

Digitaler Campus

Audit 4 im Entwicklungsprojekt WS22/23

Inhaltsverzeichnis

1. Expose
2. Stakeholderanalyse
3. Domänenmodell
4. Zielhierarchien
5. Alleinstellungsmerkmale
6. Projektrisiken
7. Erfordernisse
8. Anforderungen
9. Anwendungsmodellierungen
10. Durchgeführte POCs
11. Änderungen im Projekt
12. Projektergebnisse (Auszug)
13. Kritisch reflektiertes Prozessassessment und Fazit
14. Projektplan

Expose

- Problemstellung
- Zielsetzung, Version
- Leitthema
- Relevanz

Expose

Problemstellung

Der Campus Gummersbach der Technischen Hochschule Köln verfügt als solche über mehrere Gebäude und Räume, welche nicht immer eindeutig gekennzeichnet sind. Gerade Studenten im ersten Semester haben oft keinen Überblick welche Funktion ein Raum hat, welches Material zu Verfügung steht oder ob ein Raum bereits belegt ist.

Es existieren zwar Angebote, um Studenten die Orientierung am Campus zu erleichtern, diese sind aber meistens unzureichend oder nicht ausreichend bekannt. Im Hochschul-Planungs-System (HoPS) [1] der TH Köln sind Informationen nur oberflächlich zu finden. In der 26-Seitigen Erstsemester Broschüre [2] ist nur eine Seite dem Campus gewidmet, wo dieser nur grob kategorisiert ist.

[1] HoPS | HochschulPlanungsSystem der TH Köln. (n.d.). HoPS | HochschulPlanungsSystem der TH Köln.
Abgerufen am October 31, 2022, von <https://fhpwww.gm.fh-koeln.de/hops/modules/roomequipment/room.php?room=0215>

[2] TH Köln | Informationen für Erstsemester | Fakultät für Informatik und Ingenieurwissenschaften der TH Köln. Abgerufen am October 31, 2022, von https://www.th-koeln.de/mam/downloads/deutsch/hochschule/fakultaeten/informatik_und_ingenieurwissenschaften/erstsemesterbroschure_gesamt_august_2021.pdf

Zielsetzung, Version

Das Ziel ist es, eine Abbildung des Campus in digitaler Form darzustellen. Hierbei soll es Benutzern möglich sein, nach Räumen und Equipment zu filtern. Zum Equipment gehören z. B. die benötigte Software, Hardware oder die verfügbaren Sitzplätze. Die Raumverfügbarkeit soll auf Basis aktueller Stundenpläne und Benutzereingaben ermittelt werden.

Zusätzlich zur Filterfunktion können sich Nutzer auch direkt Informationen und Metadaten zu individuellen Räumen anzeigen lassen.

Expose

Leitthema

Device Shifting

Bislang befinden sich kleine Tafeln mit der Raumnummer, der Bezeichnung sowie gegebenenfalls dem Stundenplan des Raumes neben der Tür.

Die Informationstafeln neben jedem Raum werden durch E-Ink Paper Displays ersetzt. Über sie werden begrenzt Rauminformationen präsentiert.

Mittels eines NFC-Tags am Display sollen Anwender auf die Hauptanwendung geleitet werden. Dort erhalten sie näheren Überblick zu dem jeweiligen Raum und erhalten die Möglichkeit, sich für den entsprechenden Raum einzutragen.

Synchronisation

Sollte sich ein Anwender für den Raum eintragen, so wird der Status in der Anwendung und dem E-Ink Paper Display aktualisiert und man erhält Einblick, ob ein Raum aktuell belegt ist oder nicht.

Expose

Relevanz

Gesellschaftliche Relevanz

- Verbesserte Orientierung am Campus
- Ansehen der Hochschule wird gesteigert
- Zufriedenheit der Studierenden hebt sich
- Wege und Zeit können eingespart werden, da die Raumbelegungen direkt angezeigt werden können

Stakeholderanalyse

Stakeholderanalyse

Stakeholderanalyse

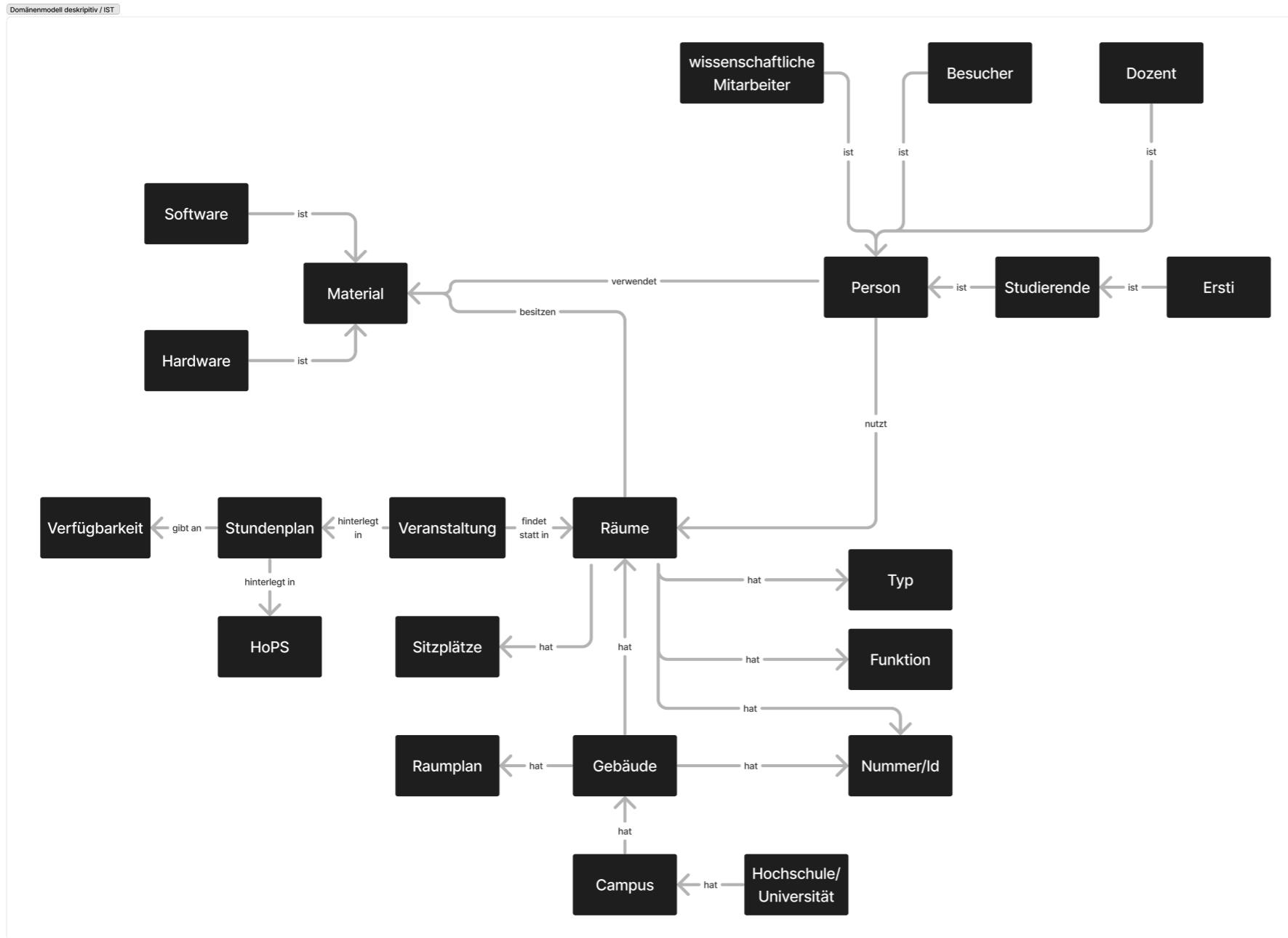
| Bezeichnung | | Bezug zum System | Objektbereich | Erfordernis / Erwartung |
|-------------------------------|---|------------------|----------------------------------|---|
| Benutzer | Studierende Wissenschaftliche Mitarbeiter Studentische Hilfskraft Pförtner | Anteil | Nutzer des Systems | Als Benutzer muss man wissen das dieses System existiert, um es nutzen zu können. |
| | | Interesse | Nutzer des Systems | Als Benutzer muss man einen Überblick über die Räume der Hochschule haben, um effizient verfügbare Räume finden zu können. |
| | | Interesse | Nutzer des Systems | Als Benutzer muss man einen Überblick über die Räume der Hochschule haben, um effizient ein benötigtes Material finden zu können. |
| | | Anspruch | Nutzer des Systems | Als Benutzer müssen die Metadaten zuverlässig sein. |
| | | Anspruch | Nutzer des Systems | Als Benutzer müssen die Daten bezüglich der Raumverfügbarkeit zuverlässig sein. |
| | | Anspruch | Nutzer des Systems | Als Benutzer müssen die Daten bezüglich der Raumverfügbarkeit stets aktuell (synchronisiert) sein. |
| | | Anspruch | Nutzer des Systems | Als Benutzer muss die Interaktion zur Bestimmung der Raumverfügbarkeit angenehm, schnell und minimal sein, um die Anwendung ungestört nutzen zu können. |
| | Dozent | Interesse | Nutzer des Systems | Als Dozent muss man einen Überblick über die Räume der Hochschule haben, um bei kurzfristigen Raumänderungen einen verfügbaren Raum finden zu können. |
| Hochschule | | Interesse | Bereitstellung von Informationen | Als Hochschule muss man Informationen zu den Räumen haben, um diese bereitstellen zu können |
| Besucher | | Anspruch | Foerderer | Als Hochschule muss das System genutzt werden, um den Weiterbetrieb des Systems zu begründen |
| | | Interesse | Besucher des Campus | Als Besucher muss man einen Überblick über die Räume der Hochschule haben, um effizient einen gesuchten Raum finden zu können. |
| primärer Stakeholder | | | | |
| sekundärer Stakeholder | | | | |
| tertiärer Stakeholder | | | | |

Domänenmodell

- Deskriptiv
- Präskriptiv

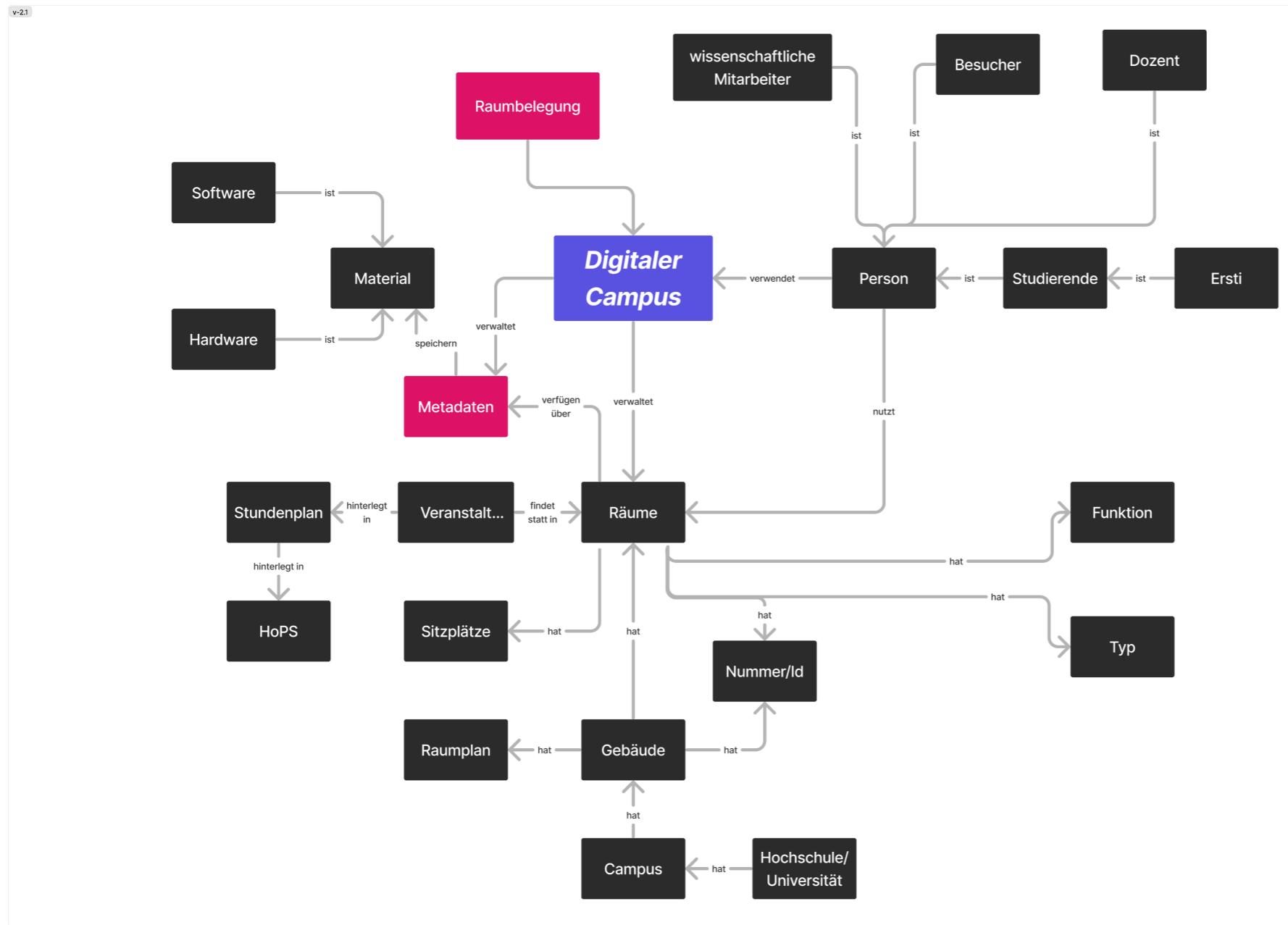
Domänenmodell

Deskriptiv



Domänenmodell

Präskriptiv



Zielhierarchien

- Strategische Ziele
- Taktische Ziele
- Operative Ziele

Zielhierarchien

Strategische Ziele

Es soll eine interaktive Abbildung des Campus Gummersbach erstellt werden. Studenten und andere Stakeholder sollen eine bessere Einsicht über das vorhandene Material und die generelle Verfügbarkeit jeden Raumes bekommen.

