem Kanban-Board hinzugefügt. To-Dos aus dem letzten Meeting werden auf ihre Erfüllung überprüft. Außerdem werden offene Fragen geklärt und Grundsatzentscheidungen gefällt. Team-Meetings werden grundsätzlich von mindestens einer Person protokolliert. Jedes Teammitglied ist verantwortlich für die Erfüllung der eigenen To-Dos und den Status des To-Dos auf dem Kanban-Board. In den wöchentlichen Meetings mit dem Betreuer wird der Fortschritt des aktuellen Sprints besprochen und offene Fragen geklärt. Meetings mit dem Betreuer werden grundsätzlich von mindestens zwei Personen protokolliert. Protokolle von Meetings werden abgeglichen und das zusammengeführte Protokoll allen Teammitgliedern bereitgestellt.

4.3 Eingesetzte Tools

Es werden ausschließlich voll digitale, papierlose Tools verwendet. Als Kanban-Board wird Jira verwendet. Für den Teamchat und die wöchentlichen Teammeetings wird Discord eingesetzt. Die wöchentlichen Meetings mit dem Betreuer werden meist in Person abgehalten, falls dies nicht möglich ist, kommt Webex zum Einsatz. Die Dokumentation und die Meeting-Protokolle sowie Notizen und Ideen und organisatorische Informationen befinden sich auf HajTex, dem Online-LaTeX-Editor von fachschaften.org, um eine einfache und zeitgleiche Bearbeitung durch alle Teammitglieder zu ermöglichen. Die Zeiterfassung wird über das zentralisierte Online-Tool Clockify verwaltet.

5 Entwicklungsumgebung

In diesem Kapitel wird die technische Umgebung beschrieben, die für die Entwicklung der Fullstack Applikation LeihSy verwendet wurde.

5.1 Eingesetzte IDE

Für die Entwicklung der Anwendung wird die IDE IntelliJ IDEA Ultimate verwendet. Die IDE bietet umfangreiche Funktionen wie Syntax-Highlighting, Autovervollständigung sowie integriertes Debugging und Git-Unterstützung. Des Weiteren unterstützt die IDE die Frontend- sowie die Backend-Entwicklung, wodurch die Entwicklung einer Fullstack-Applikation effizienter und einheitlich gestaltet werden kann.

5.2 Eingesetzte Frameworks

Für die Backend-Entwicklungen der Applikation wird das Framework Spring-Boot verwendet. Spring-Boot ist ein Java-basiertes Framework und funktioniert nach dem Prinzip “Convention over Configuration”, was den Entwicklern bei den Konfigurations-Entscheidungen Zeit spart, weil das Framework vordefinierte Standardwerte verwendet. Des Weiteren unterstützt das Framework die Entwicklung von RESTful APIs, welche für die Datenübertragung zwischen Client und Server zuständig sind. Zudem ist Spring Boot weit verbreitet und es existieren umfangreiche Dokumentation zur Benutzung des Frameworks.

Für die Frontend-Entwicklung der Applikation wird das Framework Angular verwendet. Angular ist ein TypeScript-basiertes Framework und bietet die Entwicklung einer komponentenbasierten Frontend-Architektur, welche die Anwendung in wiederverwendbare Komponenten unterteilt, was die Wartbarkeit und Effizienz während der Entwicklung steigert. Zusätzlich dazu besitzt Angular ein integriertes Routing-System, welches die Navigation in der Applikation unterstützt. Beide Frameworks werden außerdem bereits für das Inventarisierungssystem der Hochschule Esslingen, kurz InSy, verwendet und bieten somit eine gute Anschlussfähigkeit in die schon bestehende Infrastruktur.

5.3 Server

Die Web-Anwendung wird auf einem Server der bwCloud betrieben und dient dazu, die Applikation mittels Docker in einer containerisierten Umgebung bereitzustellen, wodurch neue Features einfacher integriert werden können. Des Weiteren existieren Instanzen, welche von der bereits bestehenden Infrastruktur genutzt werden.

5.4 Datenbank

Für die Speicherung und Verwaltung der Daten wird die Datenbank PostgreSQL verwendet. PostgreSQL ist ein Open-Source-Datenbankmanagementsystem, das leicht skalierbar und stabil ist. Es ermöglicht den Betrieb von relationalen Datenbanken und bearbeitet Transaktionen nach dem ACID Prinzip. Zur Visualisierung und Verwaltung der Daten wird pgAdmin verwendet, welches ebenfalls eine Open-Source-Software ist, welche speziell für PostgreSQL entwickelt wurde.

5.5 Development-Server

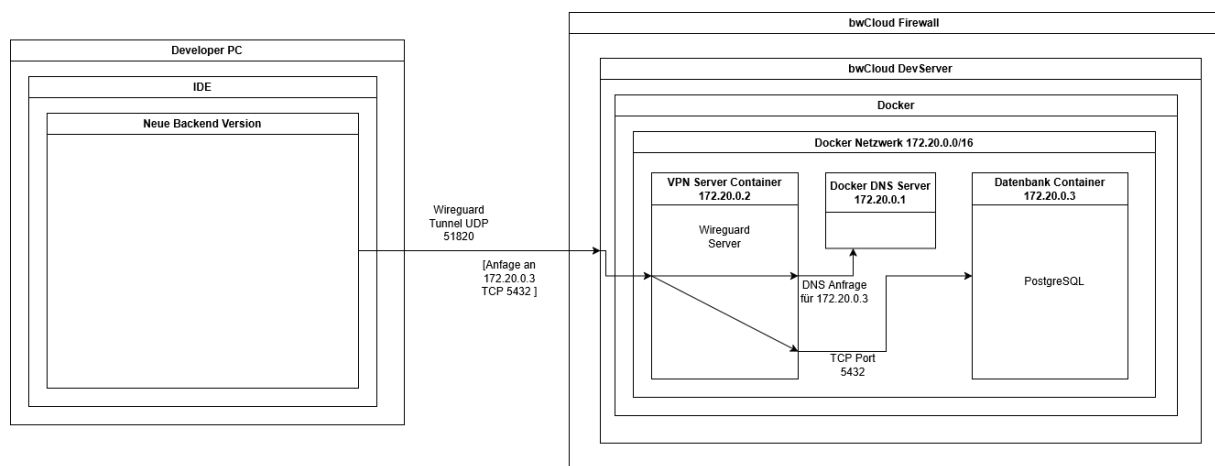


Abbildung 1: Netzwerkübersicht Development-Server

Wir haben für die Entwicklung einen eigenen Server eingerichtet, auf dem eine Datenbank läuft. Diese Datenbank ist mit Mock-Daten gefüllt, damit alle Entwickler mit denselben Informationen arbeiten können und die Ergebnisse vergleichbar bleiben.

Der Zugriff auf die Datenbank erfolgt nicht direkt, sondern über ein VPN, das wir mit WireGuard umgesetzt haben. WireGuard haben wir gewählt, weil es einfach einzurichten ist, zuverlässig funktioniert und eine sichere Verbindung bietet.

Man kann sich über WireGuard mit dem VPN verbinden. Sobald die Verbindung steht, können die Entwickler auf die Datenbank zugreifen und dort mit den vorbereiteten Mock-Daten arbeiten. So haben wir eine gemeinsame Grundlage geschaffen, die für alle gleich ist und die Entwicklung deutlich erleichtert.

Damit ist die Basis gelegt, also ein Server, eine Datenbank mit einheitlichen Testdaten und ein VPN für den sicheren Zugang. Mehr war für unseren aktuellen Stand nicht nötig, aber es sorgt dafür, dass wir stabil und konsistent arbeiten können.

5.6 Version Control management System

Für die Versionsverwaltung der Applikation wird Git verwendet, mit einem Repository auf GitHub. Dadurch werden Änderungen am Source-Code dokumentiert und eine Zusammenarbeit von mehreren Entwicklern organisiert. Außerdem kann durch das Erstellen von Branches für einzelne Feature und das Mergen durch eine Pull-Request der Fehleranfälligkeit entgegengewirkt werden.

5.6.1 CI/CD Pipeline

Zur Automatisierung des Entwicklungsprozesses wurde eine CI/CD Pipeline entworfen, siehe Abb. 1. (Stand 18.10.2025). Die Pipeline ist aktuell noch nicht vollständig integriert. Wie in der Abbildung dargestellt wird mit jedem Commit eine statische Code-Analyse und Unit Tests durchgeführt. Die den jeweiligen neu hinzugefügten Code überprüfen. Nachdem der Code in die Hauptbranch gemerged wurde, wird der Code kompiliert und gebaut. In einer Testumgebung werden Integrationstests durchgeführt und anschließend in einer Staging-Umgebung abgelegt, um manuell in Docker-Container eingefügt zu werden.

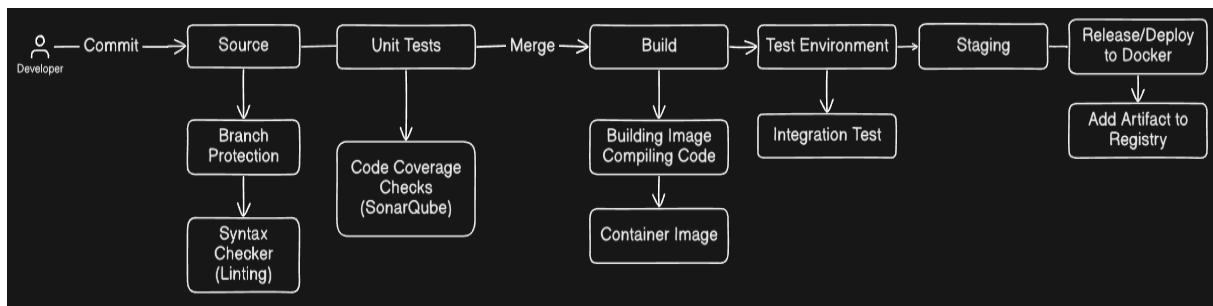


Abbildung 2: Entwurf der CI/CD Pipeline Architektur in GitHub

Dadurch wird die Codequalität konstant gehalten und die Wartbarkeit der Software verbessert. Außerdem werden Fehler durch die Tests frühzeitig erkannt und können so schneller und effizienter behoben werden.

6 UI-Prototyp

Der vollständige Prototyp ist direkt über den folgenden Figma-Link einsehbar: [Figma Dummy](#).

6.1 Farbpalette

6.1.1 Version 1 (Erstkonzept)

Schon in der ersten Version haben wir Grundfarben eingesetzt, um der Oberfläche ein stimmiges Erscheinungsbild zu geben. Zwar lag der Schwerpunkt noch auf der Struktur und dem Aufbau - also auf Suche, Listen und Detailseiten, aber eine schlichte Farbgestaltung war bereits vorhanden und wurde später beibehalten.

6.1.2 Version 2 (Wireframe)

Wir haben die Farben aus Version 1 übernommen und leicht verfeinert, um mehr Klarheit und Konsistenz zu schaffen. Dabei setzen wir auf neutrale UI-Farben:

- Primär: #0A0A0A
- Sekundär: #717182
- Ausgewählt: #E9EBEF
- Buttons: #000000
- Dazu die Statusfarben: #00A63E (verfügbar) und #C10007 (ausgeliehen).

6.1.3 Finale, farbige Version


In der finalen Version werden die hochschultypischen Farben verwendet:

- Primär: #012E58
- Sekundär: #32424A
- Ausgewählt: #32424A
- Buttons: #253359
- sowie wieder die Statusfarben: #00A63E (verfügbar) und #C10007 (ausgeliehen).

6.2 Erste Prototypen

6.2.1 Version 1 – Grundidee

- **Anmeldung & Rollen:** Eine einfache Anmeldemaske mit klarer Rollenauswahl (Studierende, Dozierende, Personal). (Siehe Abbildung 3)



HS-Esslingen

Digitales Buchausleihsystem der Hochschulbibliothek

Anmeldung
Melden Sie sich mit Ihrer Universitäts-E-Mail an

☒ Student ☐ Dozent ☐ Personal

E-Mail-Adresse

Passwort

Anmelden als Student

[Probleme beim Anmelden?](#)
[IT-Support kontaktieren](#)

Abbildung 3: Login screen

- **Suche & Trefferliste:** Suche über Bücher und Geräte mit Filtern (Medientyp, Kategorie, Campus, Verfügbarkeit, Sortierung nach „Relevanz“). Die Trefferliste zeigt direkt Verfügbarkeit, Standort und maximale Ausleihdauer. (Siehe Abbildung 4)

