

**Konzeption und Entwicklung einer  
Webanwendung zur einfachen Integration und  
Verwaltung von Augmented Reality Inhalten,  
am Beispiel des Kooperationsprojektes  
„ARlebnispfade OBK“**

**BACHELORARBEIT**

ausgearbeitet von

Anton Zaitsev

11146297

vorgelegt an der

TECHNISCHEN HOCHSCHULE KÖLN  
CAMPUS GUMMERSBACH  
FAKULTÄT FÜR INFORMATIK UND  
INGENIEURWISSENSCHAFTEN

im Studiengang  
MEDIENINFORMATIK

Erster Prüfer/in: Prof. Christian Noss  
Technische Hochschule Köln

Zweiter Prüfer/in: Prof. Dr. Matthias Böhmer  
Technische Hochschule Köln

Gummersbach, im September 2024

**Adressen:** Anton Zaitsev  
Buchenweg 4  
51643 Gummersbach  
[zaitsev.anton@outlook.com](mailto:zaitsev.anton@outlook.com)

Prof. Christian Noss  
Technische Hochschule Köln  
Advanced Media Institute  
Steinmüllerallee 1  
51643 Gummersbach  
[christian.noss@th-koeln.de](mailto:christian.noss@th-koeln.de)

Prof. Dr. Matthias Böhmer  
Technische Hochschule Köln  
Advanced Media Institute  
Steinmüllerallee 1  
51643 Gummersbach  
[matthias.boehmer@th-koeln.de](mailto:matthias.boehmer@th-koeln.de)

## **Kurzfassung**

Diese Bachelorarbeit befasst sich mit der Konzeption und Entwicklung eines spezialisierten Content-Management-Systems zur Verwaltung von Augmented Reality Inhalten für das Kooperationsprojekt „ARlebnispfade OBK“. Ziel der Arbeit ist es, eine benutzerfreundliche Lösung zu schaffen, die den spezifischen Anforderungen der AR-Inhaltserstellung und -verwaltung gerecht wird.

Die Arbeit umfasst eine detaillierte Analyse bestehender CMS-Lösungen, die Modellierung von Anforderungen und die Entwicklung einer an das Projekte angepassten Systemarchitektur. Ein besonderer Fokus liegt auf der Integration mit dem GitHub-Repository des Kooperationsprojekts und der Implementierung einer Vorschaufunktion für AR-Inhalte.

Das resultierende webbasierte CMS ermöglicht eine intuitive Verwaltung verschiedener AR-Inhaltstypen, bietet eine grundlegende AR-Vorschau und integriert sich nahtlos in die bestehenden Projektstrukturen. Die Arbeit schließt mit einer kritischen Evaluation des Systems, der Diskussion von Chancen und Risiken sowie einem Ausblick auf zukünftige Entwicklungsmöglichkeiten.

Die Ergebnisse dieser Arbeit tragen zur Effizienzsteigerung in der AR-Inhaltsverwaltung bei und bieten eine Grundlage für die Weiterentwicklung von spezialisierten CMS im Kontext digitaler Kulturvermittlung.

# **Abstract**

This bachelor thesis deals with the conception and development of a specialized content management system for the administration of augmented reality content for the collaborative project „ARlebnispfade OBK“. The aim of the thesis is to create a user-friendly solution that meets the specific requirements of AR content creation and management. The work includes a detailed analysis of existing CMS solutions, the modeling of requirements and the development of a system architecture adapted to the project. A special focus is on the integration with the GitHub repository of the cooperation project and the implementation of a preview function for AR content.

The resulting web-based CMS enables intuitive management of different AR content types, offers a basic AR preview and integrates seamlessly into the existing project structures. The paper concludes with a critical evaluation of the system, a discussion of opportunities and risks and an outlook on future development possibilities.

The results of this work contribute to increasing efficiency in AR content management and provide a basis for the further development of specialized CMS in the context of digital cultural education.

# Inhaltsverzeichnis

<b>Abkürzungsverzeichnis</b>	<b>I</b>
<b>Abbildungsverzeichnis</b>	<b>II</b>
<b>Tabellenverzeichnis</b>	<b>III</b>
<b>Quellcodeverzeichnis</b>	<b>IV</b>
<b>Gender-Hinweis</b>	<b>V</b>
<b>1. Einleitung</b>	<b>1</b>
1.1. Problemstellung und Kontext . . . . .	1
1.2. Zielsetzung . . . . .	2
1.3. Motivation . . . . .	2
1.4. Aufbau der Arbeit . . . . .	2
<b>2. Grundlagen</b>	<b>4</b>
2.1. Augmented Reality und WebAR . . . . .	4
2.2. Content-Management-Systeme . . . . .	5
2.3. Kooperationsprojekt „ARlebnispfade OBK“ . . . . .	7
<b>3. Analyse</b>	<b>8</b>
3.1. Content-Management-Systeme . . . . .	8
3.1.1. Existierende AR-CMS Lösungen . . . . .	8
3.1.2. Vergleich von Funktionalitäten und Limitationen . . . . .	8
3.1.3. Eignung für das Projekt . . . . .	10
3.2. Aktuelle Lösungsansätze im Projekt . . . . .	10
3.3. Domänenrecherche . . . . .	11
3.3.1. IST-Zustand (deskriptiv) . . . . .	11
3.3.2. SOLL-Zustand (präskriptiv) . . . . .	12
3.4. Benutzermodellierung . . . . .	13
3.4.1. Stakeholderanalyse . . . . .	14
3.4.2. User Profiles . . . . .	15
3.4.3. Personas . . . . .	16
3.5. Anforderungsanalyse . . . . .	18
3.5.1. Benutzererfordernisse . . . . .	18
3.5.2. Funktionale Anforderungen . . . . .	18
3.5.3. Non-funktionale Anforderungen . . . . .	19
3.6. Design Konzept . . . . .	20

## Inhaltsverzeichnis

<b>4. Konzeption</b>	<b>21</b>
4.1. Systemarchitektur und Technologieauswahl . . . . .	21
4.1.1. Gesamtarchitektur . . . . .	21
4.1.2. Technologieauswahl . . . . .	22
4.2. Datenmodellierung . . . . .	22
4.2.1. Vorhandene Datenstruktur . . . . .	22
4.2.2. Speicherung von Routen, POIs und Inhalten . . . . .	23
4.3. Benutzeroberfläche und User Experience . . . . .	25
4.3.1. Wireframes . . . . .	25
4.3.2. Navigationskonzept und Informationsarchitektur . . . . .	26
4.4. Schnittstellen-Design . . . . .	27
<b>5. Entwicklung</b>	<b>28</b>
5.1. Entwicklungsumgebung und Setup . . . . .	28
5.2. Backend-Entwicklung . . . . .	29
5.2.1. API-Design und Routing . . . . .	29
5.2.2. Authentifizierung und Autorisierung . . . . .	31
5.2.3. GitHub-Integration . . . . .	33
5.3. Frontend-Entwicklung . . . . .	36
5.3.1. Routing und Navigation . . . . .	37
5.3.2. Komponenten-Struktur . . . . .	39
5.4. Umsetzung der Kernfunktionalitäten . . . . .	39
5.4.1. Vorschaufunktion für AR-Inhalte . . . . .	41
<b>6. Evaluation und Diskussion</b>	<b>44</b>
6.1. Erfüllung der Anforderungen . . . . .	44
6.2. Chancen und Risiken . . . . .	46
6.3. Limitationen und Ausblick . . . . .	47
<b>7. Fazit und Ausblick</b>	<b>48</b>
<b>Literaturverzeichnis</b>	<b>50</b>
<b>A. Anhang</b>	<b>53</b>
A.1. Analyse . . . . .	53
A.1.1. User Profile - Redaktioneller Anwender . . . . .	55
A.1.2. User Profile - AR-Content-Ersteller . . . . .	56
A.1.3. Persona - Redaktioneller Anwender . . . . .	57
A.1.4. Benutzererfordernisse . . . . .	59
A.1.5. Funktionale Anforderungen . . . . .	60
A.1.6. Non-Funktionale Anforderungen . . . . .	61
A.2. Wireframes . . . . .	62
A.3. Entwicklung . . . . .	64
A.4. Evaluation . . . . .	67

# **Abkürzungsverzeichnis**

**API** Application Programming Interface

**AR** Augmented Reality

**CMS** Content-Management-System

**JWT** JSON Web Tokens

**MR** Mixed Reality

**REST** Representational State Transfer

**VR** Virtual Reality

# Abbildungsverzeichnis

2.1. Darstellung des Realitäts-Virtualitäts-Kontinuum (Milgram u. Kishino, 1994, S. 283) . . . . .	4
2.2. CMS System Architektur (nach Boiko) (Bizjak u. a., 2017) . . . . .	6
3.1. Einbindung von DecapCMS in das Projekt-Repository . . . . .	9
3.2. Domänenmodell deskriptiv . . . . .	12
3.3. Domänenmodell präskriptiv . . . . .	13
4.1. Verzeichnisstruktur GitHub-Repository . . . . .	23
4.2. Entity Relationship Diagramm der Projektdaten (Teil 1) . . . . .	24
4.3. Entity Relationship Diagramm der Projektdaten (Teil 2) . . . . .	24
4.4. Wireframe - Startseite . . . . .	25
4.5. Wireframe - POI Übersicht . . . . .	26
4.6. Wireframe - POI-Detailseite . . . . .	26
5.1. Aufbau CMS Projektverzeichnis - Backend . . . . .	28
5.2. Aufbau CMS Projektverzeichnis - Frontend . . . . .	29
5.3. Sequenzdiagramm - Authentifizierungsflow . . . . .	32
5.4. CMS Backend Response - Alle Routen . . . . .	35
5.5. CMS Frontend - JWT Authentifizierung Sequenz . . . . .	38
5.6. Benutzeroberfläche - Startseite . . . . .	40
5.7. Benutzeroberfläche - Routen Detailansicht . . . . .	40
5.8. Benutzeroberfläche - POI Detailansicht . . . . .	41
5.9. Benutzeroberfläche - Änderungen . . . . .	42
5.10. AR-Vorschaufunktion - 3D-Modell . . . . .	43
A.1. Versuch TinaCMS zu integrieren . . . . .	53
A.2. Persona - Thomas Weber (Redaktioneller Anwender) . . . . .	57
A.3. Persona - Lisa Müller (AR-Content-Erststellerin) . . . . .	58
A.4. Wireframes Seiten . . . . .	62
A.5. Wireframes AR-Vorschaufunktion . . . . .	63
A.6. GitHub Access Token Generierung . . . . .	64
A.7. Benutzeroberfläche - Login . . . . .	64
A.8. Benutzeroberfläche - Routen . . . . .	65
A.9. Benutzeroberfläche - Routen Medien und Inhalte . . . . .	65
A.10. Benutzeroberfläche - POI Detailansicht Formular . . . . .	66
A.11. Benutzeroberfläche - NFT Bilder Vorschau . . . . .	66
A.12. Weltweiter GitHub Ausfall . . . . .	67

# **Tabellenverzeichnis**

6.1. Übersicht der Erfüllung der Muss-Anforderungen . . . . .	45
6.2. SWOT-Analyse des CMS für das CMS . . . . .	46
A.1. Stakeholderanalyse . . . . .	54

# **Quellcodeverzeichnis**

1.	Backend Routing . . . . .	30
2.	Beispiel Routing - POI Routes . . . . .	30
3.	Backend Controller - „getAllRoutes“ . . . . .	31
4.	User Schema . . . . .	31
5.	Registrierung - Passwort-Hashing mit bcrypt . . . . .	33
6.	Login - Generierung JWT . . . . .	33
7.	Abrufen von Routen aus dem GitHub-Repository . . . . .	34
8.	Abfrage von Frontmatter Inhalten einer Route . . . . .	34
9.	Generierung einer Eindeutigen ID für Medien . . . . .	35
10.	Erstellung eines Pull Requests . . . . .	36
11.	Sichere Initialisierung von Octokit mit GitHub-Token . . . . .	36
12.	Implementierung der Navigation Guards . . . . .	38

## **Gender-Hinweis**

Zur besseren Lesbarkeit wird in dieser Bachelorarbeit das generische Maskulinum verwendet. Die in dieser Arbeit verwendeten Personenbezeichnungen beziehen sich – sofern nicht anders kenntlich gemacht – auf alle Geschlechter.

# 1. Einleitung

Augmented Reality (AR) bietet als Technologie die Möglichkeit, virtuelle Inhalte in die reale Umgebung einzublenden. Dabei wird AR bereits heute in verschiedenen Anwendungsbereichen eingesetzt, wie z. B. in der Unterhaltung, der Medizin, in der Lehre und im kommerziellen Umfeld (Arena u. a., 2022).

?

Ein Content-Management-System (CMS) ist eine Software, die es Benutzern ermöglicht, digitale Inhalte ohne tiefgreifende technische Kenntnisse zu erstellen, zu verwalten und zu veröffentlichen. Es bietet in der Regel eine benutzerfreundliche Oberfläche zur Erstellung und Bearbeitung von Inhalten sowie Funktionen zur Organisation, Versierung und Verteilung dieser Inhalte auf verschiedenen digitalen Plattformen (Boiko, 2005).

?

Der Fokus dieser Arbeit liegt auf der Konzeption und Entwicklung eines auf das Kooperationsprojekt „ARlebnispfade OBK“ zugeschnittenen CMS.

## 1.1. Problemstellung und Kontext

Die Integration von AR-Inhalten in bestehende Systeme stellt Entwickler und redaktionelle Anwender vor verschiedene Herausforderungen. Im Kontext des Kooperationsprojekts werden diese besonders deutlich. Das Projekt zielt darauf ab, entlang der Straße der Arbeit in Wiehl und Wipperfürth ein innovatives Konzept zur digitalen Weiterbildung mittels AR zu entwickeln. Hierbei sollen sowohl ältere, weniger technikaffine Menschen als auch jüngere, technikaffine Menschen und Schulklassen gleichermaßen an neue Technologien herangeführt und ihnen gleichzeitig der Zugang zur Geschichte und Kultur ihrer Region ermöglicht werden.

Die aktuelle Verwaltung von AR-Inhalten im Projekt unterliegt diversen Limitationen. Herkömmliche CMS sind oft nicht auf die spezifischen Anforderungen von AR-Inhalten ausgelegt, was zu ineffizienten Workflows und Einschränkungen in der Verwaltung führen kann. Die Erstellung und Verwaltung von AR-Inhalten erfordert spezielle Funktionalitäten, die über die Möglichkeiten traditioneller CMS hinausgehen. Die technischen Herausforderungen umfassen dabei nicht nur die Integration von 3D-Modellen, sondern auch der Umgang mit Inhalten wie Texten, Audios, Bildern und Videos. Organisatorisch ergeben sich zusätzliche Schwierigkeiten durch die Notwendigkeit der Zusammenarbeit verschiedener Stakeholder, darunter Entwickler und lokale Partner aus dem Oberbergischen Kreis.

Das Kooperationsprojekt steht somit vor der Aufgabe, eine spezialisierte Lösung für die Verwaltung von AR-Inhalten zu entwickeln, die sowohl den technischen Anforderungen als auch den spezifischen Bedürfnissen des Projektes gerecht wird. Diese Lösung muss nicht nur die effiziente Erstellung und Verwaltung ermöglichen, sondern vor allem auch die nahtlose Integration in bestehende Datenstrukturen und Systeme gewährleisten.

## 1.2. Zielsetzung

Das Hauptziel dieser Arbeit ist die Entwicklung eines spezialisierten CMS zur Verwaltung von AR-Inhalten innerhalb des Kooperationsprojekts. Das System soll den Workflow für redaktionelle Nutzer optimieren und die Integration von AR-Inhalten in die bestehende Anwendung erleichtern. Die Kernziele des CMS umfassen dabei die effiziente Verwaltung verschiedener AR-Inhaltstypen wie Texte, Audio, Bilder, Videos und 3D-Modelle. Zudem soll eine AR-Vorschaufunktionen bereitgestellt und eine nahtlose Integration mit dem GitHub-Repository des Projekts gewährleistet werden.

