

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: 9. Dezember 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	12
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	17
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	22
2.5.10 Epic 10: Technische Anforderungen	22
2.5.11 Epic 11: Nice-to-Have Features	22
3 Zeitmanagement	25
3.1 Zeiterfassung	26

4 Projektmanagement	26
4.1 Methode	26
4.2 Organisation	26
4.3 Eingesetzte Tools	27
5 Entwicklungsumgebung	27
5.1 Eingesetzte IDE	27
5.2 Eingesetzte Frameworks	27
5.2.1 Weitere Backend-Bibliotheken	28
5.3 Server	28
5.4 Datenbank	28
5.4.1 Entwicklungsumgebung mit H2	28
5.4.2 Spring Profiles	29
5.4.3 Datenbankverbindung	29
5.4.4 Automatische Testdaten	30
5.5 Development-Server	30
5.6 Version Control management System	31
5.6.1 CI/CD Pipeline	31
5.7 API-Testing mit Postman	33
5.7.1 Motivation	33
5.7.2 Automatisches Token-Management	33
5.7.3 Collection Variables	33
5.7.4 Test-Requests	34
5.7.5 Vorteile	34
6 UI-Prototyp	34
6.1 Farbpalette	34
6.1.1 Version 1 (Erstkonzept)	34
6.1.2 Version 2 (Wireframe)	34
6.1.3 Finale, farbige Version	35
6.2 Erste Prototypen	35
6.2.1 Version 1 – Grundidee	35
6.2.2 Version 2 – Änderungen gegenüber Version 1	40
6.3 Visuelle Umsetzung der Funktionen	46
6.3.1 Login-Seite	46
6.3.2 Geräte-Details	46
6.3.3 Warenkorb	47
7 User Research	49
7.1 Proto-Personas	49
7.1.1 Persona A – Lena Schmid (Studierende)	49
7.1.2 Persona B – Max Schmidt (IT-Admin / Systemverantwortlicher)	49
7.1.3 Persona C – Prof. Tom Fischer (Lehrender/Verleiher)	50
7.2 Interview Auswertung	51
7.3 Auswertung (Online-Umfrage, n = 28)	52
7.3.1 Stichprobe & Nutzung	52
7.3.2 Zufriedenheit (Ist-Prozess)	52
7.3.3 Größte Pain Points (Top-Nennungen)	52
7.3.4 Adoptionsbereitschaft für ein digitales System	53

7.3.5	Top-Features (max. 3 Stimmen)	53
7.3.6	Unverzichtbare Filter	54
7.3.7	Verfügbarkeit je Campus	55
7.3.8	Storno-Fenster (Abhol-/Rückgabe-Slots)	55
8	Softwarearchitektur	56
8.1	Entity-Relationship-Diagramme	56
8.1.1	Erster Entwurf	56
8.1.2	Änderungen vom ersten Entwurf zur finalen Version	56
8.1.3	Finale Umsetzung	58
8.2	Logische Sichten	60
8.2.1	Verteilungssicht	60
8.2.2	Struktursicht	62
8.2.3	Verhaltenssicht	64
8.3	API-Dokumentation	67
9	Implementierung	68
9.1	Keycloak	68
9.2	Frontend	68
9.2.1	User Stories	68
9.2.2	Verwendete Technologien & Frameworks	69
9.2.3	Struktursicht	69
10	Features	70
10.1	Produktgruppenverwaltung	70
10.2	Gegenstandsverwaltung	73
10.3	Admin-Ansicht: Gesamte Gegenstände	75
10.4	Admin-Feature: Kategorienverwaltung	77
10.5	Admin-Feature: Verleiher-Zuordnung	78
10.6	Warenkorb	79
10.7	Buchungsübersicht	83
10.7.1	Benutzeroberfläche	83
10.7.2	Backend-Implementierung	84
10.8	Authentifizierung und Benutzerverwaltung	86
10.8.1	Überblick & Architektur	86
10.8.2	JWT-Token und Security-Konfiguration	87
10.8.3	Rollensystem	88
10.8.4	Frontend-Integration	89
10.8.5	Automatische Benutzersynchronisation	90
10.9	Bildverwaltung	92
10.9.1	Anforderungen	92
10.9.2	Architektur-Entscheidung: Filesystem vs. Datenbank	93
10.9.3	Backend-Implementierung	93
10.9.4	Frontend-Integration	95
10.9.5	Deployment-Überlegungen	96
10.9.6	Zusammenfassung	96

Abbildungsverzeichnis

1	Netzwerkübersicht Development-Server	30
2	Entwurf der CI/CD Pipeline Architektur in GitHub	32
3	Login screen	36
4	Inventarkatalog	37
5	Geräte-Details Seite	38
6	Persönlicher Bereich	39
7	Aktualisierter Inventarkatalog	40
8	Kalender in Produktdetails	41
9	Abholungszeitfenster	42
10	Warenkorb	43
11	Standort des Geräts	44
12	Aktualisierter persönlicher Bereich	45
13	Aktualisierter Login screen	46
14	Geräte-Details Seite mit aktualisierten Design	47
15	Verbesserter Warenkorb	48
16	Persona Lena Schmid	49
17	Persona Max Schmidt	50
18	Persona Tom Fischer	51
19	Anteil Studenten mit kürzlichen Ausleihen	52
20	Zufriedenheit mit aktuellem Prozess	52
21	aktuelle Pain Points	53
22	Adoptionsbereitschaft für digitales System	53
23	Wichtigste Features	54
24	Unverzichtbare Filter	54
25	Wichtigkeit Verfügbarkeit nach Campus	55
26	Länge Stornierbarkeit	55
27	Erster Entwurf Entity-Relationship-Diagramm	56
28	Entity-Relationship-Diagramm	60
29	Diagramm zur Verteilungssicht	61
30	Komponentendiagramm	62
31	Sequenzdiagramm zum Ausleihprozess	65
32	Sequenzdiagramm zum Rückgabeprozess	66
33	Sequenzdiagramm zum Login	67
34	Tabellarische Übersicht aller Produkte	71
35	Formular zu Erstellung und Bearbeitung von Produkten	72
36	Bearbeitungsansicht der Produktdetails	73
37	Liste der Items im Erstellungsmenü	74
38	Formular zur Erstellung und Bearbeitung von Gegenständen	75
39	Adminansicht aller Gegenstände	76
40	Kategorienverwaltung im Admin-Dashboard	77
41	Dialog zur Verleiher-Zuordnung in der Geräteverwaltung	78
42	Produkt-Detailseite	80
43	Zum Warenkorb Hinzugefügt	81
44	Warenkorb-Seite	82
45	Warenkorb-Checkout	82
46	Buchungsverlauf eines Users	83

1 Erklärung der Aufgabe

1.1 Ist-Zustand

Aktuell erfolgt die Geräteausleihe an der Hochschule in Papierform. Um ein Gerät auszuleihen, müssen sich die Studierenden einen analogen Leihchein 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 Ressourcenverschwendungen.
- 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 gebrauchstauglichenen 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
- **Überfälligkeit-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 abgeholt 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.

2.3.0.1 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.

2.3.0.2 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.

2.3.0.3 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.

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

2.3.0.5 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.

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

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

2.3.0.8 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 ↔ 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
 - Warenkorbsicht

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 abgeholt 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
 - Überfälligkeit-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 abgeholt 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 Verleiher 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, Auto vervollständigung 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 Na-

vigation 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.

5.2.1 Weitere Backend-Bibliotheken

Neben Spring Boot werden im Backend weitere Bibliotheken eingesetzt, um die Entwicklung zu vereinfachen und den Quellcode übersichtlicher zu halten. Die Bibliothek Lombok reduziert wiederkehrenden Code, indem sie über einfache Annotationen automatisch Methoden wie Getter und Setter erzeugt. Dadurch bleibt der Code schlanker und besser lesbar, ohne dass auf Funktionalität verzichtet wird.

Die Bibliothek MapStruct wird verwendet, um Daten zwischen den JPA-Entitäten und den Data Transfer Objects (DTOs) zu übertragen. Anstatt die Felder manuell zu kopieren, werden die benötigten Mapper-Klassen aus Interfaces generiert. Das verringert die Fehleranfälligkeit, erleichtert Änderungen an den Datenklassen und unterstützt eine klare Trennung zwischen Persistenzschicht und API-Schicht.

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.

5.4.1 Entwicklungsumgebung mit H2

In der frühen Projektphase wurde für die lokale Entwicklung die In-Memory-Datenbank H2 eingesetzt. H2 ist eine leichtgewichtige, in Java geschriebene Datenbank, die keine separate Installation erfordert und bei jedem Anwendungsstart eine frische Datenbankinstanz erstellt. Dies ermöglichte uns, unabhängig voneinander zu arbeiten, ohne auf eine zentrale Datenbankinfrastruktur angewiesen zu sein.

Die H2-Konsole war während der Entwicklung unter `http://localhost:8080/h2-console` erreichbar und bot eine webbasierte Oberfläche zur direkten Inspektion der Datenbank. Dies war besonders hilfreich beim Debugging von JPA-Queries und der Überprüfung von Entity-Beziehungen.

Nachdem die PostgreSQL-Datenbank auf dem Entwicklungsserver eingerichtet und über VPN erreichbar war, wurde die Entwicklung vollständig auf PostgreSQL umgestellt. H2 wird seitdem nicht mehr aktiv verwendet, die Konfiguration bleibt jedoch als Fallback erhalten.

5.4.2 Spring Profiles

Um flexibel zwischen verschiedenen Datenbankumgebungen wechseln zu können, wurden Spring Profiles implementiert. Die Konfiguration ist auf drei Dateien aufgeteilt:

Datei	Profil	Beschreibung
application.properties	–	Basis-Konfiguration, legt aktives Profil fest
application-dev.properties	dev	H2 In-Memory-Datenbank für lokale Entwicklung
application-prod.properties	prod	PostgreSQL-Verbindung zum Entwicklungsserver

Tabelle 1: Übersicht der Konfigurationsdateien

Das aktive Profil wird in der Hauptkonfiguration festgelegt:

```
# application.properties
spring.profiles.active=prod
```

Alternativ kann das Profil beim Anwendungsstart als Parameter übergeben oder über Umgebungsvariablen gesetzt werden. Die profilspezifischen Konfigurationsdateien enthalten jeweils die Datenbankverbindungsparameter, Hibernate-Einstellungen und Logging-Konfiguration für die entsprechende Umgebung.

5.4.3 Datenbankverbindung

Die Verbindung zur PostgreSQL-Datenbank auf dem Entwicklungsserver erfolgt über ein VPN (WireGuard). Die Verbindungsparameter sind in `application-prod.properties` hinterlegt:

```
spring.datasource.url=jdbc:postgresql://<HOST>:5432/<DATABASE>
spring.datasource.username=<USERNAME>
spring.datasource.password=<PASSWORD>
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.hibernate.ddl-auto=update
```

Die Einstellung `ddl-auto=update` sorgt dafür, dass Hibernate das Datenbankschema automatisch an Änderungen in den Entity-Klassen anpasst, ohne bestehende Daten zu löschen. Für einen späteren Produktivbetrieb sollte diese Einstellung auf `validate` geändert werden, um unbeabsichtigte Schemaänderungen zu verhindern.

5.4.4 Automatische Testdaten

Für die Entwicklung mit H2 wurde ein `DataInitializer` implementiert, der beim Anwendungsstart automatisch Testdaten in die Datenbank einfügt. Die Klasse ist mit der Annotation `@Profile("dev")` versehen, sodass die Testdaten ausschließlich im Entwicklungsprofil geladen werden:

```
@Component
@Profile("dev")
public class DataInitializer implements CommandLineRunner {
    @Override
    public void run(String... args) {
        // Kategorien, Standorte, Produkte und Items anlegen
    }
}
```

Bei Verwendung des Produktionsprofils wird der `DataInitializer` nicht ausgeführt, sodass die PostgreSQL-Datenbank nicht mit Testdaten überschrieben wird. Dies stellt sicher, dass Entwicklungs- und Produktionsdaten strikt getrennt bleiben.

5.5 Development-Server

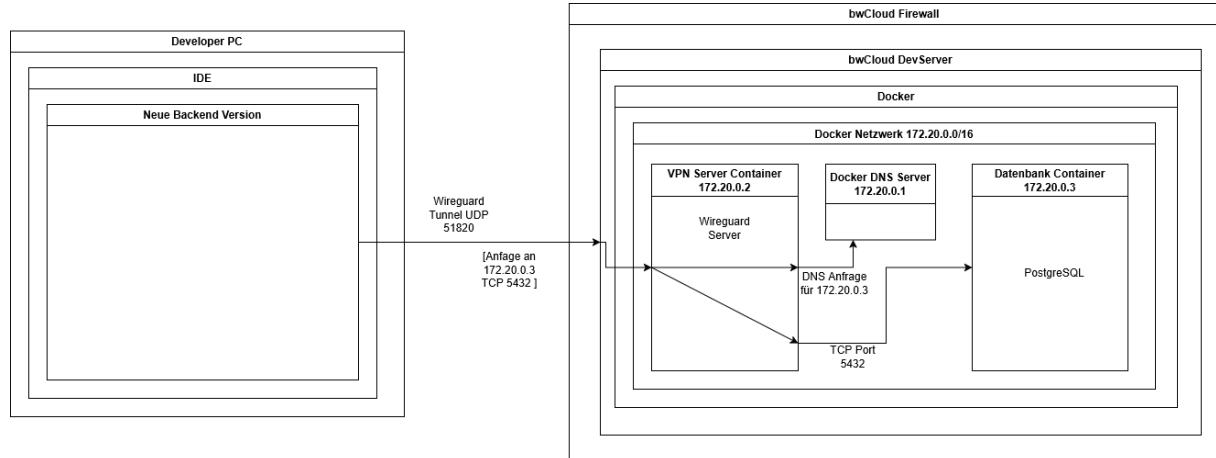


Abbildung 1: Netzwerkübersicht Development-Server

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.

Mit der Weiterentwicklung der Applikation wurde es notwendig auch Mock-Bilder Teamintern zu synchronisieren. Daher wurde im gleichen Docker Netzwerk ein Webdav-Container eingerichtet. Dieser stellt über HTTP ein Netzlaufwerk bereit das in Windows File Explorer als Netzlaufwerk hinzugefügt werden kann. HTTP ist in diesem Fall ausreichend da der Webdav-Server nur mit aktivem VPN erreichbar ist. Im Windows File Explorer kann dann eine Verknüpfung auf den Mockbilder-Ordner auf dem Webdav Netzwerk erstellt werden. Diese Verknüpfung kann in den lokalen Projektordner hinzugefügt werden. Somit hat die lokal laufende Applikation Zugriff auf die Mockbilder die auf dem Webdav-Laufwerk des Dev-Servers liegen.

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 Integrationstest durchgeführt und anschließend in einer Staging-Umgebung abgelegt, um manuell in Docker-Container eingefügt zu werden.

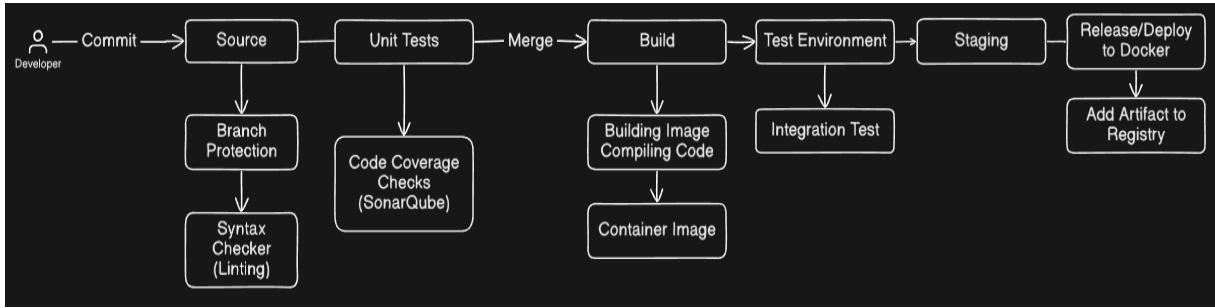


Abbildung 2: Entwurf der CI/CD Pipeline Architektur in GitHub

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.

Das automatische Deployment als Docker-Container auf dem Testserver jeder neuen main Branch ist als Github Workflow realisiert. Dazu wurde in `/.github/workflows` eine yml-Datei angelegt die folgende Schritte durchläuft:

1. Der Prozess wird erst gestartet wenn die Code Coverage und SonarQube Tests erfolgreich durchlaufen wurden.
2. Im Repository sind SSH-Key, IP Adresse und User des Servers bei bwCloud als Secret hinterlegt. Diese werden verwendet um sich per SSH mit dem Server zu verbinden.
3. Dort wird in den Deployment-Ordner gewechselt und die aktuelle main Branch aus dem Github Repository gecloned.
4. Im Repository befindet sich ein Dockerfile, mit dem auf dem Server lokal ein Docker-Image gebaut wird. Dieses wird mit der ebenfalls im Repository enthaltenen docker-compose.yml als Container auf dem Server gestartet.
5. Dann wird durch eine Probeanfrage bestätigt dass der Container ordnungsgemäß in Betrieb ist.
6. falls diese Probeanfrage scheitert wird automatisch auf dem Server ein Rollback zur letzten funktionierenden Version der main Branch durchgeführt. Diese wird ebenfalls wieder gebaut und als Container gestartet so dass immer eine funktionierende Version der Anwendung auf dem Testserver zu Testzwecken bereit steht.