Taktische Ziele

Um einen besseren Durchblick in die Entwicklung der Modelle und des Systems zu erlangen, wird innerhalb des Teams ein Kanban-Board zur Planung und taktischen Verfolgung von Aufgaben erstellt.

Operative Ziele

Kurzfristig gesehen müssen alle notwendigen Daten katalogisiert und etwaige Technologien bewertet und verglichen werden.

Die Digitale Abbildung des Campus Gummersbach und dessen Räume soll mit Dummy-Metadaten verknüpft werden. Außerdem müssen Informationen zu den verfügbaren Materialien innerhalb eines Raumes herausgesucht und gespeichert werden.

Die Raumdaten sollen in einer interaktiven Karte (2D oder 3D) für einen Nutzer einsehbar sein. Hierfür muss entschieden werden, wie die Karte innerhalb des Anwendung erfolgreich umgesetzt werden kann.

Alleinstellungsmerkmale

- Physischer Raumplan
- HoPS Raumplan
- HoPS Ausstattung
- Alternative

Alleinstellungsmerkmale

Physischer Raumplan

- Größere Datentiefe (mehr Informationen können zu einem Raum dargestellt werden)
- Interaktivität nicht gegeben
- keine Informationen zur Raumverfügbarkeit vorhanden

HoPS Raumplan

- Interaktivität nicht gegeben
- Übersichtlichkeit nicht gegeben
- keine aktuellen Informationen zur Raumverfügbarkeit vorhanden

HoPS Ausstattung

- Interaktivität nicht gegeben
- Übersichtlichkeit nicht gegeben
- Ausstattung in HoPS nicht detailliert genug
- Keine intuitive Suche/Filterung nach Ausstattung in HoPS

Alternative

<https://www.mazemap.com/>

Vorteil gegenüber mazemap:

- Spezialisierung auf Campus-Umfeld
- Integration Raumverfügbarkeit
- Integration Rauminformationen (Equipment)

Ähnliches System muss für den Campus für die Einbindung von Gebäudeplänen erstellt werden:
<https://davinci.stueber.de/floorplan.php>

Projektrisiken

- Projektumfang
- Änderungsmanagement
- Kommunikation
- Ressourcen und Projektteam / Beschaffung
- System-Architektur
- Anforderungen
- Produkt-Design
- Technisches
- Integration
- Externe Risiken

Projektrisiken

Projektumfang

- Projektumfang ungenau definiert
- Schleichende Erweiterungen die den Projektumfang sprengen

Änderungsmanagement

- Unnötige Refaktorisierung von Code
- Änderungen werden falsch priorisiert

Kommunikation

- Terminabstimmung nicht immer möglich
- Zu viel Kommunikation
- Zu wenig Kommunikation

Ressourcen und Projektteam / Beschaffung

- Erforderliche Daten sind nicht zugänglich
- Lernkurven zu hoch bzw. Erfahrung zu niedrig: neue Technologien erfordern zu viel Zeit zur Einarbeitung

System-Architektur

- Gewünschte Architektur des Systems kann nicht wie geplant umgesetzt werden
- Architektur wird zu unflexibel entworfen

Anforderungen

- Die Anforderungen wurden falsch/ungenau spezifiziert

Produkt-Design

- Das Design ist nicht intuitiv genug und wird nicht benutzt
- Die Darstellung ist nicht eindeutig. Anwender finden sich nicht zurecht
- Zu viele UI Komponenten
- Die Darstellung des Campus mit WebGL kann die Performance beeinträchtigen
- Anwendung ist entspricht nicht den W3C Accessibility Guidelines (WCAG)

Projektrisiken

Technisches

- Messung der Raumverfügbarkeit ist zu ungenau (Untersuchung der jeweiligen Möglichkeiten zur Bestimmung der Verfügbarkeit ist erforderlich)
- Eingesetzte Technologien sind nicht für den Anwendungsfall geeignet
- Technologische Funktionen sind nicht skalierbar oder performant genug
- Informationssicherheit wird nicht genug beachtet
- Technologische Funktionen des Systems werden nicht dokumentiert
- Technologische Funktionen des Systems sind nicht oder nur schwer erweiterbar
- Zu viele Ressourcen/Skripte müssen geladen werden

Integration

- Vorhandene Systeme (z.B. Transpondersystem) lassen sich nicht verwenden

Externe Risiken

- Modulziele werden verfehlt
- Produkte zum Testen der PoCs sind nicht verfügbar
- Die Hochschule verbietet das Projekt

Erfordernisse

Schablone

Als **spezifischer Benutzer** muss man **X wissen/verfügbar** haben, um **Y entscheiden/tun** zu können.

Die Stakeholder Studierende, wissenschaftliche Mitarbeiter, studentische Hilfskraft, Pförtner und Dozent werden gemäß der Stakeholderanalyse als spezifischer Benutzer “Benutzer des Systems” zusammengefasst:

Als **Benutzer des Systems** muss man **das Wissen über die Existenz verfügbar** haben, um **das System nutzen** zu können.

Als **Benutzer des Systems** muss man **eine Plattform geboten** bekommen, um **effizient verfügbare Räume finden** zu können.

Als **Benutzer des Systems** muss man **eine Plattform geboten** bekommen, um **effizient benötigtes Material finden** zu können.

Als **Benutzer des Systems** muss man **die Möglichkeit zu filtern verfügbar** haben, um **das benötigte Material innerhalb eines Raumes finden** zu können.

Als **Benutzer des Systems** muss man **die Möglichkeit zur virtuellen Belegung eines Raumes verfügbar** haben, um **die Verfügbarkeit eines Raumes im System beeinflussen** zu können.

Als **Dozent** muss man **einen Überblick über die Räume der Hochschule** verfügbar haben, um **bei kurzfristigen Raumänderungen einen verfügbaren Raum finden** zu können.

Anforderungen

- Funktionale Anforderungen
- Non-Funktionale Anforderungen

Funktionale Anforderungen

- [F10] Das System **muss** einem Benutzer die Möglichkeit bieten, die Verfügbarkeit eines Raumes einsehen zu können.
- [F20] Das System **muss** einem Benutzer die Möglichkeit bieten, das verfügbare Material/das Equipment innerhalb eines Raumes einsehen zu können.
- [F30] Das System **muss** einem Benutzer die Möglichkeit bieten, einen Raum anhand von Filterkriterien (Material/Equipment, Verfügbarkeit) zu suchen.
- [F40] Das System **muss** fähig sein, einen Link zu einem bestimmten Raum über NFC zur Verfügung zu stellen.
- [F50] Das System **muss** fähig sein, die Raumverfügbarkeit anhand des aktuellen Stundenplans festzulegen.
- [F60] Das System **muss** einem Benutzer die Möglichkeit bieten, virtuell einen Raum zu belegen.
- [F70] Das System **muss** fähig sein, eine angemeldete Raumverfügbarkeit für einen Raum automatisch nach einem Zeitraum wieder freizugeben.
- [F80] Das System **muss** fähig sein, eine Darstellung für jedes Display vor einem Raum einzeln zu generieren.
- [F90] Das System **muss** fähig sein, verschiedene Räume in einem Gebäudeplan zu markieren.
- [F100] Das System **muss** fähig sein, Tokens für die Authentifizierung von Nutzerbelegungen zu generieren.
- [F110] Das System **muss** fähig sein, Tokens für die Authentifizierung von Nutzerbelegungen zu validieren.
- [F120] Das System **muss** fähig sein, eine API zur Bereitstellung von Rauminformationsdaten zur Verfügung zu stellen.
- [F130] Das System **muss** fähig sein, asynchrone Nachrichten via PubSub Architektur zu versenden.
- [F140] Das System **sollte** einem Benutzer die Möglichkeit bieten, sich per Button vor einem Raum anzumelden.
- [F150] Das System **sollte** fähig sein, eine Fuzzy Search anhand von Benutzereingaben durchführen zu können.

- [F160] Das System **kann** fähig sein, einen Benutzer per Push-Benachrichtigung zu einer Eingabe aufzufordern.
- [F170] Das System **kann** fähig sein, einen Benutzer per E-Mail zu einer Eingabe aufzufordern.
- [F180] Das System **kann** fähig sein, einen Benutzer über die Verfügbarkeit eines Raumes per E-Mail zu informieren.
- [F190] Das System **kann** einem Benutzer die Möglichkeit bieten, sich für Updates zur Verfügbarkeit eines Raumes per Eingabe der E-Mail anzumelden.
- [F200] Das System **kann** fähig sein, den aktuellen Winkel der Sonneneinstrahlung in die Berechnung der Relevanz eines Raumes bei einer Suche mit einzubeziehen.
- [F210] Das System **kann** fähig sein, die dynamische Verfügbarkeit eines Raumes anhand von historischen Daten auszuwerten und somit Prognosen über die Verfügbarkeit in der Zukunft für verschiedene Wochentage zu treffen.

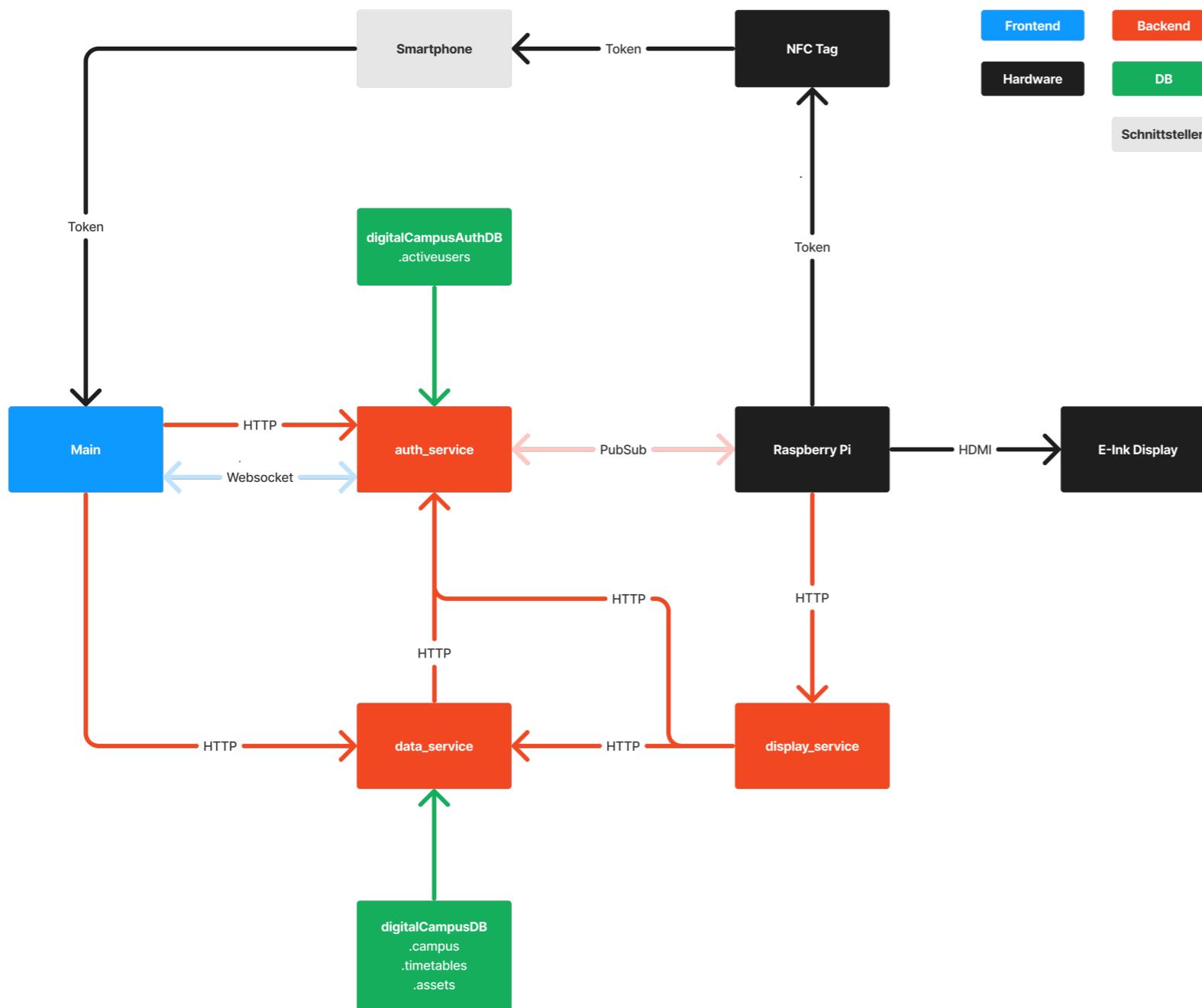
Non-Funktionale Anforderungen

- [O10] Die Datenschutzverordnung **muss** eingehalten werden.
- [O20] Die Korrektheit der Daten **muss** immer gewährleistet werden.
- [Q10] Die Suche nach Rauminformationen und Raumverfügbarkeiten **muss** so gestaltet sein, dass eine einfache und intuitive Bedienung möglich ist.
- [Q20] Die Oberfläche des Systems **sollte** so gestaltet sein, dass eine einfache und intuitive Bedienung von jedem möglich ist.