Bei der Entwicklung steht die Benutzerfreundlichkeit im Vordergrund, um auch technisch weniger versierten Nutzern die Erstellung und Verwaltung von AR-Inhalten zu ermöglichen. Das CMS zielt zudem darauf ab, die Zeit der Inhaltspflege zu reduzieren und damit einen Beitrag zum Kooperationsprojekt zu leisten. Durch die Erreichung dieser Ziele strebt die Arbeit an, nicht nur einen Mehrwert für das spezifische Projekt zu schaffen, sondern auch einen Beitrag zur Weiterentwicklung von AR-spezifischen Content-Management-Systemen im Allgemeinen zu leisten.

## 1.3. Motivation

Die Motivation für diese Arbeit entspringt dem wachsenden Potenzial von AR (ARtillery Intelligence, 2024) und dem damit verbundenen Bedarf an spezialisierten Content-Management-Systemen. Das persönliche Interesse an der Integration von neuen Technologien in Verbindung mit dem Web bildet die Grundlage für dieses Projekt.

Für die Zukunft der Medieninformatik ist die Entwicklung solcher Systeme von großer Bedeutung. Die Fähigkeit, komplexe AR-Inhalte effizient zu verwalten, wird zunehmend an Bedeutung annehmen (Borisov u. a., 2018).

Ein besonderer Anreiz liegt in der regionalen Bedeutung des Projekts für den Oberbergischen Kreis. Die Entwicklung eines maßgeschneiderten CMS trägt nicht nur zur technologischen Innovation bei, sondern leistet auch einen Beitrag zur kulturellen und bildungspolitischen Entwicklung der Region.

Die Arbeit verspricht, wertvolle Erkenntnisse für zukünftige AR-Projekte zu liefern und bietet die Möglichkeit, theoretische Konzepte der Medieninformatik in einem praxisnahen Kontext anzuwenden. Die Verbindung von technischen Herausforderungen mit den spezifischen Anforderungen des Kulturbereichs eröffnet dabei ein spannendes Forschungsfeld.

## 1.4. Aufbau der Arbeit

Diese Arbeit gliedert sich in sieben Kapitel, die einen systematischen Ansatz zur Konzeption und Entwicklung des CMS verfolgen.

Nach der Einleitung werden in Kapitel 2 die theoretischen Grundlagen zu Augmented Reality, WebAR und Content-Management-Systemen dargelegt, sowie das Kooperationsprojekt vorgestellt. Kapitel 3 widmet sich der Analyse bestehender CMS, aktueller Lösungsansätze im Projekt sowie der Benutzermodellierung und Anforderungsanalyse. Die Konzeption des Systems, einschließlich Systemarchitektur, Datenmodellierung und

## *1. Einleitung*

dem Design, wird in Kapitel 4 erarbeitet. Kapitel 5 beschreibt den Entwicklungsprozess, wobei die Implementierung des Backends und Frontends sowie die Umsetzung der Kernfunktionalitäten im Fokus stehen. Eine kritische Evaluation des entwickelten Systems erfolgt in Kapitel 6, gefolgt von einem Fazit und Ausblick in Kapitel 7. Methodisch folgt die Arbeit einem praxisorientierten Forschungsansatz, der theoretische Konzepte mit praktischer Anwendung verbindet und sich auf Methodiken aus der Menschzentrierten Gestaltung (DIN EN ISO 9241-210:2020-03, 2020) stützt.

## 2. Grundlagen

Dieses Kapitel legt die theoretische und konzeptuelle Basis für die vorliegende Arbeit. Es behandelt die zentralen Themengebiete, die für das Verständnis und die Einordnung des entwickelten CMS essentiell sind. Zunächst wird ein Überblick über Augmented Reality und dessen webbasierte Variante, WebAR, gegeben. Anschließend werden die Grundlagen und aktuellen Entwicklungen im Bereich von Content-Management-Systemen dargestellt, mit besonderem Fokus auf deren Anwendung im Kontext von AR. Abschließend wird das Kooperationsprojekt vorgestellt, welches den praktischen Anwendungskontext für die in dieser Arbeit entwickelte Lösung bildet. Diese Grundlagen schaffen den notwendigen Kontext, um die Herausforderungen und Lösungsansätze in den folgenden Kapiteln nachvollziehen zu können.

### 2.1. Augmented Reality und WebAR

Augmented Reality ermöglicht als Technologie die Wahrnehmung der realen Welt durch die Einblendung virtueller Elemente zu erweitern. Die technologischen Grundlagen von AR umfassen dabei verschiedene Komponenten, darunter Tracking-Systeme zur Positionsbestimmung, Rendering-Technologien zur Darstellung virtueller Objekte und Interaktionstechniken zur Manipulation der Objekte (Billinghurst u. a., 2015).

Im Kontext dieser Arbeit ist besonders die Unterscheidung zwischen AR und verwandten Technologien wie Virtual Reality (VR) und Mixed Reality (MR) von Bedeutung (Abbildung 2.1). Während VR den Nutzer vollständig in eine virtuelle Umgebung eintauchen lässt, kombiniert AR virtuelle Elemente mit der realen Welt (Azuma, 1997). Mixed Reality kann als Oberbegriff verstanden werden, der sowohl AR als auch VR umfasst und verschiedene Grade der Vermischung realer und virtueller Elemente beschreibt (Milgram u. Kishino, 1994).

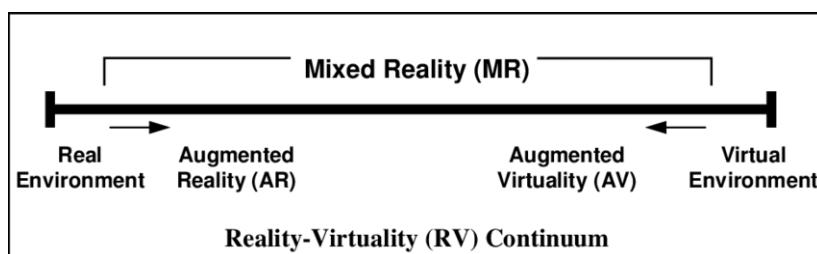


Abbildung 2.1.: Darstellung des Realitäts-Virtualitäts-Kontinuums (Milgram u. Kishino, 1994, S. 283)

WebAR stellt eine spezifische Ausprägung von Augmented Reality dar, welche es ermöglicht, AR-Erlebnisse direkt im Webbrowser zu realisieren, ohne dass eine separate

## 2. Grundlagen

Anwendung installiert werden muss. Dies eröffnet neue Möglichkeiten für die Verbreitung und Zugänglichkeit von AR-Inhalten (Qiao u.a., 2019). WebAR basiert auf Web-Standards und -Technologien wie WebXR und WebGL, die es Entwicklern ermöglichen, interaktive 3D-Grafiken und AR-Funktionalitäten in Webseiten zu integrieren (Shepiliev u.a., 2021).

Zu den Vorteilen von WebAR gegenüber nativen AR-Anwendungen zählen die plattformübergreifende Kompatibilität, die einfache Verteilung und Aktualisierung von Inhalten sowie die niedrigere Einstiegshürde für Nutzer. Allerdings bringt WebAR auch Herausforderungen mit sich, insbesondere in Bezug auf Performance und die Verfügbarkeit fortgeschrittener AR-Funktionen, die in nativen Anwendungen leichter zu realisieren sind (Rock Paper Reality, 2022). Einer der größten Nachteile von nativen Web-Technologien wie WebXR ist die schlechte Kompatibilität unter iOS. Das mobile Betriebssystem unterstützt die Technologie lediglich unter einem experimentellem Flag, welches manuell gesetzt werden muss (Can I Use, 2024).

Für die Entwicklung von WebAR-Anwendungen stehen deshalb neben den nativen Technologien auch verschiedene Frameworks und Bibliotheken zur Verfügung, ~~wie~~ zum Beispiel A-Frame, AR.js und Three.js. Diese Technologien bieten Abstraktionsebenen über die grundlegenden Web-Technologien und ermöglichen es Entwicklern, sich auf die Gestaltung der AR-Erlebnisse zu konzentrieren, ohne sich mit den komplexen Details der unterlagerten Technologien auseinanderzusetzen zu müssen.

Die historische Entwicklung von AR und WebAR zeigt eine rasante Progression von frühen Experimenten mit Head-Mounted Displays in den 1960er Jahren bis hin zu den heutigen, weitverbreiteten mobilen AR-Anwendungen und webbasierten AR-Erlebnissen. Diese Entwicklung wurde maßgeblich durch Fortschritte in der Computergrafik, Sensorik und mobilen Technologien vorangetrieben (Van Krevelen u. Poelman, 2010).

Für eine detailliertere Betrachtung der technischen Aspekte und Anwendungsmöglichkeiten von AR und WebAR sei auf das vorherige Praxisprojekt verwiesen, in dem diese Themen eingehender behandelt wurden.

Ref?

### 2.2. Content-Management-Systeme

Content-Management-Systeme stellen eine zentrale Komponente moderner Webinfrastrukturen dar. Sie bieten eine strukturierte Umgebung zur Erstellung, Verwaltung und Veröffentlichung digitaler Inhalte, ohne dass tiefgreifende technische Kenntnisse erforderlich sind (Boiko, 2005). In ihrer Grundfunktion ermöglichen CMS die Trennung von Inhalt, Struktur und Darstellung, was die Flexibilität und Wartbarkeit von Anwendungen erheblich verbessert.

Die Hauptkomponenten eines typischen CMS umfassen mehrere zentrale Elemente. An erster Stelle steht eine Datenbank, welche für die Speicherung und Organisation der Inhalte verantwortlich ist. Ergänzt wird dies durch Workflow-Management-Tools, die den Prozess der Inhaltserstellung und -genehmigung steuern. Eine weitere wichtige Komponente ist die Benutzerverwaltung, die Rollen- und Rechtekonzepte implementiert, um den Zugriff auf Inhalte und Funktionen zu kontrollieren. Hinzu kommt oftmals ein Template-System, das für die konsistente Darstellung von Inhalten sorgt. Schließlich

## 2. Grundlagen

verfügen CMS über Publikationsmechanismen, die die Veröffentlichung von Inhalten auf verschiedenen Kanälen ermöglichen. Das Zusammenspiel dieser Komponenten bildet die Grundlage für ein effektives Content-Management (siehe Abbildung 2.2).

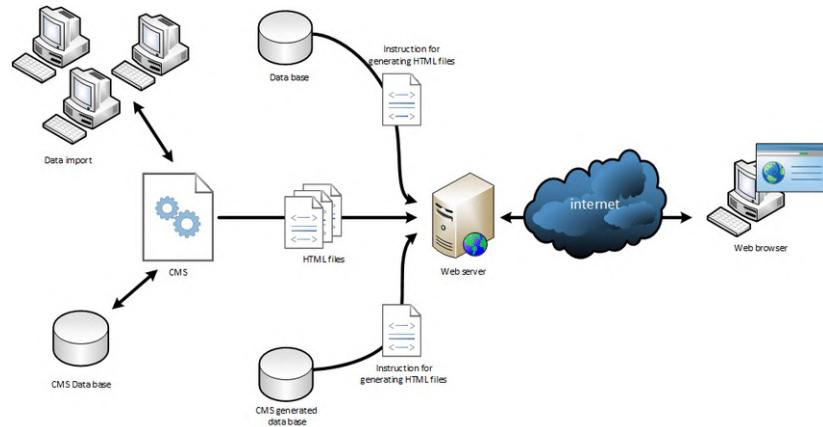


Abbildung 2.2.: CMS System Architektur (nach Boiko) (Bizjak u. a., 2017)

Im Kontext von AR-Inhalten ergeben sich für CMS besondere Anforderungen aber auch Herausforderungen. Anders als bei herkömmlichen Webinhalten müssen AR-spezifische CMS in der Lage sein, komplexe 3D-Modelle, räumliche Informationen und interaktive Elemente zu verwalten und zu präsentieren. Zudem erfordert die Vorschau von AR-Inhalten spezielle Funktionalitäten, die über die Möglichkeiten traditioneller CMS hinausgehen (Borisov u. a., 2018).

In der Entwicklung von Content-Management-Systemen lassen sich verschiedene Architekturen unterscheiden. Traditionelle, monolithische CMS bieten eine All-in-One-Lösung, bei der Backend und Frontend eng miteinander verknüpft sind. Im Gegensatz dazu setzen moderne, Headless-CMS auf eine strikte Trennung von Content-Management und Content-Delivery, was insbesondere für Multi-channel Publishing und die Integration mit AR-Anwendungen Vorteile bietet (Barker, 2016). Aktuelle Trends umfassen dabei die zunehmende Bedeutung von API-first Ansätzen, die Integration von KI-gestützten Funktionen zur Inhaltsanalyse und -erstellung sowie die Entwicklung von spezialisierten CMS für bestimmte Anwendungsbereiche wie E-Commerce oder eben AR (Murphy, 2023). Diese Trends spiegeln die wachsende Komplexität und Diversifizierung digitaler Inhalte wider und stellen die CMS-Entwicklung vor neue Herausforderungen.

Die Rolle von CMS in modernen Webanwendungen geht dabei weit über die bloße Inhaltsverwaltung hinaus. Im Kontext von AR-Anwendungen müssen CMS die Fähigkeit besitzen, räumliche Daten zu verwalten, 3D-Modelle zu integrieren und AR-spezifische Metadaten, wie Informationen zur Positionierung, Skalierung und zu Interaktionsmöglichkeiten, zu handhaben. Ein weiterer Aspekt ist die Integration von AR-Vorschaufunktionen, die es Entwicklern und redaktionellen Nutzern ermöglichen, die AR-Erlebnisse direkt im CMS zu testen und zu optimieren. Darüber hinaus ist eine robuste Versionierung und Unterstützung für kollaboratives Arbeiten von Bedeutung.

## 2. Grundlagen

Im Vergleich zu herkömmlichen Content-Management-Systemen stellen diese Anforderungen erhebliche Herausforderungen dar. Während einige existierende Systeme durch Erweiterungen oder Plugins begrenzte AR-Funktionalitäten bieten, fehlt es oft an umfassenden Lösungen, die den spezifischen Bedürfnissen von AR-Projekten gerecht werden.

### 2.3. Kooperationsprojekt „ARlebnispfade OBK“

Das Kooperationsprojekt „ARlebnispfade OBK“ stellt einen Ansatz zur Verknüpfung von digitaler Technologie und kulturellem Erbe dar. Im Rahmen dieses Vorhabens wird entlang der „Straße der Arbeit“ in Wiehl und Wipperfürth ein innovatives Konzept zur digitalen Weiterbildung mittels Augmented Reality entwickelt. Es sollen ältere, nicht technikaffine Menschen sowie jüngere, technikaffine Menschen und Schulklassen gleichermaßen an neue Technologien herangeführt und ihnen gleichzeitig den Zugang zur Geschichte und Kultur ihrer Region ermöglicht werden. Der Fokus des Vorhabens liegt somit in der Förderung einer generationsübergreifenden, inklusiven und nachhaltigen Strategie zur digitalen Weiterbildung.

Die Umsetzung des Projekts involviert diverse Stakeholder, deren Zusammenarbeit für den Erfolg des Vorhabens entscheidend ist. Zu den Hauptbeteiligten zählen die Volks hochschule Oberberg als Initiator und lokale Kultureinrichtungen, die ihr Fachwissen und ihre Inhalte beisteuern. Die Technische Hochschule Köln, vertreten durch Lehrende und Studierende, bringt ihr technisches Know-How ein und fungiert als Bindeglied zwischen akademischer Forschung und praktischer Anwendung. Diese Konstellation ermöglicht es, verschiedene Perspektiven und Expertisen zu bündeln und so ein ganzheitliches AR-Erlebnis zu schaffen, das sowohl technologisch fortschrittlich als auch kulturell fundiert ist.