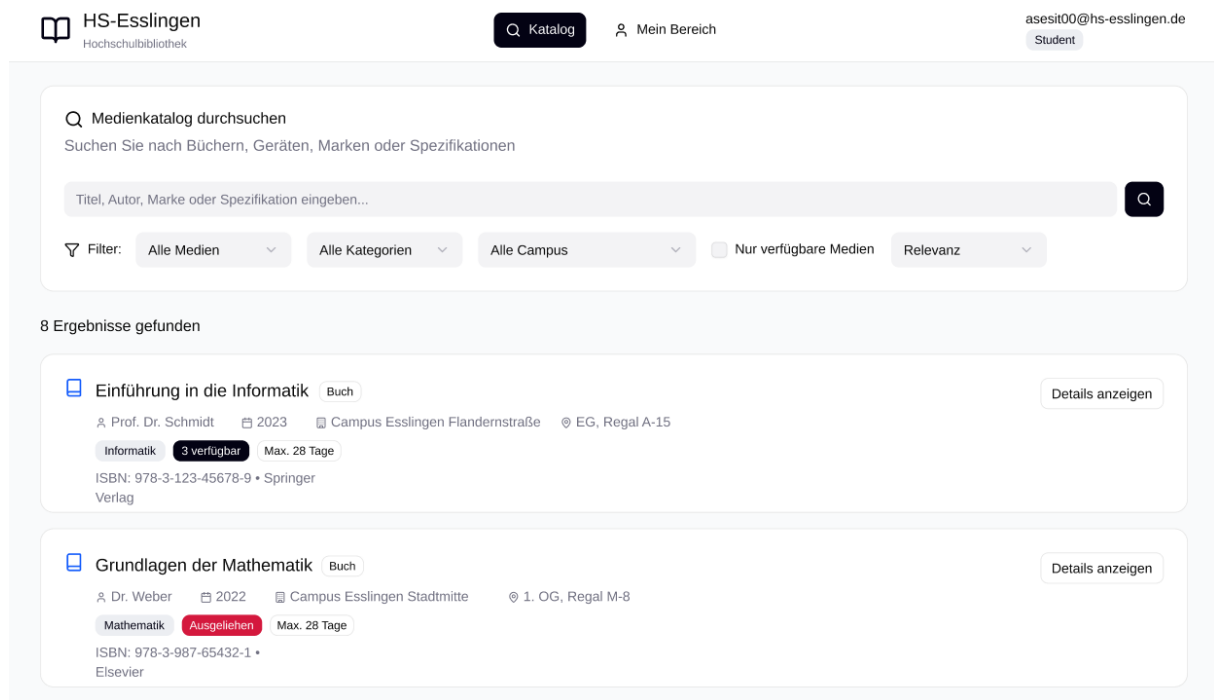


Abbildung 4: Inventarkatalog

- **Detailseite (Beispiel VR-Gerät):** Beschreibung, technische Spezifikationen, Zubehör, Verfügbarkeit je Campus, Ausleihmöglichkeit „Jetzt ausleihen“, Inventarnummer, Modell sowie die Ausleihbedingungen. Öffnungszeiten sind ebenfalls ersichtlich. (Siehe Abbildung 5)

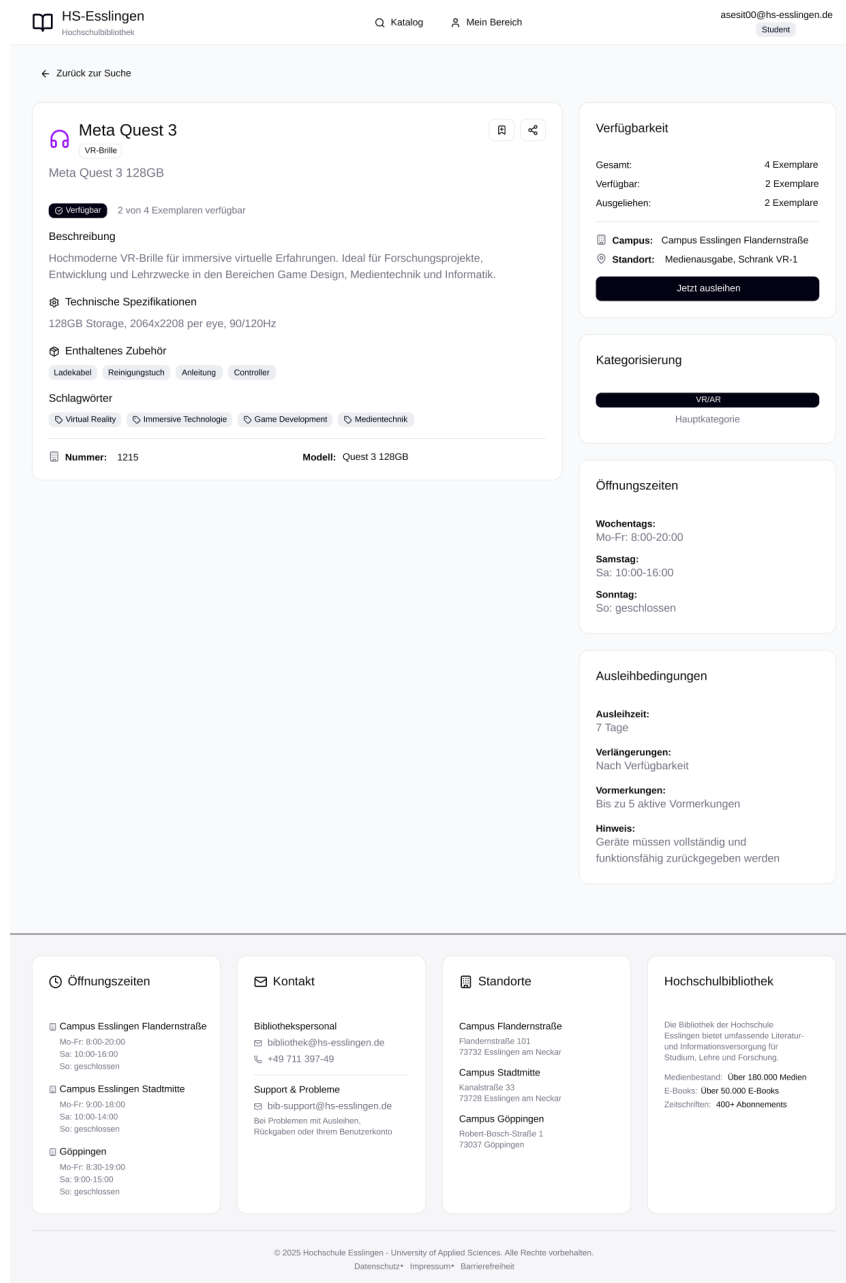


Abbildung 5: Geräte-Details Seite

- **Mein Bereich:** Übersichtsseite mit aktuellen Ausleihen, Fälligkeiten, Gebühren und Verlängerungsmöglichkeit. (Siehe Abbildung 6)

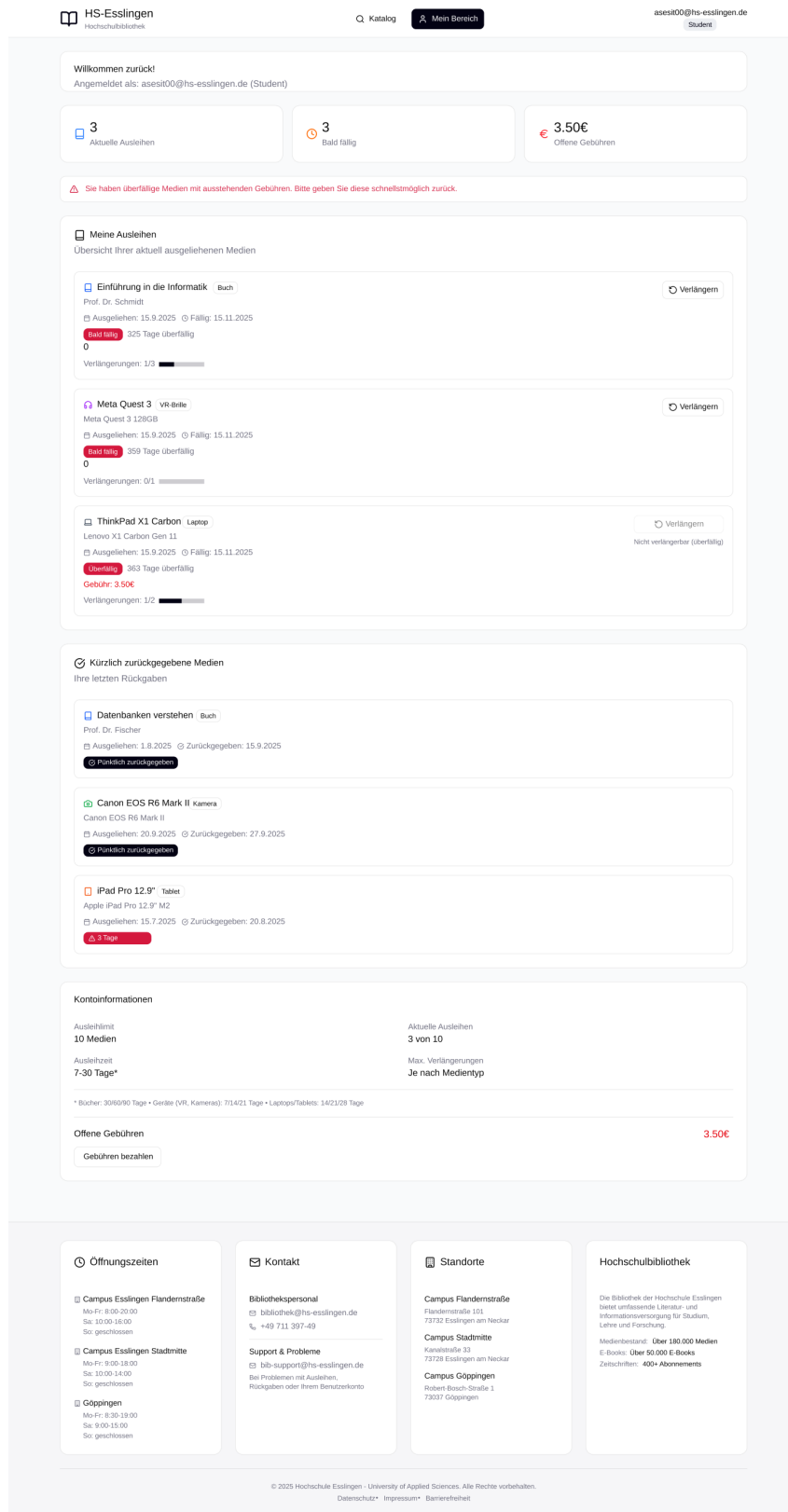


Abbildung 6: Persönlicher Bereich

6.2.2 Version 2 – Änderungen gegenüber Version 1

- **Mehr Fokus:**

- Bücher wurden komplett entfernt. Das System ist nun klar auf die Geräteausleihe ausgerichtet – nicht auf die Bibliothek.

- **Schnellere Übersicht:**

- In der Übersicht sieht man die Verfügbarkeit pro Campus direkt auf der Karteansicht. (Siehe Abbildung 7)

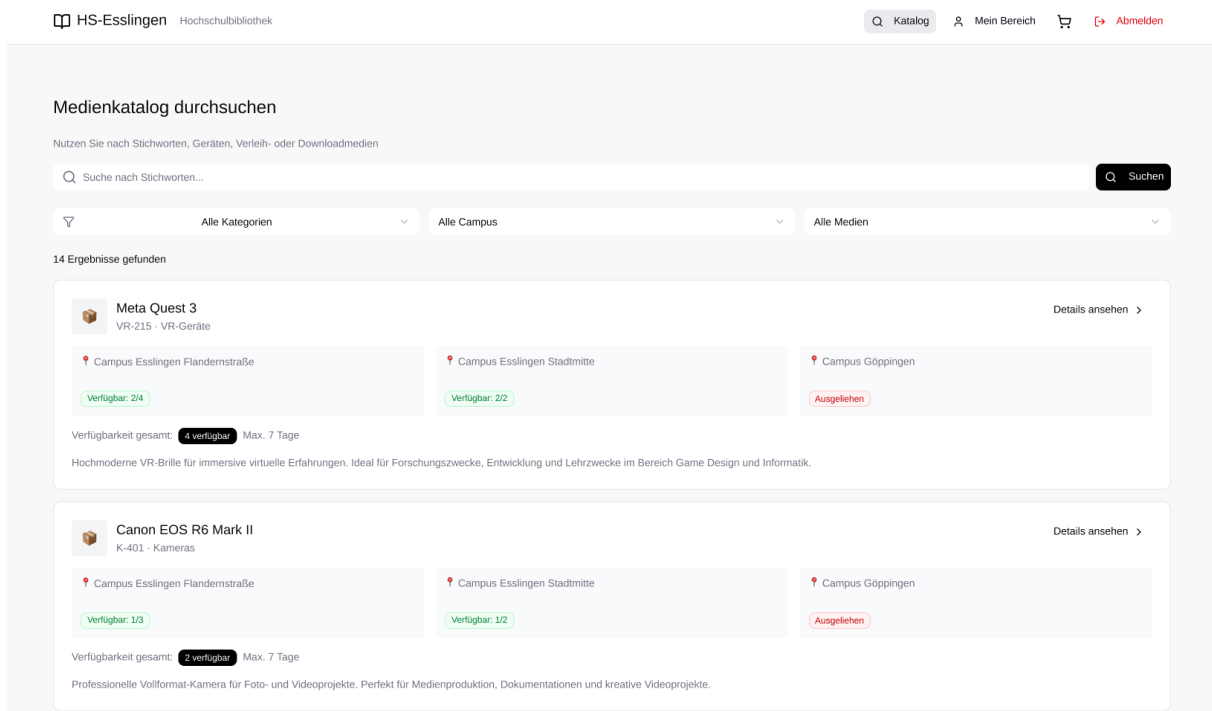


Abbildung 7: Aktualisierter Inventarkatalog

- **Gerätedetailseite sinnvoll erweitert**

- Auf Wunsch von dem Kunden wurde auf der Detailseite ein Kalender mit auswählbaren Zeitslots integriert. Nutzer können Datum und Zeitfenster direkt wählen und die Reservierung abschließen. (Siehe Abbildungen 8 und 9)

HS-Esslingen

Hochschulbibliothek

Katalog

Mein Bereich

Abmelden

Zurück zur Suche

Meta Quest 3

VR-Geräte

Verfügbar

VR-215

Beschreibung

Hochmoderne VR-Brille für immersive virtuelle Erfahrungen. Ideal für Forschungszwecke, Entwicklung und Lehrzwecke im Bereich Game Design und Informatik.