Anwendungsmodellierungen

- Architekturmodell
- Infrastruktur
- PubSub
- Sequenzdiagramm - Aufruf via NFC
- Sequenzdiagramm - Aufruf via Pi

Architekturmodell



26

Kommentar

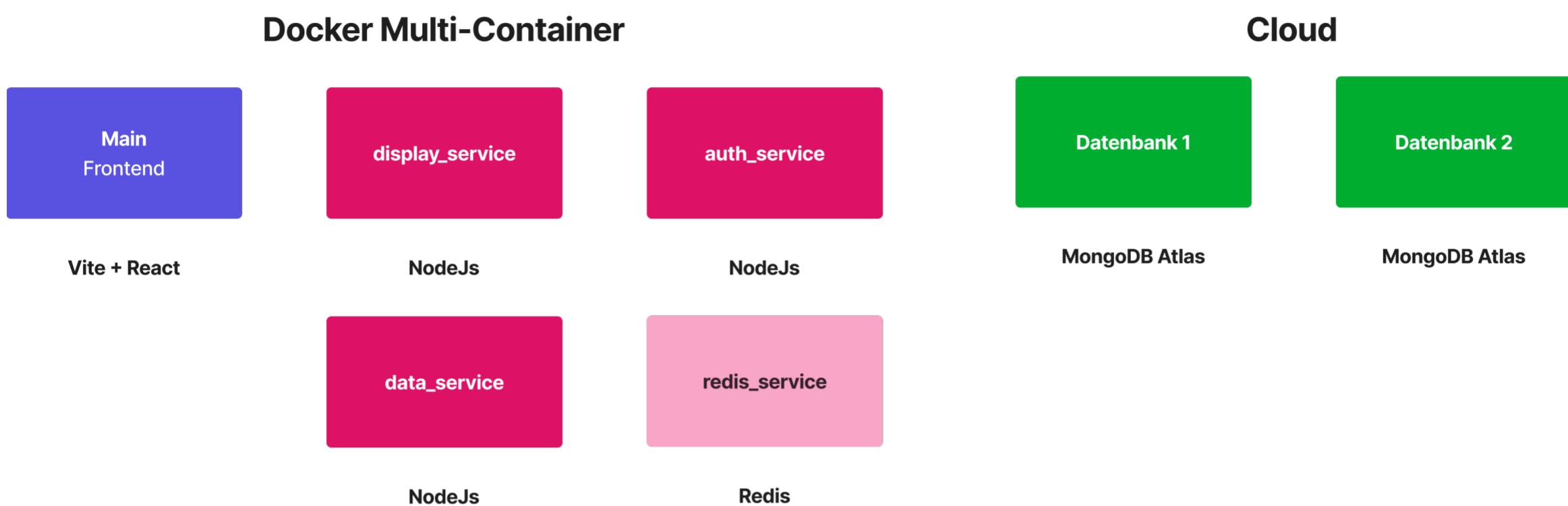
Das System wird in **vier verschiedene Bereiche** unterteilt.

1. Das **Main Frontend** agiert als zentrale Schnittstelle für Nutzerinteraktionen und bietet eine Darstellung des Campus mit Raumbelegungen und weiteren Rauminformationen an. Über dieses können Nutzer einzelne Räume betreten, verlassen und sich über die aktuelle Nutzeranzahl informieren.
2. Die in **Microservices** aufgeteilten Backend-Systeme stehen als Schnittstellen zur Datenbereitstellung, zur Generierung von Displayseiten der E-Ink Displays und zur Authentifizierung von Nutzern/Tokens zur Verfügung.
3. Die in der **MongoDB Atlas Cloud** aufgesetzten Datenbanken speichern zum einen die Raumdaten und zum anderen separat die zur Authentifizierung verwendeten Tokens.
4. Zur Hardware können die **Raspberry Pis** und die damit verbundenen **E-Ink Displays** und die **NFC-Geräte** gezählt werden.

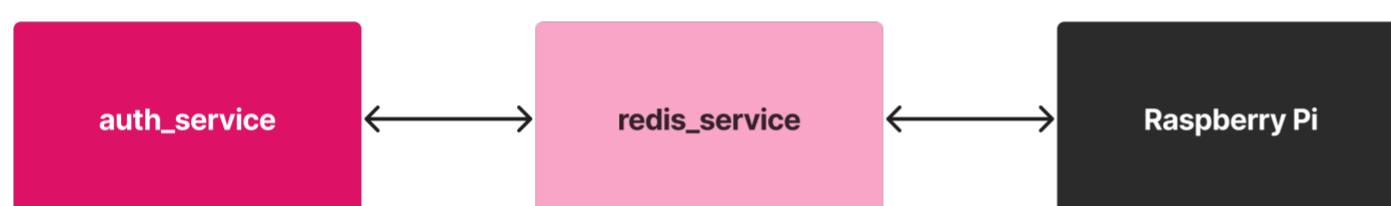
Die Kommunikation innerhalb des Systems erfolgt dabei über gängige **HTTP REST-Schnittstellen**, intern über einen **asynchronen PubSub-Broker** und über eine **WebSocketverbindung**.

Der Ablauf des System wurde im folgenden durch Sequenzdiagramme detaillierter dargestellt.

Infrastruktur



Publish-Subscribe Architektur



27

Kommentar

Die Infrastruktur des Systems wird in zwei Bereiche unterteilt:

Auf der einen Seite werden die Microservices und das Frontend über Docker Multi-Container und **docker compose**, vor allem zur vereinfachten Cross-Plattform Entwicklung, aufgesetzt.

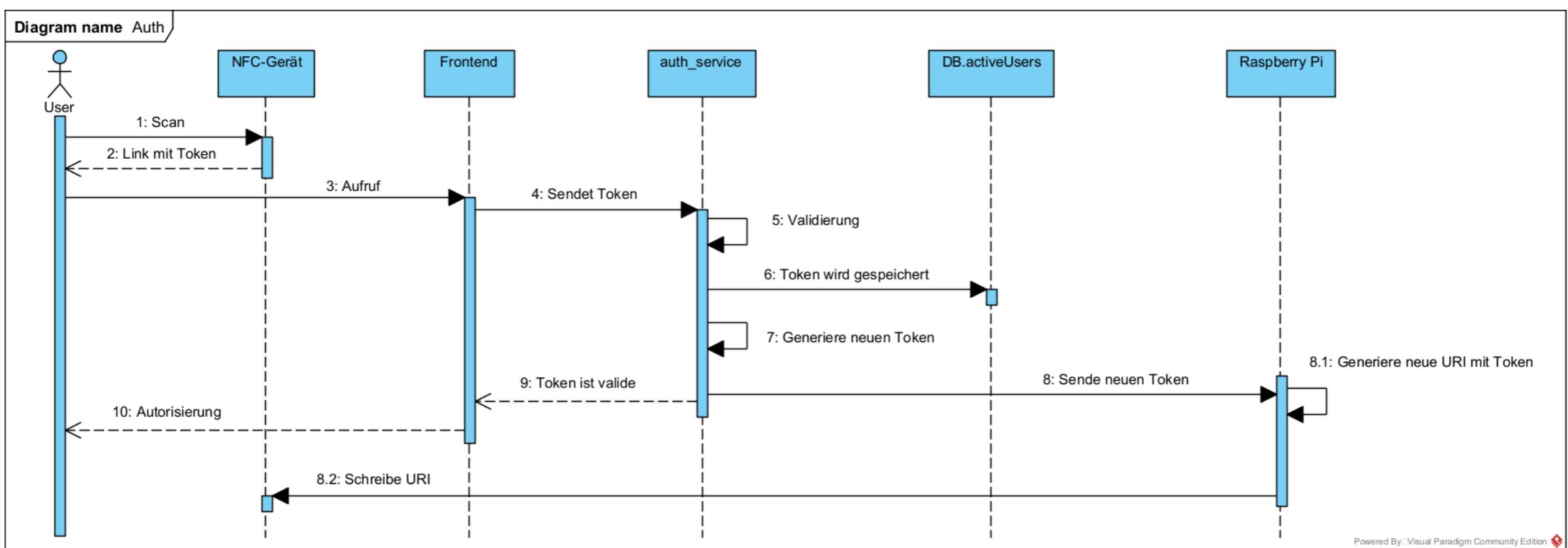
Die Datenbasis wurde über **MongoDB Atlas** in einem Cloud-Cluster integriert und kann dadurch über die Docker Container hinaus gewartet werden. Da keine sensiblen bzw. personenbezogenen Daten gespeichert werden, besteht keine Gefahr den **Datenschutz** zu verletzen. Bei einer Weiterentwicklung muss hier natürlich im Verlauf weiter genauer evaluiert werden.

Innerhalb der Architektur wird ein **Redis Publish-Subscribe Server** zur asynchronen Kommunikation zwischen den Microservices und dem Raspberry Pi verwendet.

Über diesen werden Nachrichten innerhalb eines gemeinsamen Kanals übertragen und empfangen. Hervorzuheben ist hierbei, dass jedes System, welches dem Nachrichtenkanal beigetreten ist, die Nachrichten empfangen und für sich entscheiden kann, ob Funktionen ausgeführt werden sollen.

- Der **Raspberry Pi** sendet ein Event in den Nachrichtenchannel, wenn dieser gestartet ist und einen neuen Token benötigt.
- Der **auth_service** sendet ein Event mit einem neuen Token in den Nachrichtenchannel, wenn das NFC-Gerät gescannt und ein Token von einem Nutzer verwendet wurde.
- Der **auth_service** sendet ein Event in den Nachrichtenchannel, wenn ein Nutzer einen Raum verlassen hat.

Sequenzdiagramm - Aufruf via NFC



28

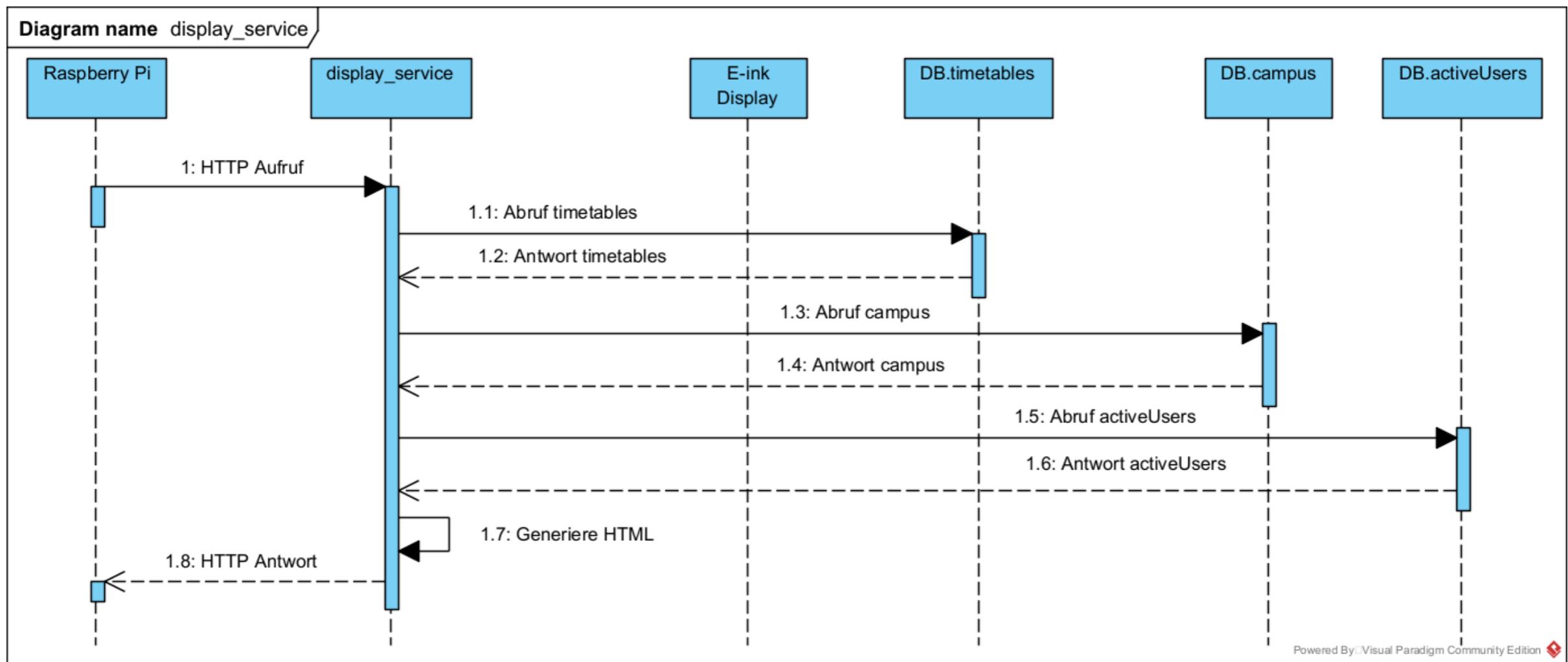
Kommentar

1. Scan: Ein Benutzer scannt den NFC-tag.
2. Link mit Token: Ein Link zur Raumbelegungskomponente des Frontends inklusive Token wird an das Endgerät des Benutzer gesendet.
3. Aufruf: Der Benutzer ruft das Frontend auf dem Endgerät auf.
4. Sendet Token: Das Frontend sendet via HTTP den Token an den auth_service.
5. Validierung: Der auth_service validiert das Token.