Die bisherigen Entwicklungen und Ergebnisse des Projekts zeigen vielversprechende Ansätze. In einer initialen Phase wurden innerhalb verschiedener Projekte an der TH Köln Konzepte und Prototypen für die Anwendung identifiziert. Anschließend wurde die Entwicklung des Projekts durchgeführt. Das Ergebnis ist eine Webanwendung, die an ausgewählten Stationen entlang der „Straße der Arbeit“ diverse AR-Erlebnisse anbietet. Zu diesen gehören unter anderem Visualisierungen historischer Gebäude und Arbeitsprozesse, interaktive Zeitreisen und die Einbindung von Audio, Videos und 3D-Modellen. Redaktionell konnte die Anwendung bisher nur direkt über die Entwickler und Einbindungen in das Repository erfolgen.

Diese Arbeit soll sich diesem Problem widmen und eine geeignete Lösung konzipieren und entwickeln.

## **3. Analyse**

In der Analyse werden bestehende Content-Management-Systeme untersucht, aktuelle Lösungsansätze im Projekt evaluiert sowie eine Domänenrecherche, eine Benutzermodellierung und eine Anforderungsanalyse durchgeführt. Diese Schritte soll es ermöglichen, ein fundiertes Verständnis der Problemstellung zu entwickeln und dienen als Basis für eine erfolgreiche Entwicklung.

### **3.1. Content-Management-Systeme**

Die Integration von Augmented Reality stellt in Content-Management-Systemen eine relativ neue Entwicklung dar, die spezifische Anforderungen an die Verwaltung und Bereitstellung von AR-Inhalten stellt. Aus diesem Grund sollen vor der Konzeption und der Entwicklung zunächst im Rahmen einer Analyse verschiedene CMS-Lösungen untersucht werden, die entweder speziell für AR-Inhalte konzipiert sind oder Potenzial für die Integration von AR-Funktionalitäten aufweisen.

#### **3.1.1. Existierende AR-CMS Lösungen**

Bestehende CMS-Lösungen lassen sich grob in drei Kategorien einteilen: AR-spezifische CMS, Headless-CMS mit AR-Potenzial und Git-basierte CMS mit AR-Potenzial. Unter den AR-spezifischen CMS-Lösungen zeichnen sich Plattformen wie Zapworks Designer, 8thWall und echo3D durch ihre spezialisierten Funktionen aus. Der Zapworks Designer bietet eine umfassende Lösung für die Erstellung und Verwaltung von AR-Erlebnissen, die sich besonders an Nutzer ohne tiefgreifende Programmierkenntnisse richtet. Die 8thWall-Plattform fokussiert sich hingegen auf WebAR-Lösungen und bietet Entwicklern mit Tools wie „Niantic Studio“ die Möglichkeit zur Erstellung von AR-Erlebnissen, die direkt im Browser funktionieren.

Im Bereich der Headless-CMS bieten Plattformen wie Contentful, PayloadCMS und Strapi flexible Systeme, die es durch ihre API-first-Ansätze theoretisch ermöglichen, AR-spezifische Inhaltstypen und Metadaten zu verwalten.

Git-basierte CMS wie DecapCMS oder Tina versprechen eine enge Integration mit bestehenden Entwicklungsworkflows, was sie für Projekte wie das Kooperationsprojekt interessant machen.

#### **3.1.2. Vergleich von Funktionalitäten und Limitationen**

Bei der detaillierten Betrachtung der Funktionalitäten zeigen sich deutliche Unterschiede zwischen den verschiedenen CMS-Typen.

AR-spezifische CMS bieten in der Regel Plattformen für das Erstellen umfassender

### 3. Analyse

AR-Erlebnisse und stellen benötigte Tools zum Tracken, zum Platzieren und zur Vorschau von AR-Inhalten. So ermöglicht 8thWall mit der eigenen Studio Plattform einen Echtzeit XR-Editor mit einer integrierten Game-Engine. Diese Plattformen bieten passende Lösungen für die Erstellung und Verwaltung von AR-Inhalten an, binden jedoch an dieses Plattformen und sind zusätzlich mit erheblichen Mehrkosten verbunden. Für die Verwaltung von 3D-Inhalten bietet echo3D eine spezialisierte Lösung, die es ermöglicht, 3D-Modelle effizient zu verwalten. Hier liegt der Fokus jedoch lediglich auf 3D-Inhalten und bietet keine weiteren AR-Integrationen.

Headless-CMS bieten keine der AR-spezifischen Funktionen nativ an, zeichnen sich aber durch ihre Flexibilität in der Datenmodellierung und der offenen Darstellung des Inhalts aus. Dies ermöglicht es theoretisch, komplexe Strukturen für AR-Inhalte zu definieren, erfordert jedoch oft zusätzliche Entwicklungsarbeit für die Integration von AR-spezifischen Funktionen. Da das bestehende Projekt inklusive dem AR-Inhalt in einem GitHub-Repository gepflegt wird, ist die Integration eines bestehenden Headless-CMS mit den Datenstrukturen erschwert.

Git-basierte CMS wie DecapCMS oder Tina haben den Vorteil der engen Integration mit diversen Versionskontrollsystmen, darunter auch GitHub. Bestehende Datenstrukturen können durch eine YAML-Datei festgelegt und in das System integriert werden. In der Analysephase dieser Arbeit wurde versucht DecapCMS und Tina in das bestehende Projekt einzubinden, wobei Tina gar nicht integriert werden konnte (Abbildung A.1). Die Anpassungsfähigkeit an die bestehende Datenstrukturen stellen sich mit DecapCMS leider als begrenzt heraus (Abbildung 3.1). Dies liegt zwar auch an dem CMS, aber vor allem an der komplexen Datenstruktur des bestehenden Projektes. Es werden zwar Möglichkeiten angeboten, durch die Entwicklung eigener React-Komponenten Erweiterungen der Benutzeroberfläche zu erstellen, jedoch wird in der Analyse auf Grund der Komplexität die Betrachtung des CMS bereits davor nicht weiter verfolgt.

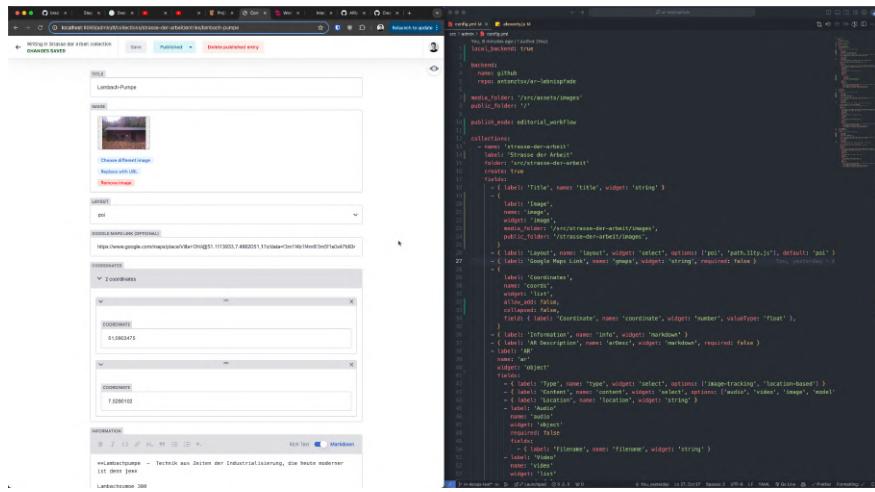


Abbildung 3.1.: Einbindung von DecapCMS in das Projekt-Repository

Innerhalb der Analyse wurde ein weiteres CMS prototypisch evaluiert. Kirby ist ein

### 3. Analyse

kostengünstiges Dateibasiertes-CMS, welches durch Plugins die Anbindung an ein Git-Repository ermöglicht und die Möglichkeit für den Betrieb im Headless-Mode bietet. Auch hier wurde das CMS lokal installiert und evaluiert, ob eine Integration in die bestehenden Datenstrukturen ohne einen erhöhten Mehraufwand durchgeführt werden kann. Da der AR-Inhalt im Projekt-Repository auf der gleichen Ebene wie der Code der Anwendung liegt, ist auch hier eine vollständige Integration nicht, bzw. nur mit einem großen Aufwand verbunden, möglich.

#### 3.1.3. Eignung für das Projekt

Für das Kooperationsprojekt erweist sich keines der untersuchten Systeme als ideale Lösung. AR-spezifische CMS wie 8thWall bieten zwar relevante Funktionen, sind aber oft zu eingeschränkt in ihrer Anpassbarkeit an die spezifischen Projektanforderungen und zudem mit hohen Kosten verbunden. Headless-CMS wie Contentful oder Strapi könnten theoretisch angepasst werden, erfordern aber einen erheblichen Entwicklungsaufwand für die Integration von AR-spezifischen Funktionen.

Die Evaluation von DecapCMS zeigt, dass Git-basierte Systeme zwar gut in den Entwicklungsworkflow integrierbar sind, aber Schwächen in der Benutzerfreundlichkeit und Anpassbarkeit aufweisen. Die erfolglose Integration von Tina CMS unterstreicht die Herausforderungen bei der Implementierung solcher Systeme in bestehende Projektstrukturen.

Aufgrund der im Projekt bestehenden Datenstrukturen zur Speicherung der AR-Inhalte, bietet sich eine Eigenentwicklung eines auf das Projekt angepassten CMS für diese Arbeit an.

## 3.2. Aktuelle Lösungsansätze im Projekt

Im Rahmen des Kooperationsprojekts wurde bisher ein pragmatischer Ansatz zur Verwaltung von AR-Inhalten verfolgt, der primär auf der Zusammenarbeit zwischen redaktionellen Anwendern und Entwicklern basiert. Dieser Prozess, obwohl funktional, weist einige Herausforderungen auf, die im Folgenden näher aufgezeigt werden.

Die derzeitige Methode zur AR-Inhaltsverwaltung folgt einem linearen Workflow. Redaktionelle Anwender, die entweder selbst die AR-Inhalte erstellen oder diese von den AR-Content-Ersteller erhalten, stellen die Daten bereit. Diese Inhalte umfassen typischerweise Texte, Bilder, Audio- und Videodateien sowie 3D-Modelle. Die Entwickler übernehmen anschließend die Aufgabe, diese Inhalte in das Projekt-Repository einzupflegen. Dieser Prozess erfordert ein hohes Maß an Koordination und Kommunikation zwischen den beteiligten Parteien, um sicherzustellen, dass die Inhalte korrekt und vollständig übertragen werden. Nach der Übertragung wird anschließend der gesamte Inhalt mittels des Static Site Generators Eleventy (11ty) neu kompiliert. Dieser Prozess generiert eine neue Version der Website, die dann manuell deployed und auf dem Webserver veröffentlicht wird.

Bei der Analyse dieses Workflows lassen sich mehrere Schwachstellen und Ineffizienzen identifizieren. Zunächst ist der Prozess stark abhängig von der Verfügbarkeit und dem Zeitmanagement der Entwickler. Dies kann zu Verzögerungen bei der Veröffentlichung

### 3. Analyse

neuer oder aktualisierter Inhalte führen. Darüber hinaus besteht ein erhöhtes Risiko für Fehler bei der manuellen Übertragung der Inhalte in das Repository. Die Notwendigkeit, für jede Änderung den gesamten Build- und Deployment-Prozess durchzuführen, führt ebenfalls zu Verzögerungen, kann aber auch mit der Verwendung eines CMS im aktuellen Projektstand nicht umgangen werden.

Die Benutzerfreundlichkeit des aktuellen Systems variiert stark zwischen den verschiedenen Stakeholdern. Für die Entwickler bietet der Prozess zwar eine hohe Kontrolle über die Inhalte und deren Integration, erfordert jedoch einen erheblichen Zeitaufwand. Redaktionelle Anwender hingegen haben nur begrenzten direkten Einfluss auf die finale Präsentation ihrer Inhalte und sind auf die Verfügbarkeit der Entwickler angewiesen. Dies kann zu Frustration und verminderter Effizienz führen.

Hinsichtlich der Skalierbarkeit und Wartbarkeit zeigt die aktuelle Lösung deutliche Limitationen. Mit wachsendem Projektumfang und zunehmender Anzahl von AR-Inhalten wird der manuelle Prozess zunehmend aufwändiger und fehleranfälliger. Die Wartbarkeit wird durch die enge Verzahnung von Inhalt und Code im Repository erschwert, was langfristig zu Herausforderungen bei der Weiterentwicklung und Anpassung des Systems führen kann.

Die Integration mit anderen Systemen und Werkzeugen im Projekt beschränkt sich derzeit auf die Nutzung des Versionskontrollsysteins Git und GitHub sowie des Static Site Generators 11ty. Innerhalb der Anwendung werden die Image Tracking Komponenten von AR.js mit A-Frame und Three.js zur grafischen Darstellung verwendet. Während dies eine solide Basis für die Versionierung und Generierung der Website bietet, fehlt es an spezialisierten Tools für die Verwaltung und Vorschau von AR-Inhalten. Dies limitiert die Möglichkeiten für redaktionelle Anwender, ihre Inhalte vor der finalen Integration zu testen und zu optimieren.

Zusammenfassend lässt sich feststellen, dass der aktuelle Lösungsansatz zwar funktional ist, jedoch signifikante Verbesserungspotenziale aufweist. Insbesondere die Effizienz des Workflows, die Benutzerfreundlichkeit für redaktionelle Anwender und die Skalierbarkeit des Systems stellen Bereiche dar, in denen Optimierungen erforderlich sind. Die Entwicklung eines spezialisierten CMS für AR-Inhalte könnte viele dieser Herausforderungen adressieren und zu einer deutlichen Verbesserung des Gesamtprozesses führen.

## 3.3. Domänenrecherche

Im Rahmen der Domänenrecherche werden zwei Modelle erstellt: ein deskriptives IST-Modell, das den aktuellen Zustand des Systems repräsentiert, sowie ein präskriptives SOLL-Modell, das die angestrebte Systemstruktur nach Implementation des CMS aufzeigt.

### 3.3.1. IST-Zustand (deskriptiv)

Das deskriptive Modell des IST-Zustands offenbart eine lineare, jedoch fragmentierte Struktur des aktuellen Arbeitsablaufs (Abbildung 3.2). AR-Content-Erststeller und redaktionelle Anwender initiieren den Prozess durch die Übermittlung von Dateien an

### 3. Analyse

die Entwickler. Diese zentrale Position der Entwickler im Workflow indiziert potenzielle Engpässe und Ineffizienzen in der Verwaltung der AR-Inhalte.

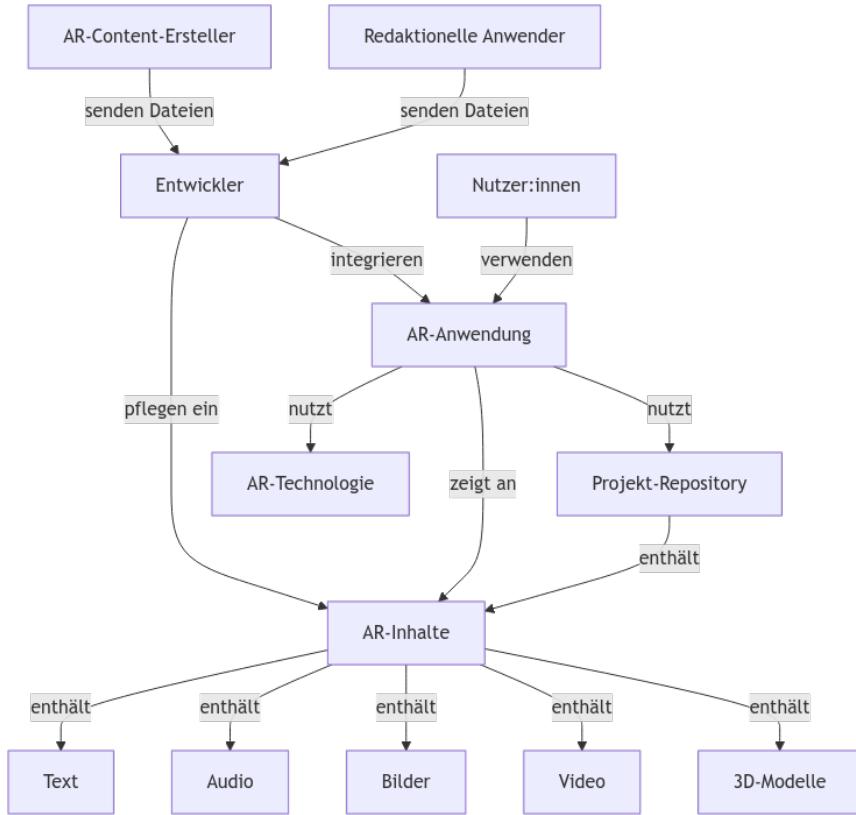


Abbildung 3.2.: Domänenmodell deskriptiv

Die Entwickler übernehmen die Verantwortung für die Integration des AR-Inhalts in das Projekt-Repository. Die Inhalte selbst umfassen diverse Medientypen wie Text, Audio, Bilder, Video und 3D-Modelle. Diese Diversität der Inhaltstypen unterstreicht die Komplexität des Contentmanagements und die Notwendigkeit eines spezialisierten Systems zur effektiven Verwaltung.