Technische Spezifikationen

Speicher:

128GB

Sensor:

256x256 pixel eye, 90/120Hz

Zubehör:

Ladekabel

Reinigungstuch

Anleitung

Controller

Enthaltenes Zubehör

Virtual Reality

Immersive Technologie

Game Development

Medientechnik

Schlagwörter

Virtual Reality

Immersive Technologie

Game Development

Medientechnik

Nummer:VR-215 · Modell:Meta Quest 3

Ausleihbedingungen

Ausleihzeit:

7 Tage

Verlängerungen:

Nach Verfügbarkeit

Vormerkungen:

Bis zu 5 aktive Vormerkungen

Hinweis:

Bis zu 5 aktive Vormerkungen. Hinweis: Geräte müssen vollständig und funktionstüchtig zurückgegeben werden

Verfügbarkeit

Gesamt:

4 Exemplare

verfügbar

4 Exemplare

ausgeliehen

Campus:

Campus Esslingen Flandernstraße

Standort:

Gebäude 01 - F 01.406

2/4 verfügbar

In den Warenkorb

Abholtermin wählen

Abholdatum:

November 2025

27	28	1	2	3	4
5	6	7	8	9	10
11	12	13	14	15	16
17	18	19	20	21	22
23	24	25	26	27	28
29	30	31	1	2	3

Done

Support

geraeteverleih@hs-esslingen.de

+49 711 397-3456

Für technische Probleme oder Fragen zur Ausleihe

Ausgabestellen

Medienausgabe Flandernstraße

Raum F-301, 3. OG

Geräteausgabe Stadtmitte

Gebäude 1, Erdgeschoss

IT-Ausgabe Göppingen

Bibliothek, Raum G-104

LeihSy

Professionelles System für Studierende und Lehrende der Hochschule Esslingen.

VR-Geräte: 12+ Headsets

Kameras: 15+ Profi-Kameras

Lichtsets & Equipment: 15+ Sets

© 2025 Hochschule Esslingen - University of Applied Sciences. Alle Rechte vorbehalten.

Datenschutz · Impressum · Barrierefreiheit

Abbildung 8: Kalender in Produktdetails

36

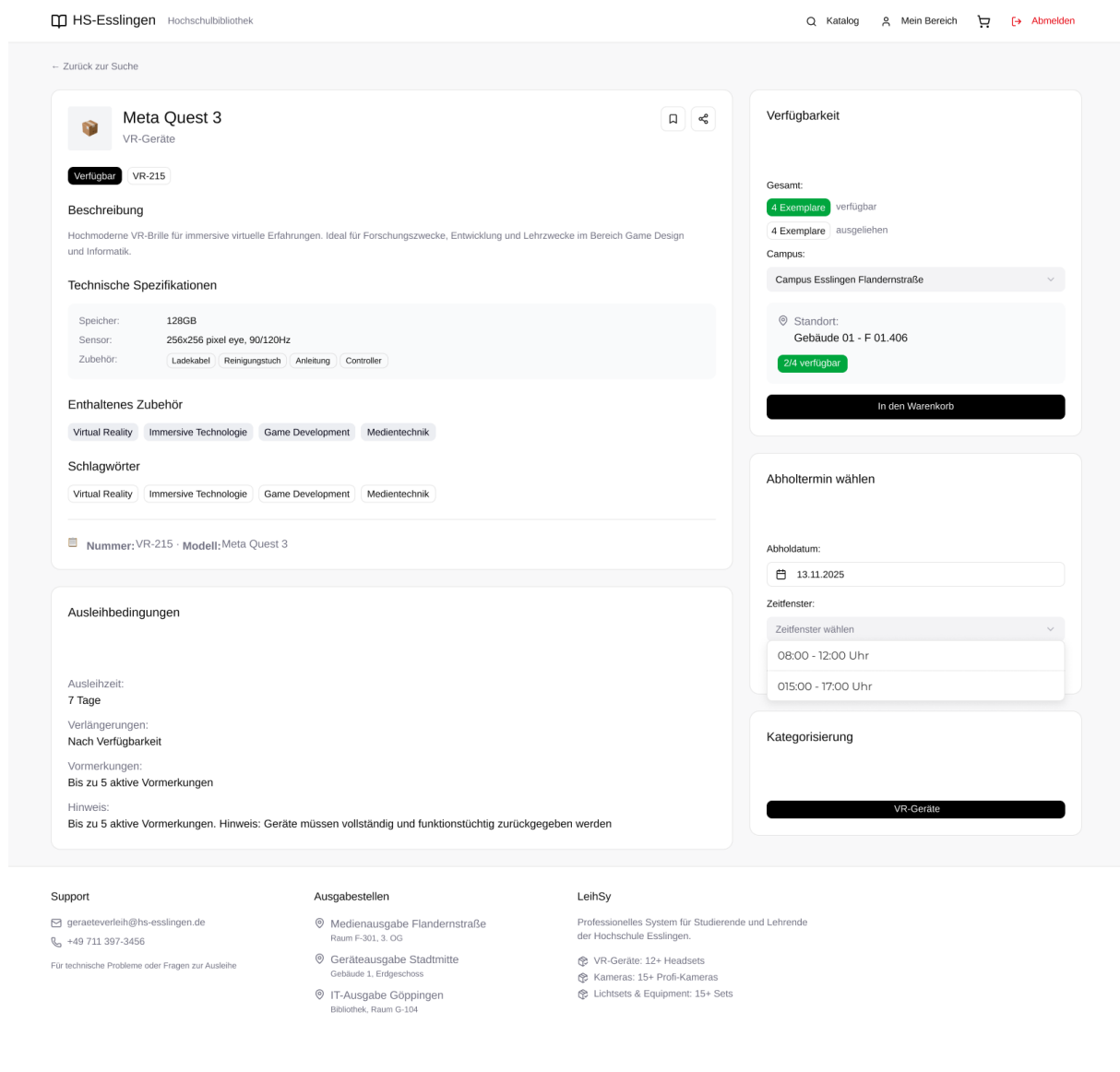


Abbildung 9: Abholungszeitfenster

- Auf Wunsch des Kunden wurde ein Warenkorb integriert. Darin können Nutzer ihre gewünschten Geräte direkt hinzufügen, einsehen und verwalten. (Siehe Abbildung 10)

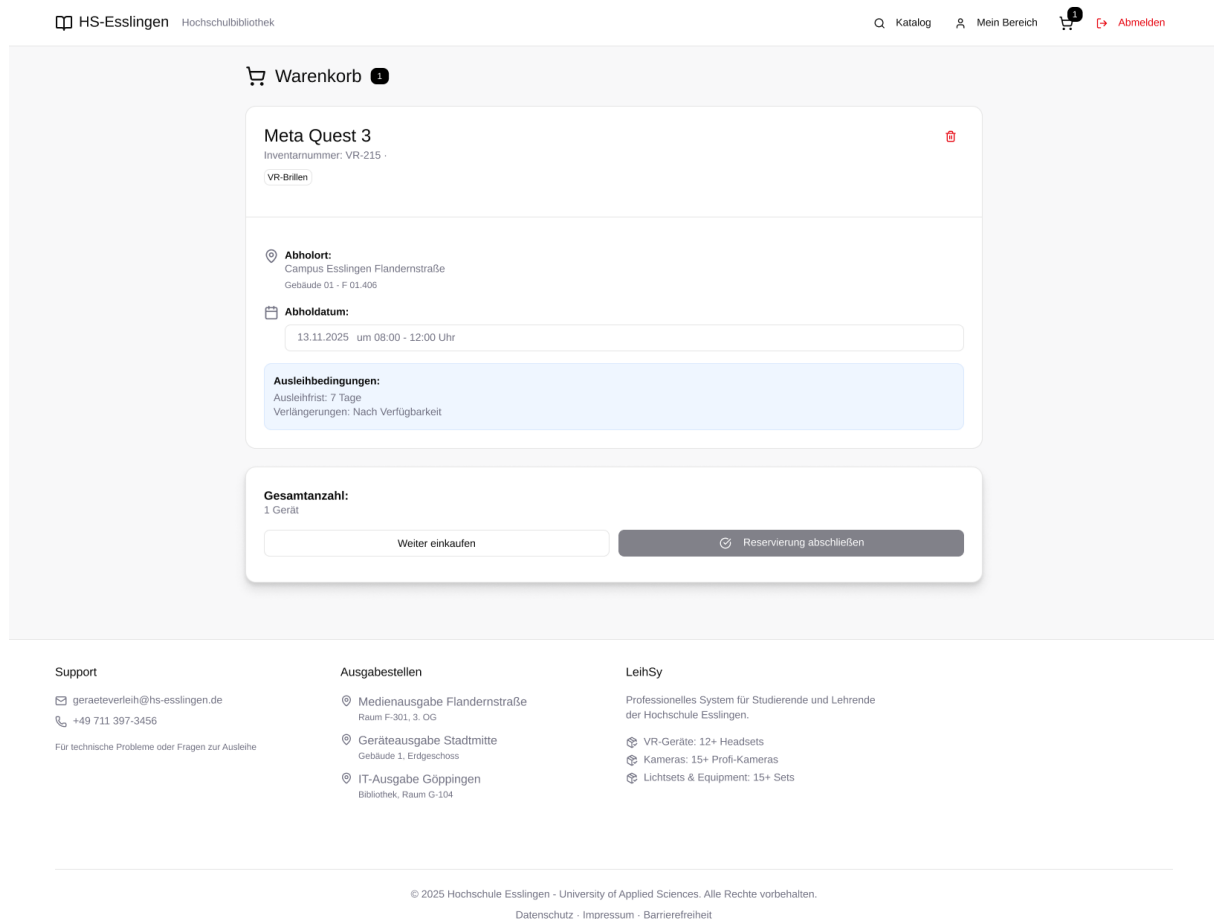


Abbildung 10: Warenkorb

- Auf der Gerätedetailseite wird oberhalb des Buttons „In den Warenkorb“ der Standort angezeigt. So erkennen die Nutzer sofort, wo sie das gewählte Gerät erhalten können. (Siehe Abbildung 11)

HS-Esslingen

Hochschulbibliothek

Katalog

Mein Bereich

Abmelden

Zurück zur Suche

Meta Quest 3

VR-Geräte

Verfügbar

VR-215

Beschreibung

Hochmoderne VR-Brille für immersive virtuelle Erfahrungen. Ideal für Forschungszwecke, Entwicklung und Lehrzwecke im Bereich Game Design und Informatik.

Technische Spezifikationen

Speicher:

128GB

Sensor:

256x256 pixel eye, 90/120Hz

Zubehör:

Ladekabel

Reinigungstuch

Anleitung

Controller

Enthaltenes Zubehör

Virtual Reality

Immersive Technologie

Game Development

Medientechnik

Schlagwörter

Virtual Reality

Immersive Technologie

Game Development

Medientechnik

Nummer:VR-215

Modell:Meta Quest 3

Ausleihbedingungen

Ausleihzeit:

7 Tage

Verlängerungen:

Nach Verfügbarkeit

Vormerkungen:

Bis zu 5 aktive Vormerkungen

Hinweis:

Bis zu 5 aktive Vormerkungen. Hinweis: Geräte müssen vollständig und funktionstüchtig zurückgegeben werden

Verfügbarkeit

Gesamt:

4 Exemplare

verfügbar

4 Exemplare

ausgeliehen

Campus:

Campus Esslingen Flandernstraße

Standort:

Gebäude 01 - F 01.406

2/4 verfügbar

In den Warenkorb

Abholtermin wählen

Abholdatum:

13.11.2025

Zeitenster:

Zeitenster wählen

08:00 - 12:00 Uhr

015:00 - 17:00 Uhr

Kategorisierung

VR-Geräte

Support

geraeteverleih@hs-esslingen.de

+49 711 397-3456

Für technische Probleme oder Fragen zur Ausleihe

Ausgabestellen

Medienausgabe Flandernstraße

Raum F-301, 3. OG

Geräteausgabe Stadtmitte

Gebäude 1, Erdgeschoss

IT-Ausgabe Göppingen

Bibliothek, Raum G-104

LeihSy

Professionelles System für Studierende und Lehrende der Hochschule Esslingen.

VR-Geräte: 12+ Headsets

Kameras: 15+ Profi-Kameras

Lichtsets & Equipment: 15+ Sets

© 2025 Hochschule Esslingen - University of Applied Sciences. Alle Rechte vorbehalten.