Bei validen Token

6. Token wird gespeichert: Der Token wird in der Datenbank gespeichert.
7. Generiere neuen Token: Generiert eine neues Token.
8. Sende neuen Token: Sendet neu generierten Token an den Raspberry Pi
 - 8.1 Generiere neue URI mit Token: Generiert neuen Link für den NFC-tag.
 - 8.2 Schreibe URI: Schreib den neu generierten Link auf den NFC-tag.
9. Token ist Valide: Das Frontend erhält eine HTTP Antwort mit code 200.
10. Autorisierung: Die Raumbelegungskomponente des Frontends wird auf dem Endgerät des Benutzers freigeschaltet.

Sequenzdiagramm - Aufruf via Pi



Kommentar

1. HTTP Aufruf: Der Raspberry Pi ruft die URL für das eigene E-Ink Display vom display_service auf.
- 1.1 Abruf timetables: Der display_service ruft die erforderlichen Stundenplandaten ab.
- 1.2 Antwort timetables: Die Daten aus der Datenbank können zwischengespeichert werden.
- 1.3 Abruf campus: Der display_service ruft die erforderlichen Campus/Rauminformationen ab.
- 1.4 Antwort campus: Die Daten aus der Datenbank können zwischengespeichert werden.
- 1.5 Abruf activeUsers: Der display_service ruft die erforderlichen aktiven Raumbelegungen ab.
- 1.6 Antwort activeUsers: Die Daten aus der Datenbank können zwischengespeichert werden.
- 1.7 Generiere HTML: Der display_service generiert mithilfe der serverseitigen Templatingsprache EJS eine für den Raum angepasste HTML Seite und generiert aus dieser ein Bild.
- 1.8 HTTP Antwort: Der Raspberry Pi erhält die Webseite als HTTP Response und kann diese auf dem E-Ink Display anzeigen.

Durchgeführte POCs

Frontend

- Digitale Darstellung Campus Gummersbach

Backend

- Seitengenerierung E-Ink Display

Hardware

- Automatisierter Seitenaufruf Raspberry Pi
- Integration ins Netzwerk
- Progammierbarer NFC Tag

Beschreibung

Es soll geprüft werden welche Darstellung sich am Besten für den Campus Gummersbach eignet.

- 2D Für die Darstellung soll eine Vektor-Grafik des Campus Gummersbach (.svg) verwendet werden.
- 3D Für die Darstellung soll ein 3D-Modell (.glTF) des Campus Gummersbach verwendet werden.
- 2.5D Für die Darstellung soll eine generische Lösung implementiert werden. Gebäude, Etagen und Räume des Campus Gummersbach werden als JSON oder XML Elementen in einem Schema definiert. Anhand von Koordinaten sollen digitale Abbildungen dieser generiert werden. Eine Skelettdarstellung ist ausreichend.

Exit-Kriterien

2D

1. Die Vektor-Grafik lässt sich auf einer Website darstellen.
2. Die Vektor-Grafik lässt sich interaktiv bedienen. (e.g. Räume werden visuell hervorgehoben)

3D

1. Ein 3D-Modell wird von einem anderen Team zur Verfügung gestellt werden.
2. Das 3D-Modell entspricht unseren Anforderungen.
3. Das 3D-Modell lässt sich auf einer Website darstellen
4. Das 3D-Modell lässt sich interaktiv bedienen. (e.g. Räume werden visuell hervorgehoben)

2.5D

1. Die Daten die für die Darstellung relevant sind, können in einem Schema beschrieben werden.
2. Überführung der Daten in XML oder JSON.
3. Eine Skellettdarstellung des Campus lässt sich aus den Daten generieren.
4. Die Skellettdarstellung lässt sich interaktiv bedienen. (e.g. Räume werden visuell hervorgehoben)

Fail-Kriterien

2D

- Die Vektor-Grafik entspricht nicht unseren Anforderungen.
- Die Vektor-Grafik lässt sich nicht einbinden.
- Die Vektor-Grafik lässt sich nicht interaktiv bedienen.

3D

- Ein 3D-Modell kann nicht zur Verfügung gestellt werden.
- Das erhaltene 3D-Modell entspricht nicht unseren Anforderungen.
- Die Erstellung eines 3D-Modell ist zu Aufwendig.
- Die eigene Erstellung des Modells ist zu Aufwendig.

2.5D

- Relevante Daten, um den Campus Gummersbach darzustellen, lassen sich nicht durch ein Schema beschreiben.
- Der Campus Gummersbach lässt sich durch ein Schema beschreiben. Die Pflege der Daten ist allerdings zu aufwendig.

Fallbacks

Die Darstellungsmöglichkeiten 2.5D, 2D und 3D werden in dieser Reihenfolge durchgeführt und dienen jeweils als Fallback für den Vorherigen.

Durchführung

2.5D

1. Der Maßstab wird durch den Gebäudeplan ermittelt.
2. Eine vereinfachte Darstellung des Campus Gummersbach wird als Vektor-Grafik erstellt.
3. **[FAIL]** Eine Skelletdarstellung lässt sich nicht durch XML realisieren.
Daten können nicht der Vektor-Grafik entnommen werden.
4. Aus der Vektor-Grafik wird ein JSON-Objekt erzeugt. JSON Schema wird erstellt.
5. **[EXIT]** Eine Skelletdarstellung lässt sich durch das JSON realisieren.

Verwendete Packages

three
react-three/fiber
react-three/drei

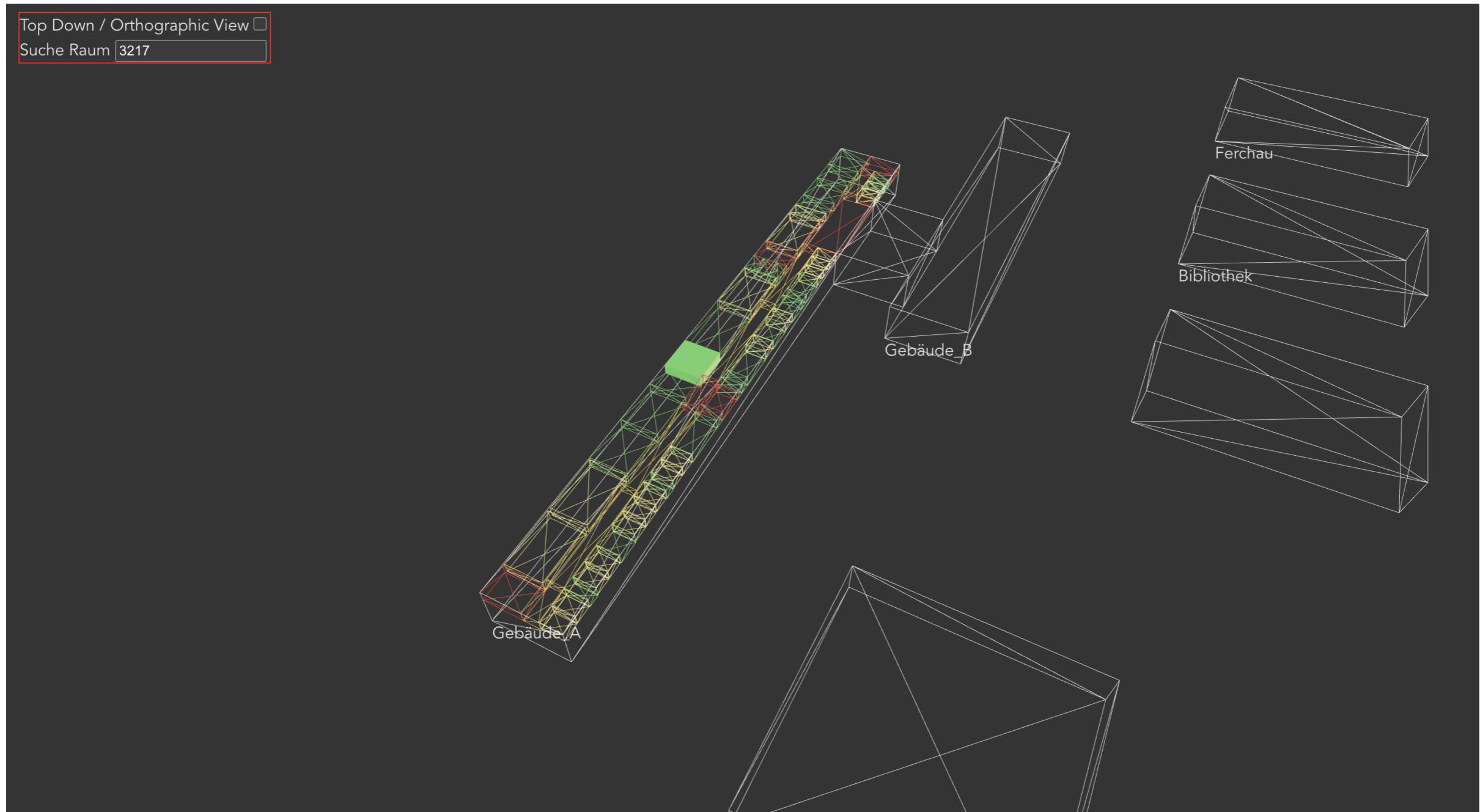
2D

1. **[EXIT]** Eine 2D Darstellung lässt sich durch die OrthographicCamera in react-three/drei simulieren.

3D

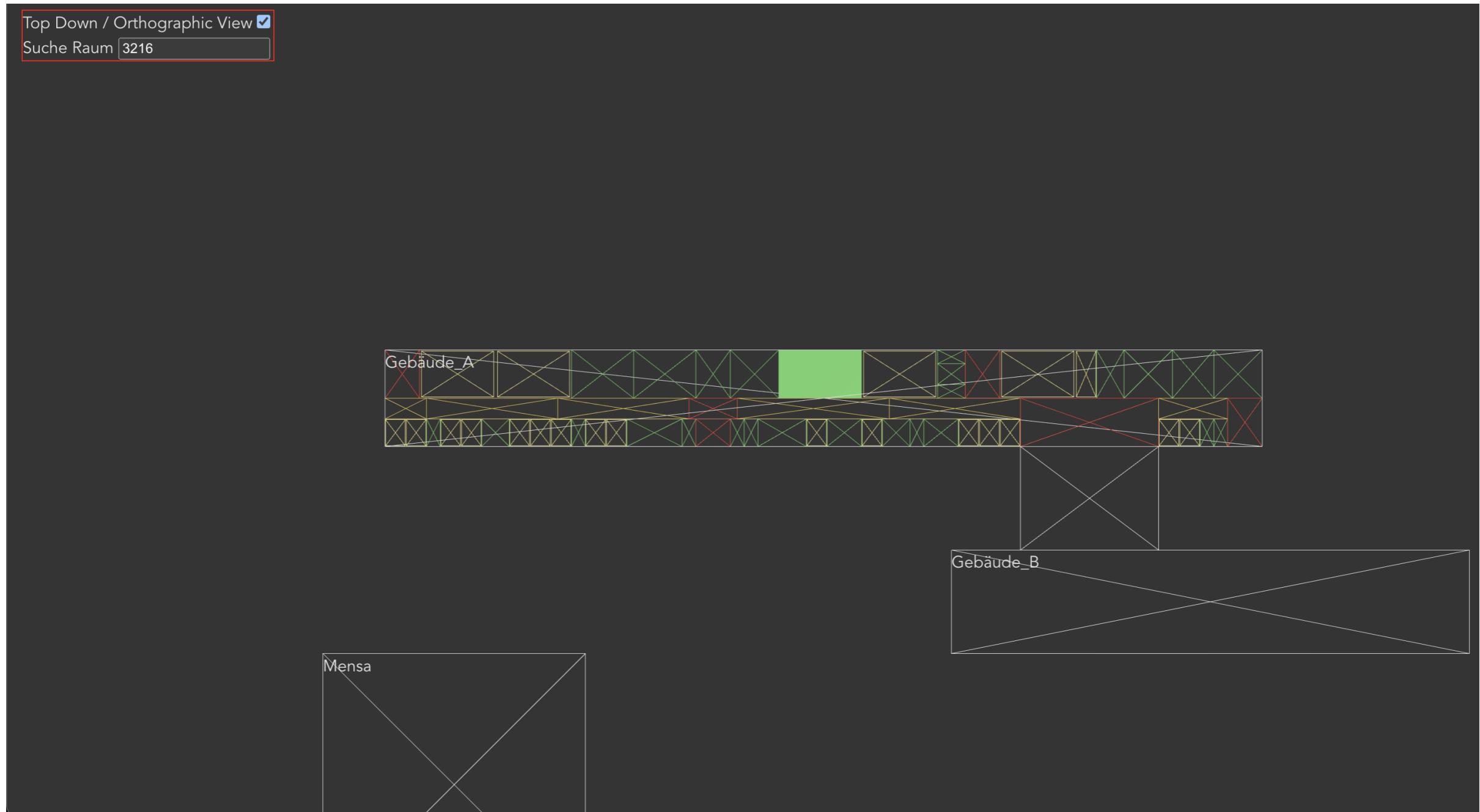
1. Modellierung einer reduzierten Darstellung eines Raumes. (Export als .gltf)
2. **[EXIT]** Modell erfolgreich in Website importiert.