#### 3.3.2. SOLL-Zustand (präskriptiv)

Das präskriptive SOLL-Modell visualisiert eine optimierte Systemarchitektur, in deren Zentrum das zu entwickelnde CMS steht. In diesem Modell erfolgt eine klare Separation der Rollen und Verantwortlichkeiten, was zu einem effizienteren Workflow führt (Abbildung 3.3).

Redaktionelle Anwender interagieren direkt mit dem CMS, was eine effizientere Content-Pflege und -Verwaltung ermöglicht. Das CMS übernimmt zentrale Funktionen wie die Speicherung der Inhalte im GitHub Projekt-Repository, die Generierung von AR-Vorschauen und die eigentliche Ansicht der Inhalte. Diese Zentralisierung der Funktio-

### 3. Analyse

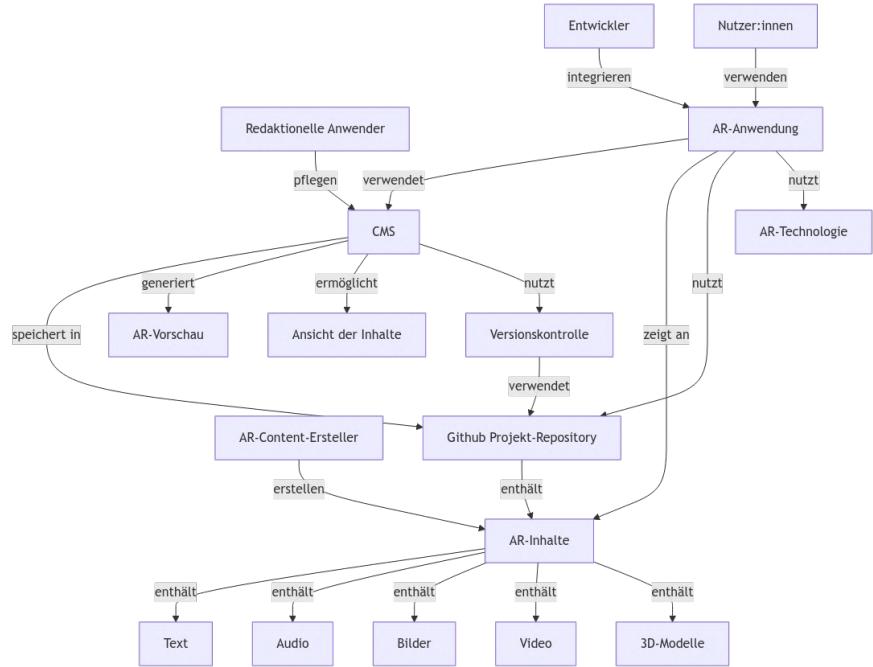


Abbildung 3.3.: Domänenmodell präskriptiv

nen im CMS reduziert die Abhängigkeit von Entwicklern für routinemäßige Content-Management-Aufgaben. Ein signifikanter Aspekt des SOLL-Modells ist die Integration einer Versionskontrolle, die direkt mit dem GitHub Projekt-Repository interagiert. Dies ermöglicht eine präzisere Nachverfolgung von Änderungen und eine verbesserte Kollaboration zwischen den verschiedenen Stakeholdern.

Die Rolle der Entwickler verschiebt sich in diesem Modell primär auf die Integration des CMS mit der AR-Anwendung. Die AR-Anwendung sowie das CMS verwenden das Projekt-Repository als zentrale Quelle für Inhalte, was eine konsistenteren und effizienteren Content-Bereitstellung ermöglicht.

Die Endnutzer interagieren in diesem Modell ausschließlich mit der AR-Anwendung, die weiterhin auf AR-Technologien zurückgreift, um die Inhalte zu präsentieren. Diese Struktur gewährleistet eine klare Trennung zwischen Content-Management und Content-Präsentation, was sowohl die Verwaltung als auch die Nutzererfahrung optimiert.

Diese Domänenmodelle illustrieren die angestrebten Verbesserungen durch die Implementation des CMS und dienen als Grundlage für die weitere Systemkonzeption und die Identifikation spezifischer Anforderungen an das zu entwickelnde System.

### 3.4. Benutzermodellierung

Die Entwicklung eines effektiven CMS für AR-Inhalte erfordert ein tiefgreifendes Verständnis der verschiedenen Nutzergruppen und ihrer spezifischen Bedürfnisse. Im Rahmen



### 3. Analyse

men dieser Benutzermodellierung werden mehrere Methoden angewandt, um ein umfassendes Bild der Nutzer und ihrer Anforderungen zu erstellen.

#### 3.4.1. Stakeholderanalyse

Wie fließt das Projekt ein?

Die durchgeführte Stakeholderanalyse identifiziert diverse Gruppen, die direkt oder indirekt von dem zu entwickelnden CMS beeinflusst werden. Diese systematische Untersuchung ermöglicht die Berücksichtigung diverser Perspektiven und Interessen im Designprozess des Systems (Tabelle A.1).

Als primäre Stakeholder werden die redaktionellen Anwender klassifiziert. Diese Einzelpersonen tragen die Hauptverantwortung für die Verwaltung und Pflege der AR-Inhalte. Ihr primäres Interesse liegt in umfassenden Funktionalitäten zur Inhaltsverwaltung, die eine übersichtliche Administration und effektive Versionskontrolle gewährleisten. In enger Verbindung stehen die AR-Content-Ersteller, deren Aufgabenbereich die Erstellung von AR-Inhalten ist. Ihre Anforderungen umfassen eine intuitive Benutzeroberfläche, die effiziente Arbeitsabläufe und die unkomplizierte Integration diverser Medientypen unterstützt.

Zu den sekundären Stakeholdern zählen Entwickler, die für die Integration und Wartung des Systems verantwortlich sind. Ihre Bedürfnisse beinhalten den Zugang zu technischer Dokumentation sowie die Erwartung einer strukturierten API und Systemerweiterbarkeit. Projektmanager, die den Gesamtfortschritt überwachen, repräsentieren ebenfalls eine sekundäre Stakeholdergruppe. Ihr Fokus liegt auf der Übersicht des Projektfortschritts und der Einhaltung von Zeitplan und Budget.

Die Qualitätssicherung nimmt eine zentrale Rolle bei der Gewährleistung der Systemqualität ein. Ihre Anforderungen an Testumgebungen und der Bedarf an verifizierbaren Funktionen und Systemstabilität fließen in die Konzeption ein. UX/UI-Designer tragen zur Gestaltung der Benutzeroberfläche bei und erwarten Flexibilität im UI-Design.

Die Kooperationspartner, die das System potenziell für eigene AR-Projekte nutzen, stellen Anforderungen an Adaptabilität, Flexibilität und Skalierbarkeit, die im Designprozess berücksichtigt werden.

Als tertiäre Stakeholder werden die Endnutzer der AR-Anwendung klassifiziert. Obgleich sie nicht in direkter Interaktion mit dem CMS stehen, ist ihre Erwartungshaltung an qualitativ hochwertige, interaktive AR-Erlebnisse ein signifikanter Faktor in der Systemkonzeption. Datenschutzbeauftragte übernehmen eine essentielle Funktion bei der Sicherstellung der DSGVO-Konformität und der Transparenz in der Datenverarbeitung.

Zusätzlich werden die Interessen des Marketings, das die AR-Inhalte für Werbezwecke zu nutzen beabsichtigt, sowie externer Technologiepartner, deren Fokus auf der Kompatibilität und Optimierung des Systems liegt, in die Analyse einbezogen.

Diese vielschichtigen Perspektiven und Anforderungen sollen im Entwicklungsprozess berücksichtigt werden und tragen zur Konzeption eines CMS bei, das den verschiedenen Bedürfnissen aller involvierten Parteien gerecht wird.

### 3. Analyse

#### 3.4.2. User Profiles

User Profiles stellen ein essentielles Instrument in der Benutzermodellierung dar. Sie dienen der detaillierten Charakterisierung typischer Nutzergruppen eines Systems und ermöglichen ein tiefgreifendes Verständnis ihrer spezifischen Bedürfnisse, Fähigkeiten und Herausforderungen. Im Kontext der Entwicklung des CMS für AR-Inhalte werden zwei zentrale User Profiles identifiziert: der redaktionelle Anwender (Unterabschnitt A.1.1) und der AR-Content-Erststeller (Unterabschnitt A.1.2).

##### **Redaktioneller Anwender**

Der redaktionelle Anwender repräsentiert eine Schlüsselrolle im Content-Management-Prozess. Diese Nutzergruppe zeichnet sich durch ein Altersspektrum von 30 bis 55 Jahren aus und verfügt typischerweise über einen Hochschulabschluss in Journalismus, Kommunikation oder verwandten Disziplinen. Ihre technische Affinität und Kenntnisse variieren von niedrig bis mittel.

Der berufliche Hintergrund dieser Gruppe umfasst 5 bis 15 Jahre Erfahrung in der redaktionellen Arbeit. Sie sind vertraut mit traditionellen und digitalen Publikationsprozessen und besitzen grundlegende Kenntnisse in Content-Management-Systemen. Ihre Kernkompetenzen liegen in starken redaktionellen und organisatorischen Fähigkeiten sowie in der Verwaltung und Pflege digitaler Inhalte.

Die primären Ziele dieser Nutzergruppe umfassen die effiziente Verwaltung und Organisation von AR-Inhalten, die Sicherstellung inhaltlicher Qualität und Konsistenz sowie die Optimierung des Workflows für die Inhalts-Erstellung und -Veröffentlichung. Sie sehen sich mit Herausforderungen konfrontiert, die den Umgang mit der Komplexität von AR-Inhalten, die Koordination zwischen Content-Erststellern und technischem Team sowie die Einhaltung gemeinsamer redaktioneller Standards betreffen.

Ihre Erwartungen an das CMS beinhalten eine übersichtliche und strukturierte Verwaltung der Inhalte, eine AR-Vorschaufunktion, effektive Versionskontrolle und Änderungsnachverfolgung sowie einfache Überprüfungs- und Freigabeprozesse.

##### **AR-Content-Erststeller**

Das User Profile des AR-Content-Erstellers charakterisiert eine jüngere, technisch versierte Nutzergruppe im Alter von 25 bis 45 Jahren. Sie verfügen typischerweise über einen Hochschulabschluss in Mediendesign, digitalen Medien oder verwandten Fachrichtungen und zeichnen sich durch eine hohe technische Affinität aus.

Ihr beruflicher Hintergrund umfasst 3 bis 10 Jahre Erfahrung in der Erstellung digitaler Inhalte. Sie sind vertraut mit verschiedenen Medienformaten, einschließlich Text, Bild, Audio, Video und 3D-Modellen, und besitzen Grundkenntnisse in AR-Technologien. Ihre Kernkompetenzen liegen im kreativen Denken, der visuellen Gestaltung und der Erfahrung mit Content-Creation-Tools.

Die Ziele dieser Nutzergruppe fokussieren sich auf die Erstellung innovativer und ansprechender AR-Erlebnisse, die Etablierung effizienter Arbeitsabläufe bei der Erstellung von Inhalten und die kontinuierliche Verbesserung der Qualität. Sie sehen sich mit Herausforderungen konfrontiert, die die Komplexität der AR-Technologie, den Zeitdruck bei der Inhaltserstellung und die Anpassung an sich ändernde AR-Standards

### 3. Analyse

und -Plattformen betreffen. Ihre Erwartungen an das CMS umfassen eine intuitive Benutzeroberfläche, die einfache Integration verschiedener Medientypen und eine Vorschaufunktion für AR-Inhalte.

Die Berücksichtigung dieser detaillierten User Profiles in der Entwicklung des CMS ermöglicht eine zielgerichtete Adressierung der spezifischen Bedürfnisse und Herausforderungen beider Nutzergruppen. Dies trägt maßgeblich zur Gestaltung eines Systems bei, das sowohl den Anforderungen der redaktionellen Anwender als auch der AR-Content-Erststeller gerecht wird und somit eine effiziente und effektive Verwaltung und Erstellung von AR-Inhalten ermöglicht.

#### 3.4.3. Personas

Aus den User Profiles lassen sich Personas identifizieren, die ein tieferes Verständnis der Bedürfnisse, Motivationen und Verhaltensweisen der Zielenutzer ermöglichen. Im Kontext des zu entwickelnden CMS für AR-Inhalte werden zwei zentrale Personas identifiziert: Thomas Weber als redaktioneller Anwender (Abbildung A.2) und Lisa Müller als Content-Erststellerin (Abbildung A.3).

##### **Thomas Weber - Redaktioneller Anwender**

Thomas Weber, 45 Jahre alt, verkörpert den typischen redaktionellen Anwender des CMS. Als Kulturbeauftragter der Stadt Gummersbach mit Wohnsitz in Bergneustadt bringt er einen fundierten Hintergrund im Kulturmanagement und eine zehnjährige Erfahrung in der Stadtverwaltung mit. Seine Verantwortlichkeiten umfassen die Koordination kultureller Projekte und die Pflege des digitalen Kulturangebots der Stadt. Thomas' primäre Ziele fokussieren sich auf die effiziente Organisation und Verwaltung digitaler Inhalte für kulturelle Projekte, die Förderung des kulturellen Erbes des Oberbergischen Kreises durch moderne Medien und die Verbesserung der Zusammenarbeit zwischen diversen lokalen Akteuren wie Museen, Vereinen und Schulen.

Seine beruflichen Herausforderungen zeichnen sich in begrenzten finanziellen und personellen Ressourcen für digitale Projekte, Schwierigkeiten bei der Integration historischer Informationen in moderne Medienformate und zeitaufwändigen Abstimmungsprozessen mit verschiedenen Interessengruppen ab.

Ein typischer Arbeitstag von Thomas beginnt mit der Sichtung von E-Mails und der Planung anstehender kultureller Veranstaltungen. Ein Großteil seiner Zeit widmet er der Prüfung und Aufbereitung eingereichter Inhalte lokaler Künstler und Historiker für digitale Plattformen. Regelmäßige Gespräche mit Vertretern lokaler Kultureinrichtungen und die Koordination übergreifender Projekte runden sein Tätigkeitsprofil ab. Thomas verfügt über grundlegende Kenntnisse in Content-Management-Systemen, Erfahrung mit Social-Media-Plattformen für kulturelle Kommunikation und Basiswissen in Bild- und Videobearbeitung.

Seine Erwartungen an das CMS umfassen eine benutzerfreundliche Oberfläche, die auch für weniger technikaffine Mitarbeiter zugänglich ist, die Möglichkeit zur einfachen Integration von historischen Daten und Geodaten, Werkzeuge zur Zusammenarbeit und Abstimmung mit verschiedenen lokalen Partnern sowie flexible Möglichkeiten zur Kategorisierung von Inhalten nach kulturellen und historischen Aspekten.