Datenschutz · Impressum · Barrierefreiheit

Abbildung 11: Standort des Geräts

- Der Bereich „Mein Bereich“ zeigt übersichtlich alle aktuellen Ausleihen, Reservierungen und offenen Gebühren. Zusätzlich können mehrere Geräte gleichzeitig verlängert werden. (Siehe Abbildung 12)

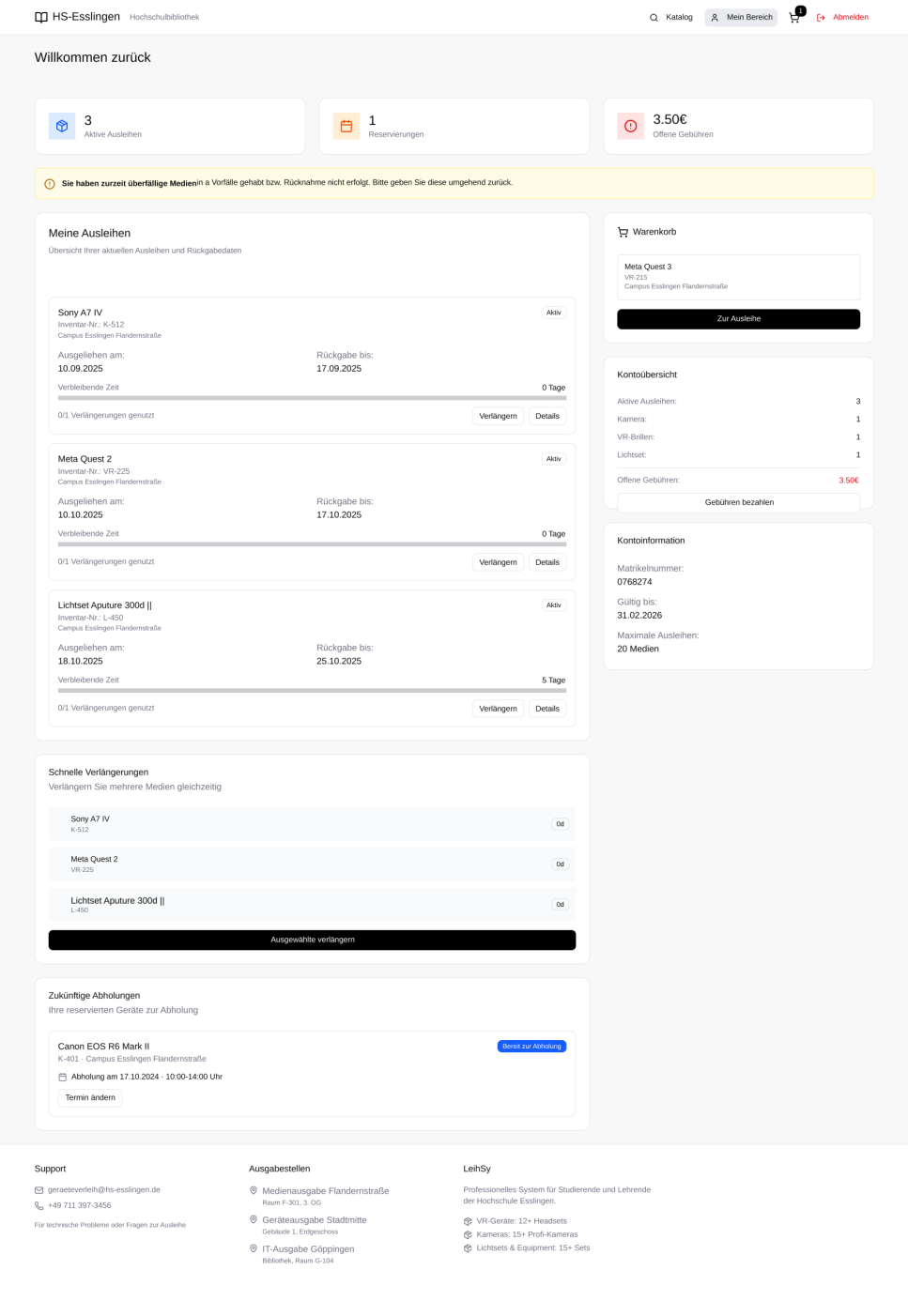
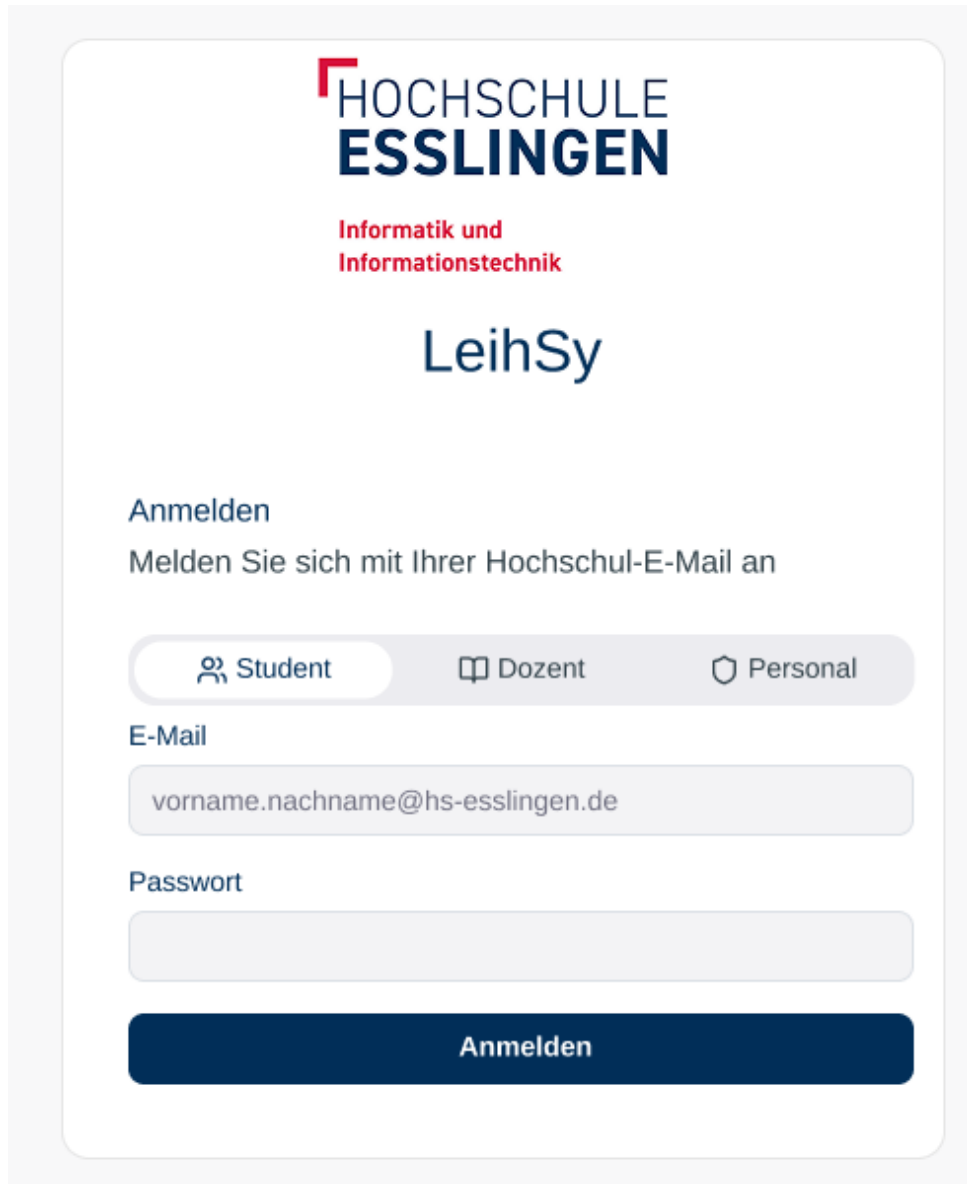


Abbildung 12: Aktualisierter persönlicher Bereich

6.3 Visuelle Umsetzung der Funktionen

6.3.1 Login-Seite

Hier ist die neue Version der Login-Seite. Sie orientiert sich am Standard Design der Hochschule Esslingen. Die blaue Farbe und das Logo wurden bewusst gewählt, sodass die Seite nicht fremd wirkt. (Siehe Abbildung 13)



The image shows a login interface for 'LeihSy' at 'HOCHSCHULE ESSLINGEN'. The header includes the university's logo and name, followed by 'Informatik und Informationstechnik' in red. Below this is the 'LeihSy' logo. The main section is titled 'Anmelden' and instructs users to log in with their university email. There are three role-based tabs: 'Student' (selected), 'Dozent', and 'Personal'. Below the tabs are input fields for 'E-Mail' (with a placeholder 'vorname.nachname@hs-esslingen.de') and 'Passwort'. A large blue 'Anmelden' button is at the bottom.

Abbildung 13: Aktualisierter Login screen

6.3.2 Geräte-Details

Oben links haben wir das Logo der Hochschule eingefügt, und die Farben wurden einheitlich angepasst. Die finale Version entspricht genau unserer Vorstellung. (Siehe Abbildung 14)

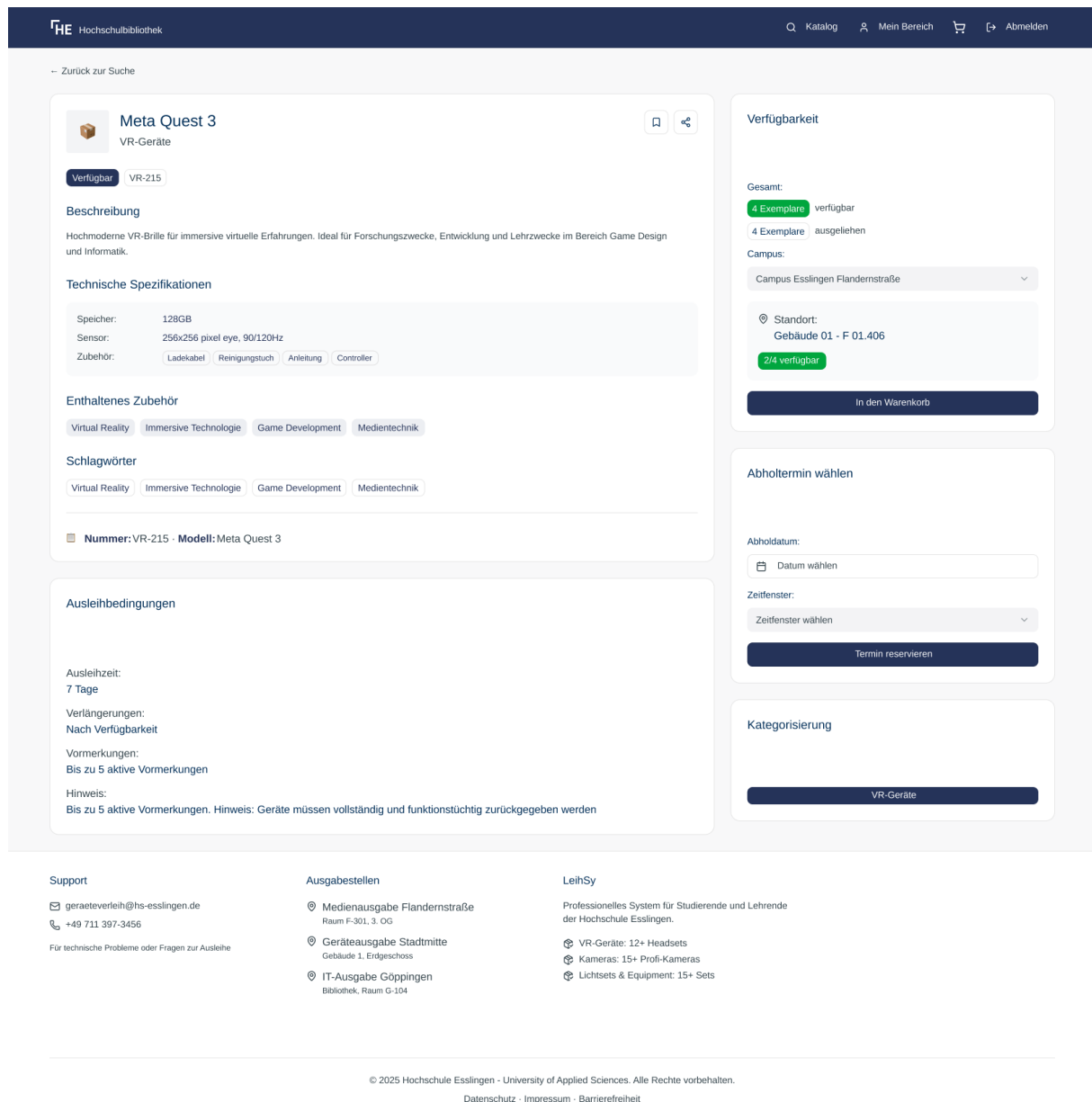





Abbildung 14: Geräte-Details Seite mit aktualisierten Design


6.3.3 Warenkorb

Wir haben das Warenkorb Logo so gestaltet, dass oben die Anzahl der hinzugefügten Geräte in Nummern dargestellt wird. Außerdem haben wir den Bereich für Terminreservierungen in Rot dargestellt, damit er sofort ins Auge fällt und die Aufmerksamkeit des Betrachters auf sich zieht. (Siehe Abbildung 15)

 Katalog

 Mein Bereich

 1

 Abmelden

Verfügbarkeit

Gesamt:

4 Exemplare


verfügbar

4 Exemplare

ausgeliehen

Campus:

Campus Esslingen Flandernstraße

 Standort:


Gebäude 01 - F 01.406

2/4 verfügbar

In den Warenkorb

Abholtermin wählen

Abholdatum:

 13.11.2025

Zeitfenster:

08:00 - 12:00 Uhr

Termin reserviert

Abbildung 15: Verbesselter Warenkorb

7 User Research

7.1 Proto-Personas

7.1.1 Persona A – Lena Schmid (Studierende)

- **Profil:** Bachelor Medieninformatik, organisiert mobil am Smartphone, wenig Geduld für Papier & Rückfragen.
- **Ziele:** schnell verfügbare Geräte finden, Slots planen, transparente Status-/Historienansicht.
- **Bedürfnisse:** Online-Reservierung mit klarer Live-Verfügbarkeit, automatische Bestätigungen & Erinnerungen, einfache Verlängerung, transparente Regeln/Gebühren.
- **Pain Points:** unklare Verfügbarkeit, Wartezeiten/keine Antwort → wünscht Auto-Storno, fehlende Erinnerungen → Überfälligkeit.