5.7 API-Testing mit Postman

5.7.1 Motivation

Für das systematische Testen der Backend-APIs wurde eine Postman-Collection erstellt. Die Herausforderung bestand darin, die Keycloak-Authentifizierung zu automatisieren, sodass nicht vor jedem Request manuell ein Token geholt werden muss.

5.7.2 Automatisches Token-Management

Ein Collection-weites Pre-request Script wurde implementiert, das folgende Aufgaben übernimmt:

- Prüfung ob ein gültiges Access Token in den Collection Variables vorhanden ist
- Validierung der Token-Gültigkeit anhand des Ablaufdatums (exp Claim)
- Automatisches Holen eines neuen Tokens von Keycloak bei Ablauf oder fehlendem Token
- Speicherung des Tokens und Ablaufdatums in Collection Variables für nachfolgende Requests

Das Script nutzt den OAuth2 Password Grant Flow (Resource Owner Password Credentials) mit dem Keycloak Token-Endpoint unter `/realms/insy/protocol/openid-connect/token`.

5.7.3 Collection Variables

Folgende Variablen wurden konfiguriert:

- `base_url`: Backend-URL (`http://localhost:8080/api`)
- `keycloak_url`: Keycloak-Server (`https://auth.insy.hs-esslingen.com`)
- `client_id`: Keycloak Client (temporär: insy-backend)
- `username`: Hochschul-Account des Test-Users
- `password`: Passwort des Test-Users
- `access_token`: Wird automatisch gefüllt vom Pre-request Script
- `token_expiration`: Unix-Timestamp des Token-Ablaufs

Alle Requests nutzen die Variable `{{access_token}}` im Authorization-Header mit Bearer-Schema.

5.7.4 Test-Requests

Die Collection enthält Requests für alle Booking-Endpoints:

- GET /bookings/users/me - Eigene Buchungen abrufen
- GET /bookings/lenders/me/pending - Offene Anfragen als Verleiher
- POST /bookings - Neue Buchung erstellen
- PUT /bookings/{id}/confirm - Buchung bestätigen mit Terminvorschlägen
- PUT /bookings/{id}/select-pickup - Abholtermin auswählen
- PUT /bookings/{id}/pickup - Ausgabe dokumentieren
- PUT /bookings/{id}/return - Rückgabe dokumentieren
- DELETE /bookings/{id} - Buchung stornieren

5.7.5 Vorteile

Dieses Setup ermöglicht effizientes Testing ohne manuelle Token-Verwaltung. Entwickler können die gesamte Booking-API mit wenigen Klicks durchspielen und den kompletten Workflow (Anfrage → Bestätigung → Ausgabe → Rückgabe) validieren.

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: #0A0AOA

- 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)

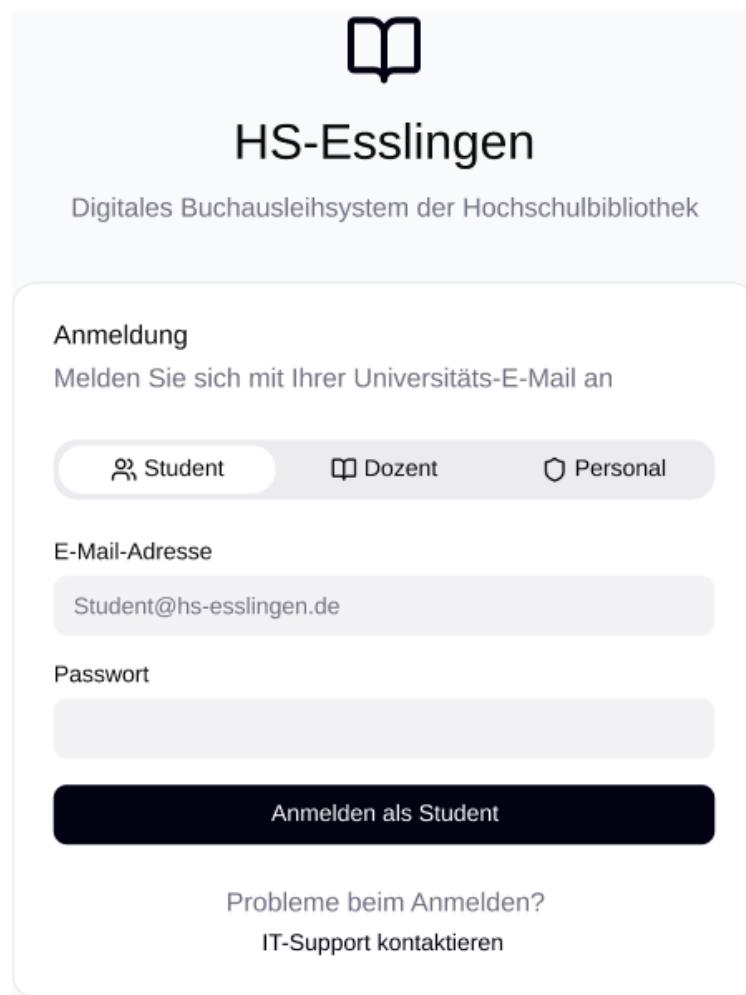
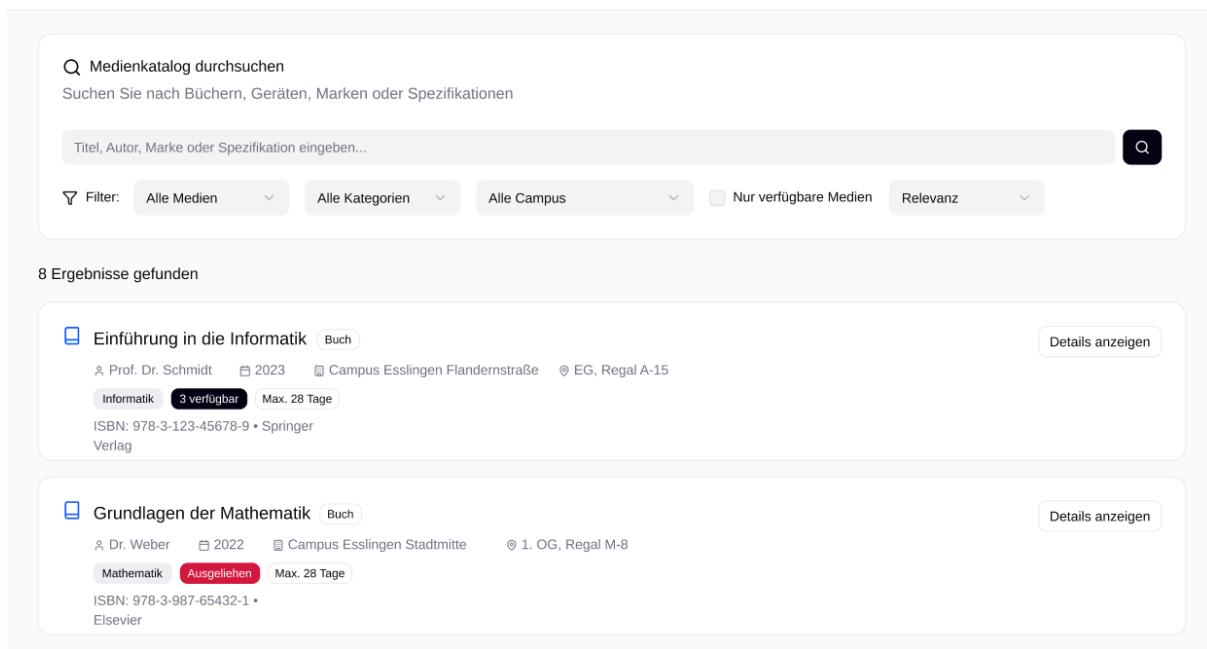


Abbildung 3: Login screen

- **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)



The screenshot shows the HS-Esslingen Medienkatalog search interface. At the top, there is a search bar with the placeholder "Titel, Autor, Marke oder Spezifikation eingeben..." and a magnifying glass icon. Below the search bar are several filter options: "Filter:" dropdown set to "Alle Medien", "Alle Kategorien" dropdown, "Alle Campus" dropdown, "Nur verfügbare Medien" checkbox checked, and "Relevanz" dropdown. A message "8 Ergebnisse gefunden" is displayed above the results. Two search results are shown in cards:

- Einführung in die Informatik** (Buch)
A Prof. Dr. Schmidt | 2023 | Campus Esslingen Flandernstraße | EG, Regal A-15
Informatik | 3 verfügbar | Max. 28 Tage
ISBN: 978-3-123-45678-9 • Springer Verlag
- Grundlagen der Mathematik** (Buch)
A Dr. Weber | 2022 | Campus Esslingen Stadtmitte | 1. OG, Regal M-8
Mathematik | Ausgeliehen | Max. 28 Tage
ISBN: 978-3-987-65432-1 • Elsevier

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)

The screenshot displays the HS-Esslingen library website's product detail page for the Meta Quest 3 VR headset. At the top, the HS-Esslingen logo and navigation links for 'Katalog' and 'Mein Bereich' are visible. The email address 'asesit00@hs-esslingen.de' is shown under 'Student'.

Product Details:

- Name:** Meta Quest 3
- Type:** VR-Brille
- Model:** Meta Quest 3 128GB
- Status:** Verfügbar (Available)
- Description:** Hochmoderne VR-Brille für immersive virtuelle Erfahrungen. Ideal für Forschungsprojekte, Entwicklung und Lehrzwecke in den Bereichen Game Design, Medientechnik und Informatik.
- Technical Specifications:** 128GB Storage, 2064x2208 per eye, 90/120Hz
- Included Accessories:** Ladekabel, Reinigungstuch, Anleitung, Controller
- Keywords:** Virtual Reality, Immersive Technologie, Game Development, Medientechnik
- Number:** 1215
- Model:** Quest 3 128GB

Availability:

- Gesamt: 4 Exemplare
- Verfügbar: 2 Exemplare
- Ausgeliehen: 2 Exemplare

Borrowing Information:

- Campus: Campus Esslingen Flandernstraße
- Standort: Medienausgabe, Schrank VR.1
- Jetzt ausleihen** (Borrow now) button

Categorization:

- VR/AR

Opening Hours:

- Wochentags:** Mo-Fr: 8:00-20:00
- Samstag:** Sa: 10:00-16:00
- Sonntag:** So: geschlossen

Borrowing Conditions:

- Ausleihzeit:** 7 Tage
- Verlängerungen:** Nach Verfügbarkeit
- Vormerkungen:** Bis zu 5 aktive Vormerkungen
- Hinweis:** Geräte müssen vollständig und funktionsfähig zurückgegeben werden

Contact Information:

- Öffnungszeiten** (Opening hours) for three campuses: Flandernstraße, Stadtmitte, and Göppingen.
- Kontakt** (Contact) information for the library staff (bibliothek@hs-esslingen.de, +49 711 397-49) and support (bibl-support@hs-esslingen.de).
- Standorte** (Locations) for the three campuses: Flandernstraße, Stadtmitte, and Göppingen.
- Hochschulbibliothek** (University Library) information: Über 180.000 Medien, Über 50.000 E-Books, 400+ Abonnements.

At the bottom of the page, a footer note reads: © 2025 Hochschule Esslingen - University of Applied Sciences. Alle Rechte vorbehalten. Datenschutz • Impressum • Barrierefreiheit.

Abbildung 5: Geräte-Details Seite

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

HS-Esslingen
Hochschulbibliothek

Katalog Mein Bereich asesit00@hs-esslingen.de Student

Willkommen zurück!
Angemeldet als: asesit00@hs-esslingen.de (Student)

3 Aktuelle Ausleihen 3 Bald fällig € 3.50€ Offene Gebühren

⚠ Sie haben überfällige Medien mit ausstehenden Gebühren. Bitte geben Sie diese schnellstmöglich zurück.

Meine Ausleihen
Übersicht Ihrer aktuell ausgeliehenen Medien

Einführung in die Informatik Buch Prof. Dr. Schmidt
Ausgeleihen: 15.9.2025 Fällig: 15.11.2025 Bald fällig 325 Tage überfällig 0 Verlängerungen: 1/3

Meta Quest 3 VR-Brille Meta Quest 3 128GB Ausgeleihen: 15.9.2025 Fällig: 15.11.2025 Bald fällig 359 Tage überfällig 0 Verlängerungen: 0/1

ThinkPad X1 Carbon Laptop Lenovo X1 Carbon Gen 11 Ausgeleihen: 15.9.2025 Fällig: 15.11.2025 Überfällig 363 Tage überfällig Gebühr: 3.50€ Verlängerungen: 1/2

Kürzlich zurückgegebene Medien Ihre letzten Rückgaben

Datenbanken verstehen Buch Prof. Dr. Fischer Ausgeleihen: 1.8.2025 Zurückgegeben: 15.9.2025 Pünktlich zurückgegeben

Canon EOS R6 Mark II Kamera Canon EOS R6 Mark II Ausgeleihen: 20.9.2025 Zurückgegeben: 27.9.2025 Pünktlich zurückgegeben

iPad Pro 12.9" Tablet Apple iPad Pro 12.9" M2 Ausgeleihen: 15.7.2025 Zurückgegeben: 20.8.2025 3 Tage

Kontoinformationen

Ausleihlimit 10 Medien Aktuelle Ausleihen 3 von 10
Ausleihzeit 7-30 Tage* Max. Verlängerungen Je nach Medientyp
* Bücher: 30/60/90 Tage • Geräte (VR, Kameras): 7/14/21 Tage • Laptops/Tablets: 14/21/28 Tage

Offene Gebühren 3.50€
Gebühren bezahlen

Öffnungszeiten Kontakt Standorte Hochschulbibliothek

Campus Esslingen Flandernstraße Mo-Fr: 8:00-20:00 Sa: 10:00-16:00 So: geschlossen
Campus Esslingen Stadtmitte Mo-Fr: 9:00-18:00 Sa: 10:00-14:00 So: geschlossen
Göppingen Mo-Fr: 8:30-19:00 Sa: 9:00-15:00 So: geschlossen

Bibliothekspersonal bibliothek@hs-esslingen.de +49 711 397-49
Support & Probleme bib-support@hs-esslingen.de Bei Problemen mit Ausleihen, Rückgaben oder Ihrem Benutzerkonto

Campus Flandernstraße Flandernstraße 101 73732 Esslingen am Neckar Campus Stadtmitte Kanalstraße 33 73728 Esslingen am Neckar Campus Göppingen Robert-Bosch-Straße 1 73037 Göppingen

Die Bibliothek der Hochschule Esslingen bietet umfassende Literatur- und Informationsdienste für Studium, Lehre und Forschung.
Medienbestand: Über 180.000 Medien
E-Books: Über 50.000 E-Books
Zeitschriften: 400+ Abonnements

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

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)

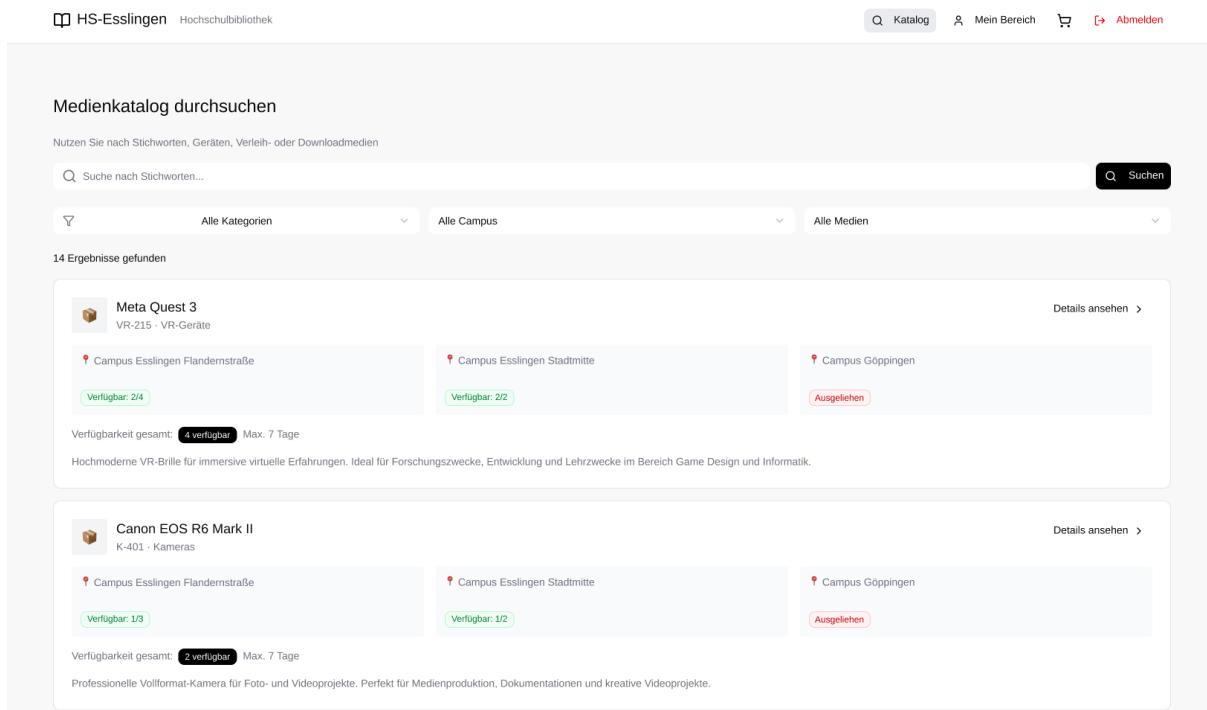


Abbildung 7: Aktualisierter Inventarkatalog

- **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)

[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](#)

© 2025 Hochschule Esslingen - University of Applied Sciences. Alle Rechte vorbehalten.
[Datenschutz](#) · [Impressum](#) · [Barrierefreiheit](#)

Abbildung 8: Kalender in Produktdetails

41

[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

Zeitfenster:

Zeitfenster wählen
08:00 - 12:00 Uhr
01:00 - 17:00 Uhr

Kategorisierung

VR-Geräte

Support

geraeieverleihs@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.