### 3. Analyse

Thomas' Motivation spiegelt sich in seinem Zitat wider: „Unser kulturelles Erbe ist ein Schatz, den wir durch moderne Technologien für alle Generationen erlebbar machen wollen.“

#### **Lisa Müller - Content-Erstellerin**

Lisa Müller, 28 Jahre alt, repräsentiert die typische Content-Erstellerin für das CMS. Als Mediengestalterin bei einer lokalen Produktionsfirma in Gummersbach bringt sie eine fundierte Ausbildung zur Mediengestalterin Bild und Ton sowie vierjährige Berufserfahrung mit. Ihre regionale Verwurzelung und ihr Engagement in lokalen Kulturprojekten unterstreichen ihre Verbundenheit mit dem Oberbergischen Kreis.

Lisas zentrale Ziele umfassen die Produktion hochwertiger visueller Inhalte, die die Geschichte und Kultur des Oberbergischen Kreises hervorheben, die Weiterentwicklung ihrer Fähigkeiten in der Erstellung verschiedener Medienformate und einen Beitrag zur Attraktivitätssteigerung der Region durch innovative Medienprojekte.

Ihre beruflichen Herausforderungen manifestieren sich in begrenzten technischen Ressourcen im Vergleich zu größeren Städten, Schwierigkeiten bei der Darstellung komplexer historischer oder kultureller Inhalte in modernen Medienformaten und zeitaufwändigen Prozessen beim Organisieren und Teilen von Inhalten mit verschiedenen Projektpartnern.

Ein typischer Arbeitstag von Lisa beginnt oft mit Außenaufnahmen an historischen Stätten oder bei lokalen Veranstaltungen. Nachmittags widmet sie sich der Bearbeitung des Materials, dem Videoschnitt oder der Fotobearbeitung. Der regelmäßige Austausch mit lokalen Partnern und Kultureinrichtungen ist integraler Bestandteil ihrer Tätigkeit.

Lisa verfügt über fortgeschrittene Fähigkeiten in Video- und Bildbearbeitungssoftware, Erfahrung im Umgang mit professionellem Kamera- und Audioequipment sowie Grundkenntnisse in Content-Management-Systemen und Social-Media-Plattformen.

Ihre Erwartungen an das CMS umfassen eine benutzerfreundliche Oberfläche für das einfache Hochladen und Organisieren verschiedener Medienformate, die Möglichkeit, Inhalte mit Geodaten und historischen Informationen zu verknüpfen, Kollaborationsfunktionen für die Zusammenarbeit mit anderen lokalen Akteuren sowie ein flexibles Tagging-System zur Kategorisierung von Inhalten nach Themen, Orten und historischen Epochen.

Lisas Motivation zeichnet sich in folgendem Zitat ab: „Ich möchte die reiche Geschichte und Kultur unserer Region durch moderne Medien zum Leben erwecken und für alle zugänglich machen.“

Die detaillierte Ausarbeitung dieser Personas ermöglicht ein tiefgreifendes Verständnis der spezifischen Bedürfnisse, Herausforderungen und Erwartungen der Hauptnutzergruppen des CMS. Diese Erkenntnisse fließen maßgeblich in den Entwicklungsprozess ein und tragen zur Gestaltung eines Systems bei, das den Anforderungen sowohl der redaktionellen Anwender als auch der Content-Ersteller gerecht wird.

### 3. Analyse

## 3.5. Anforderungsanalyse

Die Anforderungsanalyse basiert auf den Erkenntnissen aus der vorangegangenen Benutzermodellierung und zielt darauf ab, ein umfassendes Verständnis der funktionalen und nicht-funktionalen Anforderungen an das System zu entwickeln. Diese strukturierte Analyse ermöglicht es, die Bedürfnisse und Erwartungen der verschiedenen Stakeholder systematisch zu erfassen und in konkrete Systemanforderungen zu überführen. Im Folgenden werden zunächst die Benutzererfordernisse dargelegt, gefolgt von einer aus den Erfordernissen abgeleiteten Aufstellung der funktionalen und nicht-funktionalen Anforderungen an das System.

### 3.5.1. Benutzererfordernisse

Die Erfordernisse repräsentieren die grundlegenden Bedürfnisse und Erwartungen der Hauptnutzergruppen an das zu entwickelnde System. Sie bilden die Basis für die nachfolgende detaillierte Anforderungsspezifikation und lassen sich in mehrere Kernbereiche untergliedern. Die vollständige Liste der erarbeiteten Erfordernisse ist dem Anhang unter Unterabschnitt A.1.4 zu entnehmen.

Ein primäres Erfordernis der Content-Erststeller ist die Möglichkeit, verschiedene Medienformate effizient zu verwalten. Sie benötigen eine einfache und intuitive Methode, um Video-, Audio-, Bild- und 3D-Modelldateien hochzuladen und zu organisieren. Diese Fähigkeit ist entscheidend für die Erstellung vielfältiger und ansprechender AR-Erlebnisse.

Redaktionelle Anwender haben das Erfordernis, Inhalte mit geografischen und historischen Daten zu verknüpfen. Sie benötigen Werkzeuge, die es ihnen ermöglichen, AR-Inhalte präzise im räumlichen und zeitlichen Kontext zu verorten. Dieses Bedürfnis ist fundamental für die Schaffung kontextrelevanter AR-Erlebnisse, die den Nutzern ein tieferes Verständnis ihrer Umgebung vermitteln.

Redaktionelle Anwender und Content-Erststeller benötigen zudem eine Funktion zur AR-Vorschau. Sie müssen in der Lage sein, ihre Arbeit direkt im System zu visualisieren und zu überprüfen, um die Qualität ihrer AR-Inhalte sicherzustellen und Iterationszyklen zu reduzieren.

Ein weiteres zentrales Erfordernis beider Nutzergruppen ist der Zugang zu effektiven Kollaborationswerkzeugen. Angesichts der komplexen Natur von AR-Projekten müssen sie in der Lage sein, nahtlos zusammenzuarbeiten, Inhalte gemeinsam zu editieren, zu kommentieren und verschiedene Versionen zu verwalten.

Diese Kernerfordernisse der Benutzer bilden die Grundlage für die nachfolgende detaillierte Spezifikation der funktionalen und nicht-funktionalen Anforderungen an das System und stellen sicher, dass das zu entwickelnde AR-CMS die wesentlichen Bedürfnisse der Nutzer adressiert.

### 3.5.2. Funktionale Anforderungen

Die funktionalen Anforderungen definieren die spezifischen Funktionen und Verhaltensweisen, die das CMS erfüllen muss, um den Bedürfnissen der Nutzer gerecht zu werden. Diese Anforderungen leiten sich direkt aus den zuvor identifizierten Benutze-

### 3. Analyse

erfordernissen ab und bilden die Grundlage für die technische Umsetzung des Systems. Eine zentrale Anforderung an das AR-CMS ist die Fähigkeit, verschiedene Medienformate zu verarbeiten und zu verwalten [F10]. Das System muss in der Lage sein, Texte, Audios, Bilder, Videos und 3D-Modelle zu akzeptieren, zu speichern und für die weitere Bearbeitung bereitzustellen. Diese Funktionalität ist essentiell für die Erstellung vielfältiger AR-Erlebnisse und unterstützt direkt die Arbeit der Content-Ersteller. Eine weitere kritische Funktion ist die Bereitstellung einer AR-Vorschau für hochgeladene Inhalte [F30]. Diese Vorschaufunktion ermöglicht es Nutzern, die AR-Erlebnisse direkt im System zu visualisieren und zu überprüfen, bevor sie veröffentlicht werden. Dies trägt wesentlich zur Qualitätssicherung bei und reduziert den Aufwand für nachträgliche Korrekturen.

Das System muss zudem in der Lage sein, den Bearbeitungsstatus von Inhalten zu verfolgen und anzuzeigen [F80]. Diese Funktionalität unterstützt die Zusammenarbeit im Team und ermöglicht es den redaktionellen Anwendern, den Fortschritt der Inhaltsproduktion effektiv zu überwachen.

Eine weitere entscheidende Anforderung ist die Fähigkeit des Systems, mit den bestehenden Datenstrukturen des Projektes zu interagieren [F160]. Dies gewährleistet eine nahtlose Integration in die bestehende Infrastruktur des Kooperationsprojekts und ermöglicht eine effiziente Nutzung vorhandener Ressourcen.

Schließlich muss das System in der Lage sein, verschiedene Routen eines ARlebnispfades zu verwalten [F170] und die AR-Spots einer Route zu organisieren [F180]. Diese Funktionen sind zentral für die Strukturierung und Organisation der AR-Erlebnisse im Kontext des Projekts.

Für eine vollständige Liste aller funktionalen Anforderungen, einschließlich der „sollte“ und „kann“ Anforderungen, wird auf den Anhang unter Unterabschnitt A.1.5 verwiesen.

#### 3.5.3. Non-funktionale Anforderungen

Die non-funktionalen Anforderungen definieren die Qualitätsmerkmale und Rahmenbedingungen, unter denen das CMS operieren muss. Sie sind ebenso wichtig wie die funktionalen Anforderungen, da sie die Gesamtleistung, Benutzerfreundlichkeit und Zuverlässigkeit des Systems maßgeblich beeinflussen.

Eine zentrale non-funktionale Anforderung ist die Bereitstellung einer intuitiven Benutzeroberfläche [O10]. Das System muss so gestaltet sein, dass es auch von Nutzern mit begrenzten technischen Kenntnissen effektiv bedient werden kann. Dies ist entscheidend für die breite Akzeptanz und effiziente Nutzung des Systems durch verschiedene Nutzergruppen.

Das System muss zudem in der Lage sein, mit wachsenden Datensätzen und Nutzerzahlen umzugehen, ohne signifikant an Performance zu verlieren [O30]. Diese Skalierbarkeit ist entscheidend für die langfristige Nutzbarkeit und den Erfolg des CMS, insbesondere im Kontext des Kooperationsprojekts.

Die Einhaltung aller relevanten Datenschutzbestimmungen, insbesondere der DSGVO, ist eine weitere kritische Anforderung [O40]. Das System muss sicherstellen, dass personenbezogene Daten angemessen geschützt und verarbeitet werden, um rechtliche Compliance zu gewährleisten und das Vertrauen der Nutzer zu wahren.

### 3. Analyse

Schließlich muss das CMS mit gängigen Webbrowsersn und mobilen Geräten kompatibel sein [O50]. Diese breite Kompatibilität ist entscheidend, um einen barrierefreien Zugang zum System für alle Nutzer zu gewährleisten, unabhängig von ihrem bevorzugten Gerät oder Browser.

Für eine vollständige Liste aller non-funktionalen Anforderungen, einschließlich der „sollte“ und „kann“ Anforderungen, wird auf den Anhang unter Unterabschnitt A.1.6 verwiesen.

## 3.6. Design Konzept

Das Design Konzept für das CMS definiert die grundlegenden Prinzipien und Zielsetzungen für die Systementwicklung unter Berücksichtigung funktionaler und nicht-funktionaler Anforderungen sowie potenzieller Risiken.

Die Kernziele des Design Konzepts umfassen die Benutzerfreundlichkeit, eine effiziente Verwaltung von Inhalten sowie eine nahtlose Integration und Skalierbarkeit. Die intuitive Benutzeroberfläche soll die Erstellung und Verwaltung von AR-Inhalten auch für technisch weniger versierte Nutzer ermöglichen. Das System zielt darauf ab, verschiedene AR-Inhaltstypen wie Text, Audio, Bilder, Videos und 3D-Modelle effizient zu verwalten. Eine reibungslose Integration zwischen dem CMS, dem GitHub-Repository und der AR-Anwendung soll konsistenten Datenfluss und kollaboratives Arbeiten fördern. Die Implementierung einer AR-Vorschau ermöglicht das direkte Testen von Inhalten im System, was die Qualitätskontrolle verbessert und Nachkorrekturen minimiert. Das CMS wird auf Skalierbarkeit ausgelegt, um mit wachsenden Inhaltsmengen und steigenden Nutzerzahlen umgehen zu können und so langfristige Effizienz zu gewährleisten.

Bei der Entwicklung des CMS müssen mehrere potenzielle Risiken berücksichtigt werden. Die technische Komplexität der Integration verschiedener Systeme (CMS, GitHub, AR-Anwendung) stellt eine zentrale Herausforderung dar. Es besteht zudem das Risiko mangelnder Benutzerakzeptanz, falls Nutzer das neue System als zu komplex empfinden. Performance-Probleme bei hoher Datenlast oder Nutzerzahl sowie Kompatibilitätsschwierigkeiten mit verschiedenen Endgeräten sind weitere mögliche Herausforderungen. Die Abhängigkeit von GitHub als zentrales Repository birgt potenzielle Risiken. Nicht zuletzt muss der Schulungsaufwand für Nutzer in der Projektplanung berücksichtigt werden.

Dieses Design Konzept bildet einen anpassungsfähigen Rahmen für die Entwicklung des CMS, der die Hauptanforderungen adressiert und gleichzeitig potenzielle Herausforderungen berücksichtigt. Es kann im Projektverlauf weiter verfeinert und an spezifische Erkenntnisse angepasst werden.

# 4. Konzeption

Dieses Kapitel verbindet die vorangegangene Analyse und die nachfolgende Entwicklung des Content-Management-Systems. Aufbauend auf den Erkenntnissen aus der Benutzermodellierung und der Anforderungsanalyse wird in diesem Abschnitt ein detaillierter Ablauf entwickelt, der die Grundlage für die praktische Umsetzung des Systems legt.

Die Konzeption gliedert sich dabei in mehrere Kernbereiche: Zunächst werden die Systemarchitektur und die Technologieauswahl erörtert. Darauf aufbauend erfolgt der Entwurf des Datenmodells, welches die Verwaltung der AR-Inhalte ermöglicht. Die Gestaltung der Benutzeroberfläche und der User Experience steht im Fokus des dritten Abschnitts, mit dem Ziel, ein System zu konzipieren, das nicht nur funktional, sondern auch intuitiv bedienbar ist. Abschließend wird das Schnittstellendesign erarbeitet, um eine nahtlose Integration des CMS in die bestehende Projektstruktur des Kooperationsprojekts zu gewährleisten.

## 4.1. Systemarchitektur und Technologieauswahl

Die Konzeption der Systemarchitektur und die Auswahl geeigneter Technologien sind entscheidend für die Entwicklung des CMS. Basierend auf den Anforderungen des Projekts und den spezifischen Herausforderungen der Verwaltung von AR-Inhalten wird eine Architektur entworfen, die Flexibilität, Skalierbarkeit und einfache Wartbarkeit gewährleistet.

### 4.1.1. Gesamtarchitektur

Für das CMS wird eine klassische Architektur mit klarer Trennung von Frontend und Backend gewählt. Diese Entscheidung ermöglicht eine unabhängige Entwicklung und Skalierung beider Systemteile. Die Kommunikation zwischen Frontend und Backend erfolgt über eine Representational State Transfer (REST) Application Programming Interface (API), was eine standardisierte, zustandslose Interaktion gewährleistet und die Integration zusätzlicher Anwendungen oder Dienste erleichtert (Masse, 2011).

Eine zentrale Komponente der Architektur ist die Verwendung von GitHub als primären Service für die Datenspeicherung, da das Kooperationsprojekt die eigene Struktur in einem Repository festhält. Diese Herangehensweise nutzt die Vorteile der Versionskontrolle und des kollaborativen Workflows von GitHub für die Verwaltung von AR-Inhalten, wobei der Pull-Request-Mechanismus eine strukturierte Überprüfung und Freigabe von Änderungen ermöglicht. Zur Vereinfachung des Deployments und zur Gewährleistung konsistenter Entwicklungs- und Produktionsumgebungen wird eine Docker-basierte Containerisierung eingesetzt. Dies erleichtert zum einen die initiale

## 4. Konzeption

Einrichtung und konsistente Developer Experience, zum anderen auch die spätere Skalierung und das Deployment der Anwendung.