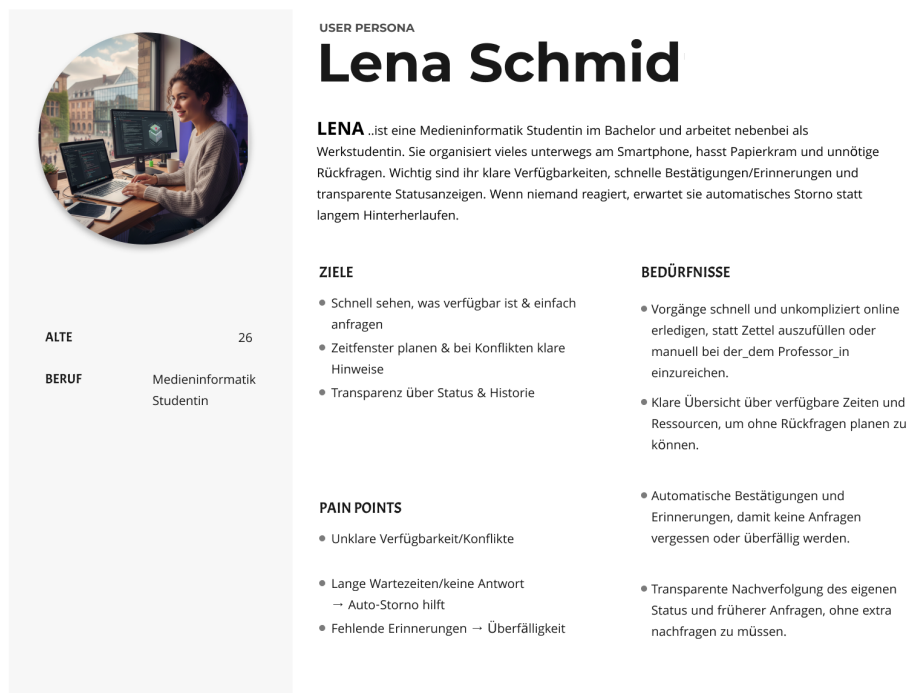



Abbildung 16: Persona Lena Schmid

7.1.2 Persona B – Max Schmidt (IT-Admin / Systemverantwortlicher)

- **Profil:** IT-Administrator (Hochschule), prozess- und sicherheitsorientiert.
- **Ziele:** Benutzer/Rollen zentral steuern, Inventar & Sets pflegen, geringe Supportlast.

- **Bedürfnisse:** SSO (Hochschul-Account), klares Rechte, sauberes Protokoll, Medienmanagement.
- **Pain Points:** Datenmüll, mangelnde Nachvollziehbarkeit, aufwändige Pflege.



ALTE 55

BERUF IT-Administratorin / Systemverantwortliche für das Hochschul-Leihsystem

USER PERSONA

Max Schmidt

Max Schmidt ... ist ein zielorientierter IT-Administrator, der an einer Hochschule arbeitet. Er hat eine Leidenschaft für Technologie, Sicherheit und stabile IT-Systeme. Trotz der oft komplexen und stressigen Aufgaben im Hochschul-IT-Bereich achtet Max darauf, den Überblick zu behalten und seine Projekte effizient umzusetzen. Neben seiner Arbeit interessiert er sich für neue Softwarelösungen, Automatisierung und die Optimierung von Prozessen, um die digitale Infrastruktur der Hochschule zu verbessern.

ZIELE

- Benutzer, Rollen & Berechtigungen zentral steuern
- Inventar anlegen/bearbeiten/sets pflegen, Bilder managen
- Gute Nutzererfahrung für alle Anwender
- Weniger Supportaufwand durch einfache Bedienung

BEDÜRFNISSE

- Single Sign-On (SSO): Nutzer sollen sich mit ihrem Hochschul-Account anmelden können – ohne zusätzliche Logins.
- Rechte- und Rollenmanagement: Klare Zugriffsebenen für Admins, Mitarbeitende und Studierende.
- Inventar-Formulare, Set-Erstellung mit Auto-Nummerierung

PAIN POINTS

- Datenmüll/Dubletten → Single Source (Keycloak), sparsame Datenspeicherung
- Nachvollziehbarkeit → unveränderbares Protokoll
- Medienmanagement → Größen/Formate/Thumbnails

Abbildung 17: Persona Max Schmidt

7.1.3 Persona C – Prof. Tom Fischer (Lehrender/Verleiher)

- **Profil:** Professor/Verantwortlicher für Leihgeräte (z. B. VR).
- **Ziele:** Anfragen schnell prüfen, saubere Ausgabe/Rückgabe vor Ort, Überblick über eigene Ressourcen.
- **Bedürfnisse:** Dashboard offener Anfragen mit Filter, Vorschläge für Alternativtermine, E-Mail-Trigger für Abholung/Rückgabe, zentrale Plattform (ideal integrierbar in Hochschul-Umgebung).
- **Pain Points:** viele Anfragen gleichzeitig → Priorisierung schwer, keine klare Verfügbarkeit, kein Überblick über bestehende Reservierungen.

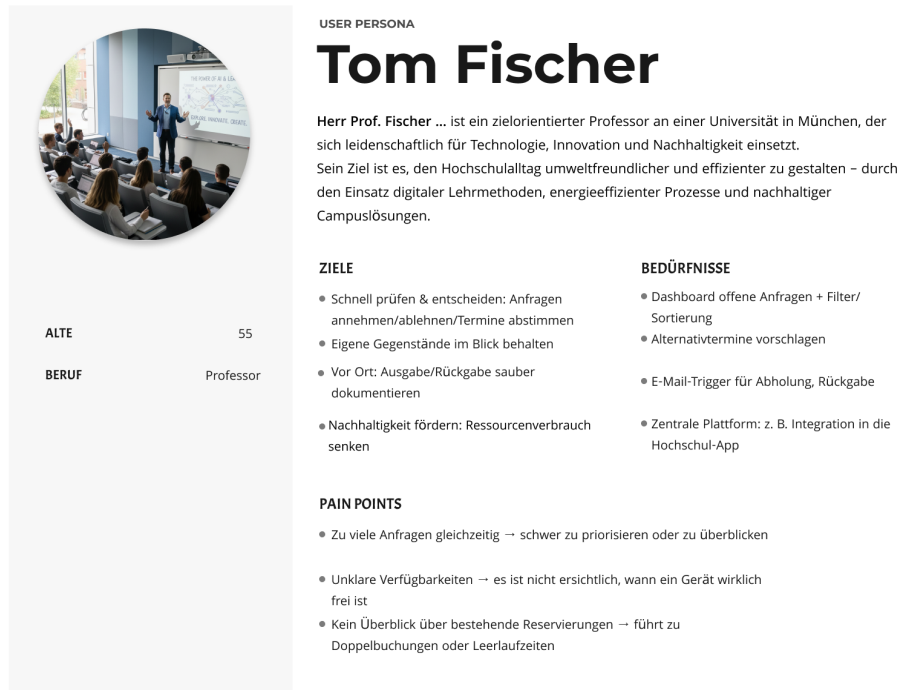


Abbildung 18: Persona Tom Fischer

7.2 Interview Auswertung

Für das Projekt „Leihsy“ wurden Studierende der Hochschule interviewt. Dabei wurde Ihnen unser UI-Prototyp gezeigt, und es wurden folgende Fragen gestellt.

- „Finde eine VR-Brille, die am Campus Flandernstraße verfügbar ist.“
- „Ist klar, wo das Gerät abgeholt wird (Standort)?“
- „Ist *Bald fällig* vs. *Überfällig* eindeutig?“
- „Wenn du eine Sache ändern könntest, was zuerst?“
- 1–5-Rating: „Wie einfach war es, X zu erledigen?“

Bisher sind alle Studierenden gut mit dem UI-Prototyp zurechtgekommen und konnten alle wichtigen Funktionen problemlos finden. Die Farbgestaltung wurde als passend empfunden, und auf den ersten Blick war alles klar erkennbar. Daher haben wir in unserem Ranking 4,5 von 5 Punkten erhalten.

Der halbe Punkt Abzug ergibt sich aus der Darstellung des Raums für die Geräteabholung. Da über dem Raum der Campus mit einem Standort-Emoji angezeigt wird, war es den Studierenden nicht sofort klar, wo genau sich der Raum befindet. Das Emoji hat die Aufmerksamkeit vom Raum leicht abgelenkt.

7.3 Auswertung (Online-Umfrage, n = 28)

7.3.1 Stichprobe & Nutzung

- In den letzten 24 Monaten ausgeliehen: 6 (21 %); nicht ausgeliehen: 22 (79 %).
- Genannte Kategorien (Mehrfachauswahl): VR-Brillen 4, Sonstiges 2, Kamera 0.

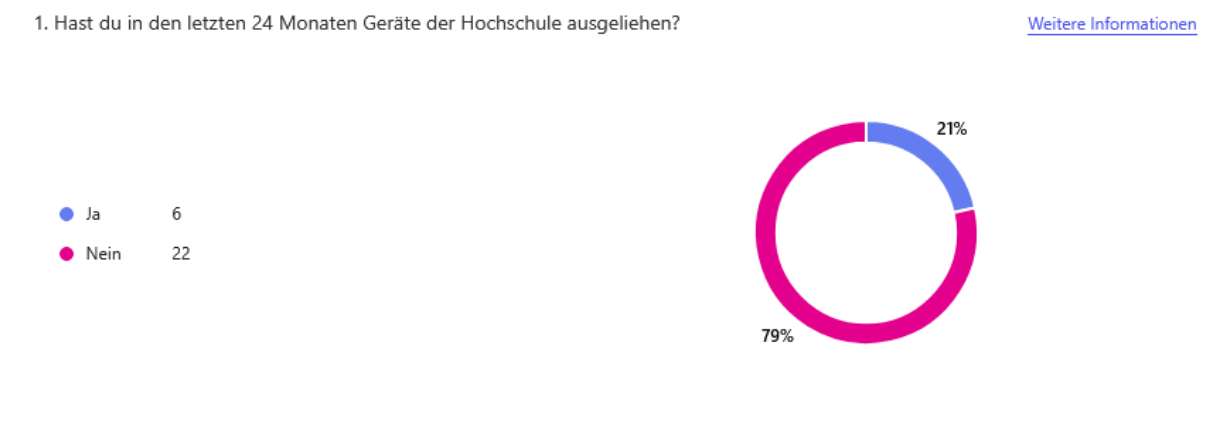


Abbildung 19: Anteil Studenten mit kürzlichen Ausleihen

7.3.2 Zufriedenheit (Ist-Prozess)

Ø-Bewertung: 3,00 (mittelmäßig) → klarer Verbesserungsbedarf.

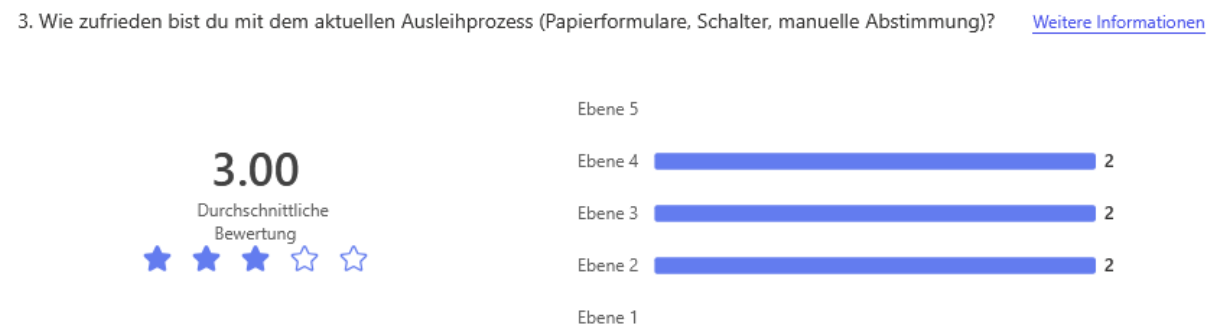


Abbildung 20: Zufriedenheit mit aktuellem Prozess

7.3.3 Größte Pain Points (Top-Nennungen)

- Formulare/Unterschriften 5
- Unklare Verfügbarkeit 3

- Rückgabe unflexibel 3
- Abholung nicht planbar 2

⇒ Papier & manuelle Abstimmung sind die Hauptbremser; Echtzeit-Infos & flexible Slots fehlen.

4. Was nervt dich am meisten?

[Weitere Informationen](#)

Unklare Verfügbarkeit	3
Formulare/Unterschriften	5
Abholung nicht planbar	2
Rückgabe unflexibel	3
Sonstiges	0

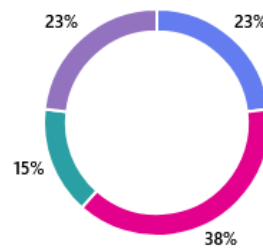


Abbildung 21: aktuelle Pain Points

7.3.4 Adoptionsbereitschaft für ein digitales System

- Ja 13 + Eher ja 12 → ~89 % positiv.
- Unsicher 2 (~7 %), Nein 1 (~4 %).

⇒ Die hohe Nutzungsintention spricht klar für die Umsetzung.

5. Wenn es ein zentrales, digitales System gäbe (Suchen, Verfügbarkeit je Campus, Online-Reservierung, Abhol-/Rückgabe-Slots): würdest du es nutzen?