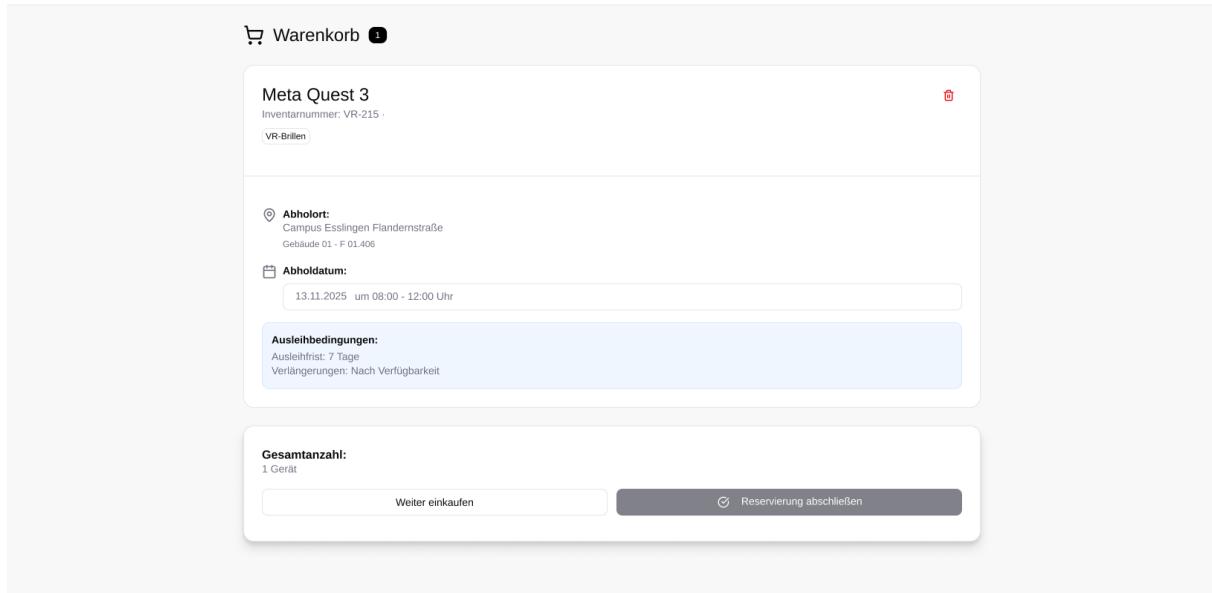
VR-Geräte: 12+ Headsets
Kameras: 15+ Profi-Kameras
Lichtsets & Equipment: 15+ Sets

© 2025 Hochschule Esslingen - University of Applied Sciences. Alle Rechte vorbehalten.
Datenschutz · Impressum · Barrierefreiheit

Abbildung 9: Abholungszeitfenster

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

42

**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 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)

[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

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:

Zeitfenster wählen

08:00 - 12:00 Uhr
01:00 - 17:00 Uhr

Kategorisierung

VR-Geräte

Support

geraeieverleih@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)

HS-Esslingen Hochschulbibliothek

Katalog Mein Bereich Abmelden

Willkommen zurück

3 Aktive Ausleihen

1 Reservierungen

3.50€ Offene Gebühren

Sie haben zurzeit überfällige Medien in Vorlage gehabt bzw. Rücknahme nicht erfolgt. Bitte geben Sie diese umgehend zurück.

Meine Ausleihen

Übersicht Ihrer aktuellen Ausleihen und Rückgabedaten

Sony A7 IV
Inventar-Nr.: K-512
Campus Esslingen Flandernstraße
Ausgeliehen am: 10.09.2025
Rückgabe bis: 17.09.2025
Verbleibende Zeit 0 Tage
0/1 Verlängerungen genutzt Verlängern Details

Meta Quest 2
Inventar-Nr.: VR-225
Campus Esslingen Flandernstraße
Ausgeliehen am: 10.10.2025
Rückgabe bis: 17.10.2025
Verbleibende Zeit 0 Tage
0/1 Verlängerungen genutzt Verlängern Details

Lichtset Aputure 300d II
Inventar-Nr.: L-450
Campus Esslingen Flandernstraße
Ausgeliehen am: 18.10.2025
Rückgabe bis: 25.10.2025
Verbleibende Zeit 5 Tage
0/1 Verlängerungen genutzt Verlängern Details

Schnelle Verlängerungen
Verlängern Sie mehrere Medien gleichzeitig

Sony A7 IV
K-512

Meta Quest 2
VR-225

Lichtset Aputure 300d II
L-450

Ausgewählte verlängern

Zukünftige Abholungen

Ihre reservierten Geräte zur Abholung

Canon EOS R6 Mark II
K-401 - Campus Esslingen Flandernstraße
Abholung am 17.10.2024 - 10:00-14:00 Uhr
Bereit zur Abholung
Termin ändern

Warenkorb

Meta Quest 3
VR-215
Campus Esslingen Flandernstraße

Zur Ausleihe

Kontoubersicht

Aktive Ausleihen: 3
Kamera: 1
VR-Brillen: 1
Lichtset: 1
Offene Gebühren: 3.50€
Gebühren bezahlen

Kontoinformation

Matrikelnummer: 0768274
Gültig bis: 31.02.2026
Maximale Ausleihen: 20 Medien

Support

geraeiterverleih@hs-esslingen.de
+49 711 397-3456

Für technische Probleme oder Fragen zur Ausleihe

Ausbagstellen

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 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)

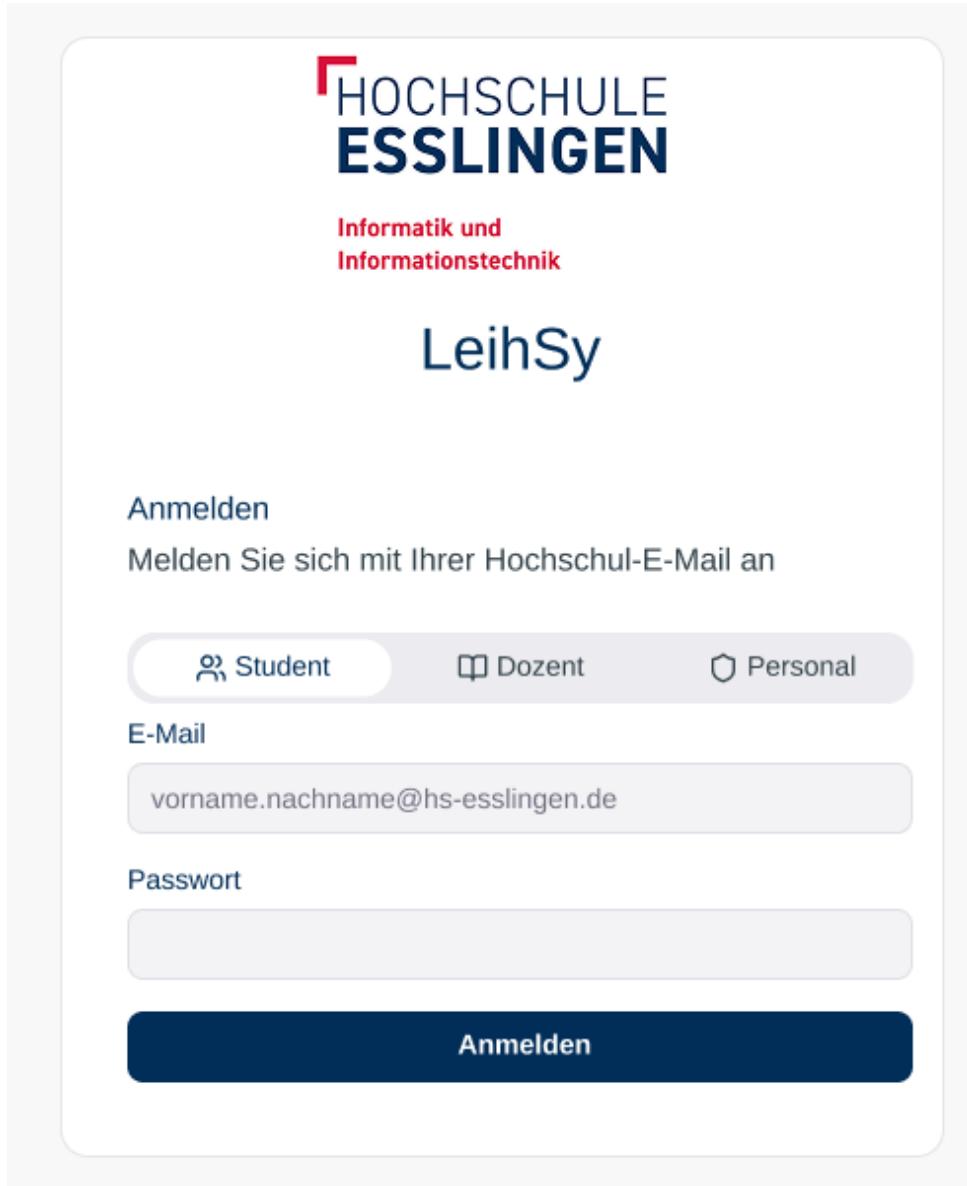


Abbildung 13: Aktualisierter Login screen

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)

The screenshot shows a product detail page for a 'Meta Quest 3 VR-Geräte'. Key details include:

- Description:** Hochmoderne VR-Brille für immersive virtuelle Erfahrungen. Ideal für Forschungszwecke, Entwicklung und Lehrzwecke im Bereich Game Design und Informatik.
- Technical Specifications:**
 - Speicher: 128GB
 - Sensor: 256x256 pixel eye, 90/120Hz
 - Zubehör: Ladekabel, Reinigungstuch, Anleitung, Controller
- Included Accessories:** Virtual Reality, Immersive Technologie, Game Development, Medientechnik
- Keywords:** Virtual Reality, Immersive Technologie, Game Development, Medientechnik
- Borrowing Conditions:**
 - 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
- Availability:** 4 Exemplare verfügbar, 4 Exemplare ausgeliehen
- Campus:** Campus Esslingen Flandernstraße
- Reservation:** Standort: Gebäude 01 - F 01.406, 2/4 verfügbar
- Categorization:** VR-Geräte

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)



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: Verbesserter 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.



ALTE 26
BERUF Medieninformatik Studentin

USER PERSONA
Lena Schmid

LENA ist eine Medieninformatik Studentin im Bachelor und arbeitet nebenbei als Werkstudentin. Sie organisiert vieles unterwegs am Smartphone, hasst Papierkram und unnötige Rückfragen. Wichtig sind ihr klare Verfügbarkeiten, schnelle Bestätigungen/Erinnerungen und transparente Statusanzeigen. Wenn niemand reagiert, erwartet sie automatisches Storno statt langem Hinterherlaufen.

ZIELE	BEDÜRFNISSE
<ul style="list-style-type: none">• Schnell sehen, was verfügbar ist & einfach anfragen• Zeitfenster planen & bei Konflikten klare Hinweise• Transparenz über Status & Historie	<ul style="list-style-type: none">• Vorgänge schnell und unkompliziert online erledigen, statt Zettel auszufüllen oder manuell bei der dem Professor_in einzureichen.• Klare Übersicht über verfügbare Zeiten und Ressourcen, um ohne Rückfragen planen zu können.
PAIN POINTS	
<ul style="list-style-type: none">• Unklare Verfügbarkeit/Konflikte• Lange Wartezeiten/keine Antwort → Auto-Storno hilft• Fehlende Erinnerungen → Überfälligkeit	<ul style="list-style-type: none">• Automatische Bestätigungen und Erinnerungen, damit keine Anfragen vergessen oder überfällig werden.• Transparente Nachverfolgung des eigenen Status und früherer Anfragen, ohne extra nachfragen zu müssen.

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.

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.

ALTE	55
BERUF	IT-Administratorin / Systemverantwortliche für das Hochschul-Leihsystem

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.

USER PERSONA

Tom Fischer

Herr Prof. Fischer ... ist ein zielorientierter Professor an einer Universität in München, der sich leidenschaftlich für Technologie, Innovation und Nachhaltigkeit einsetzt. Sein Ziel ist es, den Hochschulalltag umweltfreundlicher und effizienter zu gestalten – durch den Einsatz digitaler Lehrmethoden, energieeffizienter Prozesse und nachhaltiger Campuslösungen.

ALTE	55
BERUF	Professor

ZIELE

- Schnell prüfen & entscheiden: Anfragen annehmen/ablehnen/Termine abstimmen
- Eigene Gegenstände im Blick behalten
- Vor Ort: Ausgabe/Rückgabe sauber dokumentieren
- Nachhaltigkeit fördern: Ressourcenverbrauch senken

BEDÜRFNISSE

- Dashboard offene Anfragen + Filter/Sortierung
- Alternativtermine vorschlagen
- E-Mail-Trigger für Abholung, Rückgabe
- Zentrale Plattform: z. B. Integration in die Hochschul-App

PAIN POINTS

- Zu viele Anfragen gleichzeitig → schwer zu priorisieren oder zu überblicken
- Unklare Verfügbarkeiten → es ist nicht ersichtlich, wann ein Gerät wirklich frei ist
- Kein Überblick über bestehende Reservierungen → führt zu Doppelbuchungen oder Leerlaufzeiten

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.

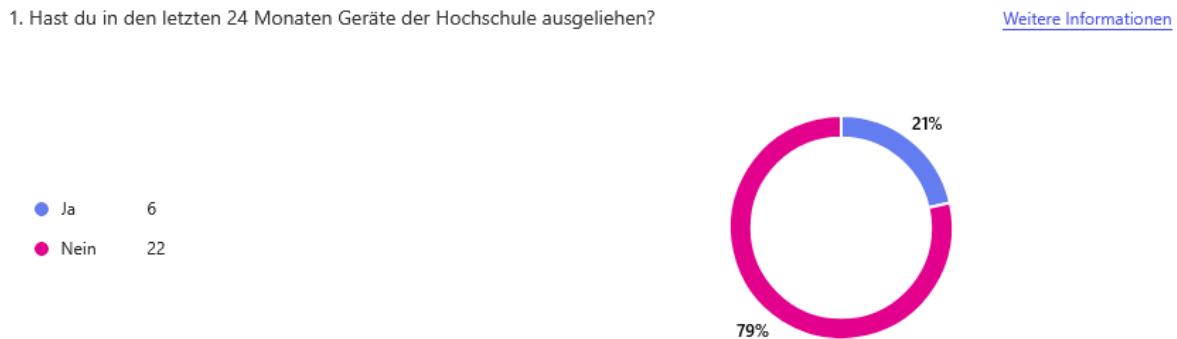


Abbildung 19: Anteil Studenten mit kürzlichen Ausleihen

7.3.2 Zufriedenheit (Ist-Prozess)

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

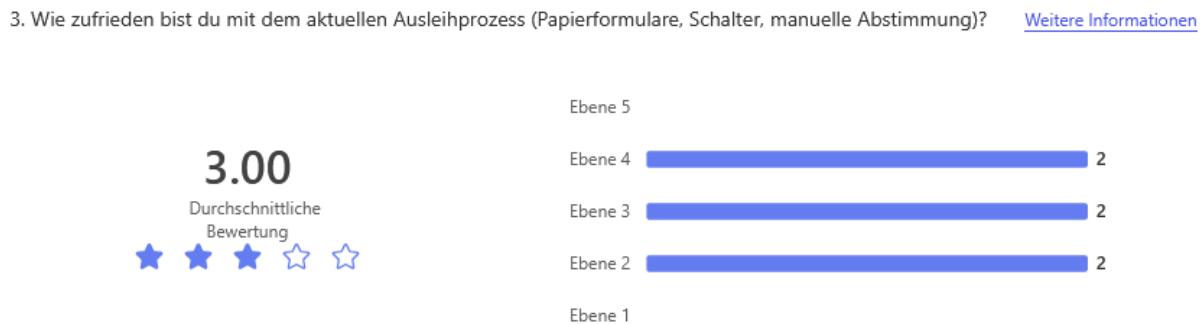


Abbildung 20: Zufriedenheit mit aktuellem Prozess

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

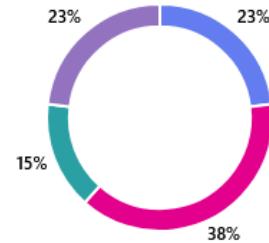


Abbildung 21: aktuelle Pain Points

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

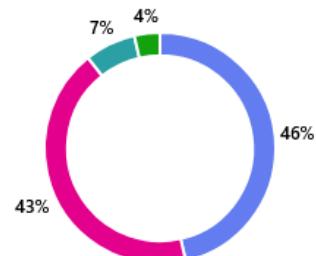


Abbildung 22: Adoptionsbereitschaft für digitales System

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](#)