### 4.1.2. Technologieauswahl

Die Auswahl der Technologien erfolgt unter Berücksichtigung der Anforderungen des CMS sowie des eigenen Know-Hows. Für das Frontend soll Vue.js 3 in Kombination mit TypeScript eingesetzt werden, was ein modernes, reaktives Javascript-Framework ist und die Entwicklung von Komponentenbasierten Benutzeroberflächen ermöglicht. Als Build-Tool kommt Vite zum Einsatz, das durch schnelles Hot-Module-Replacement die Entwicklererfahrung verbessert. Das UI-Styling wird durch Tailwind CSS realisiert, ein Utility-First CSS-Framework, das eine flexible und effiziente Gestaltung der Benutzeroberfläche ermöglicht.

Das Backend basiert auf Node.js mit Express.js und TypeScript, eine Kombination, die eine bewährte, schnelle Laufzeitumgebung mit verbesserter Typsicherheit bietet. Für die Datenbankschicht wird MongoDB eingesetzt, das mit seiner flexiblen, dokumentenbasierten Struktur gut zu den variablen Anforderungen der Benutzerverwaltung passt. Die Nutzung des GitHub-Repositories als Datenspeicher für alle Inhalte gilt als primäre Anforderung. Die Verwendung dessen ermöglicht eine transparente Versionskontrolle und einen kollaborativen Workflow für Inhaltsänderungen.

Der gesamte Technologie-Stack umfasst weitere wichtige Komponenten wie Pinia für State Management, Vue Router für das clientseitige Routing, JWT für die Authentifizierung und Zod für die Schema-Validierung im Backend. Für die Entwicklung und das Deployment kommen Docker, docker-compose und Render.com zum Einsatz, während VS Code, ESLint, Prettier und Git als Entwicklungswerkzeuge dienen.

Diese sorgfältig ausgewählte Kombination von Technologien bildet eine solide Grundlage für die Entwicklung eines leistungsfähigen, skalierbaren und wartbaren CMS. Sie ermöglicht eine effiziente Umsetzung der Projektanforderungen und bietet gleichzeitig die notwendige Flexibilität für zukünftige Erweiterungen und Anpassungen.

## 4.2. Datenmodellierung

Die vorhandenen Datenstrukturen des Kooperationsprojektes werden in diesem Schritt evaluiert und dargestellt. Die Analyse der bestehenden Strukturen der Inhaltsspeicherung und die Konzeption einer ersten Datenmodellierung sind entscheidend für die im Kapitel 5 durchzuführende Entwicklung des CMS.

### 4.2.1. Vorhandene Datenstruktur

Das Kooperationsprojekt nutzt ein GitHub-Repository zur Speicherung und Versionierung der Inhalte. Die Struktur des Repositories ist hierarchisch aufgebaut und folgt einem definierten Organisationsprinzip. Der Hauptordner „src/“ enthält, neben dem Entwicklungscode für die eigentliche Anwendung, alle relevanten AR-Inhalte, wobei sich innerhalb dieses Ordners Unterordner für jede Route befinden. Jede Route wird durch einen eigenen Ordner repräsentiert, welcher immer eine „index.md“-Datei für allgemeine Informationen über die Route selbst und weitere „.md“-Dateien für die

jeweiligen Point of Interest (POI)s enthält. Medieninhalte sind in spezifischen Unterordnern organisiert, wobei der Ordner „images/“ für Bilder vorgesehen ist und einen „small/“ Unterordner für Thumbnails enthält. AR-spezifische Medien werden im Ordner „ar-media/“ gespeichert, der wiederum in die Unterordner „audios/“, „videos/“, „models/“ und „images/“ unterteilt ist (Abbildung 4.1).

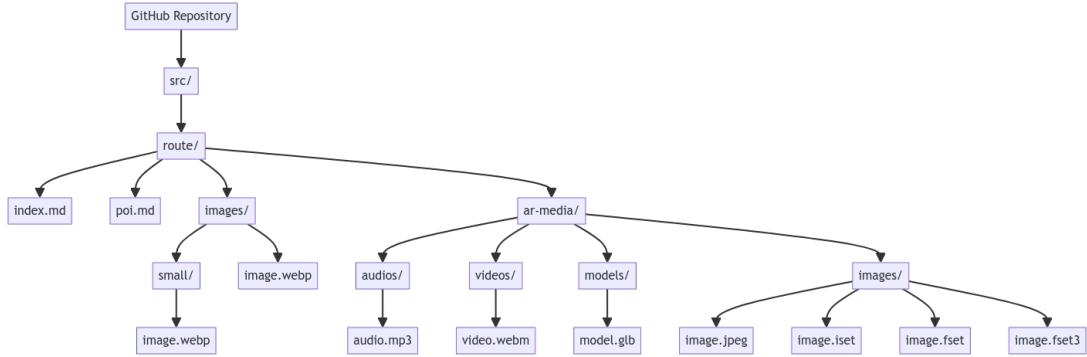


Abbildung 4.1.: Verzeichnisstruktur GitHub-Repository

Diese Struktur bietet Vor- sowie Nachteile. Die Routen werden zusammen mit den POIs und den dazugehörigen AR-Inhalten getrennt gespeichert und bilden, wie in einer Datenbank, einzelne Datenblöcke. Die Speicherung der Daten erfolgt jedoch zusammen mit dem Anwendungscode, wodurch keine klare Trennung zwischen den Routen und der Anwendungslogik vorhanden ist.

#### 4.2.2. Speicherung von Routen, POIs und Inhalten

Basierend auf der vorhandenen Struktur des CMS kann ein Datenmodell mit den Entitäten und den dazugehörigen Beziehungen zueinander konzipiert werden.

Routen werden als übergeordnete Entitäten betrachtet und enthalten grundlegende Informationen wie Id, Titel, Layout, Bild und Typ. Diese Informationen werden in der „index.md“-Datei der jeweiligen Route gespeichert.

Jeder POI wird durch eine eigene „.md“-Datei repräsentiert und enthält detaillierte Informationen wie Titel, Bild, Typ, Layout, Google Maps-Link, Koordinaten, allgemeine Informationen und AR-spezifische Beschreibungen. Die Id für einen POI wird dabei über den Dateinamen der „.md“-Datei abgeleitet.

AR-spezifische Inhalte werden in einer AR-Konfiguration zusammengefasst, die den Typ des AR-Erlebnisses, den Inhalt und den Standort definiert. Diese Konfiguration kann verschiedene Medientypen wie Audio, Video oder 3D-Modelle einschließen, die in den entsprechenden Unterordnern von „ar-media/“ gespeichert werden (Abbildung 4.2).

Eine Route kann mehrere POIs enthalten und kann somit durch eine 1:n-Beziehung dargestellt werden. Gleichzeitig ist jeder POI genau einer Route zugeordnet und somit

#### 4. Konzeption

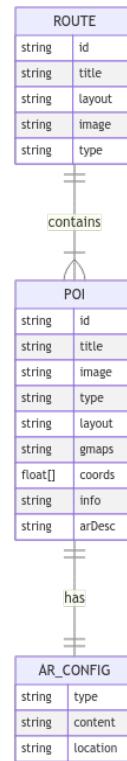


Abbildung 4.2.: Entity Relationship Diagramm der Projektdaten (Teil 1)

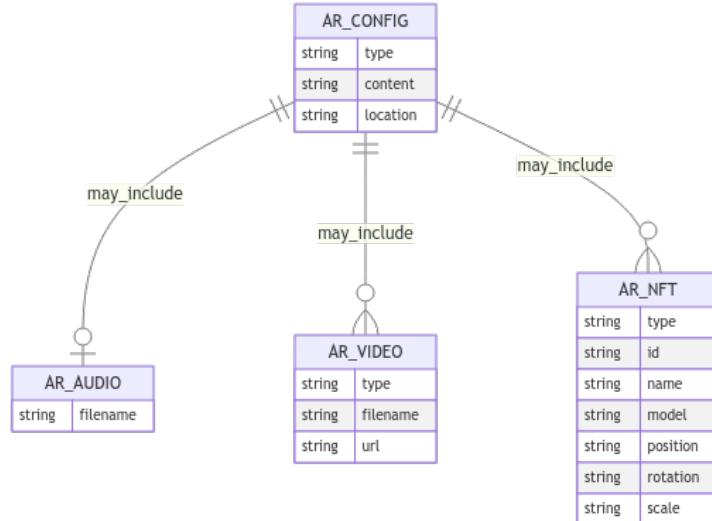


Abbildung 4.3.: Entity Relationship Diagramm der Projektdaten (Teil 2)

eine 1:1-Beziehung. Jeder POI verfügt über eine eigene AR-Konfiguration, was ebenfalls einer 1:1-Beziehung entspricht. Die AR-Konfiguration selbst kann verschiedene

#### 4. Konzeption

Medientypen wie Audio, Video oder NFT beinhalten, was wiederum eine 1:n-Beziehung darstellt (Abbildung 4.3).

Diese Datenmodellierung spiegelt die vorhandene Projektstruktur innerhalb des GitHub-Repositories wieder und stellt für die weitere Bearbeitung die Datengrundlage für die Entwicklung des CMS dar.

### 4.3. Benutzeroberfläche und User Experience

Basierend auf den Erkenntnissen aus vorherigen Abschnitten werden Wireframes für die wichtigsten Funktionsbereiche des Systems erstellt und dienen als visuelle Repräsentation der Benutzeroberfläche.

#### 4.3.1. Wireframes

Zu den Hauptansichten des CMS gehören die Startseite mit einer Übersicht der verfügbaren Routen, eine Ansicht der POIs einer Route, eine POI Detailseite und eine AR-Vorschau-Funktion. Diese Ansichten sollen die Kernfunktionalitäten des Systems repräsentieren und berücksichtigen zudem die spezifischen Anforderungen an die Erstellung und Verwaltung der AR-Inhalte.

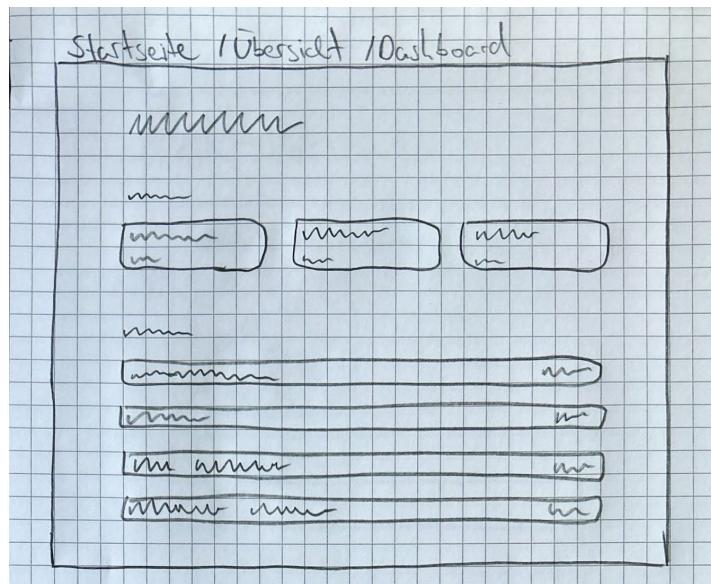


Abbildung 4.4.: Wireframe - Startseite

Die Startseite bietet einen schnellen Überblick über die verfügbaren Routen und zeigt die letzten Änderungen an (Abbildung 4.4). Dies ermöglicht den Nutzern einen effizienten Einstieg in ihre Arbeit und hält sie über aktuelle Entwicklungen auf dem Laufenden.

Die Routen-POI-Übersicht visualisiert die einzelnen Stationen einer Route und ermöglicht eine intuitive Navigation und Verwaltung der AR-Inhalte (Abbildung 4.5). Die

#### 4. Konzeption

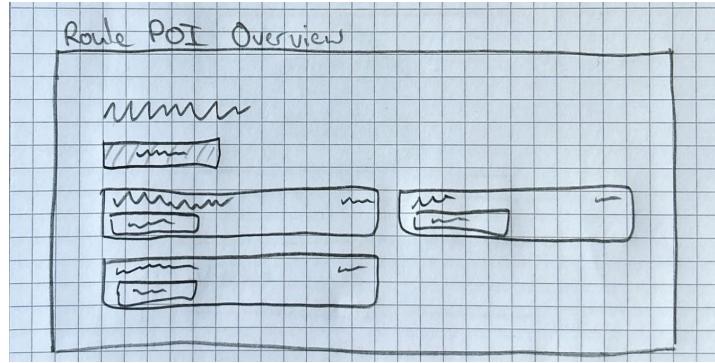


Abbildung 4.5.: Wireframe - POI Übersicht

POI-Detailansicht stellt eine Übersicht über alle bereitstehenden Informationen eines Spots dar (Abbildung 4.6).

Die AR-Vorschau-Funktion integriert verschiedene Medientypen wie QR-Codes, 3D-Modelle, Videos, Audio und Bilder in einer einzigen Ansicht, was eine umfassende Kontrolle und Optimierung der AR-Erlebnisse ermöglicht (Abbildung A.5).

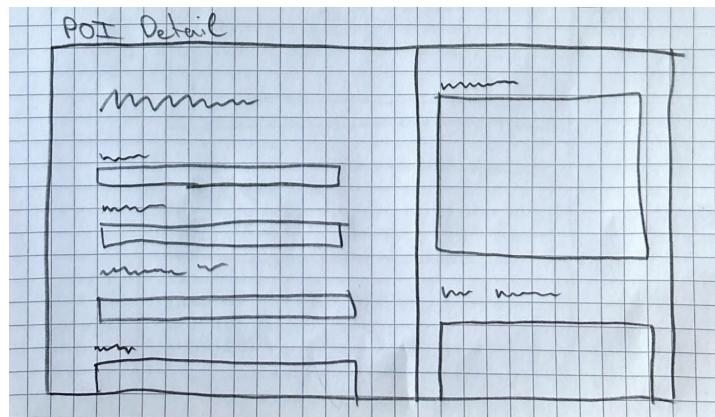


Abbildung 4.6.: Wireframe - POI-Detailseite

#### 4.3.2. Navigationskonzept und Informationsarchitektur

Das Navigationskonzept folgt dem hierarchischen Ansatz des Projektrepositories, welches die natürliche Struktur der AR-Inhalte widerspiegelt. Von der Übersicht der Routen können Nutzer zu den einzelnen POIs navigieren und von dort aus die spezifischen AR-Inhalte bearbeiten und vorschauen. Diese Struktur entspricht dem mentalen Modell der Nutzer und fördert eine intuitive Navigation durch das System (Carroll u. Olson, 1988).

Die Informationsarchitektur wird so konzipiert, dass sie die wichtigsten Informationen und Funktionen auf jeder Ebene leicht zugänglich macht. Auf der Startseite werden beispielsweise die verfügbaren Routen und die letzten Änderungen platziert, während

auf der POI-Detailseite alle relevanten Informationen und Bearbeitungsmöglichkeiten übersichtlich in einem Formular angeordnet sind.

#### 4.4. Schnittstellen-Design

In diesem Abschnitt soll das Schnittstellen-Design erarbeitet werden, welches die Kommunikation zwischen den Systemkomponenten und externen Diensten ermöglichen soll. Die konzeptionelle Planung der API-Endpunkte orientiert sich an den zuvor genannten REST-Prinzipien. Dabei sollen Endpunkte für alle wesentlichen Ressourcen des Systems definiert werden, einschließlich Routen, POIs und AR-Medien. Die Endpunkte sind so gestaltet, dass sie die grundlegenden CRUD-Operationen unterstützen und gleichzeitig spezifische Datei Funktionen, wie die Verwaltung von AR-Inhalten, ermöglichen.

Für den Datenaustausch zwischen Frontend und Backend wird JSON als Format gewählt, da hiermit eine einfache Integration innerhalb von JavaScript-basierte Anwendungen garantiert werden kann (Bourhis u. a., 2017).