[Weitere Informationen](#)

Ja	13
Eher ja	12
Unsicher	2
Eher nein	0
Nein	1

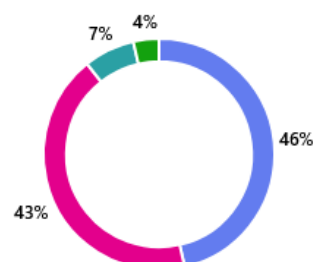


Abbildung 22: Adoptionsbereitschaft für digitales System

7.3.5 Top-Features (max. 3 Stimmen)

- Erinnerungen vor Rückgabe – 17

- Live-Verfügbarkeit – 16
- Online-Verlängerung – 12
- Standort-Filter (Campus) – 11
- Maximale Ausleihdauer sichtbar – 8
- Transparentes Gebühren-/Zubehör-Listing – 7

6. Was wäre dir dabei am wichtigsten? (max. 3 auswählen)

[Weitere Informationen](#)

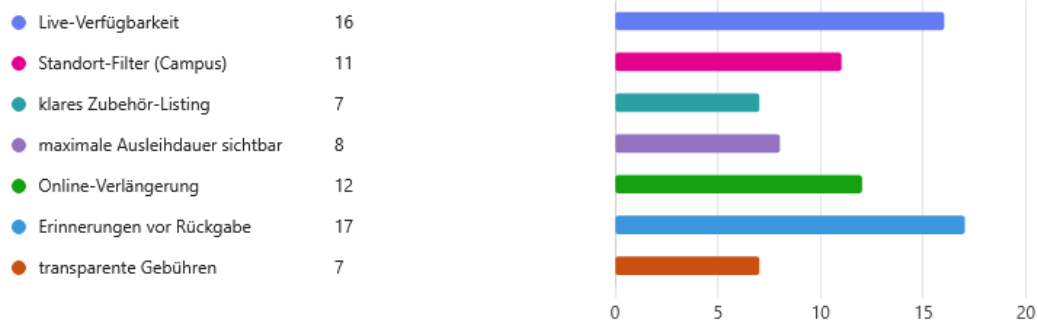


Abbildung 23: Wichtigste Features

7.3.6 Unverzichtbare Filter

- Nur verfügbare Geräte – 19 (Must-Have)
- Maximale Ausleihdauer – 5
- Campus – 4

7. Welche Filter wären für dich unverzichtbar? (Mehrfachauswahl möglich)

[Weitere Informationen](#)



Abbildung 24: Unverzichtbare Filter

7.3.7 Verfügbarkeit je Campus

- sehr hilfreich – 14 (50 %)
- eher hilfreich – 10 (35,7 %)
- eher nicht hilfreich – 3 (10,7 %)
- gar nicht hilfreich – 1 (3,6 %)

⇒ Erkenntnis: Insgesamt 85,7 % der Befragten empfinden die Anzeige als hilfreich. Die Funktion gilt somit als klarer Mehrwert und sollte im System standardmäßig integriert werden.

8. Wie hilfreich ist die Anzeige „Verfügbarkeit je Campus“ für deine Planung?

[Weitere Informationen](#)

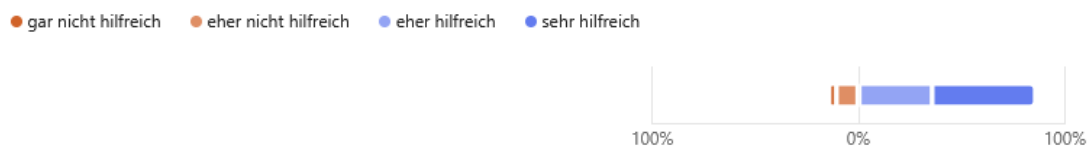


Abbildung 25: Wichtigkeit Verfügbarkeit nach Campus

7.3.8 Storno-Fenster (Abhol-/Rückgabe-Slots)

- bis 24 h vorher – 14 (50 %)
- bis 12 h vorher – 8 (29 %)
- bis 5 h vorher – 6 (21 %)

⇒ Empfehlung: Standard 24 h, ggf. 12 h für „Kurzfristig“.

9. Wie kurzfristig sollen Abhol-/Rückgabe-Slots stornierbar sein?

[Weitere Informationen](#)



Abbildung 26: Länge Stornierbarkeit

8 Softwarearchitektur

8.1 Entity-Relationship-Diagramme

Das in Abbildung 26 dargestellte Entity-Relationship-Modell beschreibt die strukturellen Zusammenhänge der in *LeihSy* verwalteten Entitäten und bildet die Grundlage für die relationale Implementierung der Datenbank. Das Modell verfolgt das Ziel, die Prozesse rund um die Verwaltung von Ausleihen, Produkten und zugehörigen Ressourcen konsistent und nachvollziehbar abzubilden.

Zentrale Entitäten Die Entität **Users** repräsentiert alle im System registrierten Benutzer. Neben Identifikationsmerkmalen umfasst sie Informationen wie Name, Budget sowie Metadaten zur Erstellung und Historisierung. Die Authentifizierung erfolgt nicht direkt im System, sondern über das externe Identitätsmanagement *Keycloak*, das für jede Benutzerinstanz anhand einer 1:1-Beziehung eindeutige Rollen- und E-Mail-Informationen bereitstellt.

Die Entität **Bookings** bildet den Kernprozess der Ausleihverwaltung ab. Eine Buchung wird von einer Benutzerinstanz initiiert und kann mehrere Items enthalten. Neben zeitlichen Attributen wie etwa Start-, End- und Rückgabedatum verwaltet die Entität Zustandsinformationen sowie potenzielle Vorschlags- oder Ausleihzeitpunkte. Die Beziehung zwischen **Users** und **Bookings** ist als 1:n ausgeprägt, da ein Benutzer mehrere Buchungen anlegen kann.

Ressourcen und Kategorien Mit der Entität **Products** werden alle ausleihbaren Produktarten strukturiert erfasst. Jedes Produkt ist einer Kategorie zugeordnet und kann mehrere physische Einheiten, sogenannte *Items*, besitzen. Die Entität enthält unter anderem Beschreibungen, Preisangaben, ein Bildverweisfeld sowie Verweise auf Kategorien und Standorte. Die Beziehung zwischen **Products** und **Items** ist folglich 1:n. Darüber hinaus kann ein Produkt Teil eines Sets sein.

Die physische Ebene wird durch die Entität **Items** repräsentiert. Jedes Item ist eine konkrete Instanz eines Produkts und verweist über einen Fremdschlüssel auf **InSy**. Zusätzlich werden Eigentumsinformationen, Inventarnummern sowie Metadaten zur Bestandsführung gespeichert.

Die Klassifizierung der Produkte erfolgt über die Entität **Categories**, die eine hierarchisch flache Struktur abbildet. Kategorien ermöglichen eine semantische Gruppierung der Produktpalette und unterstützen die spätere Filter- und Suchlogik im System.

Sets und Standorte Ein weiteres strukturelles Element stellen **Sets** dar. Ein Set ist eine definierte Kombination mehrerer Produkte, die gemeinsam ausgeliehen werden können. Die Beziehung ist n:m, sodass ein Set mehrere Produkte enthalten kann und ein

Produkt mehreren Sets zugeordnet werden kann. Die Umsetzung erfolgt über eine assoziative Entität mit Verweisen auf Produkte sowie entsprechenden Metadaten.

Standortinformationen werden über die Entität **Locations** repräsentiert. Dies ermöglicht die räumliche Zuordnung einzelner Produkte und Items. Jeder Standort kann mehreren Produkten zugeordnet sein, während jedes Produkt genau einem Standort zugewiesen ist (1:n-Beziehung).

Externe Systeme Die Entität **InSy** ist ein externes Inventarisierungssystem, welches Produktinformationen zur Verfügung stellt.

Konsistenz und Historisierung Mehrere Entitäten enthalten optionale **deleted_at**-Attribute, die eine logische Löschung ermöglichen. Damit wird eine einfache Datenhaltung unterstützt, ohne physische Datensätze vollständig zu entfernen oder die Struktur der Tabellen zu verändern. Ergänzend werden **created_at**- und **updated_at**-Attribute zur transparenten Nachvollziehbarkeit von Zustandsänderungen eingesetzt.

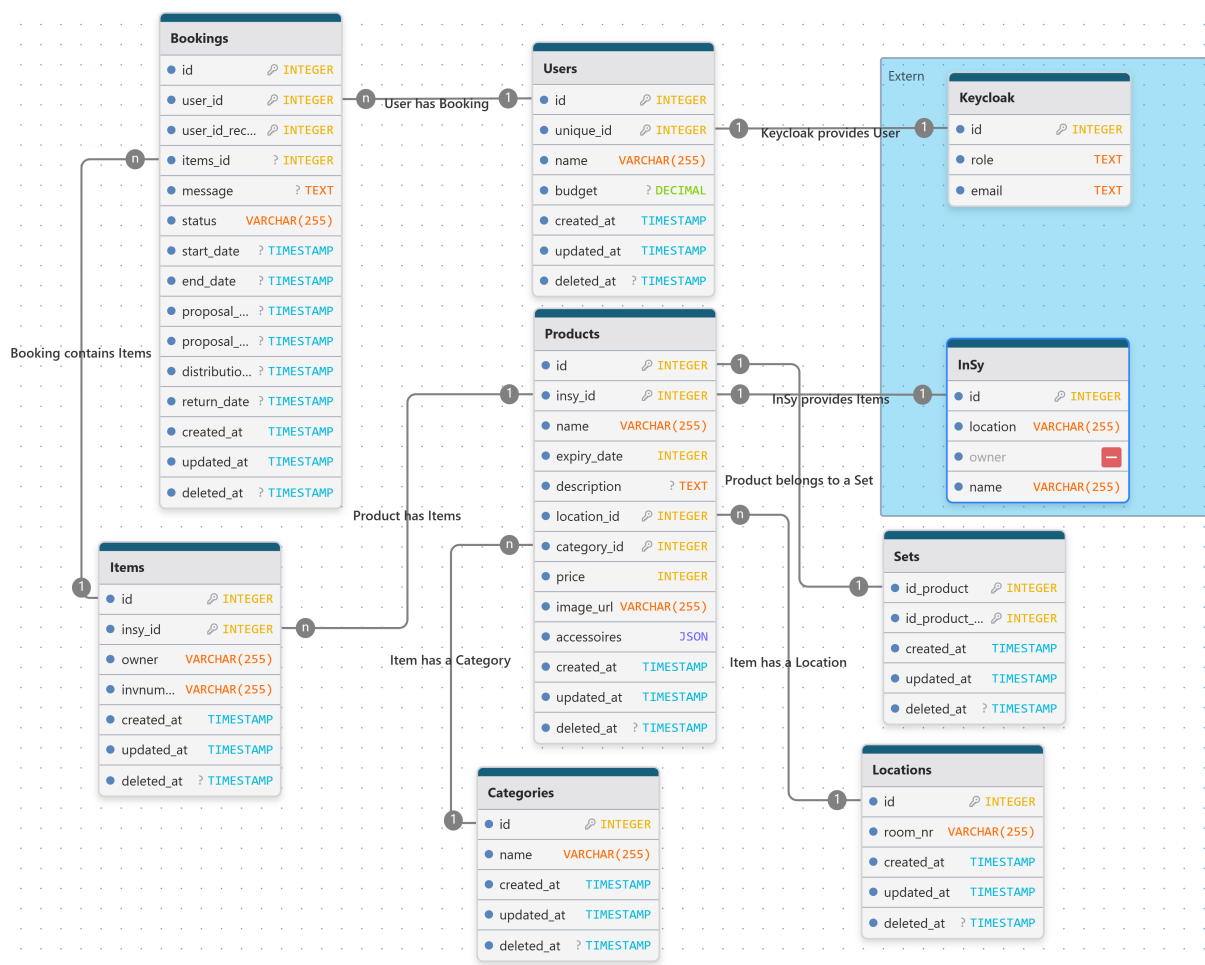


Abbildung 27: Entity-Relationship-Diagramm

8.2 Logische Sichten

8.2.1 Verteilungssicht

Die Anwendung folgt einem klassischen Client-Server-Architekturmodell. Der Client ist der Webbrowser, der auf dem Endgerät des Benutzers ausgeführt wird. Möchte der Benutzer die Anwendung aufrufen, sendet sein Browser zunächst HTTPS-Anfragen an einen Webserver, der als Docker-Container auf einem bwCloud-Server betrieben wird. Alle eingehenden Anfragen passieren zuvor einen Reverse Proxy, der die Requests anhand der Ziel-URL an die jeweils zuständigen Container weiterleitet.