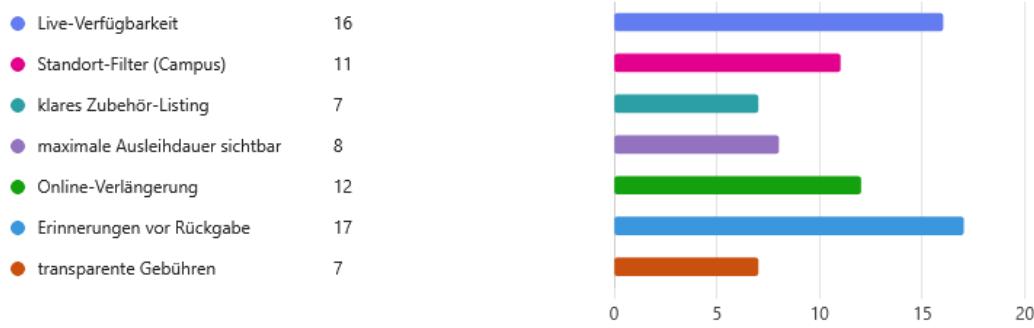


Abbildung 23: Wichtigste Features

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](#)

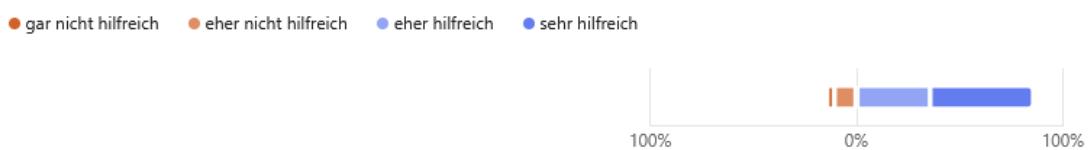


Abbildung 25: Wichtigkeit Verfügbarkeit nach Campus

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

8.1.1 Erster Entwurf

Das in Abbildung 27 dargestellte Entity Relationship Diagramm zeigt unseren ersten Entwurf der Datenbank Strukturierung

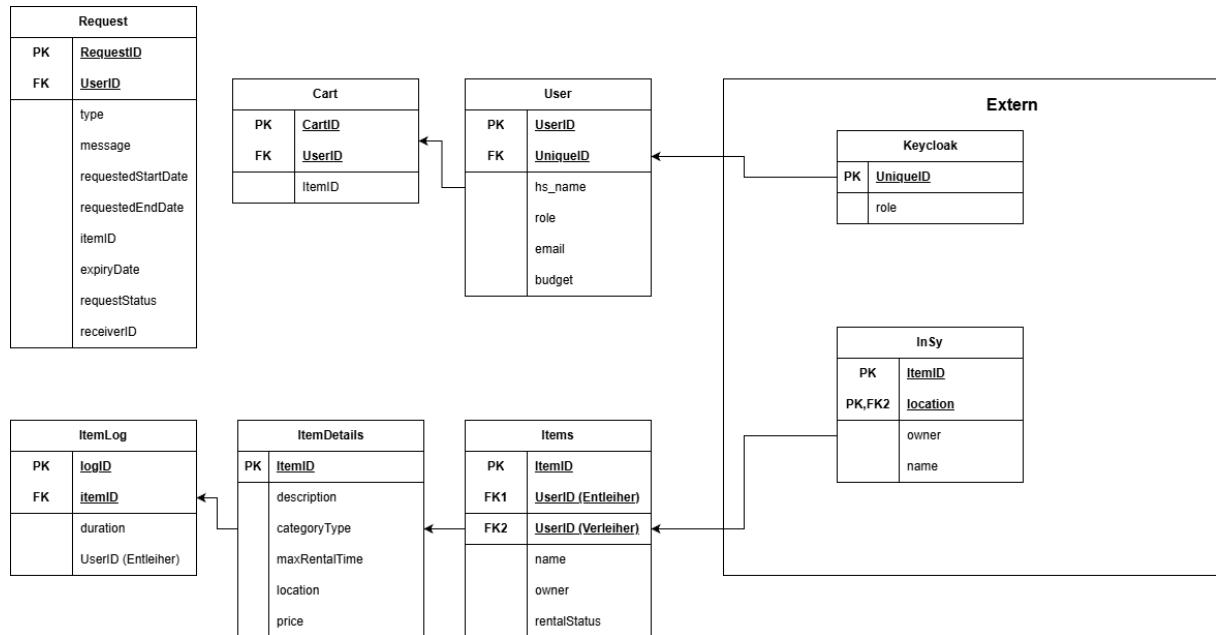


Abbildung 27: Erster Entwurf Entity-Relationship-Diagramm

8.1.2 Änderungen vom ersten Entwurf zur finalen Version

Im Vergleich zum in Abbildung 27 dargestellten ersten Entwurf wurde das Entity-Relationship-Modell in mehreren Bereichen strukturell überarbeitet, normalisiert und an die funktionalen Anforderungen des Systems angepasst. Die wichtigsten Änderungen lassen sich wie folgt zusammenfassen:

8.1.2.1 1. Umbenennung zentraler Entitäten

- Die Entität **Request** wurde durch die Entität **Bookings** ersetzt. Dabei wurden Attribute wie Start-, End- und Rückgabedatum, Status und Nachrichten in ein einheitliches Buchungsmodell überführt.
- Die frühere Entität **User** wurde zu **Users** umbenannt und um Historisierungsattribute (`created_at`, `updated_at`, `deleted_at`) erweitert.

- Die Beziehung zur externen Authentifizierung (`Keycloak`) wurde neu modelliert und verwendet nun einen expliziten Primärschlüssel anstelle der im Entwurf genutzten `UniqueID`-Referenz.

8.1.2.2 2. Neuorganisation der Produkt- und Itemstruktur

- Die im Entwurf bestehenden Entitäten `Items`, `ItemDetails`, `ItemLog` und `InSy` wurden vollständig restrukturiert.
- Die Entität `Products` ersetzt die zuvor getrennten Entitäten `Items` und `ItemDetails` als übergeordnete Produktbeschreibung.
- Physische Einheiten eines Produkts wurden in eine neue, klar abgegrenzte Entität `Items` ausgelagert, die nur produkt- und inventarspezifische Attribute enthält.
- Attribute wie `categoryType`, `maxRentalTime`, `price` und `location` wurden in die Produkt- bzw. Standortentitäten überführt und dort normalisiert.

8.1.2.3 3. Einführung neuer Entitäten

- **Categories:** Eine neue Entität zur Klassifizierung von Produkten, die im Entwurf lediglich als Attribut existierte.
- **Locations:** Eine eigene Entität zur räumlichen Zuordnung der Produkte; ersetzt das frühere einfache Attribut `location`.
- **Sets:** Neue n:m-Assoziation zur Gruppierung mehrerer Produkte zu vordefinierten Sets.

8.1.2.4 4. Beziehungsanpassungen und Normalisierung

- Die im Entwurf teilweise unklaren oder nicht normalisierten Beziehungen wurden zu eindeutigen 1:n- oder n:m-Beziehungen restrukturiert.
- Die frühere direkte Bindung zwischen `Items` und `User` (Verleiher/Entleiher) wurde vollständig entfernt; Buchungen regeln nun alle Nutzerinteraktionen.
- Die externe Entität `InSy` wurde funktional reduziert und stellt nun lediglich Stammdaten für Items bereit.

8.1.2.5 5. Vereinheitlichung der Zeit- und Statusfelder

- Einführung systemweiter Felder für Historisierung: `created_at`, `updated_at`, `deleted_at`.
- Vereinheitlichung der Statusparameter: `requestStatus` aus dem Entwurf wurde durch das Attribut `status` in `Bookings` ersetzt.

8.1.2.6 6. Entfernte oder ersetzte Entitäten

- Cart wurde vollständig entfernt und vom Buchungskonzept ersetzt.
- ItemLog wurde nicht übernommen; Logik wird künftig über Buchungen oder Auditing gelöst.
- ItemDetails ging in Products und Categories auf.

Diese Anpassungen führten zu einer klaren Trennung zwischen Produktdefinition, physischen Einheiten, Benutzerinteraktionen und externen Systemen. Dadurch ist die finale Version konsistenter, normalisierter und deutlich besser wartbar als der ursprüngliche Entwurf.

8.1.3 Finale Umsetzung

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.

8.1.3.1 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.

8.1.3.2 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.

8.1.3.3 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).

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

8.1.3.5 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.

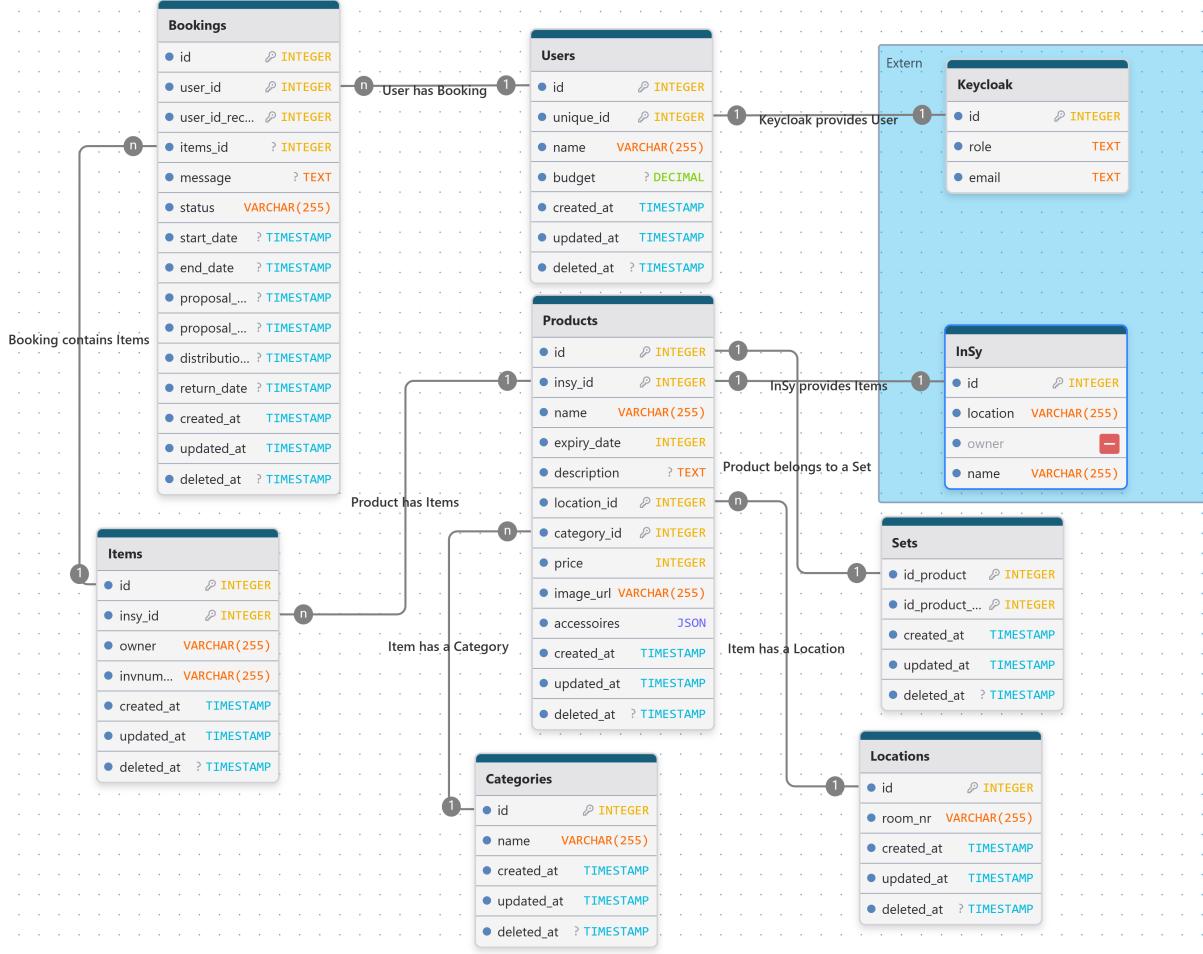


Abbildung 28: Entity-Relationship-Diagramm

8.2 Logische Sichten

8.2.1 Verteilungssicht

Die Anwendung folgt einem klassischen Client-Server-Architekturmmodell. Der Client ist der Webbrower, 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.

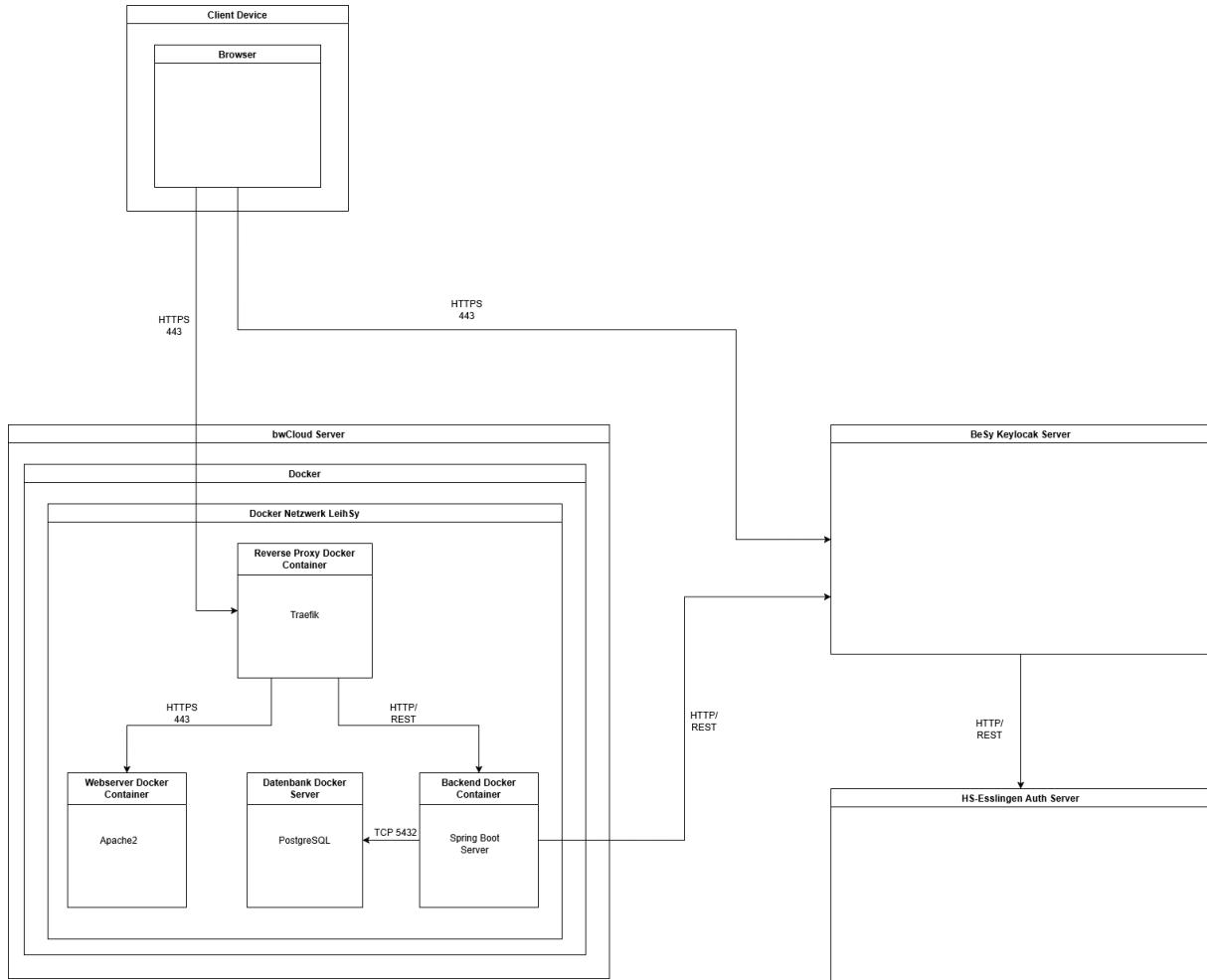


Abbildung 29: Diagramm zur Verteilungssicht

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. 30 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.

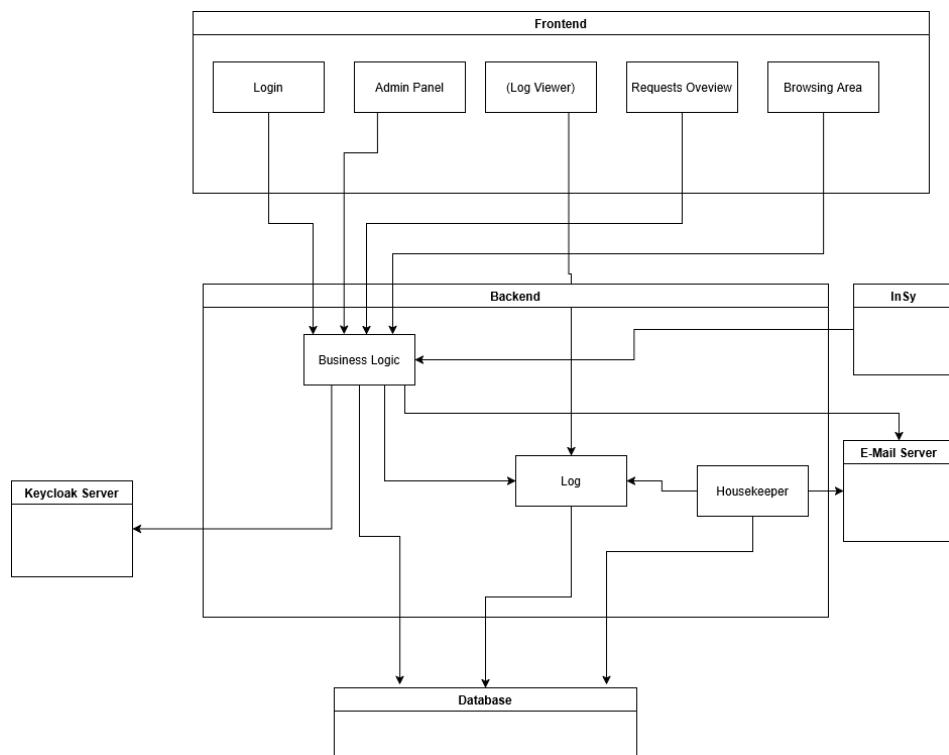


Abbildung 30: Komponentendiagramm

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

8.2.3.1 Sequenzdiagramme In Abb 31 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 Verleiher erfolgt wird die Anfrage storniert.