Digitale Darstellung Campus Gummersbach



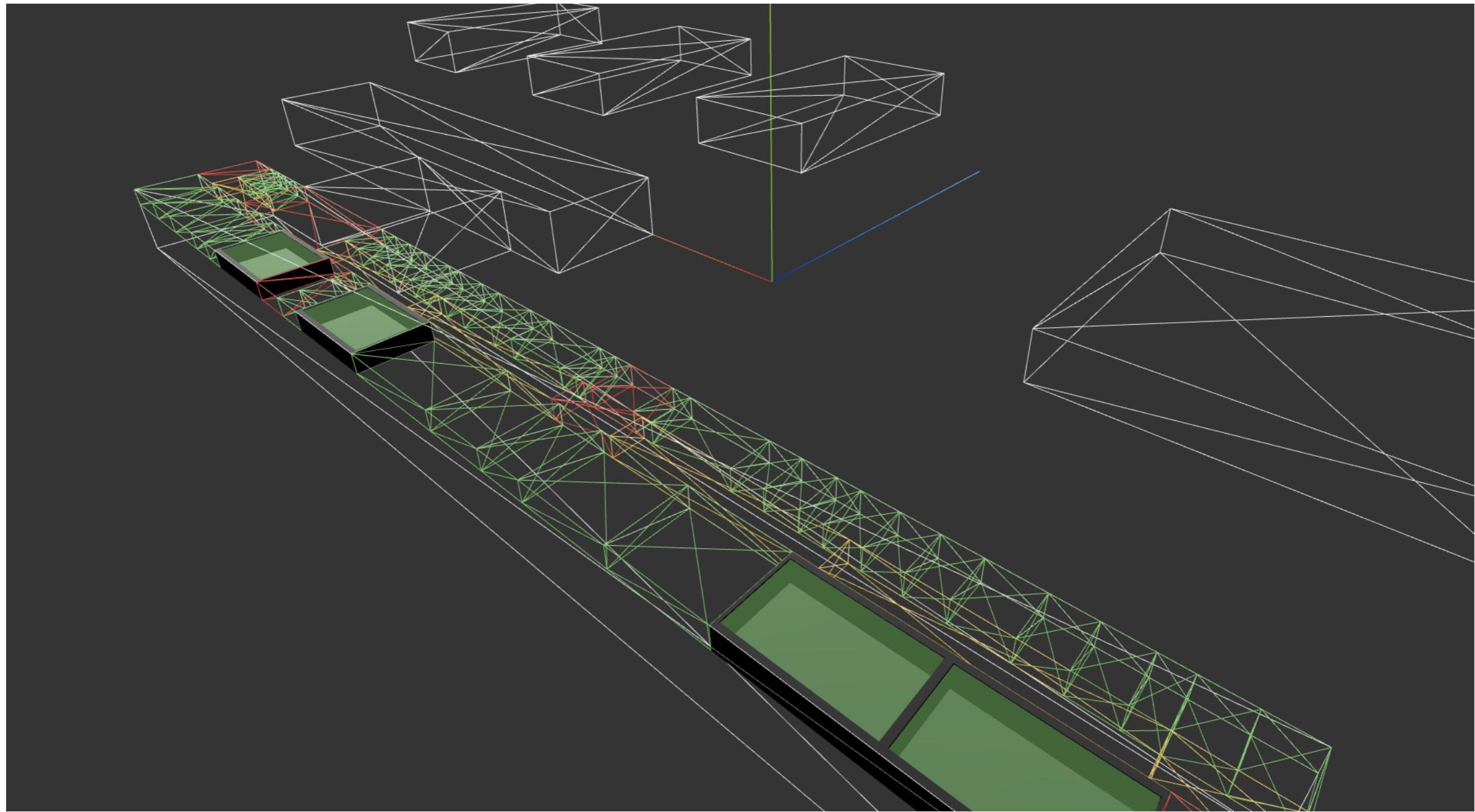
2.5D - Eine Skelletdarstellung des Campus lässt sich in die Szene integrieren.

Digitale Darstellung Campus Gummersbach



2D - Es lässt sich die Orthografische Ansicht wechseln.

Digitale Darstellung Campus Gummersbach



3D - Es lassen sich 3D Modelle nachträglich in die Szene einfügen.

Beschreibung

Ein E-Ink Display soll vor einem Raum platziert werden, Daten über einen PubSub-Channel aus dem Backend erhalten und diese verarbeiten.

Der Fokus liegt hierbei auf dem Backend welches dynamisch eine einfache, an ein E-Ink Display angepasste, Webseite generiert und diese bereitstellt.

Exit-Kriterien

1. Das Backend kann aktuelle Daten über den Raumzustand empfangen.
2. Eine Seite kann dynamisch für einen Raum generiert werden.
3. Bei Aktualisierungen zur z.B. Raumbelegung kann das Backend dies verarbeiten und eine neue Seite bereitstellen. (Relevanz für den funktionalen Prototypen)

Fail-Kriterien

- Die Datenbasis gibt nicht die erforderlichen Informationen zu einem Raum her.
- Die Seite ist nicht auf die Darstellung auf einem E-Ink Display angepasst.

Fallbacks

Ein Display sollte selber Informationen darstellen, wenn z.B. die generierte Seite nicht erreicht werden konnte.

Durchführung

Im ersten Schritt der Durchführung wurde die Darstellung auf einem E-Ink / e-Paper Display recherchiert.

Hervorzuheben sind hierbei die **Limitierungen der Technologie** und die damit verbundenen Limitierungen in der Erstellung des Designs:

- **Kontrast** ist wichtig (Begrenzte Farbwiedergabe)
- **Keine Animationen** (Ghosting und Smearing sollen vermieden werden)
- **Fokus auf HTML** und kein Nachladen von Javascript (Lightweight)

Entwicklung Wireframe



[4] Go Make Things | Building websites that work on an e-ink Kindle, Aufgerufen am October 15 2022, von <https://gomakethings.com/building-websites-that-work-on-an-e-ink-kindle/>

[5] Reddit | Tips to make web browsing much better on E-Ink , Aufgerufen am October 15 2022, von https://www.reddit.com/r/eink/comments/lkc0ea/tip_to_make_web_browsing_much_better_on_eink/

Durchführung

Für die Generierung der Seite wird im Backend, genauer auf dem display_service, auf die **Templating Sprache EJS** gesetzt. Mit dieser kann die Seite **serverseitig gerendert** werden und der Client erhält die passende HTML-Datei.

- Die Seite kann **erfolgreich** auf dem Backend **generiert** werden. Die Exit Kriterien wurden erfüllt.
- Die **Darstellung** wurde auf ein E-Ink Display **angepasst**, konnte jedoch aufgrund der Verfügbarkeit noch nicht auf einem Display getestet werden.
- Die **dynamische Aktualisierung** von Seiten wird für den funktionalen Prototypen implementiert, wenn auch die erforderlichen Services für Audit 4 verfügbar sind.

Stundenplan Templating

```
● ● ●  
1 <table>  
2   <thead>  
3     <tr>  
4       <th class="time-col">Uhrzeit</th>  
5       <th>Montag</th>  
6       <th>Dienstag</th>  
7       <th>Mittwoch</th>  
8       <th>Donnerstag</th>  
9       <th>Freitag</th>  
10    </tr>  
11  </thead>  
12  <tbody>  
13    <% for (let i = 8; i < 21; i++) { %>  
14    <% let hour = i.toString().padStart(2, "0") %>  
15    <tr>  
16      <td class="time-col"><%= hour %>:00</td>  
17      <td><%= monday.find(t => t.time === `${hour}:00`)??.name || " " %></td>  
18      <td><%= tuesday.find(t => t.time === `${hour}:00`)??.name || " " %></td>  
19      <td><%= wednesday.find(t => t.time === `${hour}:00`)??.name || " " %></td>  
20      <td><%= thursday.find(t => t.time === `${hour}:00`)??.name || " " %></td>  
21      <td><%= friday.find(t => t.time === `${hour}:00`)??.name || " " %></td>  
22    </tr>  
23    <% } %>  
24  </tbody>  
25 </table>
```

Beschreibung

Der Raspberry Pi 3 (Rpi3) soll eine Website auf einem E-ink Display aufrufen. Der Aufruf soll nach Erhalten eines Events in nodejs geschehen, d. h. der Aufruf muss innerhalb von nodejs stattfinden.

Exit-Kriterien

- Die Seite wird aufgerufen und auf dem Display angezeigt.

Fail-Kriterien

- Der Aufruf der Seite funktioniert nicht
- Die Seite wird aufgerufen, aber nicht auf dem Display gezeigt
- Das Programm funktioniert, nur wenn es lokal aufgerufen wird, nicht via ssh

Fallbacks

- Falls keine Möglichkeit besteht, die Seite via nodejs zu öffnen, muss ein shell Skript geschrieben werden, welches diese Aufgabe übernimmt.
- Das Programm muss mittels angeschlossener Tastatur auf dem pi gestartet werden

Durchführung

Eine Library wurde gefunden, welche die benötigte Funktionalität anbietet: [open-npm](#)

1. Der Rpi3 wurde via HDMI mit einem Monitor verbunden (Wir warten auf die Lieferung eines E-ink Displays)
2. Es wurde eine Verbindung zu dem Rpi3 via ssh aufgebaut.
3. Eine simple Integration der library wurde vorgenommen und getestet.
4. **[FAIL]** Kriterium 1 tritt ein, die Seite wird nicht geöffnet. Die console.log Statements werden allerdings ausgeführt.
5. Derselbe Code wurde auf lokal auf der Maschine ausgeführt und funktioniert vollständig (inklusive Aufrufen der Website)
6. **[EXIT]** Eine Tastatur wurde an den Rpi3 angeschlossen, um die Datei lokal ohne ssh auszuführen. Dies ist erfolgreich und das Kriterium tritt ein.

```
const open = require('open');
const domain = 'https://www.th-koeln.de/';
(async function() {
  console.log("pi code in progress");
  await open(domain);
  console.log("pi code done");
})();
```

Es wird angenommen, dass die Ausführung via ssh nicht funktioniert wegen fehlender X11 forwarding Konfiguration bei der ssh Verbindung. Der Fallback, das Programm mittel Tastatur auszuführen, ist nach aktuellem Stand ausreichend, alternativ kann ein cronjob auf dem PI eingerichtet werden welcher das node Programm automatisch beim hochfahren des PI's starten.

[6] [open | Browser automation](#), Aufgerufen am December 04, 2022, von <https://github.com/sindresorhus/open>

Kommentar

Da die "open" Library keine Möglichkeit bietet eine aufgerufene Seite neu zu laden, wurde diese durch Puppeteer ersetzt. Puppeteer ist eine Browser Automatisierungs Software und wird auf dem Raspberry Pi im headful Modus gestartet um die Stundenpläne abzubilden.

[7] [puppeteer | Browser automation](#), Aufgerufen am February 21, 2023, von <https://github.com/puppeteer/puppeteer/tree/main>

Beschreibung

Ein Raspberry Pi 3 (Rpi3) soll in das System integriert werden und über ein lokales Netzwerk von den anderen Services ansprechbar sein.