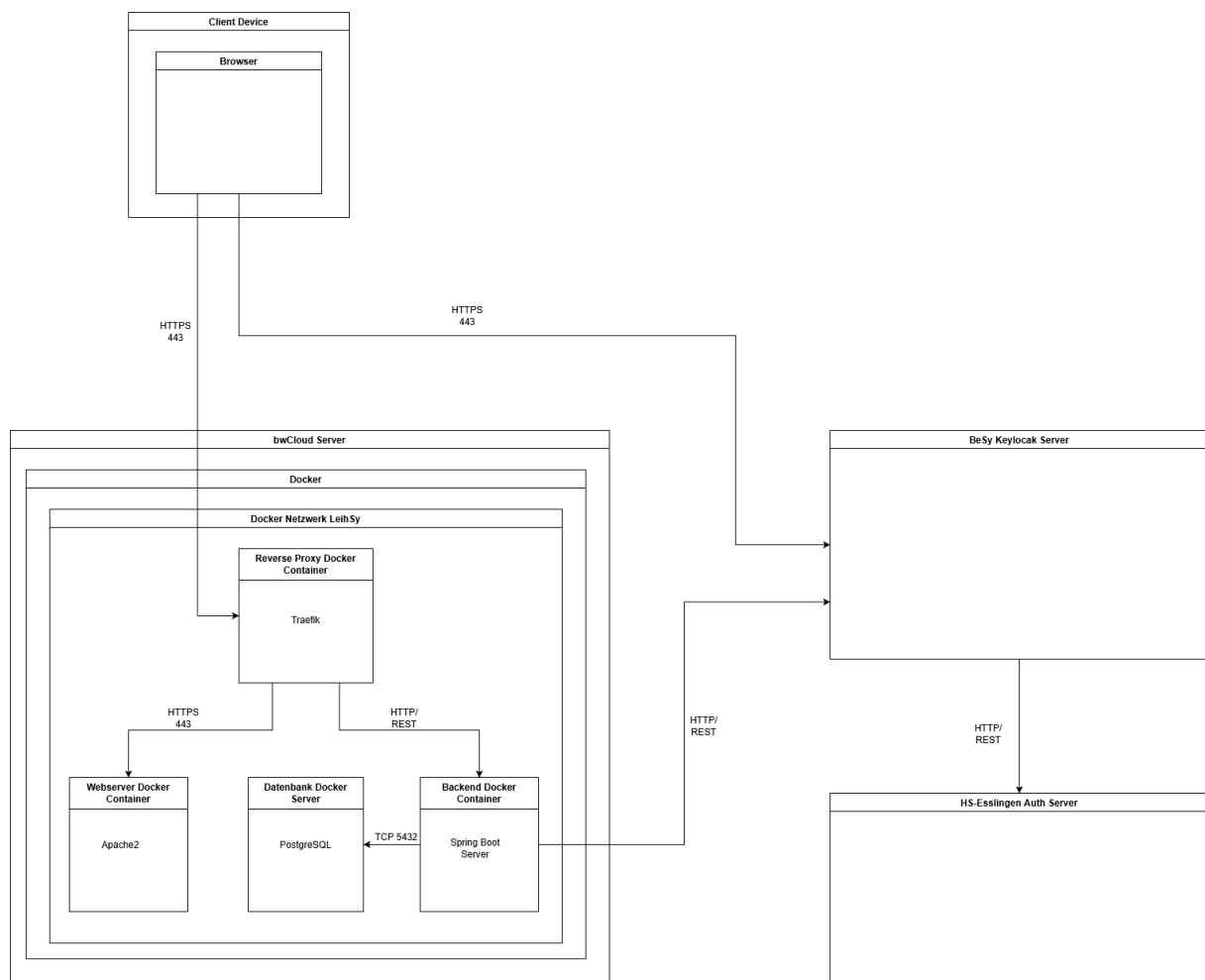


Abbildung 28: Diagramm zur Verteilungssicht

Der Webserver stellt dem Browser die benötigten statischen Ressourcen – HTML-, CSS- und JavaScript-Dateien – zur Verfügung, welche die Benutzeroberfläche der Anwendung aufbauen. Für den Zugriff auf die Daten des Ausleihsystems sendet der Browser anschließend HTTP/REST-Anfragen an die in der JavaScript-Anwendung definierten API-Endpunkte des Backends. Diese Anfragen werden ebenfalls vom Reverse Proxy empfangen und intern an den Backend-Container weitergeleitet.

Zur Persistierung von Daten kommuniziert das Backend über den TCP-Port 5432 mit der PostgreSQL-Datenbank. Da sich sowohl Datenbank-Container als auch Backend, Webserver und Reverse Proxy im gleichen Docker-Netzwerk befinden, erfolgt die Kommunikation direkt über dieses interne Netzwerk und ist von außen isoliert.

Für die Authentifizierung nutzt das System einen zentralen Keycloak-Server, der gemeinsam mit dem Projekt BeSy betrieben wird. Möchte ein Benutzer sich anmelden, erhält dessen Browser vom Backend eine Redirect-URL zum Keycloak-Auth-Endpoint. Der Browser ruft diesen Endpoint über HTTPS auf und der Benutzer authentifiziert sich dort mittels seines Hochschulaccounts.

Nach erfolgreicher Anmeldung leitet Keycloak den Browser mit einem Authorization Code zurück zum Backend, das diesen Code serverseitig gegen Access- und ID-Tokens eintauscht. Für die weitere Kommunikation speichert das Backend die Sitzung (z. B. mittels Session-Cookie) oder validiert eingehende Tokens lokal anhand des öffentlichen Schlüssels von Keycloak.

8.2.2 Struktursicht

Das Komponentendiagramm in Abb. 29 beschreibt die Architektur des LeihSy-Systems, das aus einem modular aufgebauten Frontend, einem funktionsorientierten Backend sowie mehreren externen Diensten besteht. Die Darstellung zeigt die Interaktionen der einzelnen Komponenten und verdeutlicht ihre funktionalen Abhängigkeiten.

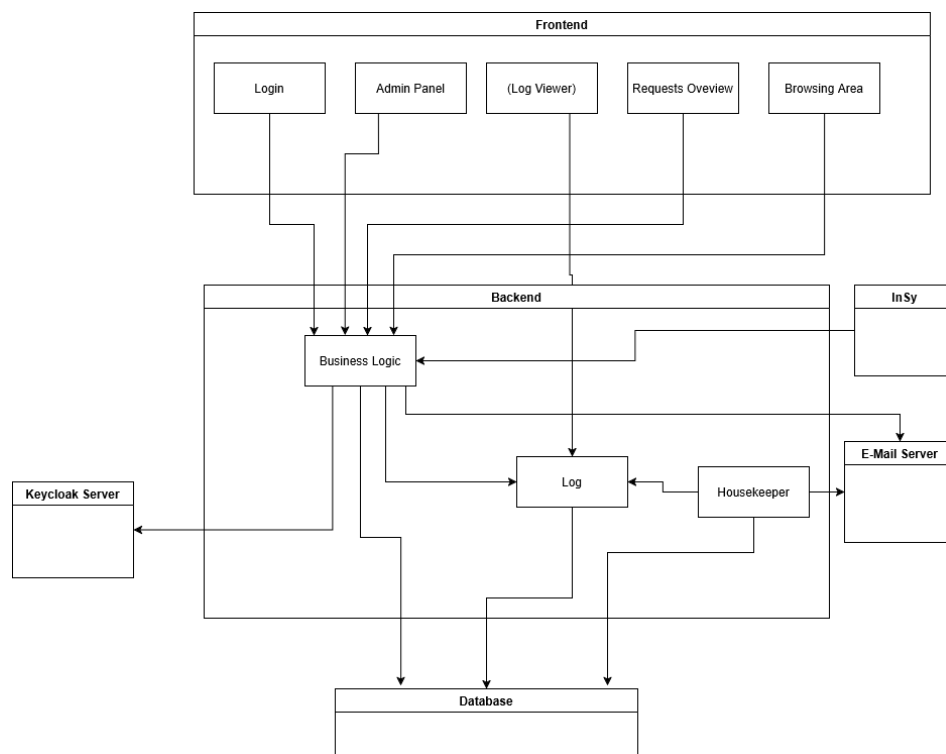


Abbildung 29: Komponentendiagramm

Das Frontend bildet die Präsentationsschicht des Systems und dient als Schnittstelle zwischen den Benutzerinnen und Benutzern und der Anwendungslogik. Es umfasst mehrere eindeutig abgegrenzte Funktionsbereiche:

- **Login:** Komponente zur Benutzerauthentifizierung.
- **Admin Panel:** Administrationsoberfläche für Konfigurations- und Verwaltungsaufgaben.
- **Log Viewer:** Optionaler oder eingeschränkter Zugriff auf System- und Anwendungsprotokolle.
- **Requests Overview:** Übersichtskomponente zur Darstellung und Verwaltung eingehender Anfragen.
- **Browsing Area:** Bereich zur Recherche oder Durchsicht von Datenobjekten.

Alle Frontend-Komponenten interagieren ausschließlich über definierte Schnittstellen mit dem Backend und enthalten keine eigene Geschäftslogik.

Das Backend stellt den zentralen Verarbeitungs- und Koordinationsmechanismus des Gesamtsystems dar. Es implementiert die Geschäftslogik und übernimmt sämtliche Datenzugriffe.

Die Komponente Business Logic bildet das funktionale Kernstück der Architektur. Ihre Hauptaufgaben umfassen:

- die Verarbeitung aller durch das Frontend ausgelösten Anfragen,
- die Integration externer Systeme,
- die Steuerung aller Datenzugriffe auf die Datenbank,
- die Weiterleitung relevanter Informationen an Logging- und Housekeeping-Dienste.

Die Log-Komponente aggregiert und persistiert systemweit auftretende Ereignisse, Fehler und Statusinformationen. Sie unterstützt damit:

- Systemmonitoring,
- Fehlerdiagnose,
- Nachvollziehbarkeit von Vorgängen.

Der Housekeeper führt periodische Hintergrundprozesse aus, darunter:

- die Bereinigung veralteter oder temporärer Einträge,

- die Ausführung zeitgesteuerter Systemaufgaben,
- die Interaktion mit dem E-Mail-Server zum Versand automatisierter Benachrichtigungen,
- die Dokumentation relevanter Abläufe über die Log-Komponente.

Der Keycloak-Server dient der Authentifizierung und Autorisierung. Anmeldevorgänge werden vom Backend an Keycloak delegiert. Die resultierenden Identitäts- und Berechtigungsinformationen werden anschließend von der Business Logic verarbeitet.

Das angebundene Inventarisierungssystem InSy übermittelt bereits inventarisierte Gegenstände über POST-Requests an das Backend. Die Business Logic validiert und verarbeitet diese Daten und integriert sie in das interne Datenmodell.

Der E-Mail-Server wird primär durch den Housekeeper angesteuert. Typische Anwendungsfälle sind:

- der Versand automatisierter Benachrichtigungen,
- die Übermittlung von Fehlermeldungen,
- die regelmäßige Statuskommunikation.

Die Datenbank dient als persistente Speicherkomponente des Systems. Sowohl die Business Logic als auch die Log-Komponente führen direkte Lese- und Schreiboperationen auf der Datenbank aus. Persistiert werden alle langfristig relevanten Informationen wie:

- Datenobjekte und Anfragen,
- Status- und Verlaufsdaten,
- systeminterne Strukturen, sofern diese nicht extern (z. B. in Keycloak) verwaltet werden.

Diese Struktur gewährleistet eine klare Trennung der Verantwortlichkeiten, hohe Wartbarkeit sowie eine robuste und nachvollziehbare Systemarchitektur.

- die Bereinigung veralteter oder temporärer Einträge,
- die Ausführung zeitgesteuerter Systemaufgaben,
- die Interaktion mit dem E-Mail-Server zum Versand automatisierter Benachrichtigungen,
- die Dokumentation relevanter Abläufe über die Log-Komponente.

8.2.3 Verhaltenssicht

Sequenzdiagramme In Abb 30 wird der Ausleihe Prozess modelliert, in welcher der Nutzer ein Gegenstand sucht, den Gegenstand seinem Warenkorb hinzufügt und diesen anschließend bestellt. Der Verleiher kann die Ausleih-Anfrage ablehnen oder zustimmen. Bei Letzterem schickt der Verleiher seine Termine an welchen er Zeit hat. Falls keine Reaktion des Verleihers erfolgt wird die Anfrage storniert.

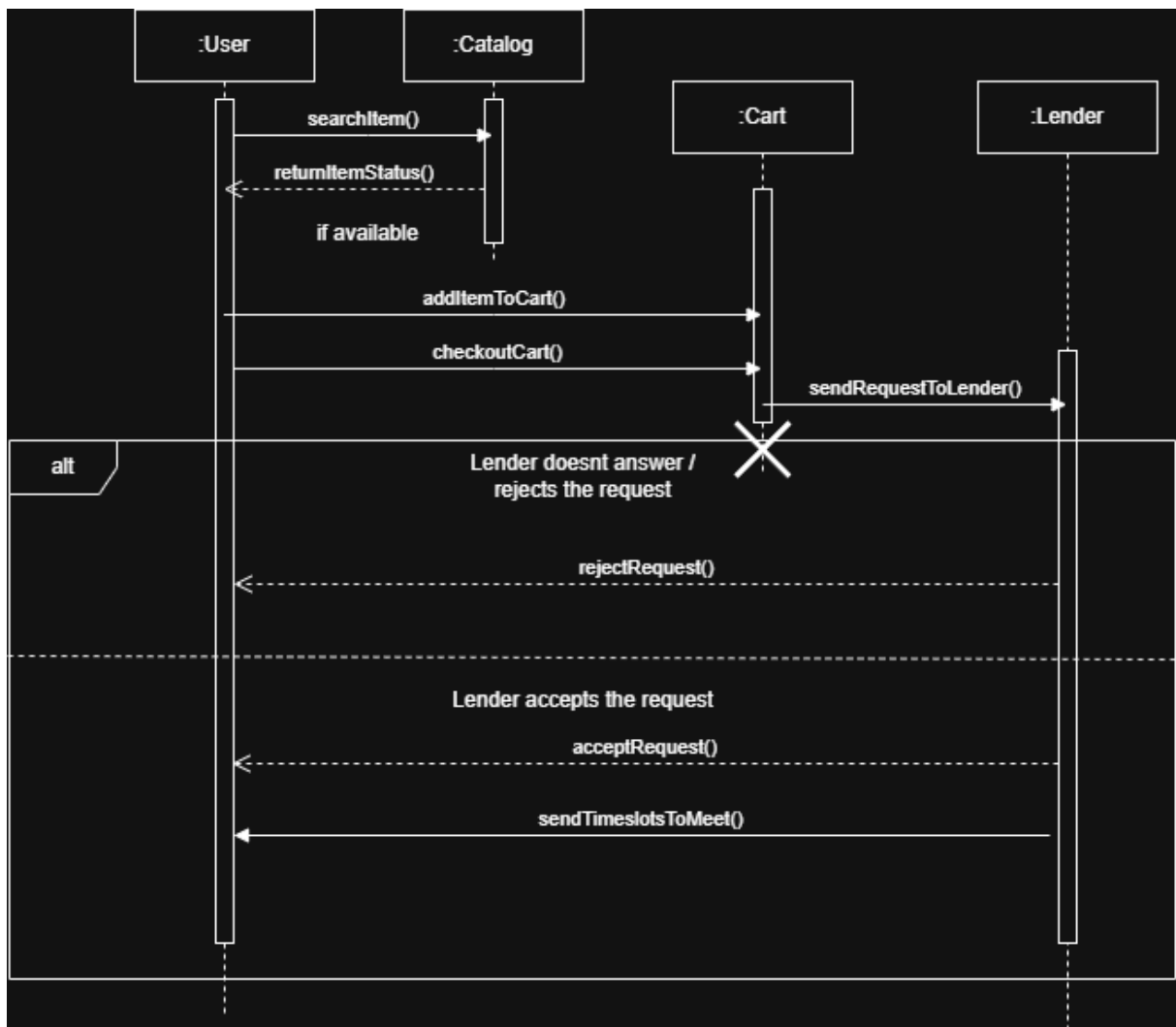


Abbildung 30: Sequenzdiagramm zum Ausleihprozess

Bei der Rückgabe bekommt der Nutzer eine Nachricht des Verwaltungssystem, dass der Gegenstand bald zurückgegeben werden müsse. Anschließend vereinbart der Ausleiher einen Termin mit dem Verleiher. Während des Termins schickt der Nutzer eine Bestätigungsanfrage an den Verleiher, dass der Verleiher die Gegenstände erhalten hat und aktualisiert das Verwaltungssystem.