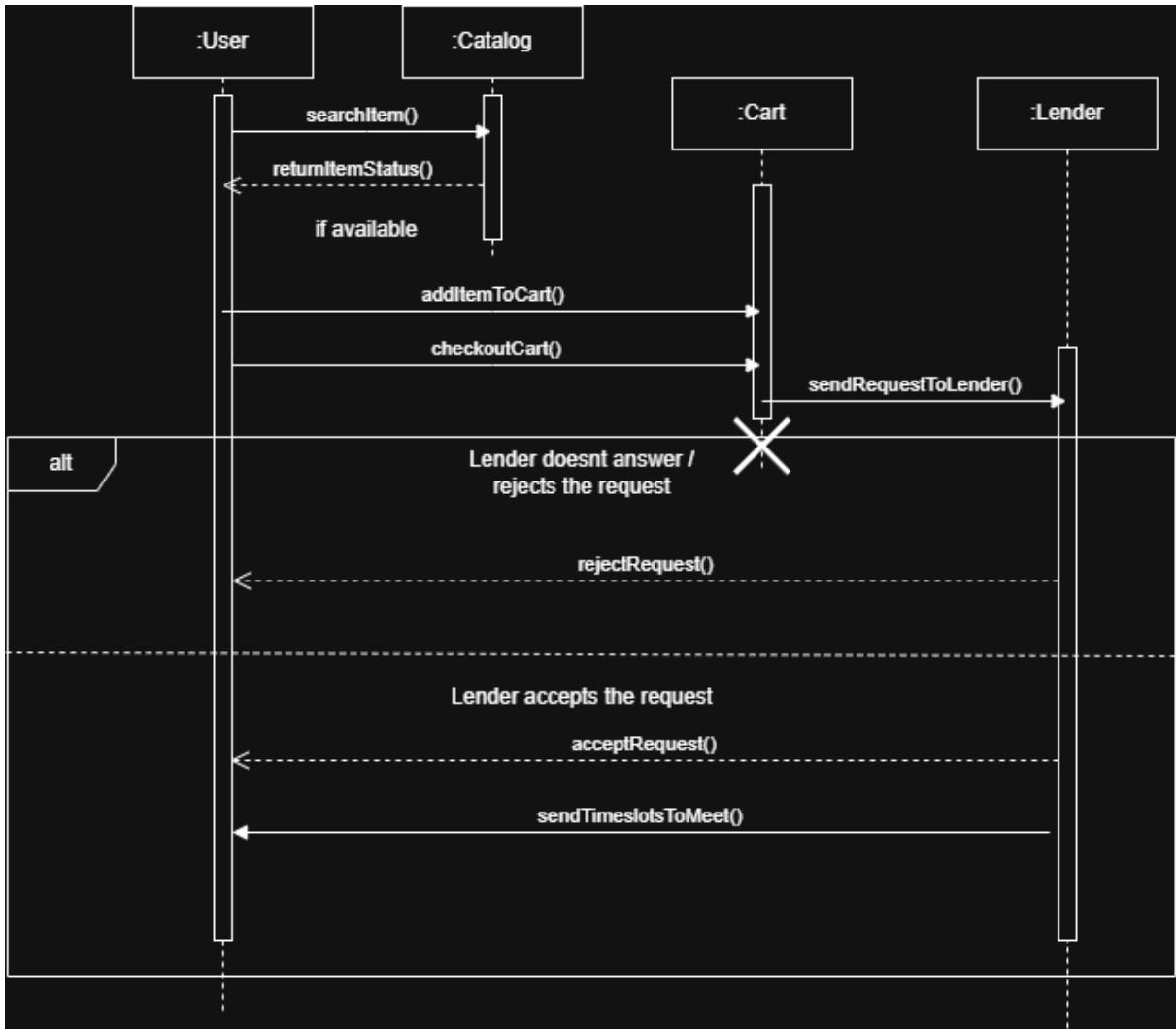


Abbildung 31: Sequenzdiagramm zum Ausleihprozess

Bei der Rückgabe bekommt der Nutzer eine Nachricht des Verwaltungssystems, 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, das der Verleiher die Gegenstände erhalten hat und aktualisiert das Verwaltungssystem.

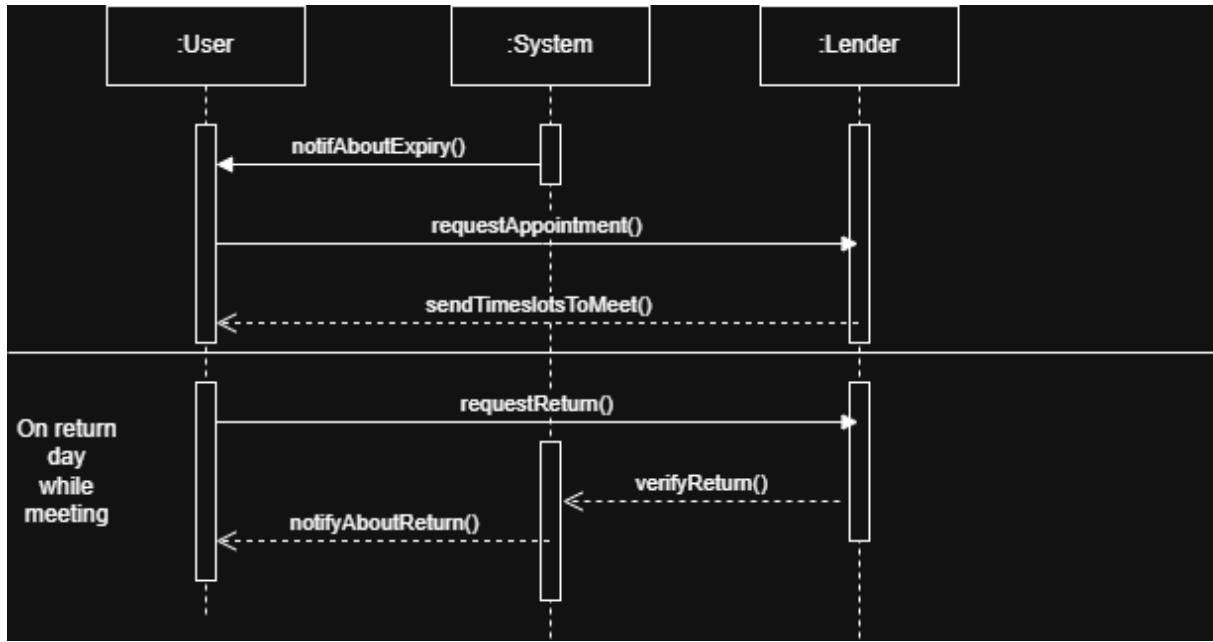


Abbildung 32: Sequenzdiagramm zum Rückgabeprozess

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.

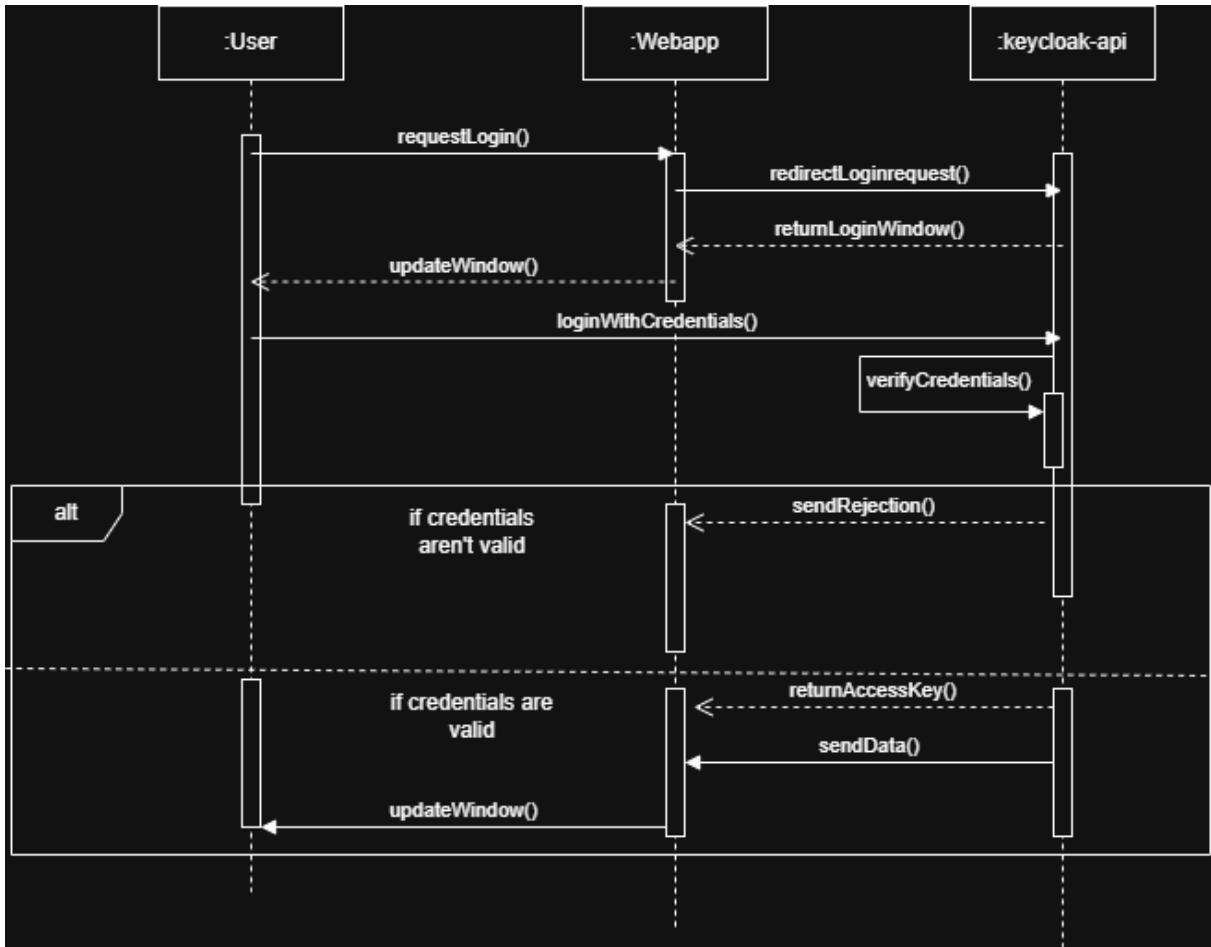


Abbildung 33: Sequenzdiagramm zum Login

8.3 API-Dokumentation

Die API-Dokumentation wird halbautomatisch durch Swagger erstellt. Sie wird von Swagger unter der URL `http://<Spring Boot-IP>:8080/swagger-ui/index.html` bereitgestellt. Dazu wurde Swagger für Spring Boot als Dependency in Maven hinzugefügt. Swagger analysiert die REST-Controller von Spring Boot automatisch und erstellt eine Übersicht über die API-Endpunkte, gibt anhand der Return-Values in den Controllern Beispieldaten aus und zeigt an, welche Parameter in einer REST-Anfrage enthalten sein können oder müssen. Die Beschreibungen zu den Parametern und der Bedeutung von Antwortcodes können händisch als Annotations in den REST-Controllern hinzugefügt werden. Diese werden dann ebenfalls automatisch von Swagger erkannt und in der API-Dokumentation dargestellt. Dies hat unter anderem den Vorteil dass die API-Dokumentation auch im Sourcecode enthalten ist und ein Nachschlagen in der Dokumentation gegebenenfalls übersprungen werden kann.

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

9.2.2.1 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.

9.2.2.2 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.

10 Features

10.1 Produktgruppenverwaltung

Die Ansicht in Abbildung 34 zeigt eine tabellarische Übersicht aller im System vorhandenen Produkte. Sie dient der zentralen Verwaltung des Produktbestands und ermöglicht den schnellen Zugriff auf Bearbeitungs- und Löschfunktionen.

Im oberen Bereich befindet sich ein Suchfeld, über das Produkte nach Name, Beschreibung oder Kategorie gefiltert werden können.

Die Tabelle enthält folgende Spalten:

- **ID:** Eindeutige Produkt-ID.
- **Name:** Bezeichnung des Produkts inklusive Kurzbeschreibung.
- **Kategorie:** Zugeordnete Produktkategorie.
- **Standort:** Aktueller Lager- oder Einsatzort.
- **Preis/Tag:** Tagesmietpreis in Euro.
- **Aktionen:** Schaltflächen zum Bearbeiten oder Löschen des Produkts.

Die Listenansicht ist scrollbar ausgeführt und ermöglicht dadurch auch bei großen Produktbeständen eine übersichtliche Darstellung.

Vorhandene Produkte						
<input type="text"/> Suche nach Name, Beschreibung, Kategorie...						
ID	Name	Kategorie	Standort	Preis/Tag	Aktionen	
1	Meta Quest Pro High-End Mixed-Reality-Headset mit Eye-Tracking un...	VR-Equipment	VR-Labor F01.403	8.00 €		
2	Valve Index Premium VR-System mit 144Hz Display und Finger-Tra...	VR-Equipment	VR-Labor F01.403	12.00 €		
3	Pico 4 Standalone VR-Headset mit hoher Auflösung und Panc...	VR-Equipment	VR-Labor F01.403	6.00 €		
4	Canon EOS R6 Mark II Professionelle Vollformatkamera mit 24 MP und 6K V...	Foto-Equipment	F01.402 (KEIM)	18.00 €		
5	Nikon Z6 III Hybrid-Kamera mit 24 MP und beeindruckender Low-Li...	Foto-Equipment	Bibliothek Flandernstraße	17.00 €		
6	DJI Ronin SC Kompakter 3-Achsen Gimbal für spiegellose Kameras ...	Foto-Equipment	F01.402 (KEIM)	10.00 €		
7	Shure SM7B Studio-Mikrofon, ideal für Podcasts, Streaming und...	Audio-Equipment	F01.402 (KEIM)	4.00 €		

Abbildung 34: Tabellarische Übersicht aller Produkte

Abbildung 35 zeigt das Interface zum Anlegen neuer Produktgruppen. Diese Funktion ermöglicht das zentrale Erstellen neuer Produktgruppen, denen anschließend einzelne Gegenstände zugeordnet werden.

Das Formular enthält folgende Felder:

- **Name:** Bezeichnung des Produkts (z. B. Kameramodell).
- **Kategorie:** Zuordnung zu einer übergeordneten Kategorie.
- **Beschreibung:** Ausführliche Beschreibung und technische Details.
- **Produktbild:** Upload eines Bildes im Format JPG, PNG oder WebP.
- **Zubehör:** Liste optionaler Komponenten wie Kabel, Netzteile oder Controller.

Produktgruppen verwalten

Erstelle, bearbeite oder lösche Produktgruppen

Neues Produkt anlegen

Allgemeine Informationen Produkt Details

Name *
z.B. Canon EOS R5

Kategorie *
Kategorie wählen

Beschreibung
Detaillierte Beschreibung des Produkts...

Produktbild
Datei auswählen Keine ausgewählt
JPG, PNG oder WebP, max. 5MB

Zubehör
["Controller", "Kabel", "Netzteil"]

Abbildung 35: Formular zu Erstellung und Bearbeitung von Produkten

Abbildung 36 zeigt die Ansicht zur Bearbeitung produktspezifischer Detailinformationen. Diese Funktion dient der Pflege ausleihrelevanter Parameter einzelner Produkte und ergänzt die Produktgruppenverwaltung um administrative Metadaten.

Das Formular enthält folgende Felder:

- **Max. Ausleihdauer (Tage):** Maximale Anzahl an Tagen, für die ein Produkt ausgeliehen werden darf.
- **Preis / Tagessatz (€):** Täglicher Mietpreis des Produkts.
- **Raumnummer:** Angabe des physischen Lager- oder Einsatzortes des Produkts.
- **Location ID:** Automatisch ermittelte Standortkennung, die aus der angegebenen Raumnummer generiert wird.

Pflichtfelder sind entsprechend gekennzeichnet. Über die Schaltflächen „Aktualisieren“ und „Abbrechen“ können Änderungen gespeichert oder verworfen werden.

Produkt bearbeiten

Allgemeine Informationen Produkt Details

Max. Ausleihdauer (Tage) *	10	Preis / Tagessatz (€) *	8,00 €
Raumnummer *	VR-Labor F01.403	Location ID	3
Wird automatisch aus Raumnummer ermittelt			
Aktualisieren		Abbrechen	

Vorhandene Produkte

Suche nach Name, Beschreibung, Kategorie...