Exit-Kriterien

- Der Rpi3 kann in einer lokal laufenden nodejs Umgebung, Nachrichten empfangen, welche über den Redis-PubSub Messenger versendet werden und selber Nachrichten verschicken.

Fail-Kriterien

- Der Rpi3 lässt sich nicht integrieren

Fallbacks

Teile des Systems müssen deployed werden, damit der Rpi3 über das Internet Zugriff hat.

Durchführung

1. Das aktuelle Standard Raspberry Pi OS wurde auf dem Rpi3 installiert.
2. Nodejs wurde auf dem Rpi3 installiert
3. "Hello World" Programm um node zu testen
4. Einfügen der "redisPubSub.js" + Konfiguration um Verbindung zu Docker Netzwerk herzustellen.
5. **[EXIT]** Ausführen der redisPubSub.js um festzustellen ob die Verbindung klappt.

Beschreibung

Der Raspberry Pi soll ein einen Nachricht (Link + Token) an ein NFC board senden, damit Geräte mit NFC-lese Funktion diese Nachricht via Scannen des NFC-Boards erhalten können.

Exit-Kriterien

- Die Nachricht wird auf das NFC-Board übertragen.
- Ein NFC-lese Gerät scannt das Board und liest die übertragene Nachricht.

Fail-Kriterien

- Die Übertragung der Nachricht an das NFC Board funktioniert nicht.
- Ein NFC-lese Gerät scannt das Board und bekommt eine andere Nachricht.

Fallbacks

- Ein QR-code mit der selben Nachricht wird angeboten.
- Eine URI wird auf dem E-Ink Display angezeigt.
- Funktionalität für ein NFC-Board wird implementiert, bei der Durchführung werden NFC-Tags manuell beschrieben.

Durchführung

1. Installieren des aktuellsten OS auf dem Pi (Debian - Bullseye)
2. Update der Software (via apt-get update / apt-get upgrade)
3. Aufstecken der NFC-HAT Hardware auf dem Pi.
4. Installieren der notwendigen Software auf dem Pi.
5. [FAIL] Aufführung der Software => Der Pi kann keine Verbindung zum HAT aufbauen
6. Ältere Version der Software installieren
7. [FAIL] Aufführung der Software => Der Pi kann keine Verbindung zum HAT aufbauen
8. Alte OS Version (Debian - Wheezy) auf zweitem Raspberry Pi (Model 1B) installieren
9. Software + Hardware auf zweitem Pi installieren
10. [FAIL] Aufführung der Software => Der Pi kann keine Verbindung zum HAT aufbauen
11. Alte OS Version (Debian - Wheezy) auf drittem Raspberry Pi (Model 1B) installieren
12. Software + Hardware auf drittem Pi installieren
13. [FAIL] Aufführung der Software => Der Pi kann keine Verbindung zum HAT aufbauen

Fazit

Es wurde sich mit der Architektur des NFC-HAT auseinandergesetzt. Es stellt sich heraus, dass der Daemon der NFC-Software keine Verbindung mit dem Databus des NFC-HAT aufbauen kann. Es wird von einem Hardware Fehler ausgegangen.

Der programmierbare NFC-HAT Aufsatzt wird nun durch NFC-Tags ersetzt, welche über das Handy beschrieben werden.

Änderungen im Projekt

- NFC-HAT
- E-Ink Display

Änderungen im Projekt

NFC-HAT

Da der NFC-HAT nicht eingebunden werden konnte, ist im Verlauf der Projektdurchführung folgende Änderung vorgenommen worden:

- NFC-Tags für Räume müssen manuell mit dem Handy beschrieben werden

Alle Event-Abläufe finden trotzdem wie modelliert statt, der Pi empfängt alle benötigten Daten um diese an einen betriebsfähigen NFC-HAT anbinden zu können.

E-Ink Display

Da das E-Ink Display nicht eingebunden werden konnte, ist im Verlauf der Projektdurchführung folgende Änderung vorgenommen worden:

- Die auf dem E-Ink Display darzustellenden Informationen werden auf einem handelsüblichen Display mit HDMI Anschluss dargestellt

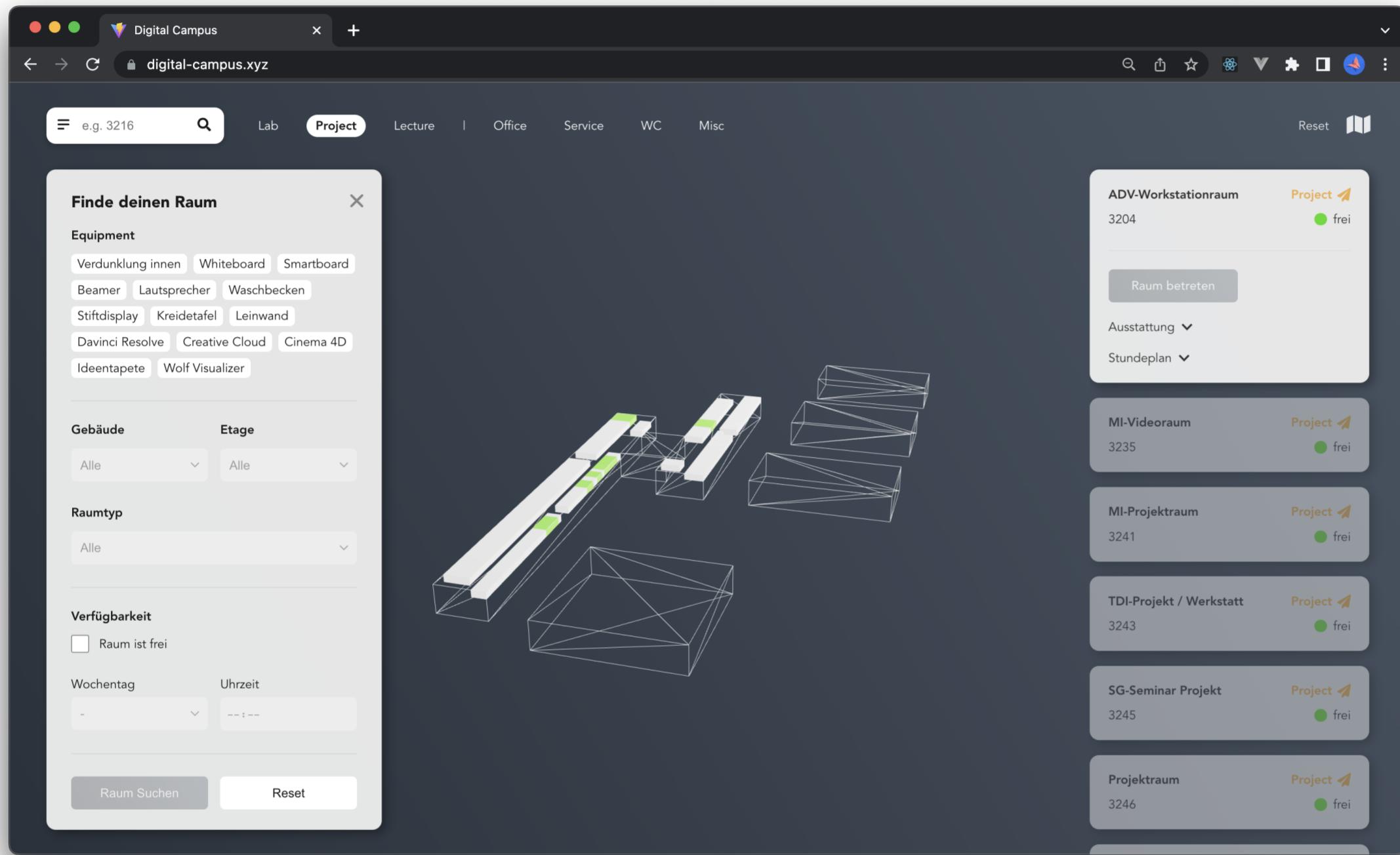
Alle E-Ink spezifischen Limitierungen wurden bei der Entwicklung der darzustellenden Informationen beachtet, sodass eine Umstellung auf ein E-Ink Display mit HDMI Anschluss keinen erhöhten Mehraufwand erfordert.

Projektergebnisse (Auszug)

- Anwendung
- Raumsuche - Formular
- Raum betreten / verlassen
- Generierung Display

Projektergebnisse (Auszug)

Anwendung



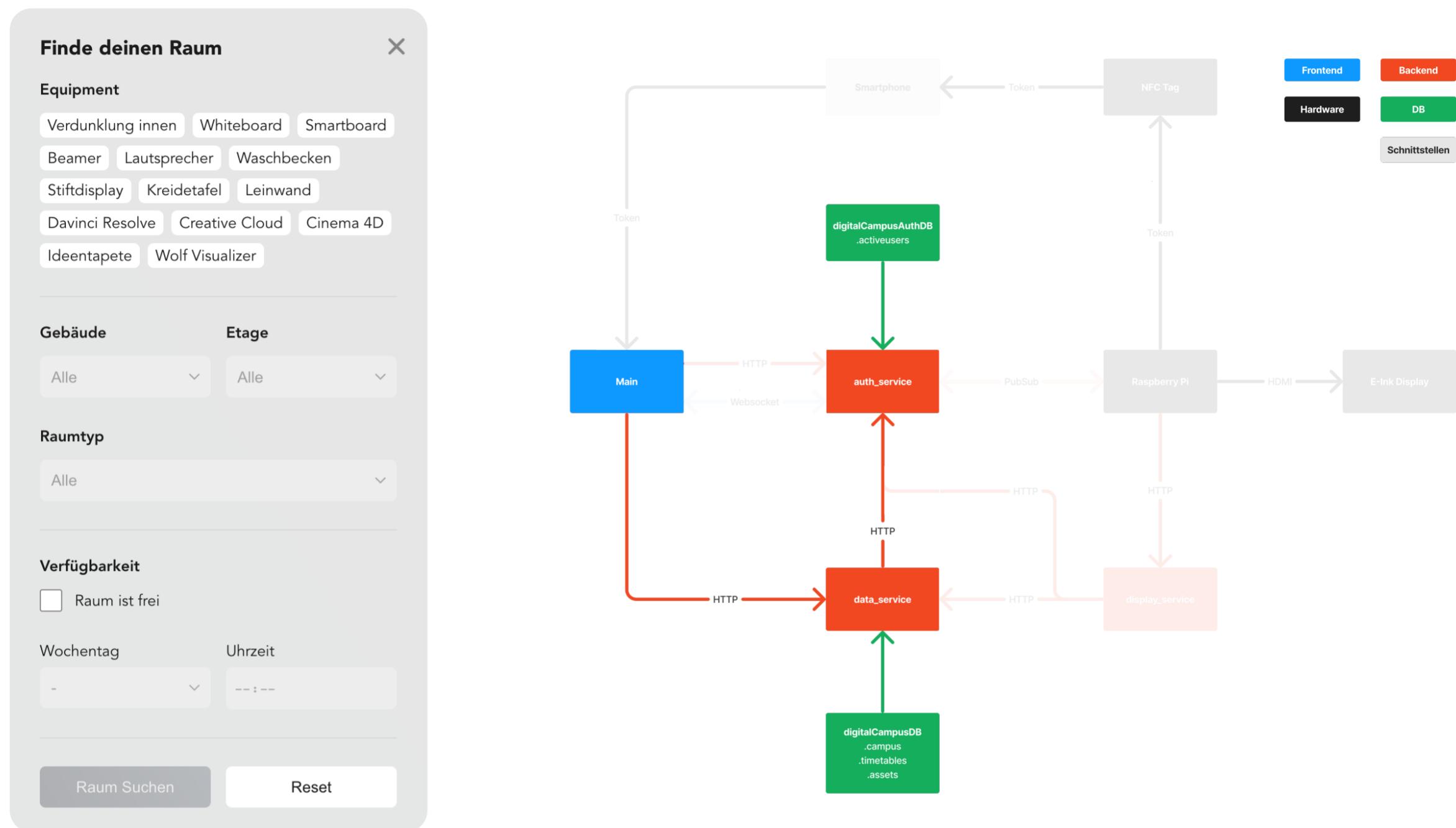
Kommentar

Der Funktionsumfang der Anwendung beinhaltet folgende wesentliche Bestandteile:

- Formular Suche nach Equipment, Gebäude, Etage, Raumtyp, Verfügbarkeit, Wochentag, Uhrzeit
- Reactive Suche nach Raumtyp, Raumnummer, Standort
- Betreten eines Raumes inkl. persistente Speicherung dessen im Browser eines Nutzers
- Digitale Abbildung des Campus Gummersbach
- Persistente Datenspeicherung

Projektergebnisse (Auszug)