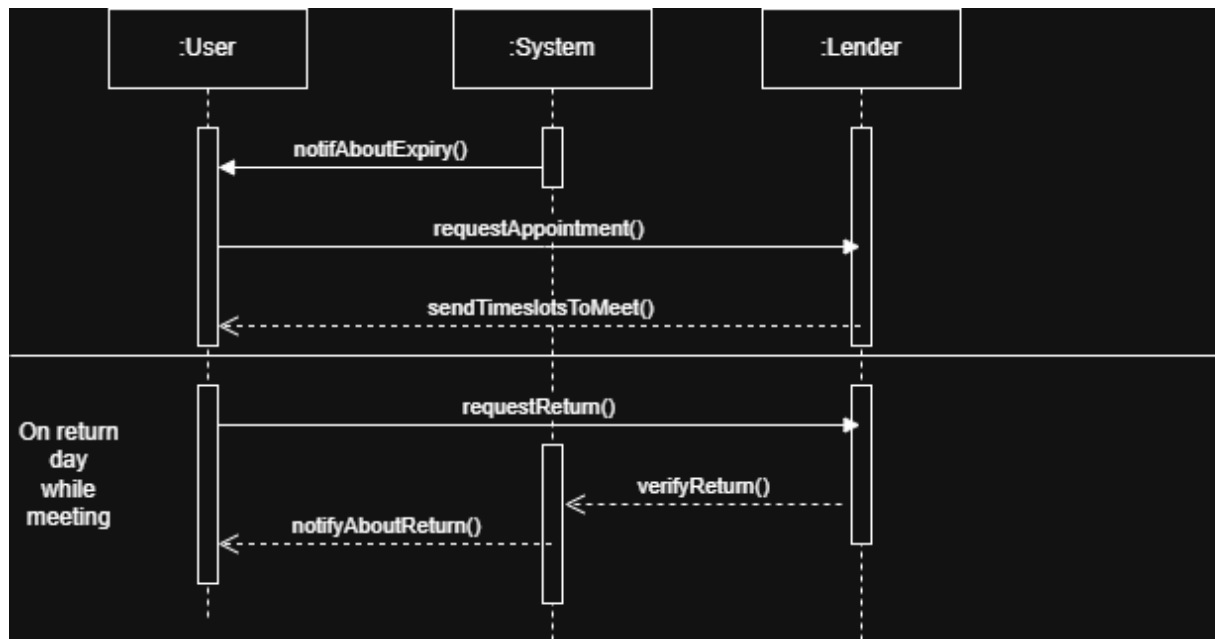


Abbildung 31: Sequenzdiagramm zum Rückgabeprozess

Der Login funktioniert über die KeyCloak API und muss dementsprechend von der Anwendung an KeyCloak übermittelt werden. Anschließend überprüft KeyCloak die Anmeldeinformationen. Wenn die Anmeldeinformationen falsch sind, wird der Zugriff verweigert. Wenn die Anmeldeinformationen korrekt sind, wird der Zugriff erlaubt und die WebApp lädt mit den Daten von KeyCloak die jeweiligen Profildaten in die Anwendung.

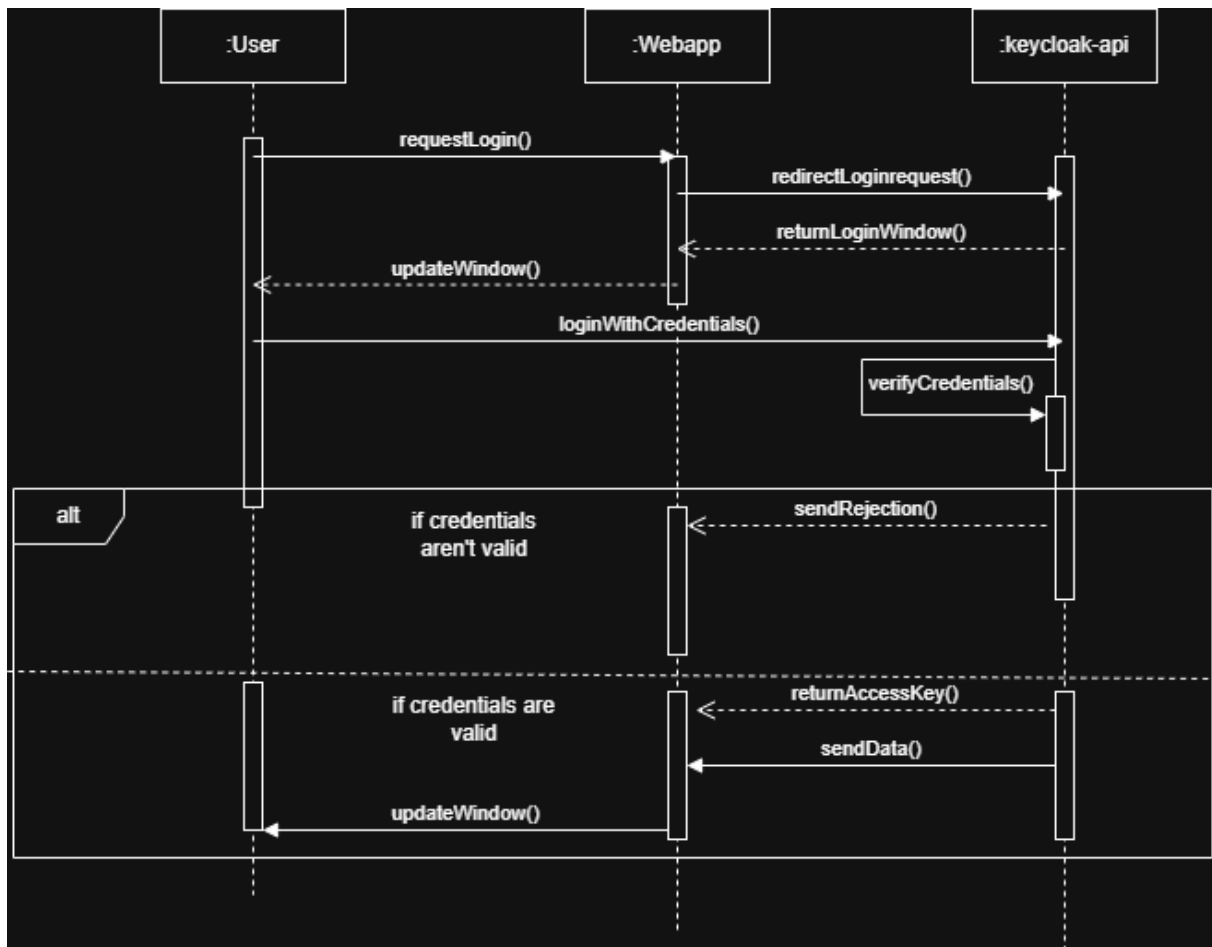


Abbildung 32: Sequenzdiagramm zum Login

9 Implementierung

9.1 Keycloak

Für die Benutzer- und Rollenverwaltung wird die Open-Source-Software Keycloak eingesetzt. Das im Projekt verwendete Realm wird bereits von BeSy und InSy benutzt. Im Realm „Hochschule Esslingen“ wurden für das Projekt drei Clients eingerichtet, die jeweils verschiedene technische Einsatzumgebungen abbilden:

- **leihsy-frontend-dev** - Client für die lokale Entwicklungsumgebung,
- **leihsy-frontend-prod** - Client für den produktiven Betrieb der Webanwendung,
- **leihsy-frontend-test** - Client für Tests und Integrationsumgebungen.

Alle Clients verwenden OpenID Connect (OIDC) als Authentifizierungsstandard, um die Identität der Benutzer sicherzustellen. Für die Verwaltung der Zugriffsrechte wird darüber hinaus OAuth 2.0 genutzt. Über die im Realm definierten Rollen kann Keycloak

Berechtigungen an die Anwendung übergeben, welche diese dann zur Zugriffskontrolle weiterverarbeitet.

Die für das Projekt eingerichteten Rollen lassen sich in drei Kategorien gliedern:

- **User**
- **Lender**
- **Admin**

Die Rolle User stellt die Standardrolle dar, die allen Personen zugewiesen wird, die sich im System anmelden können. Die Rolle Lender repräsentiert Verleiher. Sie umfasst alle Berechtigungen der Standardrolle sowie zusätzlich den Zugriff auf das erweiterte Verleiher-Dashboard. Die Rolle Admin ermöglicht die Verwaltung von Benutzern in Keycloak und von ausleihbaren Gegenständen innerhalb der Anwendung.

9.2 Frontend

9.2.1 User Stories

- Medienkatalog filtern & Zeitraum prüfen
 - Als Studierende*r möchte ich auf der Katalogseite alle Geräte sehen und den Katalog nicht nur nach Kategorien (z. B. VR-Geräte, Kameras), sondern auch nach einem konkreten Datumszeitraum, nach Campus sowie nach „nur verfügbare Geräte“ filtern können, damit ich sofort erkenne, welche Geräte für mein geplantes Projektfenster tatsächlich verfügbar sind.
- Technische Spezifikationen einsehen
 - Als Studierende*r möchte ich detaillierte technische Daten (z. B. Sensorgröße, Speicher) und das enthaltene Zubehör einsehen können, um die Eignung eines Geräts für mein Vorhaben zu bewerten.
- Mobile Nutzung
 - Als mobile Nutzer*in möchte ich über ein für Smartphones optimiertes Menü navigieren, um auch unterwegs Verfügbarkeiten bequem prüfen zu können.
- Gerätedetails und Ausleihbarkeit prüfen
 - Als Studierende*r möchte ich auf der Detailseite eines Geräts ein Abholdatum wählen können. Das System soll mir dabei visuell (z. B. durch eine rote Markierung) anzeigen, wenn das Gerät an diesem Datum nicht verfügbar ist, damit ich Zeit spare und direkt einen verfügbaren Termin wählen kann.

9.2.2 Verwendete Technologien & Frameworks

Angular (Architektur & Kern)

Das Frontend basiert auf dem Angular Framework in der modernen Ausführung. Architektonisch wird konsequent auf den Standalone-Komponenten-Ansatz gesetzt. Dies bedeutet, dass auf die klassische Modul-Struktur verzichtet wird, was die Anwendung modularer und wartbarer macht. Für das Zustandsmanagement (z. B. Anmeldestatus, Warenkorb-Zähler) kommen Signals zum Einsatz. Diese ermöglichen eine hochperformante, reaktive Aktualisierung der Benutzeroberfläche, sobald sich Daten ändern. Der Zugriffsschutz für bestimmte Bereiche (z. B. Admin-Ansicht) wird durch Routing-Guards gewährleistet, die vor dem Laden einer Seite die Berechtigungen des Nutzers prüfen.

PrimeNG & Design-System

Zur Erstellung der Benutzeroberfläche wird die Komponentenbibliothek PrimeNG verwendet, konfiguriert mit dem modernen Aura-Theme. Dies stellt sicher, dass alle interaktiven Elemente wie Kalender, Dropdown-Menüs und Eingabefelder ein einheitliches, professionelles Design aufweisen. Für das Layout und die Feinjustierung des Designs wird **Tailwind CSS** genutzt. Dies ermöglicht ein vollständig responsives Design, das sich dynamisch an verschiedene Bildschirmgrößen (Desktop, Tablet, Smartphone) anpasst, ohne dass komplexe separate Stylesheets notwendig sind.

9.2.3 Struktursicht

Die Anwendung ist als Single Page Application (SPA) konzipiert. Die Struktur gliedert sich logisch in drei Ebenen:

1. **Seiten-Komponenten:** Diese repräsentieren ganze Ansichten (z. B. „Katalogseite“, „Detailseite“). Sie sind für die Kommunikation mit der Datenlogik verantwortlich.
2. **UI-Komponenten:** Wiederverwendbare Bausteine wie Kopfzeile, Fußzeile oder Informationskarten für Campus-Standorte. Sie dienen rein der Darstellung und erhalten ihre Daten von den übergeordneten Seiten.
3. **Dienste:** Im Hintergrund agierende Einheiten, die die Datenhaltung und Geschäftslogik kapseln. Sie bilden die Schnittstelle zwischen Komponenten und den später folgenden Backend-Daten.