ID	Name	Kategorie	Standort	Preis/Tag	Aktionen
1	Meta Quest Pro High-End Mixed-Reality-Headset mit Eye-Tracking un...	VR-Equipment	VR-Labor F01.403	8,00 €	
2	Valve Index Premium VR-System mit 144Hz Display und Finger-Tra...	VR-Equipment	VR-Labor F01.403	12,00 €	
3	Pico 4 Standalone VR-Headset mit hoher Auflösung und Panc...	VR-Equipment	VR-Labor F01.403	6,00 €	
4	Canon EOS R6 Mark II	Foto-Equipment	F01.402 (KFIM)	18,00 €	

Abbildung 36: Bearbeitungsansicht der Produktdetails

10.2 Gegenstandsverwaltung

Die Ansicht in Abb.37 stellt alle im System erfassten Produkte in einer übersichtlichen, kategorisierten Listenstruktur dar. Ziel der Seite ist es, dem Nutzer einen schnellen Überblick über alle verfügbaren Produkte sowie deren zugehörige Einzelgegenstände (Instanzen) zu ermöglichen.

Im oberen Bereich befindet sich ein Suchfeld, über das nach Produktnamen, Kategorie oder Standort gefiltert werden kann.

Jedes Produkt wird in einem eigenen Abschnitt dargestellt. Der Kopfbereich eines Produktes enthält folgende Informationen:

- den Produktnamen als Titel,
- die zugehörige Kategorie,
- den Standort des Produktes,
- die Anzahl verfügbarer bzw. insgesamt vorhandener Gegenstände.

Über ein Aufklapp-Element kann der Nutzer die Liste der Einzelgegenstände des Produktes einsehen. Für jeden Gegenstand werden folgende Attribute angezeigt:

- **ID** des Gegenstandes,
- **Inventarnummer**,
- **Besitzer** bzw. Verantwortlicher,
- **Verleiher** inklusive E-Mail-Adresse und System-ID,
- **Status** (z. B. „Verfügbar“),
- **Aktionen** zum Bearbeiten oder Löschen.

Im rechten oberen Bereich eines Produktabschnitts befindet sich ein Button zum Hinzufügen neuer Gegenstände. Die Darstellung ermöglicht eine schnelle Einschätzung der Verfügbarkeit jedes Produktes sowie eine effiziente Verwaltung einzelner Einheiten.

The screenshot shows a user interface for managing products and their items. At the top, there's a search bar labeled "Suche nach Produktnamen, Kategorie...". Below it, a section titled "Produkte und ihre Gegenstände" lists items under categories like VR-Equipment. Each item entry includes a title, category, location, availability status (e.g., "3 / 3 verfügbar"), and a blue "+" button. A detailed table below shows items for the "Meta Quest Pro" category, listing ID, Inventarnummer, Besitzer, Verleiher, Status (all green "Verfügbar"), and actions (edit and delete icons). Other sections for "Valve Index", "Pico 4", "Canon EOS R6 Mark II", and "Nikon Z6 III" show similar structures with varying availability counts (1/1, 5/5, 1/1, 2/2).

ID	Inventarnummer	Besitzer	Verleiher	Status	Aktionen
2	VR-PRO-002	Christian Haas	admin@hs-esslingen.de (ID: 5)	Verfügbar	
31	VR-PRO-003	Christian Haas	admin@hs-esslingen.de (ID: 5)	Verfügbar	
1	VR-PRO-001	Christian Haas	admin@hs-esslingen.de (ID: 5)	Verfügbar	

Abbildung 37: Liste der Items im Erstellungsmenü

In Abbildung 38 ist das Administrationsformular zur Bearbeitung einzelner physischer Gegenstände dargestellt. Dieses Menü ermöglicht das Anlegen, Ändern oder Löschen eines bestimmten Inventarobjekts.

Das Formular umfasst:

- **Inventarnummer-Präfix:** Grundkennzeichnung, an die automatisch eine Nummer angehängt wird.

- **Besitzer User ID:** Optionale Zuordnung zu einem Besitzer.
- **Verleiher User ID / Benutzername:** Angabe des administrativen Verantwortlichen.
- **Produkt:** Zugehörige Produktgruppe.
- **Verfügbarkeitsstatus:** Markierung, ob das Gerät aktuell ausgeliehen werden kann.

Einzelne Gegenstände verwalten

Erstelle, bearbeite oder lösche einzelne physische Gegenstände zu jedem Produkt

Gegenstand bearbeiten

Inventarnummer Präfix *

Basisname für die Inventarnummer (Zahl wird automatisch hinzugefügt)

Besitzer User ID

Verleiher User ID *

Verleiher Benutzername *

Gefunden: admin@hs-esslingen.de

Produkt *

Verfügbar

Aktualisieren Abbrechen

Abbildung 38: Formular zur Erstellung und Bearbeitung von Gegenständen

10.3 Admin-Ansicht: Gesamte Gegenstände

Die Ansicht in Abb. 39 zeigt eine vollständige Übersicht über sämtliche Gegenstände im System. Sie dient Administratoren dazu, alle Produktinstanzen unabhängig vom übergeordneten Produkt zentral zu verwalten.

Im Header werden drei Kennzahlen hervorgehoben dargestellt:

- **Gesamtanzahl der Gegenstände,**
- **Anzahl der aktuell verfügbaren Gegenstände,**
- **Anzahl der ausgeliehenen Gegenstände.**

Darunter befindet sich ein globales Suchfeld, mit dem nach Produktnamen, Kategorien oder Inventarnummern gesucht werden kann.

Die Gegenstände sind nach ihren zugehörigen Produkten gruppiert. Jede Produktgruppe zeigt:

- den Produktnamen,
- die Kategorie,
- den Standort,
- die Produkt-ID,
- die Anzahl verfügbarer Gegenstände.

Innerhalb jeder Produktgruppe werden die einzelnen Gegenstände in Tabellenform dargestellt. Die Tabelle enthält folgende Spalten:

- **Inventarnummer**,
- **Besitzer**,
- **Status** (mit visueller Hervorhebung),
- **Letzte Aktualisierung** mit Datum und Uhrzeit.

Die admin-spezifische Darstellung ermöglicht eine schnelle Kontrolle der Systembestände, erleichtert das Auffinden bestimmter Gegenstände und unterstützt die effiziente Verwaltung großer Inventare.

Alle Gegenstände (Admin-Ansicht)

Übersicht über alle Gegenstände im System

Angemeldet als: admin@hs-esslingen.de

The screenshot shows the 'Alle Gegenstände (Admin-Ansicht)' page. At the top, there's a summary table with three rows: 'Gesamt Gegenstände' (34), 'Verfügbar' (34 with a green checkmark icon), and 'Ausgeliehen' (0 with a red circle icon). Below this is a search bar. The main content area displays two detailed item tables.

Meta Quest Pro		Verfügbar 3 / 3	
↳ VR-Equipment	↳ VR-Labor F01.403		
ID: 1			
Inventarnummer	Besitzer	Status	Zuletzt aktualisiert
VR-PRO-002	Christian Haas	Verfügbar	23.11.2025 11:18
VR-PRO-003	Christian Haas	Verfügbar	25.11.2025 18:04
VR-PRO-001	Christian Haas	Verfügbar	23.11.2025 11:18

Valve Index		Verfügbar 1 / 1	
↳ VR-Equipment	↳ VR-Labor F01.403		
ID: 2			
Inventarnummer	Besitzer	Status	Zuletzt aktualisiert
VR-INDEX-001	KEIM	Verfügbar	23.11.2025 11:18

Abbildung 39: Adminansicht aller Gegenstände

10.4 Admin-Feature: Kategorienverwaltung

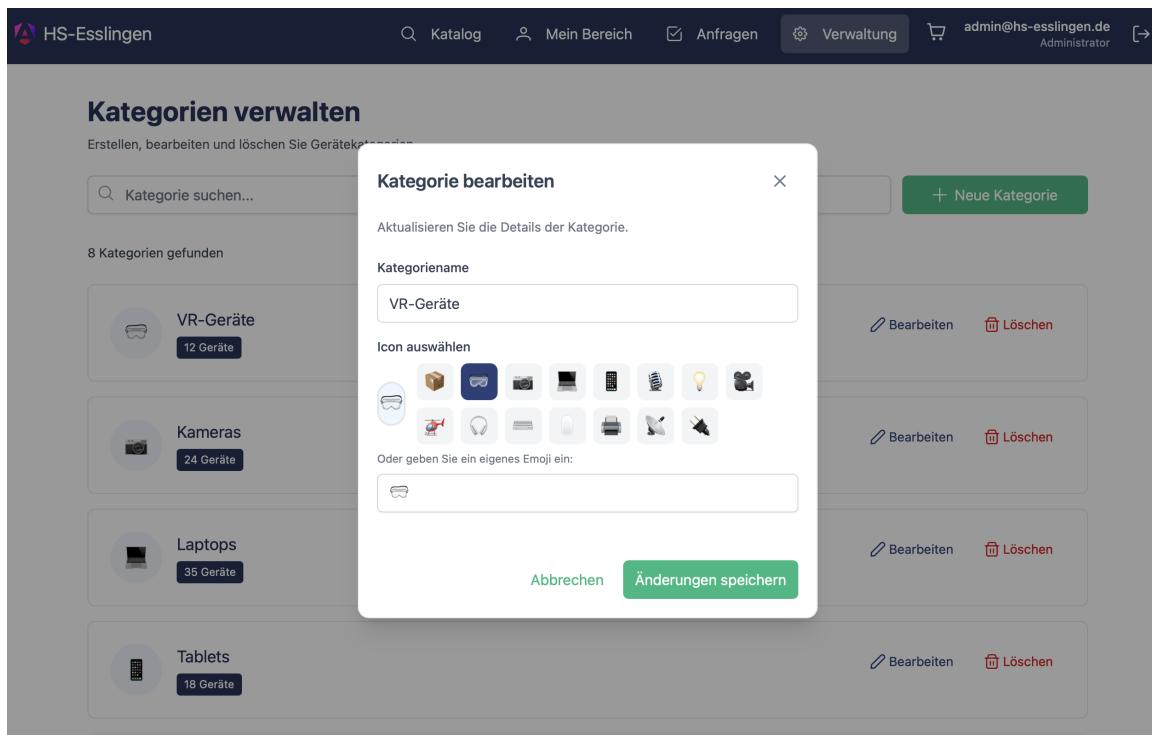


Abbildung 40: Kategorienverwaltung im Admin-Dashboard

Die Kategorienverwaltung ist Teil des Admin-Dashboards und ermöglicht es Administratoren, Gerätetypen im System übersichtlich zu verwalten. Das Feature wurde mit Angular umgesetzt und nutzt **PrimeNG**-Komponenten für die Benutzeroberfläche.

10.4.0.1 Funktionen

- Anzeige aller vorhandenen Kategorien
- Suchfunktion zur Filterung der Kategorienliste
- Erstellen neuer Kategorien über eine Eingabe
- Bearbeiten bestehender Kategorien (Name & Icon)
- Löschen einer Kategorie, sofern keine Geräte zugeordnet sind

10.4.0.2 Datenmodell

Die Kategorien sind typisiert über ein Interface mit folgendem Aufbau:

```
interface Category {  
    id: string;
```

```

name: string;
icon: string;
deviceCount: number;
}

```

Jede Kategorie besitzt einen Identifier, einen Namen, ein Icon (in diesem Fall als Emoji) und die Anzahl der verknüpften Geräte. Besonders `deviceCount` ist relevant, da Kategorien nicht gelöscht werden dürfen, wenn noch Geräte enthalten sind.

Zum Testen werden Beispiel-Daten genutzt:

```

categories: Category[] = [
  { id: '1', name: 'VR-Geräte', icon: '', deviceCount: 12 },
  { id: '2', name: 'Kameras', icon: '', deviceCount: 24 },
  ...
];

```

10.5 Admin-Feature: Verleiher-Zuordnung

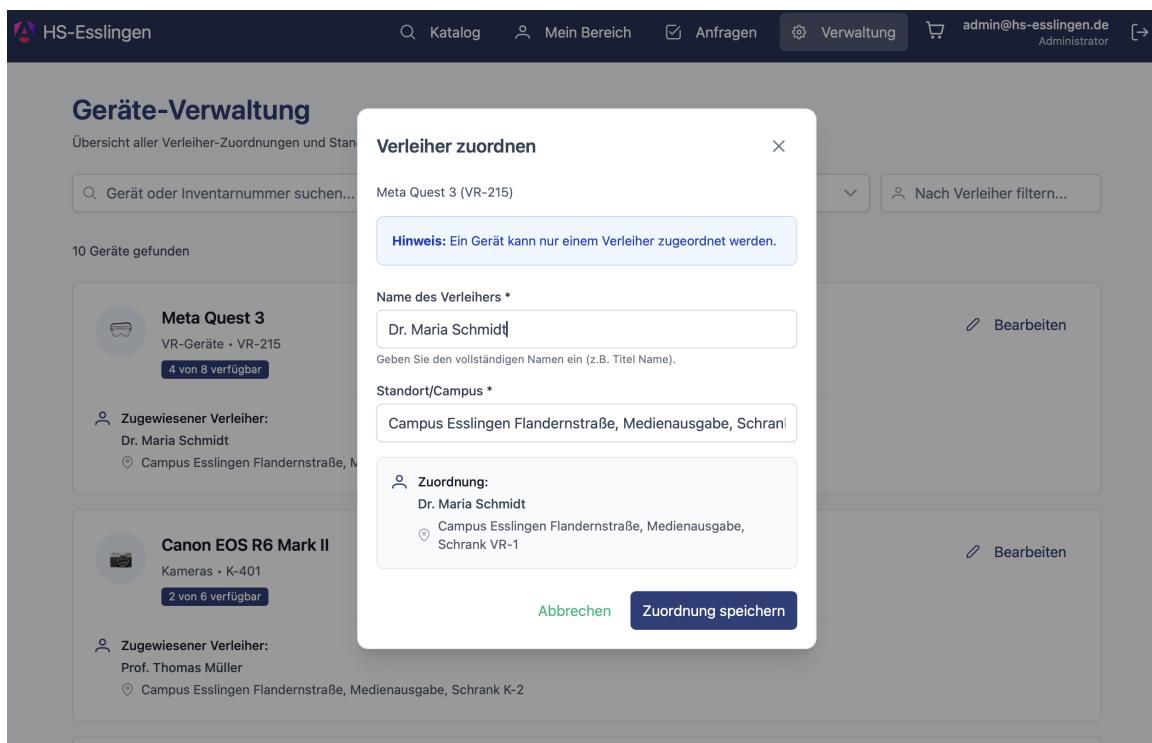


Abbildung 41: Dialog zur Verleiher-Zuordnung in der Geräteverwaltung

Die Verleiher-Zuordnung im Admin-Bereich ermöglicht es, Geräte im System zu verwalten und jedem Gerät einen Verleiher inklusive Standort/Campus zuzuordnen. Auf der Gerä-

teliste werden Name, Kategorie, Inventarnummer, Verfügbarkeit und aktueller Verleiher angezeigt.

Über einen **Bearbeiten**-Button pro Gerät kann ein Dialog geöffnet werden (siehe Abbildung 41), in welchem der Name des Verleiher sowie der Standort eingetragen oder geändert werden kann. Unterhalb der Felder wird eine Zusammenfassung der Zuordnung angezeigt, bevor diese gespeichert wird.

10.5.0.1 Funktionen

- Übersicht aller Geräte (z. B. *Meta Quest 3, Canon EOS R6 Mark II*)
- Suchfeld zur Suche nach Gerätenamen oder Inventarnummern
- Filter nach Kategorie (VR-Geräte, Kameras, Laptops, usw.)
- Optionaler Filter nach Verleihernamen
- Bearbeiten und Ändern der Verleiher-Zuordnung über Dialog
- Aktualisierung der Geräteliste nach dem Speichern

10.6 Warenkorb

Der Warenkorb stellt eine Benutzeroberfläche bereit um gesammelte gewünschte Ausleihanfragen abzusenden. Dazu wird zunächst ein Produkt auf der jeweiligen Produkt-Detailseite aufgerufen. Auf der rechten Seite kann das gewünschte Abholdatum ausgewählt werden. Mit einem Click auf "Zum Warenkorb hinzufügen" kann das Produkt zum Warenkorb hinzugefügt werden.

The screenshot shows a product detail page for a 'Meta Quest Pro' VR-Equipment. The top navigation bar includes links for 'Katalog', 'Mein Bereich', and 'Student'. The main content area is divided into several sections:

- Product Information:** Shows the product name 'Meta Quest Pro' and category 'VR-Equipment'. A 'Verfügbar' (Available) button is present.
- Description:** A brief description stating it's a High-End Mixed-Reality-Headset with Eye-Tracking and Gesichtserkennung.
- Technical Specifications:** A large, mostly empty section.
- Accessories:** Lists 'Enthaltenes Zubehör' (Included accessories): '2x Touch Pro Controller', 'Ladedock', 'USB-C Kabel', and 'Transporttasche'.
- Keywords:** Lists 'Schlagwörter' (Keywords): 'VR-Equipment'.
- Barcode:** A placeholder for 'Nummer:' with a barcode icon.
- Ausleihbedingungen:** Shows 'Ausleihzeit:' (Loan duration) as '10 Tage' (10 days).
- Availability:** Shows 'Gesamt:' (Total) as '5 Exemplare' (5 copies) 'verfügbar' (available). It also specifies the 'Campus' as 'Campus Esslingen Flandernstraße' and the 'Standort' (Location) as 'VR-Labor F01.403'.
- Date:** A field for 'Abholdatum' (Pickup date) set to '10.12.2025' with a calendar icon.
- Action Buttons:** A dark blue button labeled 'In den Warenkorb' (Add to cart).