Raumsuche - Formular



50

Kommentar

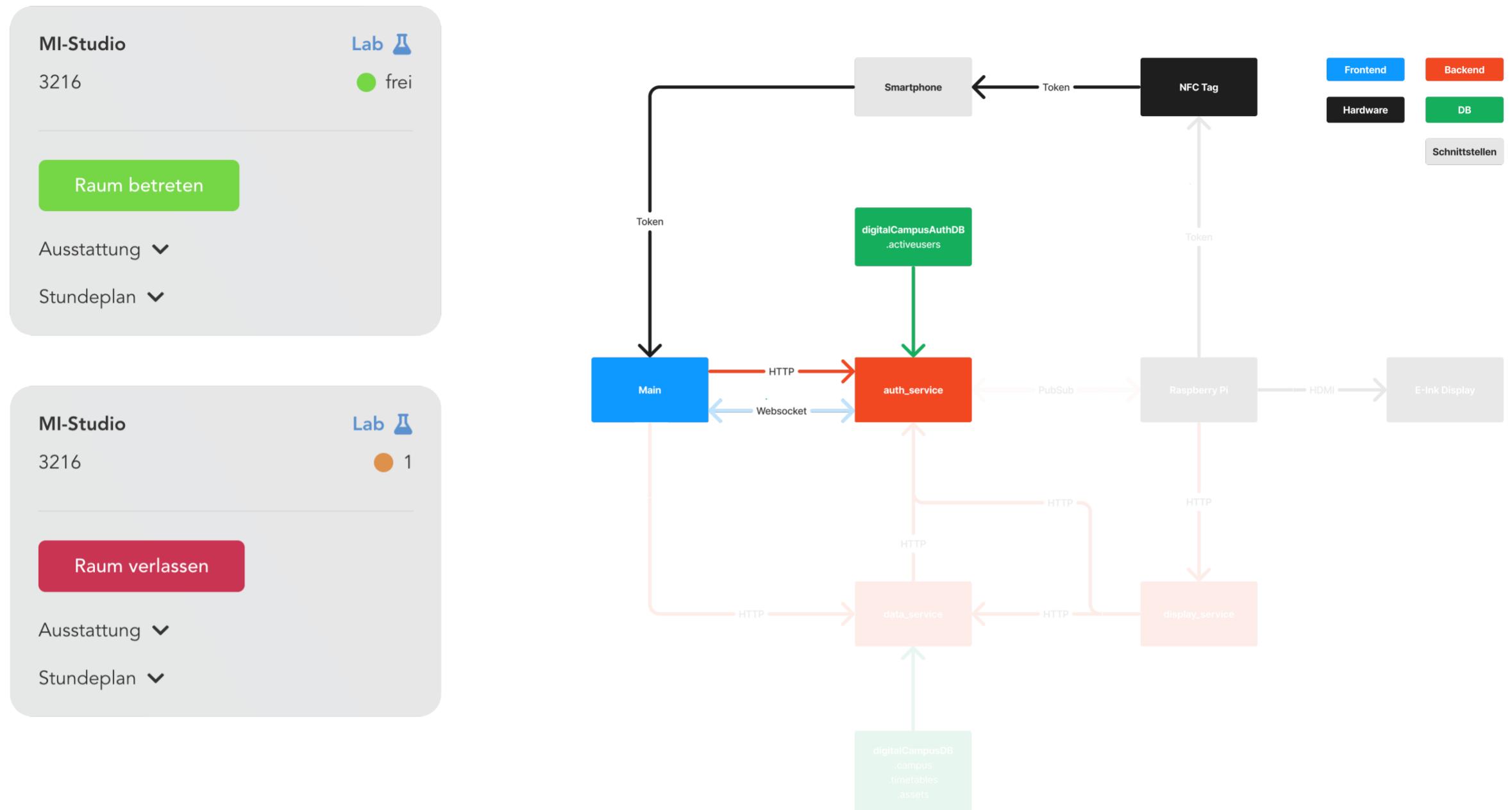
Bei der Suche nach einem Raum kann dabei innerhalb eines dynamischen Formulars nach den gewünschten Kriterien gefiltert werden:

- das benötigte Equipment
- das Gebäude in dem gesucht werden soll
- die Etage in der gesucht werden soll
- der Raumtyp nach welchem gesucht werden soll
- die generelle Verfügbarkeit eines Raumes (ob eine Vorlesung in diesem Raum statt findet oder ob andere Nutzer bereits über die Anwendung einem Raum beigetreten sind) nach Wochentag und Uhrzeit

Das Frontend sendet beim Absenden des Formulars einen Request an den **data_service**. Dieser fragt die Datenbank nach den benötigten Campusdaten und den **auth_service** nach der aktuellen dynamischen Nutzerbelegung eines Raumes ab. Die Daten werden innerhalb des Services aufbereitet und den gesuchten Kriterien nach gefiltert und an das Frontend zurückgesendet. Dieses kann dem Nutzer alle den gesetzten Filtern nach passenden Räume anzeigen.

Projektergebnisse (Auszug)

Raum betreten / verlassen



51

Kommentar

Der beschriebene NFC-Tag kann von Nutzern gescannt und anschließend die URL mit dem integrierten Token für einen Raum im Browser geöffnet werden. Das Frontend öffnet die richtige Raumseite und ermöglicht es, den Raum zu betreten. Ein Nutzer bekommt dabei nichts von dem Tokenaustausch und der Authentifizierung mit.

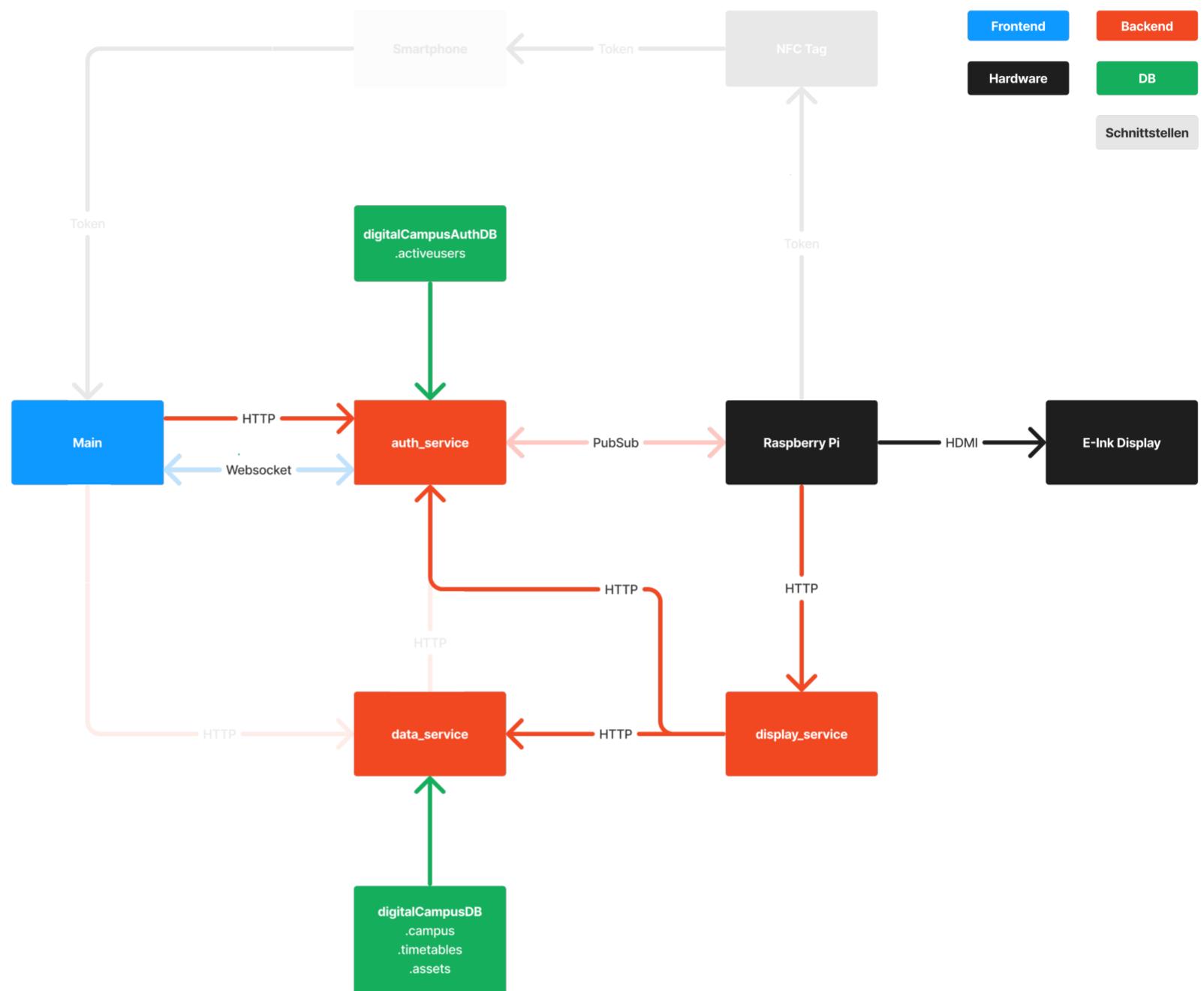
Nach dem Betreten eines Raumes kann ein Nutzer den Raum auch wieder verlassen. Beim Scan eines NFC-Tags eines anderen Raumer, erhält der Nutzer die Möglichkeit den neuen Raum direkt zu betreten. Der alte Raum wird dabei automatisch von der Anwendung verlassen.

Projektergebnisse (Auszug)

Generierung Display

| R 3.216 | | | | | | Labor |
|-----------|----------------------|-------------------------|-----------------|-------------------------|--------------------------|-------|
| MI-Studio | | | | | | |
| Uhrzeit | Montag | Dienstag | Mittwoch | Donnerstag | Freitag | |
| 08:00 | | | | | | |
| 09:00 | | | | Studioproduktion - U... | | |
| 10:00 | Praxisprojektseminar | | | Studioproduktion - U... | Storytelling und narr... | |
| 11:00 | Grundlagen des Web | Praxisprojektseminar | Projekt I & III | Studioproduktion - U... | Storytelling und narr... | |
| 12:00 | Grundlagen des Web | Praxisprojektseminar | Projekt I & III | Studioproduktion - U... | Audiovisuelles Medie... | |
| 13:00 | Entwicklungsprojekt | Einführung in die Me... | Projekt I & III | Audiovisuelles Medie... | Audiovisuelles Medie... | |
| 14:00 | Entwicklungsprojekt | Einführung in die Me... | Projekt I & III | Audiovisuelles Medie... | Audiovisuelles Medie... | |
| 15:00 | Entwicklungsprojekt | Einführung in die Me... | Projekt I & III | Audiovisuelles Medie... | Audiovisuelles Medie... | |
| 16:00 | | Einführung in die Me... | Projekt I & III | Audiovisuelles Medie... | Audiovisuelles Medie... | |
| 17:00 | | | | Audiovisuelles Medie... | | |
| 18:00 | | | | Audiovisuelles Medie... | | |
| 19:00 | | | | Audiovisuelles Medie... | | |
| 20:00 | | | | | | |

Keine Personen im Raum



52

Kommentar

Die für das E-Ink Display generierte Seite wird vom Display Service via HTML-template erstellt. Hierfür sendet der display_service einen Request an den data_service und den auth_service ab um sich alle für die Generierung notwendigen Daten (Stundenplan und aktive Nutzer in einem Raum) zu holen.

Der display_service bietet zwei unterschiedliche Endpunkte an:

- /timetable/campus/{name}/rooms/{number}
- /display/campus/{name}/rooms/{number}

Der "display" Endpunkt enthaelt neben dem Stundenplan noch Informationen zum Raumnamen,-typ , -nummer und die Anzahl der aktiven Nutzer im Raum.

Der generierte Stundenplan wird standartmäßig im .bmp Format als Bild gesendet oder mit dem Query Parameter "?type=html" als HTML .

Prozessassessment und Fazit

- Reflexion
- Erkenntnisse Technologien

Prozessassessment und Fazit

In dieser Reflexion wird die Gruppenarbeit für das Projekt "Digital Campus" betrachtet, welches im Rahmen des Entwicklungsprojekts 22/23 entwickelt wurde. Das Ziel dieser Reflexion ist es, den Ablauf des Projektes anhand der ursprünglichen Zielsetzung zu bewerten und Schlüsse über unsere Arbeit zu ziehen.

Als ursprüngliches Ziel des Projektes stand eine digitale Abbildung des Campus Gummersbach, welche zur Orientierung und Wegfindung dienen sollte. Da die Einbindung des Leitthemas sich innerhalb dieser initialen Zielsetzung für schwierig erwies, musste der Fokus des Projektes neu gesetzt werden.

Um dem Leitthema nun zu entsprechen, wurde die ursprüngliche Zielsetzung um die Themen Synchronisation und Device Shifting ergänzt. Hierzu soll die Verfügbarkeit eines Raumes anhand von Sensoren und dem jeweiligen Stundenplan ermittelt werden. Die Verfügbarkeit soll anschließend auf einem E-Ink Display, am Raum angezeigt werden. Verbunden mit einem Raspberry Pi und einem programmierbaren NFC-Tag, soll man anschließend weiterführende Informationen zu dem jeweiligen Raum in einer externen Anwendung anzeigen lassen. Zusätzlich bietet die Anwendung die Möglichkeit, eine Raumsuche anhand von verschiedenen Parametern wie Verfügbarkeit, Ausstattung, Standort, Raumtyp und Raumnummer durchzuführen.