Für die Authentifizierung und Autorisierung wird ein tokenbasierter Ansatz mittels JSON Web Tokens (JWT) gewählt (Jones u. a., 2015). JWT bieten dabei eine sichere und zustandslose Authentifizierung, die gut zu dem gewählten Austausch innerhalb der Systemstruktur passen. Zudem werden verschiedene Benutzerrollen definiert, um eine Zugriffskontrolle auf API-Ressourcen und Funktionen in der Benutzeroberfläche zu ermöglichen.

Bei der Planung der Integrationsstrategien mit dem Kooperationsprojekt steht die nahtlose Einbindung in die bestehende Infrastruktur im Vordergrund. Besondere Aufmerksamkeit wird der Integration mit dem GitHub-Repository des Projekts gewidmet. Hierfür werden spezielle API-Endpunkte konzipiert, die es ermöglichen, Änderungen an AR-Inhalten direkt als Pull-Requests im Repository zu erstellen und zu verwalten. Dies gewährleistet eine transparente Versionskontrolle und erleichtert den kollaborativen Workflow bei der Inhaltserstellung.

Das Schnittstellen-Design berücksichtigt zudem die spezifischen Anforderungen der AR-Funktionalitäten. Es werden dedizierte Endpunkte für die Verwaltung und den Abruf von AR-Inhalten konzipiert. Abschließend wird bei der Konzeption der Schnittstellen auch die Dokumentation und Testbarkeit berücksichtigt. Eine umfassende API-Dokumentation wird im Verlauf der Entwicklung gepflegt und kann im Wiki des CMS-Repositories eingesehen werden (Zaitsev, 2024a).

# 5. Entwicklung

Dieses Kapitel beschreibt den Entwicklungsprozess des CMS und umfasst die Entwicklungsumgebung und das Setup, die Backend- und Frontend-Entwicklung sowie die Umsetzung der Kernfunktionalitäten. Hervorgehoben werden die Integration mit GitHub und die Implementierung der AR-Vorschaufunktion. Die gewählten Technologien und Methoden werden erläutert und ihre Anwendung im Kontext des Projekts dargestellt. Die Entwicklung des Systems wird in einem Repository auf GitHub verwaltet (Zaitsev, 2024b).

## 5.1. Entwicklungsumgebung und Setup

Das Projekt ist in die zwei Hauptkomponenten Frontend und Backend unterteilt. Diese befinden sich in separaten Verzeichnissen innerhalb des GitHub-Repositories. Im Backend-Verzeichnis liegt im „src/“-Verzeichnis die serverseitige Logik, darunter eine klassische Aufteilung der API in „routes“, „controllers“ und „models“. Zudem werden hier die Datenbankverbindung, Schema, die Services und Views definiert (Abbildung 5.1).

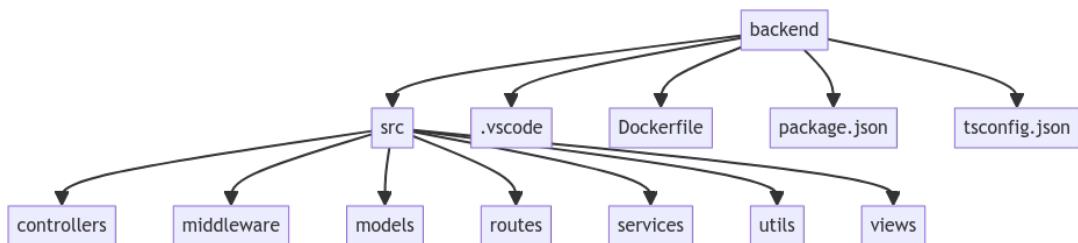


Abbildung 5.1.: Aufbau CMS Projektverzeichnis - Backend

Das Frontend-Verzeichnis enthält im „src/“-Verzeichnis den Code für alle Frontend relevanten Komponenten des Systems. Darunter fallen die Components, Routing-Konfigurationen, Views, das State-Management, die Logik zur Verbindung mit der Backend-API und sonstige UI-Elemente (Abbildung 5.2). Im Backend als auch im Frontend werden Umgebungsvariablen über lokale „.env“-Dateien in die Anwendungen importiert.

Als Texteditor bzw. Entwicklungsumgebung wird Visual Studio Code, unterstützt durch verschiedene Erweiterungen zur Optimierung des Entwicklungsprozesses, verwendet. Die Endpunkte der Backend-API werden mithilfe der API-Development-Plattform Insomnia getestet, wobei Google Chrome als Webbrowser für Tests und Debugging der Benutzeroberfläche eingesetzt wird.

Das eigentliche Projekt-Setup erfolgt mittels Docker Compose, was eine konsisten-

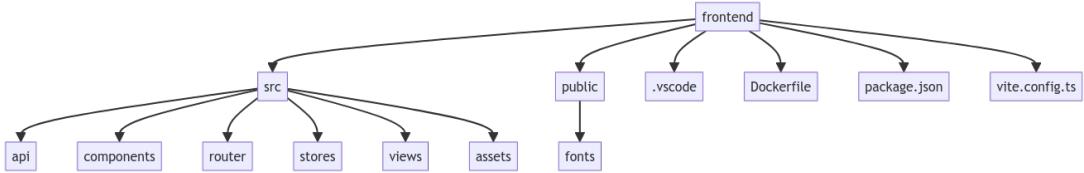


Abbildung 5.2.: Aufbau CMS Projektverzeichnis - Frontend

te Entwicklungsumgebung über verschiedene Systeme hinweg gewährleistet. Frontend, Backend und eine MongoDB werden in separaten Containern gestartet. Lokal wird eine MongoDB-Instanz direkt über Docker Compose initialisiert, während in der Produktionsumgebung eine MongoDB Atlas Cloud-Instanz zum Einsatz kommt.

Die Versionskontrolle wird über Git und GitHub realisiert. Da die Entwicklung von einem einzelnen Entwickler durchgeführt wird, erfolgten alle Änderungen direkt im Main-Branch. Hierdurch können Änderungen schneller und ohne weiteren Overhead angewandt werden. Für zukünftige Entwicklungen wird jedoch ein Feature-Branch-Workflow mit Pull Requests empfohlen (Loeliger u. McCullough, 2012).

Hinsichtlich der CI/CD-Pipeline wird im Entwicklungsprozess ein schrittweiser Ansatz gewählt. Zunächst konzentriert sich die Entwicklung auf das lokale Testing und Deployment. Im weiteren Verlauf wird eine automatisierte Deployment-Strategie über Render als kostenfreien Cloud-Anbieter implementiert. Diese löst bei Pushes in den Main-Branch einen automatisierten Build und ein Deployment aus, wobei nur geänderte Systemkomponenten neu deployed werden.

## 5.2. Backend-Entwicklung

Die Implementierung des Backends bildet den ersten Schritt innerhalb der Entwicklung des Content-Management-Systems.

Für das Backend wird ein moderner und vertrauter Technologie-Stack gewählt, der auf Node.js als plattformübergreifende Open-Source-JavaScript-Laufzeitumgebung und Express als Web-Framework basiert. Die Verwendung von TypeScript als Programmiersprache ermöglicht eine Typsicherheit und erleichtert die Wartbarkeit des Codes (Bierman u. a., 2014).

In diesem Abschnitt wird die Implementierung der Serverarchitektur und der grundlegenden Strukturen erläutert, die für die Bereitstellung der AR-Inhalte des Kooperationsprojekts von Bedeutung sind.

### 5.2.1. API-Design und Routing

Bei der Entwicklung der API-Struktur und des Routings wird besonders Wert auf ein konsistentes Design gelegt, das den RESTful-Prinzipien folgt (Fielding, 2000).

Die Implementierung der Routing-Mechanismen erfolgt mithilfe von Express, wobei die Routen in separate Module aufgeteilt werden, um die Übersichtlichkeit und Wartbarkeit zu verbessern. Jede Route wird einem spezifischen Controller zugeordnet, der die eigentliche Logik für die jeweilige Operation enthält. Dies ermöglicht eine klare

## 5. Entwicklung

Trennung von Zuständigkeiten und erleichtert zukünftige Erweiterungen.

```
1 // Public status route
2 router.use('/', statusRoutes);
3
4 // Auth routes
5 router.use('/auth', authRoutes);
6
7 // CMS routes
8 router.use('/routes', routeRoutes);
9 router.use('/routes', poiRoutes);
10 router.use('/routes', arMediaRoutes);
11 router.use('/routes', imageRoutes);
12 router.use('/pull-requests', pullRequestRoutes);
13 router.use('/ar-preview', arPreviewRoutes);
```

Quellcode 1: Backend Routing

Die Routen können dabei in zwei Bereiche aufgeteilt werden (Quellcode 1): Routen, die öffentlich aufrufbar sind und Routen, die erst nach einer Authentifizierung eine erfolgreiche Response zurücksenden. Zu den öffentlich Routen gehören eine einfache Status Rückmeldung des Servers sowie die Authentifizierungsrouten selbst, wozu eine „/register“ und eine „/login“ Route gehören.

Nach einer erfolgreichen Authentifizierung können die CMS-spezifischen Routen aufgerufen werden (Quellcode 2). Dazu gehören die Endpoint-Routen für AR-Routen, AR-POIs, AR-Medien, AR-Bilder, Pull Requests und die AR-Vorschau.

```
1 router.get('/:routeId/pois', authMiddleware,
2   → poiController.getPOIsForRoute);
3 router.get('/:routeId/pois/:poiId', authMiddleware,
4   → poiController.getPOIById);
5 router.post('/:routeId/pois', authMiddleware,
6   → poiController.createPOI);
7 router.put('/:routeId/pois/:poiId', authMiddleware,
8   → poiController.updatePOI);
9 router.delete('/:routeId/pois/:poiId', authMiddleware,
10  → poiController.deletePOI);
```

Quellcode 2: Beispiel Routing - POI Routes

Jedem API-Endpoint wird ein Controller zugewiesen. Diese haben die Aufgabe, die eigentliche Anwendungslogik beim Aufruf einer Route bereitzustellen sowie die verar-

beitete Response zurück an das Frontend zu senden.

```

1  export async function getAllRoutes(req: Request, res: Response, next:
2    → NextFunction) {
3    try {
4      const routes = await routeService.getAllRoutes();
5      res.json(routes);
6    } catch (error) {
7      next(error);
8    }
}

```

Quellcode 3: Backend Controller - „getAllRoutes“

Zur Optimierung der Codestruktur und zur Vermeidung von Redundanzen werden Middleware-Funktionen implementiert. Diese übernehmen gemeinsam genutzte Funktionalitäten wie die Authentifizierungsprüfungen (siehe Unterabschnitt 5.2.2) oder das Error-Handling. Diese Middleware-Funktionen können direkt in die Routen integriert werden und damit eine konsistente Handhabung von Anfragen gewährleisten.

### 5.2.2. Authentifizierung und Autorisierung

Die Datenpersistenz von Accountdaten wird durch MongoDB, einer dokumentenorientierte NoSQL-Datenbank, gewährleistet. Zur vereinfachten Interaktion mit der Datenbank kommt Mongoose als Object Document Mapper (ODM) zum Einsatz. Das Benutzermodell wird durch ein Mongoose-Schema definiert (Quellcode 4). Dieses Schema legt die Struktur der Benutzerdaten fest und ermöglicht eine typsichere Interaktion mit der Datenbank.

```

1  const userSchema = new mongoose.Schema<IUser>({
2    username: { type: String, required: true, unique: true },
3    password: { type: String, required: true },
4    role: { type: String, required: true },
5  });

```

Quellcode 4: User Schema

Die Implementierung der Benutzeroauthentifizierung erfolgt mithilfe von JWT. Der Prozess beginnt mit der Registrierung eines neuen Benutzers, bei der Benutzername, Passwort und Rolle erfasst werden. Nach abgeschlossener Registrierung werden anschließend beim Login die Anmelddaten überprüft, und bei erfolgreicher Authentifizierung ein JWT generiert und an den Benutzer gesendet (Abbildung 5.3).

## 5. Entwicklung

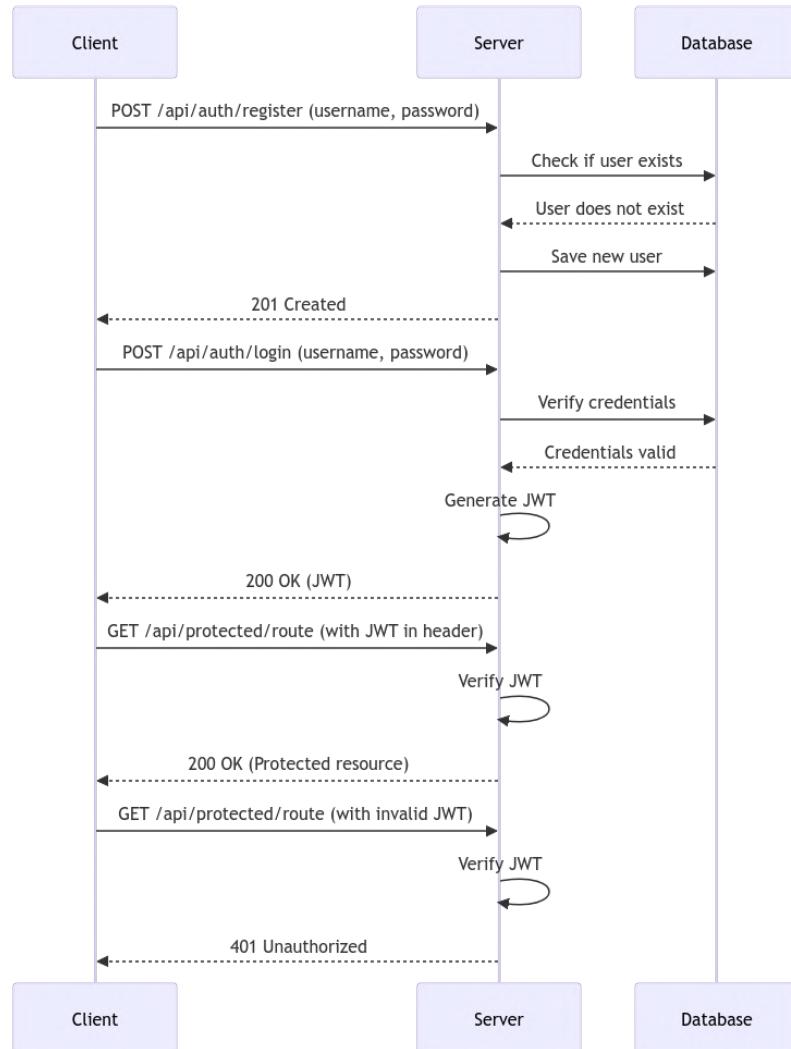


Abbildung 5.3.: Sequenzdiagramm - Authentifizierungsflow

In der aktuellen Implementierung ist die Gültigkeitsdauer des Tokens auf eine Stunde begrenzt. Für zukünftige Erweiterungen des Systems sollten fortgeschrittenere Methoden des Session-Managements, wie beispielsweise Refresh-Tokens oder Cookies, in Betracht gezogen werden.

Zur Gewährleistung der Sicherheit werden Passwörter mittels bcrypt gehasht, bevor sie in der Datenbank gespeichert werden. Bcrypt bietet einen adaptiven Hashing-Algorithmus, der durch ein Salting und mehrfache Iterationen einen robusten Schutz gegen Brute-Force- und Rainbow-Table-Angriffe bietet (Provost u. Mazieres, 1999).