Abbildung 42: Produkt-Detailseite

Dabei wird auch das gewählte Datum in den Warenkorb übertragen. Im Header der Seite zeigt das Warenkorb-Icon ein Badge mit der Anzahl der Produkte die sich im Warenkorb befinden und der Button SZum Warenkorb hinzufügen" wird grün und ändert den Text in SZum Warenkorb hinzugefügt".

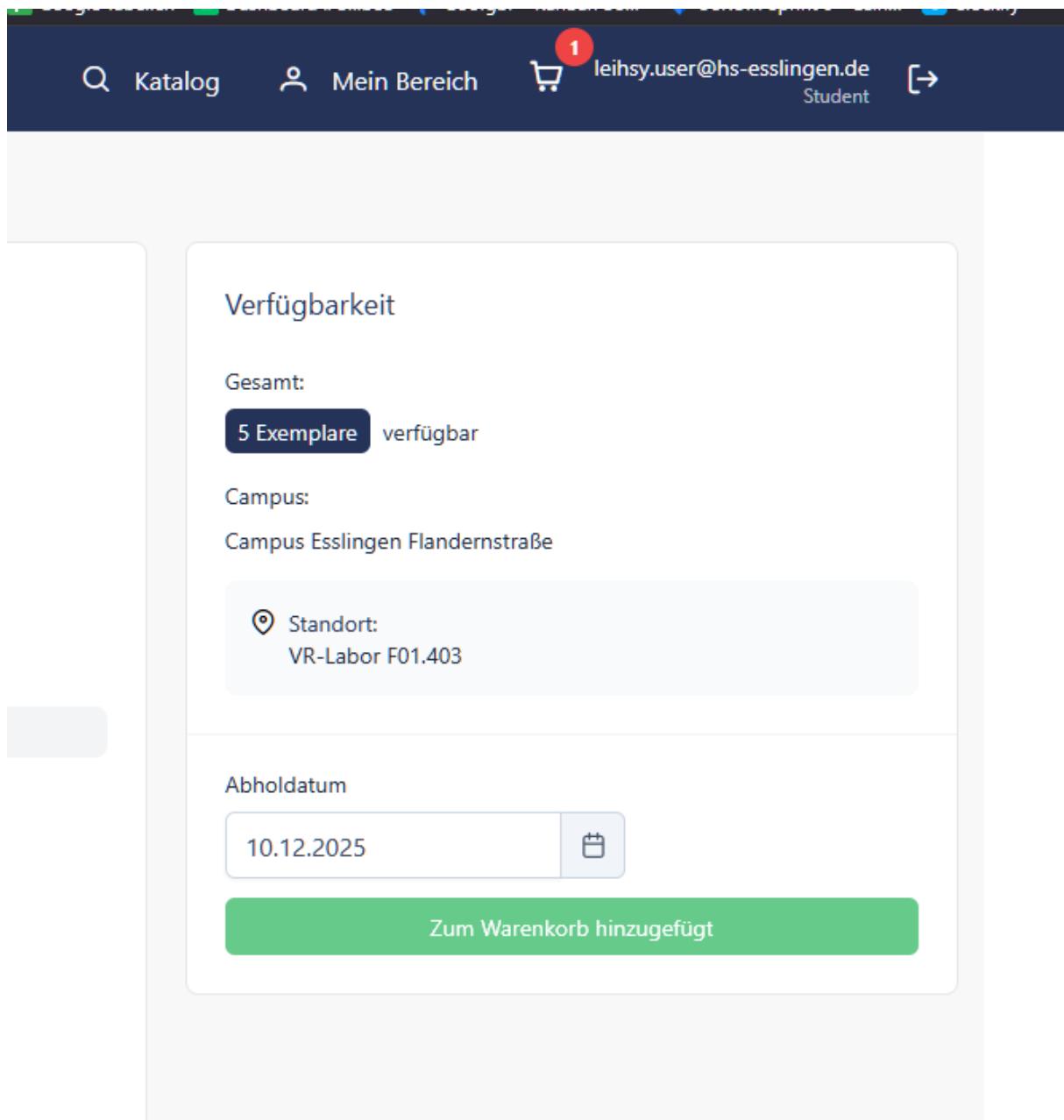


Abbildung 43: Zum Warenkorb Hinzugefügt

Aus technischer Sicht wird bei dem Click auf SZum Warenkorb hinzufügen" die Funktion addItem des Cart-Services aufgerufen. Dieser speichert die Produkt-ID, das Abholdatum und die neu generierte Warenkorb-ID des Items (die dazu dient Einträge im Warenkorb zu unterscheiden) im lokalen Speicher des Browsers. Dies sorgt für eine verbesserte Latenz und verringert den Verarbeitungsaufwand im Backend. Außerdem wird die Observable itemCount aktualisiert die beispielsweise vom Header genutzt wird um das Badge mit der Anzahl der Produkte im Warenkorb immer aktuell und automatisch auch über mehrere Tabs hinweg synchronisiert anzuzeigen. Mit einem Click auf das Warenkorb-Icon im Header gelangt man auf die Warenkorb-Seite. Hier werden alle Produkte angezeigt die sich

aktuell im Warenkorb befinden sowie das zuvor ausgewählte Abholdatum.

The screenshot shows the HS-Esslingen library website's shopping cart page. At the top, there is a dark header bar with the HS-Esslingen logo, a search icon, a user menu with the email 'leihsysuser@hs-esslingen.de' and status 'Student', and a cart icon with a red notification bubble containing the number '1'. Below the header, the page title 'Warenkorb' is displayed with a count of '1'. The main content area contains a product card for 'Meta Quest Pro'. The card includes the product name, inventory number '1', a category dropdown, and a red trash can icon in the top right corner. Below the card, there are two sections: 'Abholort:' with the address 'Campus Esslingen Flandernstraße VR-Labor F01.403' and 'Abholdatum:' with a date input field set to '10.12.2025' and a calendar icon. A third section, 'Ausleihbedingungen:', contains the text 'Ausleihfrist: 10 Tage' and 'Verlängerungen: Nach Verfügbarkeit'. At the bottom of the page, there is a summary section titled 'Gesamtanzahl:' showing '1 Produkt(e)'. It includes a checkbox for accepting terms and conditions, a 'Weiter suchen' button, and a large blue 'Ausleihanfrage abschicken' button.

Abbildung 44: Warenkorb-Seite

In der rechten oberen Ecke jedes Produkts befindet sich ein roter Mülleimer, mit dessen Anklicken das entsprechende Produkt aus dem Warenkorb gelöscht wird. Dazu wird im Cart-Service die Funktion `removeItem` aufgerufen die den entsprechenden Eintrag anhand seiner Warenkorb-ID aus dem lokalen Browser-Speicher löscht und den `itemCount` aktualisiert. Es werden weitere Informationen zu dem jeweiligen Produkt angezeigt, beispielsweise die Inventarnummer, der Abholort an dem sich das Produkt befindet und die maximale Ausleihdauer. Unter den Produkten befindet sich eine Checkbox mit der der Entleiher versichert, das Produkt ünbeschädigt, vollständig und rechtzeitig zurückzugeben". Nur wenn diese Zusicherung erteilt wurde kann der Button zum Absenden der Ausleihanfrage(n) betätigt werden.

This screenshot shows a simplified version of the shopping cart summary from Abbildung 44. It displays the same 'Gesamtanzahl:' section with '1 Produkt(e)'. Below it is a checkbox labeled 'Ich akzeptiere die Ausleihbedingungen und verpflichte mich, den/die Artikel unbeschädigt, vollständig und rechtzeitig zurückzugeben.' followed by a checked checkmark. At the bottom, there are two buttons: a white 'Weiter suchen' button and a dark blue 'Ausleihanfrage abschicken' button.

Abbildung 45: Warenkorb-Checkout

Dieser sendet für jedes Produkt im Warenkorb einzeln eine Ausleihanfrage an das Backend. Somit vereinfacht sich die Behandlung der Ausleihanfragen im Backend verglichen mit Sammelaufträgen oder Ähnlichem. Nach dem Absenden der Ausleihanfragen wird der Warenkorb mit der Funktion clearCart des Cart-Services im lokalen Browser-Speicher gelöscht.

10.7 Buchungsübersicht

10.7.1 Benutzeroberfläche

Abbildung 46 zeigt die Benutzeroberfläche der Buchungsübersicht. Hier werden alle bestehenden, ausstehenden oder stornierten Buchungen eines Benutzers tabellarisch dargestellt. Über ein zentrales Suchfeld können Buchungen nach Produktnamen, Inventarnummer, Verleiher oder Status gefiltert werden.

Die Tabelle enthält folgende Spalten:

- **Produkt:** Anzeigename des gebuchten Geräts.
- **Inventarnummer:** Eindeutige Kennung des physischen Geräts.
- **Verleiher:** Verantwortlicher Nutzer bzw. Administrator.
- **Status:** Buchungsstatus wie „Ausstehend“ oder „Storniert“.
- **Abholung / Rückgabe:** Geplante Zeiträume der Leihe.
- **Erstellt am:** Erstellungszeitpunkt der Buchung.

Am unteren Rand befindet sich eine Pagination, über die zwischen mehreren Buchungsseiten navigiert werden kann.

PRODUKT	INVENTARNUMMER	VERLEIHER	STATUS	ABHOLUNG	RÜCKGABE	ERSTELLT AM
Meta Quest Pro	VR-PRO-001	admin@hs-esslingen.de	Ausstehend	10.12.2025	15.12.2025	04.12.2025, 13:05
Meta Quest Pro	VR-PRO-001	admin@hs-esslingen.de	Storniert	10.12.2025	15.12.2025	04.12.2025, 16:48
Meta Quest Pro	VR-PRO-001	admin@hs-esslingen.de	Storniert	10.12.2025	15.12.2025	04.12.2025, 16:48

Abbildung 46: Buchungsverlauf eines Users

10.7.2 Backend-Implementierung

Dieses Kapitel dokumentiert die Implementierung des Buchungssystems im LeihSy-Backend. Der Fokus liegt auf der Umstellung von Entity-basierten Responses auf das DTO-Pattern, der korrekten Status-Verwaltung und der Integration mit dem Security-System.

10.7.2.1 DTO-Pattern und Circular Reference

Ausgangssituation:

Der BookingController gab Booking-Entities direkt als Response zurück. Dies führte zu einem Circular Reference Problem: Jackson folgte allen bidirektionalen JPA-Beziehungen (Booking → Item → Product → Items → Bookings) und generierte JSON-Responses mit über 8000 Zeilen.

Lösungsansätze:

Zwei Ansätze wurden evaluiert:

Option A - JsonIgnoreProperties: Annotations direkt an Entity-Feldern würden die Serialisierung steuern. Nachteil: Vermischt Persistenz-Logik mit Präsentations-Logik.

Option B - DTO-Pattern (gewählt): Entities bleiben sauber, DTOs enthalten nur benötigte Felder als flache Struktur. MapStruct übernimmt automatisches Mapping zwischen Entity und DTO.

Durchgeführte Änderungen:

- BookingDTO mit Lombok-Annotations optimiert (@Data, @Builder, @NoArgsConstructor, @AllArgsConstructor)
- Code-Reduktion von ca. 150 auf 30 Zeilen durch Wegfall manueller Getter/Setter
- BookingService: Alle public Methoden geben nun BookingDTO statt Booking zurück
- BookingMapper (MapStruct) bildet verschachtelte Entity-Beziehungen auf flache DTO-Felder ab
- BookingController: Alle Endpoints verwenden nun DTOs in Request/Response

Ergebnis:

JSON-Response wurde von 8000+ auf ca. 50 Zeilen reduziert. Entities bleiben frei von Serialisierungs-Logik. Klare Trennung zwischen Datenbank-Modell (Entity) und API-Modell (DTO).

10.7.2.2 Status-Verwaltung über Timestamps

Ausgangssituation:

Initial wurde versucht, den Booking-Status als Enum zu implementieren und explizit zu setzen. Dies führte zu Compile-Fehlern, da die Booking-Entity den Status als berechnetes Transient-Feld implementiert hatte.

Konzept der berechneten Status:

Der Status wird nicht in der Datenbank gespeichert, sondern zur Laufzeit aus den vorhandenen Timestamp-Feldern berechnet. Die Logik folgt einer Prioritäts-Kaskade: deletedAt → CANCELLED, returnDate → RETURNED, distributionDate → PICKED_UP, confirmedPickup → CONFIRMED, ansonsten → PENDING.

Statt einen Status zu setzen, werden die entsprechenden Timestamp-Felder manipuliert:

Gewünschter Status	Aktion
PENDING	Alle Timestamps auf NULL
CONFIRMED	confirmedPickup setzen
PICKED_UP	distributionDate setzen
RETURNED	returnDate setzen
CANCELLED	deletedAt setzen

Tabelle 2: Status-Steuerung über Timestamps

Vorteile dieses Ansatzes:

- Single Source of Truth: Status ergibt sich eindeutig aus Timestamps
- Automatischer Audit-Trail für alle Status-Wechsel
- Keine Inkonsistenzen durch manuelle Status-Pflege möglich
- Vereinfachte Datenbank-Queries

10.7.2.3 Architektur-Entscheidungen

DTO-Pattern als Standard:

Die konsequente Verwendung von DTOs für alle API-Responses verhindert nicht nur Circular Reference Probleme, sondern bietet weitere Vorteile: Trennung von Datenbank- und API-Struktur, gezielte Datenauswahl pro Endpoint, Schutz interner Strukturen vor Breaking Changes und bessere Performance durch reduzierte Datenmenge.

Berechnete Status statt gespeicherter Enum:

Die Entscheidung für berechnete Status basierend auf Timestamps statt eines gespeicherten Enum-Wertes eliminiert mögliche Inkonsistenzen. Jeder Status-Wechsel ist automatisch mit einem Zeitstempel dokumentiert (Audit-Trail) und die Logik ist zentralisiert in der Entity-Klasse.

MapStruct für DTO-Mapping:

MapStruct wurde gewählt, da es zur Compile-Zeit Code generiert (keine Reflection zur Laufzeit) und typsicher ist. Die Annotation-basierte Konfiguration ist übersichtlich und ermöglicht komplexe Mappings wie verschachtelte Entity-Beziehungen auf flache DTO-Felder.

Lombok zur Reduktion von Boilerplate:

Unter Einsatz von Lombok (siehe Abschnitt 5.2.1) wurden die DTOs und Services mit Annotations wie `@Data`, `@Builder` und `@RequiredArgsConstructor` versehen. Dies reduzierte den Code in den Booking-Komponenten um ca. 80% und verbesserte die Lesbarkeit erheblich.

API-Dokumentation mit Swagger:

Der BookingController wurde mit vollständigen OpenAPI-Annotations versehen: `@Tag` für Gruppierung, `@Operation` für Endpoint-Beschreibungen, `@ApiResponses` für Status-Codes und `@Schema` für Request-DTOs. Die Dokumentation ist unter `/swagger-ui.html` verfügbar.

10.7.2.4 Zusammenfassung

Komponente	Vorher	Nachher
BookingDTO	Manuelle Getter/Setter (ca. 150 Zeilen)	Lombok Annotations (ca. 30 Zeilen)
BookingService	Return Type: Booking Entity	Return Type: BookingDTO
BookingController	Entities in Response führten zu Circular Reference	DTOs liefern sauberes, flaches JSON
Status-Verwaltung	Versuch Enum zu setzen führte zu Compile-Fehler	Timestamps setzen, Status wird berechnet
UserService	getCurrentUser() Methode fehlte	Implementiert mit Auto-User-Creation
API-Dokumentation	Keine Swagger-Annotations	Vollständige OpenAPI-Dokumentation

Tabelle 3: Übersicht aller Änderungen am Booking-System

10.8 Authentifizierung und Benutzerverwaltung

10.8.1 Überblick & Architektur

Das LeihSy-System implementiert eine moderne OAuth2-basierte Authentifizierung mit JSON Web Tokens (JWT). Die Authentifizierung erfolgt über den zentralen Keycloak-Server der Hochschule Esslingen, was eine nahtlose Integration mit bestehenden RZ-Accounts ermöglicht.

10.8.1.1 Komponenten

Die Security-Architektur besteht aus drei Hauptkomponenten:

- **Keycloak Identity Provider:** Zentrale Authentifizierungsstelle der Hochschule Esslingen unter `auth.insy.hs-esslingen.com`. Verwaltet Benutzerkonten, Rollen und erstellt JWT-Tokens nach erfolgreichem Login.
- **Angular Frontend:** Übernimmt die Token-Verwaltung im Browser. Nutzt die `keycloak-js` Bibliothek für den OAuth2 Authorization Code Flow mit PKCE.
- **Spring Boot Backend:** Validiert eingehende JWT-Tokens und prüft Berechtigungen. Konfiguriert als OAuth2 Resource Server.

10.8.1.2 Authentifizierungsablauf

Der Login-Prozess folgt dem OpenID Connect (OIDC) Standard:

1. Benutzer ruft Frontend auf (`http://localhost:4200`)
2. Angular leitet zur Keycloak Login-Seite weiter
3. Benutzer authentifiziert sich mit RZ-Account
4. Keycloak erstellt JWT-Token und sendet es an Frontend
5. Frontend speichert Token im Browser (SessionStorage)
6. Bei jedem Backend-Request wird Token im Authorization-Header mitgesendet
7. Backend validiert Token und extrahiert Benutzerinformationen

10.8.2 JWT-Token und Security-Konfiguration

Ein JSON Web Token besteht aus drei Base64-kodierten Teilen (Header, Payload, Signature):

```
{  
  "sub": "a980c70e-...",           // Keycloak User-ID  
  "preferred_username": "admin",    // Username  
  "email": "admin@hs-esslingen.de",  
  "realm_access": {  
    "roles": ["admin", "user"]      // Zugewiesene Rollen  
  },  
  "iss": "https://auth.insy.hs-esslingen.com/realms/insy",  
  "exp": 1733331200                // Ablaufzeit (Unix Timestamp)  
}
```

Die Token-Signatur wird mit RSA-256 erstellt und kann vom Backend mit dem Public Key von Keycloak validiert werden.