Nach Festlegen der Anforderungen und der Systemarchitektur wurden die einzelnen Komponenten entwickelt und fertiggestellt. Nach Fertigstellung der Softwarekomponenten und Schnittstellen wurde getestet, den NFC-HAT und das E-Ink Display auf den schon im System angebundenen Raspberry PI zu integrieren. Aufgrund diverser Schwierigkeiten bei der Integration wurde der NFC-HAT durch einen manuell beschreibbaren NFC-Tag und das E-Ink Display durch ein HDMI-Display ersetzt.

Das Endergebnis der Entwicklung ist ein in Microservices aufgeteiltes System, welches alle benötigten Funktionalitäten gemäß der Zielsetzung des Projektes beinhaltet.

Prozessassessment und Fazit

Dank guter Organisation, regelmäßigen Meetings und Einteilung der Arbeitsbereiche innerhalb des Teams erwies sich die Umsetzung des Projektes stets angenehm. Aufgaben wurden innerhalb von Sprints abgearbeitet und nach Fertigstellung den anderen Mitgliedern vorgestellt. Dies hat bewirkt, dass trotz Aufgabenteilung alle Mitglieder über den aktuellen Stand verschiedener Bereiche inhaltlich sowie organisatorisch informiert waren. Durch einen einheitlich festgelegten Git-Workflow konnten Änderungen an der Codebasis während der Entwicklung getestet und den anderen Mitgliedern zur Verfügung gestellt werden. Durch regelmäßige Teilnahme an den Open-Spaces konnten Probleme und Fragestellungen, die sich während der Bearbeitung des Projektes ergeben haben, als Feedback im weiteren Verlauf umgesetzt werden.

Aufgrund der Verschiebung des Leitthemas, den zu integrierenden IoT-Geräten (E-Ink Display, Raspberry Pi, programmierbarer NFC-Tag) sowie dem unterschätzten Aufwand, hat sich der Umfang des Projektes schlechend erweitert. Ebenfalls wurden die benötigten Geräte verspätet bestellt, wodurch die unvorhergesehenen Probleme nicht früh genug abgefangen werden konnten und keine Alternativen bestellt wurden. Obwohl die regelmäßigen Treffen einen guten Überblick über alle Bereiche des Projektes gegeben haben, wurden auch diese von allen Mitgliedern diskutiert. Dies hatte zur Folge, dass viel Zeit mit nicht zielführenden Diskussionen verschwendet haben. Ein präziseres Datenbankmodell hätte außerdem die Integration von vielen Funktionalitäten erleichtert und würde den Aufwand bei zukünftigen Änderungen verringern.

Insgesamt sind wir mit dem Ergebnis des Projektes sehr zufrieden. Auftretende Schwierigkeiten und Herausforderungen konnten bewältigt und neue Erkenntnisse aus diesen gewonnen werden. Außerdem konnten wertvolle Erfahrungen gesammelt werden, um für zukünftige Projekte fundierte Entscheidungen in Hinblick auf die Modellierung von Software, der Wahl von Technologien und die Organisation der Arbeitsteilung treffen zu können.

Erkenntnisse Technologien

| Frontend | Backend | Database | Hardware | Other |
|---|--|--|---|--|
| Pains Was hat nicht so gut geklappt? | <ul style="list-style-type: none">• Unzureichendes Screen Design• Abfrage der Daten• Statemanagement | <ul style="list-style-type: none">• Bereitsstellung der Daten• Mongoose | <ul style="list-style-type: none">• Modellierung | <ul style="list-style-type: none">• Kompatibilität (alte Hardware, alte Software, keine Dokumentation)• Lange Einarbeitung |
| Gains Was würde ich genauso wieder machen? | <ul style="list-style-type: none">• Vite• JWT• Jotai• Socket.io | <ul style="list-style-type: none">• Microservice Architecture• JWT• Socket.io• Redis PubSub | <ul style="list-style-type: none">• gemeinsame Datenbasis• Cloud Deployment• NoSQL intuitiv | <ul style="list-style-type: none">• Debian (Unix)• Networking• Redis PubSub |
| Future Was würde ich nächstes anders? | <ul style="list-style-type: none">• TypeScript• Next (SSR)• GraphQL | <ul style="list-style-type: none">• TypeScript• tRPC• GraphQL | <ul style="list-style-type: none">• SQL Database• Local Database | <ul style="list-style-type: none">• Docker Compose• Deployment• Microservices• Git Workflow (Github Projects, Branches) |

Projektplan

- Audit 1
- Audit 2
- Audit 3
- Audit 4

Projektplan

Audit 1

▼ Audit 1 11 Nov 07, 2022 - Nov 10, 2022

| | | | | | | | |
|----|--|---|--------------------------|-----|------|--|--------------------------|
| 1 | <input checked="" type="checkbox"/> Domänenmodell #5 |  antonztsv and foseph | <button>Audit 1</button> | 2 | 3 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 2 | <input checked="" type="checkbox"/> Zielhierarchien #6 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 0.5 | 0.5 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 3 | <input checked="" type="checkbox"/> Erste Risiken #7 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 0.5 | 0.33 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 4 | <input checked="" type="checkbox"/> Problemstellung #9 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 1 | 0.75 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 5 | <input checked="" type="checkbox"/> Alleinstellungsmerkmale #12 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 1 | 0.5 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 6 | <input checked="" type="checkbox"/> Zielsetzungen sowie Begründung des Vorgehens #10 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 1 | 0.75 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 7 | <input checked="" type="checkbox"/> Spezifikation des ersten technischen/architekturellen Proof-of-Concept (PoC) #13 |  antonztsv | <button>Audit 1</button> | 0.5 | 0.5 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 8 | <input checked="" type="checkbox"/> Projektplan #8 |  foseph | <button>Audit 1</button> | 0.5 | 1 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 9 | <input checked="" type="checkbox"/> Projektplan / Deliverables Audit 2 #11 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | 0.5 | 0.33 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 10 | <input checked="" type="checkbox"/> PowerPoint - Audit 1 #14 |  calvinhnzr | <button>Audit 1</button> | 2 | 2 | <input checked="" type="checkbox"/> Done | <button>audit 1</button> |
| 11 | <input checked="" type="checkbox"/> Erste Risiken #22 |  antonztsv, calvinhnzr, and foseph | <button>Audit 1</button> | | | <input type="button" value="New"/> | <button>audit 1</button> |

Projektplan

Audit 2

| Audit 2 7 Nov 14, 2022 - Dec 08, 2022 | | | | | | | | | |
|---------------------------------------|---|---|--|-----|------|--|--|--|--|
| 12 | <input checked="" type="checkbox"/> Iteration Domänenmodell präskriptiv #36 |  antonztsv, calvinhnzr, and foseph | <input type="button" value="Audit 2"/> | 0.5 | 1 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 13 | <input checked="" type="checkbox"/> Stakeholderanalyse #35 |  foseph | <input type="button" value="Audit 2"/> | 1 | 3 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 14 | <input checked="" type="checkbox"/> Präsentation Audit 2 #55 |  antonztsv, calvinhnzr, and foseph | <input type="button" value="Audit 2"/> | 1 | 1.5 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 15 | <input checked="" type="checkbox"/> Projektplan Audit 2 #19 |  antonztsv, calvinhnzr, and foseph | <input type="button" value="Audit 2"/> | 1 | 1 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 16 | <input checked="" type="checkbox"/> Projektrisiken #39 |  antonztsv, calvinhnzr, and foseph | <input type="button" value="Audit 2"/> | 1 | 0.75 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 17 | <input checked="" type="checkbox"/> Alleinstellungsmerkmale #37 |  calvinhnzr | <input type="button" value="Audit 2"/> | 1.5 | 0.75 | <input checked="" type="button" value="Done"/> | <input type="button" value="audit 2"/> | | |
| 18 | <input checked="" type="checkbox"/> Cleanup Repository #41 |  antonztsv | <input type="button" value="Audit 2"/> | 1 | 1 | <input checked="" type="button" value="Done"/> | | | |

Projektplan

Audit 3

| Audit 3 12 Dec 12, 2022 - Jan 12 | | | | | | | | | |
|----------------------------------|---|---|--|--|-----|-----|---|--|---|
| 19 | <input checked="" type="checkbox"/> Kernfragen #42 |  antonztsv, calvihnrz, and floseph | <input type="button" value="Audit 3"/> | | 1.5 | 1.5 | <input checked="" type="button"/> Ready | | <input type="button" value="audit 2"/> <input type="button" value="audit 3"/> |
| 20 | <input checked="" type="checkbox"/> Fachperspektiven-Spezifische Leitfragen #43 |  antonztsv, calvihnrz, and floseph | <input type="button" value="Audit 3"/> | | 1.5 | 1.2 | <input checked="" type="button"/> Ready | | <input type="button" value="audit 3"/> |
| 21 | <input checked="" type="checkbox"/> Digitale Darstellung Campus Gummersbach #27 |  calvihnrz | <input type="button" value="Audit 3"/> | | 20 | 16 | <input checked="" type="button"/> Done | | <input type="button" value="proof of concept"/> |
| 22 | <input checked="" type="checkbox"/> Erfordernisse #64 |  antonztsv | <input type="button" value="Audit 3"/> | | 1 | 1 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 23 | <input checked="" type="checkbox"/> Anforderungen #65 |  antonztsv | <input type="button" value="Audit 3"/> | | 1 | 1 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 24 | <input checked="" type="checkbox"/> Überarbeitung Modellierung Architektur #86 |  antonztsv | <input type="button" value="Audit 3"/> | | 1 | 4 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 25 | <input checked="" type="checkbox"/> Raumverfügbarkeit - E-Ink Display Seitengenerierung #30 |  antonztsv | <input type="button" value="Audit 3"/> | | 3.5 | 4.5 | <input checked="" type="button"/> Done | | <input type="button" value="proof of concept"/> |
| 26 | <input checked="" type="checkbox"/> Modellierungen iterieren #58 |  antonztsv, calvihnrz, and floseph | <input type="button" value="Audit 3"/> | | 2 | 3 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 27 | <input checked="" type="checkbox"/> Deliverables für den 4. Audit (Projektplan) #60 |  antonztsv, calvihnrz, and floseph | <input type="button" value="Audit 3"/> | | 0.5 | 0.5 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 28 | <input checked="" type="checkbox"/> Modellierung der Anwendungslogik #59 |  antonztsv and floseph | <input type="button" value="Audit 3"/> | | 1 | 1.5 | <input checked="" type="button"/> Done | | <input type="button" value="audit 3"/> |
| 29 | <input checked="" type="checkbox"/> Integration eines Raspberry Pi in das Netzwerk #106 |  floseph | <input type="button" value="Audit 3"/> | | 2 | 0.5 | <input checked="" type="button"/> Done | | <input type="button" value="proof of concept"/> |
| 30 | <input checked="" type="checkbox"/> Automatisierter Seitenaufruf durch Raspberry Pi #109 |  floseph | <input type="button" value="Audit 3"/> | | 2 | 2 | <input checked="" type="button"/> Done | | <input type="button" value="proof of concept"/> |

Audit 4

| Audit 4 7 Jan 16 - Feb 23 Current | | | | | | | | | |
|-----------------------------------|--|---|---------|---|----|--------|------------------|---------|--|
| 31 | ✓ Implementierung auth_service #113 |  antonztv | Audit 4 | 3 | 6 | ✓ Done | backend | feature | |
| 32 | ✓ Integration des NFC board an den Raspberry Pi #108 |  floseph | Audit 4 | 1 | 12 | ✓ Done | proof of concept | | |
| 33 | ✓ Integration E-Ink Display #112 |  floseph | Audit 4 | 1 | 8 | ✓ Done | audit 4 | | |
| 34 | ✓ Implementierung der Kommunikationsschnittstellen #115 |  antonztv and floseph | Audit 4 | 2 | 4 | ✓ Done | backend | feature | |
| 35 | ✓ Verwaltung aktiver Belegungen von Räumen und Weitergabe durch API #114 |  antonztv | Audit 4 | 3 | 4 | ✓ Done | backend | feature | |
| 36 | ✓ Erstellung Poster / ggf. Präsentation #119 |  antonztv, calvinhnzr, and floseph | Audit 4 | 3 | 10 | ✓ Done | audit 4 | | |
| 37 | ✓ Fazit und kritisch reflektiertes Prozessassessment #120 |  antonztv, calvinhnzr, and floseph | Audit 4 | 1 | 2 | ✓ Done | audit 4 | | |

Kommentar

Die fehlenden Proof of Concepts aus Audit 3 wurden in Audit 4 mit der bestellten Hardware durchgeführt.

Die Anwendung konnte mit den spezifizierten Funktionalitäten, ausgenommen der nicht integrierbaren Hardware, erfolgreich entwickelt und abgeschlossen werden (siehe Folie "Änderungen im Projekt").

Die letzten Schritte innerhalb der Entwicklung des Prototypen waren mit einem erhöhtem Mehraufwand gegenüber dem geschätzten zeitlichen Aufwand verbunden.