10.8.2.1 SecurityConfig Klasse

Die zentrale Security-Konfiguration erfolgt in `SecurityConfig.java`:

- **CORS-Konfiguration:** Erlaubt Cross-Origin Requests vom Frontend (`localhost:4200`)
- **CSRF-Schutz deaktiviert:** Da JWT-Tokens im Authorization-Header übertragen werden, ist der Cookie-basierte CSRF-Schutz nicht erforderlich
- **OAuth2 Resource Server:** Konfiguriert JWT-Validierung mit Keycloak als Issuer
- **Rollen-Mapping:** Konvertiert Keycloak-Rollen (`admin`) zu Spring Security Authorities (`ROLE_admin`)

10.8.2.2 Autorisierungsregeln

Die URL-Muster werden mit spezifischen Berechtigungen versehen:

Endpoint-Muster	Berechtigung
<code>/api/products/**</code>	Öffentlich (permitAll)
<code>/api/categories/**</code>	Öffentlich (permitAll)
<code>/api/admin/**</code>	Nur Admin-Rolle
<code>/api/bookings/lenders/**</code>	Admin oder Lender-Rolle
Alle anderen	Authentifizierung erforderlich

Tabelle 4: Zugriffskontrolle nach Endpoint-Mustern

10.8.3 Rollensystem

Das System unterscheidet drei Benutzerrollen:

- **user:** Standard-Rolle für Studierende. Kann Gegenstände durchsuchen und Buchungen erstellen.
- **lender:** Rolle für Verleiher. Kann zusätzlich Buchungsanfragen bearbeiten und Ausgabe/Rückgabe dokumentieren.
- **admin:** Administrator-Rolle. Vollzugriff auf alle Funktionen inklusive Gegenstandsverwaltung und Statistiken.

Die Rollenzuweisung erfolgt zentral in Keycloak über Realm Roles. Ein Benutzer kann mehrere Rollen gleichzeitig haben.

10.8.4 Frontend-Integration

10.8.4.1 AuthService

Der AuthService kapselt die Keycloak-Integration:

- Initialisiert Keycloak-Client mit `keycloak-js` Bibliothek
- Verwaltet Token-Lifecycle (Abruf, Refresh, Ablauf)
- Stellt Methoden zur Rollen-Prüfung bereit (`hasRole()`)
- Extrahiert Benutzerinformationen aus Token-Claims

10.8.4.2 Route Guards

Angular Route Guards schützen Frontend-Routes vor unberechtigtem Zugriff. Der `AuthGuard` prüft den Login-Status, während der `RoleGuard` die erforderlichen Rollen validiert. Bei fehlender Berechtigung erfolgt eine Umleitung zur Login- oder Fehlerseite. Die grundlegende Funktionsweise von Guards im Angular-Framework wird in Abschnitt 9.2 beschrieben.

10.8.4.3 Token-Validierung

Bei jedem Backend-Request durchläuft der JWT-Token folgende Validierungsschritte:

1. **Signatur-Prüfung:** Validierung mit Keycloak Public Key (RSA-256)
2. **Issuer-Prüfung:** Token muss von konfiguriertem Keycloak stammen
3. **Ablauf-Prüfung:** `exp`-Claim darf nicht überschritten sein
4. **Rollen-Extraktion:** `realm_access.roles` wird in Spring Security Authorities konvertiert

Bei fehlgeschlagener Validierung antwortet das Backend mit HTTP 401 Unauthorized.

10.8.4.4 Automatische Benutzererstellung

Beim ersten Login eines Benutzers wird automatisch ein User-Datensatz in der lokalen Datenbank angelegt:

- `unique_id`: Keycloak Subject (UUID)
- `name`: Aus Token-Claim `preferred_username`
- `email`: Aus Token-Claim `email`

Die Methode `UserService.getCurrentUser()` extrahiert die Keycloak-ID aus dem Security Context und holt oder erstellt den entsprechenden User.

10.8.4.5 Vorteile der JWT-basierten Authentifizierung

- **Stateless:** Backend muss keine Sessions speichern
- **Skalierbar:** Jeder Backend-Server kann Token validieren
- **Single Sign-On:** Nutzer ist automatisch in allen HS-Anwendungen eingeloggt
- **Standardisiert:** OAuth2 und OIDC sind etablierte Standards

10.8.4.6 Implementierte Schutzmaßnahmen

- Token-Signaturprüfung verhindert Token-Manipulation
- Kurze Token-Laufzeit (Standard: 5 Minuten) minimiert Missbrauchsrisiko
- HTTPS erzwingt verschlüsselte Übertragung (in Produktion)
- CORS-Whitelist beschränkt erlaubte Origins

10.8.4.7 Keycloak-Konfiguration

Das System nutzt den gemeinsamen Keycloak-Server mit dem InSy-Inventarsystem:

- **Realm:** insy
- **Backend Client-ID:** insy-backend (temporär, leihsy-backend-dev geplant)
- **Frontend Client-ID:** leihsy-frontend-dev
- **Token Endpoint:** /realms/insy/protocol/openid-connect/token

Die Keycloak-Verbindungsparameter sind in `application.yml` hinterlegt und können pro Umgebung (dev/prod) angepasst werden.

10.8.5 Automatische Benutzersynchronisation

10.8.5.1 Übersicht

Dieses Kapitel dokumentiert die Implementierung der automatischen Benutzersynchronisation zwischen dem Keycloak Identity Provider und der LeihSy-Datenbank. Die Synchronisation stellt sicher, dass Benutzer automatisch in der lokalen Datenbank angelegt werden, sobald sie sich erstmals über Keycloak authentifizieren.

10.8.5.2 Ausgangssituation

Vor der Implementierung mussten Benutzer manuell in der LeihSy-Datenbank angelegt werden, bevor sie das System nutzen konnten. Dies führte zu einem Medienbruch: Obwohl die Authentifizierung über Keycloak erfolgte, existierte der Benutzer nicht in der lokalen Datenbank und konnte daher keine Buchungen durchführen.

10.8.5.3 Lösungsansatz

Die implementierte Lösung synchronisiert Benutzerdaten automatisch beim ersten Login. Dabei werden die relevanten Informationen aus dem JWT-Token extrahiert und in der lokalen Datenbankpersistiert.

10.8.5.4 Backend-Komponenten

Im Backend wurde ein neuer REST-Endpoint `GET /api/users/me` implementiert. Dieser Endpoint gibt die Daten des aktuell authentifizierten Benutzers zurück. Falls der Benutzer noch nicht in der Datenbank existiert, wird er automatisch angelegt.

Die Benutzeridentifikation erfolgt über die eindeutige Keycloak-ID (Subject Claim), die im JWT-Token enthalten ist. Zusätzlich werden der Benutzername und die E-Mail-Adresse aus dem Token extrahiert.

Ein Security-Filter wurde hinzugefügt, der bei jedem authentifizierten API-Request prüft, ob der Benutzer bereits existiert. Falls nicht, wird ein neuer Datensatz angelegt. Existiert der Benutzer bereits, werden geänderte Stammdaten wie der Name aktualisiert.

10.8.5.5 Frontend-Komponenten

Im Frontend wurde der AuthService erweitert. Dieser verwaltet nun zusätzlich zum Keycloak-Authentifizierungsstatus auch die Benutzerdaten aus der lokalen Datenbank.

Ein HTTP-Interceptor wurde implementiert, der automatisch das JWT-Token als Bearer-Token an alle API-Requests anhängt. Dies ermöglicht die serverseitige Authentifizierung ohne manuelle Token-Verwaltung in den einzelnen Services.

Die Initialisierungsreihenfolge beim Anwendungsstart wurde angepasst: Zuerst wird Keycloak initialisiert, danach erfolgt der Aufruf des `/api/users/me` Endpoints zur Benutzersynchronisation.

10.8.5.6 Race Condition bei der Initialisierung

Eine zentrale Herausforderung war die korrekte Reihenfolge der Initialisierung. Der AuthService im Frontend wurde ursprünglich vor Abschluss der Keycloak-Initialisierung instanziert, wodurch der Authentifizierungsstatus fälschlicherweise als `false` erkannt wurde.

Die Lösung besteht darin, die Initialisierungslogik aus dem Konstruktor zu entfernen und stattdessen eine explizite `initialize()`-Methode zu verwenden. Diese wird erst aufgerufen, nachdem Keycloak vollständig initialisiert wurde.

10.8.5.7 Injection Context in asynchronen Funktionen

Angular's `inject()`-Funktion kann nur synchron innerhalb eines Injection Context aufgerufen werden. In asynchronen Funktionen geht dieser Kontext nach dem ersten `await` verloren.

Die Lösung besteht darin, alle benötigten Dependencies am Anfang der Funktion zu injizieren, bevor asynchrone Operationen ausgeführt werden.

10.8.5.8 Rollenverwaltung

Die Benutzerrollen werden ausschließlich in Keycloak verwaltet und bei jedem Request aus dem JWT-Token extrahiert. Eine lokale Speicherung der Rollen in der LeihSy-Datenbank ist nicht erforderlich.

Die Extraktion erfolgt aus zwei Token-Bereichen:

- **Realm-Rollen:** Globale Berechtigungen aus dem Claim `realm_access.roles`
- **Client-Rollen:** Anwendungsspezifische Berechtigungen aus `resource_access`

Die Bedeutung und Berechtigungen der einzelnen Rollen (`admin`, `lender`, `user`) sind in Abschnitt 10.8.3 dokumentiert.

10.8.5.9 Ergebnis

Nach der Implementierung werden Benutzer vollautomatisch beim ersten Login angelegt. Der Prozess ist für den Endbenutzer transparent und erfordert keine zusätzlichen Registrierungsschritte. Die Datenbank-ID des Benutzers steht nach erfolgreicher Synchronisation für alle weiteren API-Aufrufe zur Verfügung.

10.9 Bildverwaltung

Dieses Kapitel dokumentiert die Implementierung des Bild-Upload-Systems für Produktbilder im LeihSy-System. Das System ermöglicht Administratoren und Verleiher, Bilder für Produkte hochzuladen, die anschließend im Katalog angezeigt werden.

10.9.1 Anforderungen

Gemäß User Story US-009 soll das System Bilder speichern und ausliefern können. Die Anforderungen umfassen:

- Bildupload mit Validierung (max. 5MB, Formate: JPG, PNG, WebP)
- Speicherung im Filesystem des Servers
- REST-Endpoints für Upload, Download und Löschen
- Integration in das Admin-Dashboard zur Produkterstellung

- Anzeige der Bilder im Produktkatalog

10.9.2 Architektur-Entscheidung: Filesystem vs. Datenbank

Für die Speicherung von Bildern wurden zwei Optionen evaluiert:

Option A – Datenbank (BLOB): Bilder werden als Binary Large Object direkt in der Datenbank gespeichert. Vorteile sind die einfache Datensicherung und transaktionale Konsistenz. Nachteile sind jedoch die erhöhte Datenbankgröße, langsamere Queries und höherer Speicherverbrauch.

Option B – Filesystem (gewählt): Bilder werden im Dateisystem gespeichert, die Datenbank enthält nur den URL-Pfad als String. Diese Lösung bietet bessere Performance beim Ausliefern von Bildern, einfachere Skalierbarkeit und die Möglichkeit, einen CDN vorzuschalten. Die Datenbank bleibt schlank und performant.

Die Entscheidung fiel auf das Filesystem, da für ein Verleihsystem mit Produktbildern die Performance beim Laden der Katalogseite wichtiger ist als transaktionale Konsistenz der Bilddaten.

10.9.3 Backend-Implementierung

10.9.3.1 Konfiguration

Die Upload-Konfiguration erfolgt in der `application.properties`:

```
spring.servlet.multipart.enabled=true
spring.servlet.multipart.max-file-size=5MB
spring.servlet.multipart.max-request-size=5MB
app.upload.dir=uploads/images/
```

Die Klasse `FileStorageConfig` erstellt beim Anwendungsstart automatisch das Upload-Verzeichnis, falls es nicht existiert. Dies verhindert Laufzeitfehler beim ersten Upload.

10.9.3.2 ImageService

Der `ImageService` kapselt die gesamte Logik für Dateispeicherung und -verwaltung:

- `saveImage(MultipartFile, String productName)`: Validiert die Datei und speichert sie mit einem aus dem Produktnamen generierten Dateinamen
- `loadImage(String filename)`: Lädt ein Bild aus dem Filesystem
- `deleteImage(String filename)`: Löscht ein Bild
- `validateImage(MultipartFile)`: Prüft Dateigröße und Dateityp

Die Validierung stellt sicher, dass nur erlaubte Dateitypen (JPG, PNG, WebP) mit maximal 5MB akzeptiert werden. Bei Verstößen wird eine `FileNotFoundException` geworfen.

10.9.3.3 Dateinamenskonvention

Ursprünglich wurden UUIDs als Dateinamen verwendet (z.B. 550e8400-e29b-41d4-a716-44665544). Dies wurde geändert zu produktnamenbasierten Dateinamen für bessere Nachvollziehbarkeit:

```
"Valve Index"      → valve-index.jpg  
"Meta Quest Pro"  → meta-quest-pro.jpg  
"Canon EOS R6 II" → canon-eos-r6-ii.jpg
```

Der Produktname wird dabei in Kleinbuchstaben konvertiert und Sonderzeichen werden durch Bindestriche ersetzt. Da jedes Produkt genau ein Bild hat, sind keine UUIDs für Eindeutigkeit erforderlich.

10.9.3.4 ImageController

Der REST-Controller stellt drei Endpoints bereit:

Methode	Endpoint	Beschreibung
POST	/api/images/upload	Lädt ein Bild hoch und gibt die URL zurück
GET	/api/images/{filename}	Liefert das Bild als Binärdaten aus
DELETE	/api/images/{filename}	Löscht das angegebene Bild

Tabelle 5: REST-Endpoints der Bildverwaltung

Der GET-Endpoint setzt den korrekten `Content-Type` Header basierend auf der Dateiendung, sodass Browser das Bild direkt anzeigen können.

10.9.3.5 Integration mit ProductService

Beim Erstellen oder Aktualisieren eines Produkts wird das Bild automatisch verarbeitet:

1. Frontend sendet `FormData` mit Produktdaten und Bilddatei
2. `ProductController` empfängt Multipart-Request
3. `ProductService` ruft `ImageService.saveImage()` auf

4. Rückgabewert (Dateiname) wird als `imageUrl` im Product gespeichert
5. Datenbank enthält nur den Pfad: `/api/images/valve-index.jpg`

10.9.4 Frontend-Integration

10.9.4.1 ProductService-Erweiterung

Der Angular `ProductService` wurde um Multipart-Upload-Support erweitert:

```
createProduct(product: ProductDTO, image: File | null): Observable<Product> {
  const formData = new FormData();
  formData.append('product', new Blob([JSON.stringify(product)], {
    type: 'application/json'
}));
  if (image) {
    formData.append('image', image);
  }
  return this.http.post<Product>(`.${this.apiUrl}/products`, formData);
}
```

Das Produkt wird als JSON-Blob gesendet, das Bild als separater File-Teil im `FormData`-Objekt.

10.9.4.2 Admin-Dashboard

Im Admin-Product-Dashboard wurde ein Bild-Upload-Bereich hinzugefügt:

- File-Input mit `accept="image/*"` für Dateiauswahl
- Bildvorschau vor dem Speichern mittels `FileReader`
- Validierung von Dateityp und -größe im Frontend
- Signal-basierte Zustandsverwaltung: `selectedFile`, `imagePreview`

Bei der Bearbeitung eines bestehenden Produkts wird das aktuelle Bild als Vorschau angezeigt. Ein neues Bild ersetzt das alte automatisch.

10.9.4.3 Katalog-Anzeige

Die Katalogseite lädt Produktbilder über eine `getImageUrl()`-Methode: Die Methode behandelt verschiedene Fälle: fehlende Bilder, absolute URLs (externe Bilder) und relative API-Pfade (interne Bilder).

10.9.5 Deployment-Überlegungen

Für den Produktivbetrieb auf dem Server müssen Docker Volumes konfiguriert werden, damit Bilder Container-Neustarts überleben:

```
# docker-compose.yml
services:
  backend:
    volumes:
      - ./uploads:/app/uploads
```

Ohne Volume-Mapping würden alle hochgeladenen Bilder beim Neustart des Containers verloren gehen, da Docker Container standardmäßig zustandslos sind.

10.9.6 Zusammenfassung

Das implementierte Bild-Upload-System ermöglicht eine vollständige Verwaltung von Produktbildern. Die Architekturentscheidung für Filesystem-Speicherung bietet optimale Performance für den Anwendungsfall eines Produktkatalogs. Die klare Trennung zwischen ImageService (Dateiverwaltung) und ProductService (Geschäftslogik) ermöglicht einfache Wartung und spätere Erweiterungen wie Thumbnail-Generierung oder CDN-Integration.