Die Zugriffskontrolle auf API-Ebene wird durch die zuvor benannte Middleware realisiert, die bei jeder Anfrage den JWT verifiziert. Der Token muss dabei bei jeder gesicherten Anfrage im Authorization-Header mit übergeben werden. Die Benutzerrolle wird in der aktuellen Version des Systems nicht überprüft, kann aber in zukünftigen Iterationen ohne erhöhten Mehraufwand integriert werden.

```

1 const hashedPassword = await bcrypt.hash(password, 10);
2 const newUser = new User({
3     username,
4     password: hashedPassword,
5     role,
6 });
7
8 await newUser.save();

```

Quellcode 5: Registrierung - Passwort-Hashing mit bcrypt

```

1 const user = await User.findOne({ username });
2 if (user && (await bcrypt.compare(password, user.password))) {
3     const token = jwt.sign(
4         { userId: user._id, username: user.username, role: user.role },
5         JWT_SECRET,
6         { expiresIn: '1h' },
7     );
8     res.json({ token });
9 }

```

Quellcode 6: Login - Generierung JWT

### 5.2.3. GitHub-Integration

Die GitHub-Integration zwischen dem CMS und dem Kooperationsprojekt ist mitunter einer der wichtigsten Meilensteine innerhalb der Entwicklung. Erst durch eine erfolgreiche Implementierung dessen, können die Datenstrukturen des GitHub-Repositories auch tatsächlich verwendet werden. Für die Integration wird dabei die von GitHub zur Verfügung gestellte REST-API genutzt. Die technische Einbindung im Code findet über „Octokit“, dem offiziellen GitHub API-SDK für JavaScript und anderen Programmiersprachen, statt. Das SDK bietet eine einfache und gut dokumentierte Schnittstelle zur GitHub API (GitHub, 2024).

Die Github API wird dabei grundsätzlich kostenfrei zur Verfügung gestellt. Einschränkungen gibt es hier in Form von Rate-Limitierungen bei den Aufrufen, die in primäre und sekundäre Limits unterteilt werden können. So werden z. B. die Anfragen an die API auf 5.000 pro Stunde beschränkt und fallen unter die primären Limits. Zu den sekundären Limits zählen die Anzahl der Anfragen auf einen spezifischen Endpunkt, die Anzahl der gleichzeitigen Anfragen, die Anzahl der Anfragen pro Minute und die Anzahl der Anfragen, die neuen Content erstellen (GitHub, 2022).

Der Prozess der programmatischen Interaktion mit dem Repository umfasst mehrere Schritte, die es ermöglichen, auf verschiedene Inhalte im GitHub-Repository zuzugreifen. Im CMS werden für den Zugriff auf die API sogenannte Services im „services/-“

## 5. Entwicklung

Verzeichnis angelegt. Diese werden von den Controllern der jeweiligen Routing-Endpunkte aufgerufen und liefern die Daten zurück.

```
1 const { data: contents } = await octokit.repos.getContent({
2   owner,
3   repo,
4   path: 'src',
5 });
```

Quellcode 7: Abrufen von Routen aus dem GitHub-Repository

Zunächst werden die Services für die Abfrage von Routen und POIs aus dem Repository implementiert. Hierfür werden die Services „routeService.ts“ und „poiService.ts“ angelegt. Generell erfolgt die Abfrage der Routen und POIs über zwei Anfragen an die GitHub-API, wobei zuerst der Content eines Verzeichnisses, z. B. einer Route oder eines POIs, und anschließend der eigentliche Content der Markdown-Dateien, abgefragt werden.

```
1 const { data: file } = await octokit.repos.getContent({
2   owner,
3   repo,
4   path: `src/${routeId}/index.md`,
5 });
6
7 const content = Buffer.from(file.content, 'base64').toString('utf8');
8 const { data: frontmatter } = matter(content);
```

Quellcode 8: Abfrage von Frontmatter Inhalten einer Route

Für die Abfrage von Routen werden alle Inhalte des „src“-Verzeichnisses abgerufen und anschließend weiterverarbeitet (Quellcode 7). Dabei werden im Backend einfach-heitshalber alle existierenden Routen in der Konfiguration gespeichert, damit nicht alle Verzeichnisse überprüft werden müssen. Für jede verfügbare Route müssen anschließend die Inhalte der „index.md“-Dateien abgerufen werden. Da die Inhalte im Frontmatter-Format gespeichert werden, werden die Frontmatter-Felder mit einem externen Package ausgelesen und im festgelegten JSON Format weiterverarbeitet (Quellcode 8). Als Response erhält man alle Routen und deren Metadaten (Abbildung 5.4).

Für die Abfrage von POIs innerhalb einer Route werden ähnliche Strukturen wie bei den Routen implementiert.

Für die Abfrage der Bilder und der AR-Medien eines POIs werden ebenfalls spezifische Services angelegt, welche sich nur auf die Verarbeitung der Medien fokussieren. Da die

## 5. Entwicklung

```
1 [ 
2   {
3     "id": "strasse-der-arbeit",
4     "title": "Straße der Arbeit",
5     "layout": "path.11ty.js",
6     "image": "strasse-der-arbeit.webp",
7     "type": "metadata"
8   },
9   {
10    "id": "wiehl",
11    "title": "Wiehl",
12    "layout": "path.11ty.js",
13    "image": "wiehl.webp",
14    "type": "metadata"
15  },
16  {
17    "id": "wipperfuerth",
18    "title": "Wipperfürth",
19    "layout": "path.11ty.js",
20    "image": "wipperfuerth.webp",
21    "type": "metadata"
22  }
23 ]
```

Abbildung 5.4.: CMS Backend Response - Alle Routen

Medieninhalte zum Teil in Unterverzeichnissen hinterlegt werden, wird für die Anfrage eine Rekursive Abfrage durchgeführt. Hierdurch kann sichergestellt werden, dass z. B. alle Bilder, die auch im „small/-“-Verzeichnis gespeichert werden, innerhalb einer Abfrage abgerufen werden können.

Da die Bilder und AR-Inhalte teilweise ähnliche oder sogar gleiche Dateibezeichnungen haben, muss für die eindeutige Identifizierung ein Hash generiert werden (Quellcode 9). Damit das Backend mit dem Upload von Dateien umgehen kann, wird die Express Middleware „Multer“ installiert. Diese macht es über den „multipart/form-data“-Header einfach, die hochgeladenen Dateien über den Body eines Requests weiterzuverwenden.

```
1 function generateConsistentId(path: string): string {
2   return crypto.createHash('md5').update(path).digest('hex');
3 }
```

Quellcode 9: Generierung einer Eindeutigen ID für Medien

Für die Verwaltung von Änderungen an den Markdowndateien oder Medieninhalten wird wie zuvor ein angepasster Service zur Erstellung und Handhabung von Pull Requests erstellt. Bei einer Änderung wird dabei ein neuer einmaliger Branch und an-

## 5. Entwicklung

schließend direkt ein neuer Pull Request mit den Änderungen erstellt (Quellcode 10). Dieser Workflow bietet ein direktes Feedback und die Möglichkeit, Änderungen vor Veröffentlichung einzusehen sowie an- oder abzulehnen. Ein potenzieller Nachteil ist, dass immer nur eine Änderung pro Pull Request erstellt wird und diese nicht gesammelt gespeichert werden. Die Handhabung der GitHub-Credentials, wie auch alle weiteren

```
1 // updateRoute (routeService)
2 await pullRequestService.createPullRequest('main', branchName,
3   ↪ prTitle, prDescription);
4
5 // createPullRequest (pullRequestService)
6 const { data: pullRequest } = await octokit.pulls.create({
7   owner,
8   repo,
9   title,
10  head: headBranch,
11  base: baseBranch,
12  body,
13});
```

Quellcode 10: Erstellung eines Pull Requests

Variablen, URLs und Secrets, gelten als sensiblen Informationen und werden in einer separaten Datei für Umgebungsvariablen (.env) gespeichert und über die „.gitignore“ aus der Versionskontrolle entfernt. Der Zugriff auf diese Daten erfolgt ausschließlich über den „process.env“-Mechanismus von Node.js und anschließend über den Aufruf durch eine globale Konfigurationsdatei (Quellcode 11).

```
1 const octokit = new Octokit({
2   auth: config.githubPersonalAccessToken,
3 });
```

Quellcode 11: Sichere Initialisierung von Octokit mit GitHub-Token

Der personalisierte GitHub-Token muss bei der Einrichtung des CMS in den Umgebungsvariablen angegeben werden. Damit kein persönlicher GitHub-Account die Änderungen am Projekt-Repository vornimmt, sollte hier in einer öffentlichen Version in den Einstellungen der GitHub-Organisation die entsprechende Einstellung für die Access-Tokens gesetzt werden (Abbildung A.6).

### 5.3. Frontend-Entwicklung

Das Frontend stellt die Schnittstelle zwischen Nutzern des CMS und den über das Backend verwalteten Daten des Kooperationsprojekts dar. In diesem Kapitel wird der

## *5. Entwicklung*

Prozess der Frontend-Entwicklung aufgezeigt, wobei vor allem auf die Umsetzung der Benutzeroberfläche und den Interaktionsmöglichkeiten mit dieser eingegangen wird.

Für die Entwicklung des Frontends wird wie auch im Backend ein moderner und vor allem vertrauter Technologie-Stack gewählt, der durch seine Flexibilität und Performance punktet. Als Basis dient Vue.js 3, ein progressives Web-JavaScript-Framework, das sich durch seine reaktive Komponenten-Architektur auszeichnet. Auch hier wird TypeScript für eine verbesserte Typsicherheit und erhöhte Codequalität verwendet. Als Build-Tool kommt Vite zum Einsatz, das durch schnelles Hot Module Replacement und optimierte Builds die Entwicklererfahrung signifikant verbessert.

Bei der Umsetzung der Benutzeroberfläche wurde im ersten Schritt die Entwicklung eines eigenen Designsystems in Erwägung gezogen. Nach sorgfältiger Abwägung wurde sich für die Verwendung von Tailwind CSS, einem Utility-First CSS-Framework, entschieden. Diese Wahl ermöglicht eine effiziente und flexible Gestaltung der Benutzeroberfläche bei gleichzeitiger Gewährleistung von Konsistenz und Wartbarkeit.

Die Unterstützung von verschiedenen Gerätetypen und eines responsiven Designs kann durch die konsequente Nutzung der Grid-Funktionalitäten von Tailwind hergestellt werden. Dieser Ansatz gewährleistet eine optimale Darstellung der Anwendung auf unterschiedlichen Endgeräten.

Durch die Verwendung von Vue.js wird in der Entwicklung auf den Einsatz von wiederverwendbaren UI-Komponenten gesetzt. Dies fördert unter anderem die Wiederverwendbarkeit von Code, die Konsistenz des Designs sowie die Wartung und Erweiterung der Anwendung. Für das globale State Management wird Pinia eingesetzt, wobei der Schwerpunkt zunächst auf der Verwaltung der Authentifizierungsdaten liegt. Hierbei ist anzumerken, dass eine umfassendere Nutzung des Stores, z. B. für die Verwaltung der CMS-Inhalte selbst, in dieser Version nicht zum Einsatz kommt und daher für zukünftigen Iterationen in Betracht gezogen werden sollte.

Die Implementierung des Client-Side Routings erfolgte mittels Vue Router. Durch den Einsatz von Client-Side Routing kann eine Single Page Application (SPA) bereitgestellt werden, welche dynamisch auf die Eingaben eines Nutzers reagiert und das Gefühl einer klassischen Webseite vermittelt. Für die Anbindung an die Backend-API wurde eine angepasste Fetch-Funktion entwickelt, die die Authentifizierung über den JWT-Token automatisch handhabt. Dieser Ansatz vereinfacht die API-Aufrufe erheblich, da die Authentifizierung nicht bei jedem einzelnen Aufruf manuell implementiert werden muss.

In den folgenden Abschnitten werden die spezifischen Aspekte der Frontend-Entwicklung, einschließlich des Routings und der Navigation der Komponenten-Struktur, detailliert erläutert.

### **5.3.1. Routing und Navigation**

Das Routing und die Navigation im Frontend basieren auf Vue-Router, der offiziellen Routing-Bibliothek für Vue.js. Diese ermöglicht ein Client-Side Routing ohne vollständiges Neuladen der Anwendung. Die Navigationsstruktur umfasst dabei die Hauptbereiche: Startseite, Routen, Routen-Detailseite, Routen-POIs, POI-Detailseite und die Änderungsseite. Jeder Bereich wird durch eine eindeutige Route repräsentiert, was eine

## 5. Entwicklung

intuitive Benutzerführung gewährleistet.

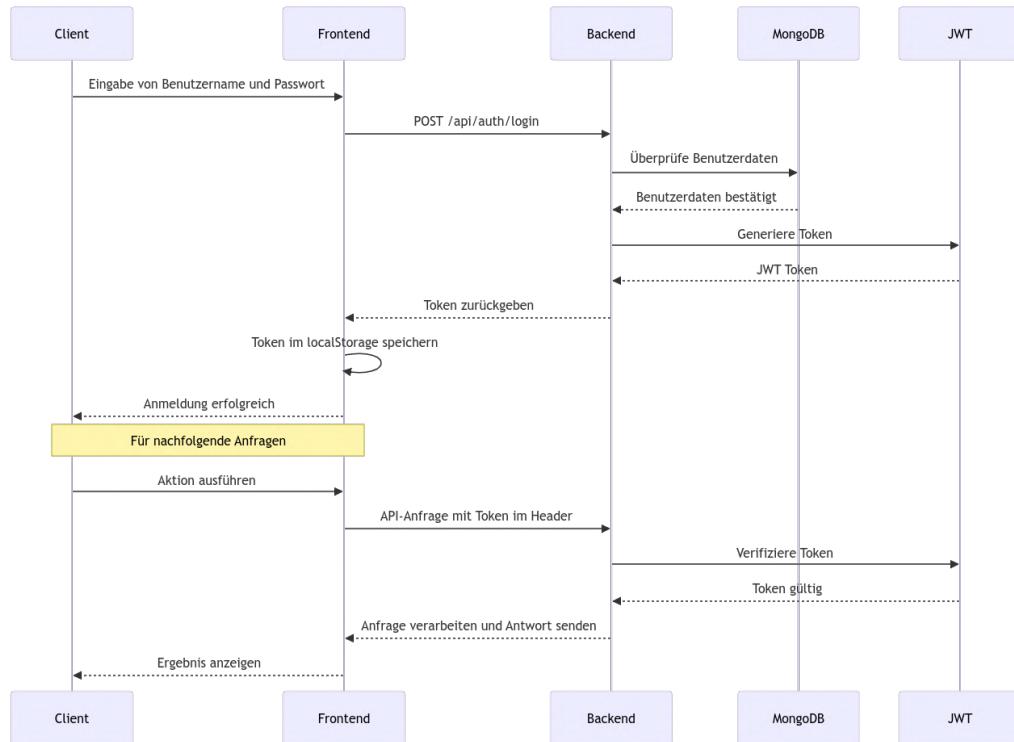


Abbildung 5.5.: CMS Frontend - JWT Authentifizierung Sequenz

Ein Kernaspekt der Implementierung sind die Navigation Guards (Quellcode 12). Diese überprüfen bei jeder Navigation das Vorhandensein eines gültigen JWT im Local Storage des Browsers. Fehlt der Token oder ist er abgelaufen, erfolgt eine automatische Umleitung zur Login-Seite (Abbildung 5.5). Dies stellt sicher, dass nur authentifizierte Benutzer auf geschützte Bereiche zugreifen können.

```

1 router.beforeEach((to, from, next) => {
2   const authStore = useAuthStore();
3   if (to.matched.some((record) => record.meta.requiresAuth) &&
4     !authStore.isAuthenticated) {
5     next('/login');
6   } else {
7     next();
8   }
9 });
  
```

Quellcode 12: Implementierung der Navigation Guards

Der JWT-Lebenszyklus wird hierbei aktiv verwaltet: Bei der Anmeldung wird der

## 5. Entwicklung

Tokens gespeichert, bei der Abmeldung entfernt. Ein zusätzlicher Mechanismus löscht abgelaufene Tokens bei der nächsten Benutzerinteraktion, was eine erneute Anmeldung erforderlich macht. Diese Maßnahme erhöht die Sicherheit gegen unbefugte Zugriffe. Die Implementierung dynamischer Routen und Parameterübergabe ermöglicht eine flexible Navigation, insbesondere für die Darstellung spezifischer Inhalte wie einzelner POIs oder Medienelemente. Routenparameter erlauben den direkten Aufruf von Detailansichten über die URL, was die Benutzerfreundlichkeit und die Möglichkeiten zur Verlinkung verbessert.

### 5.3.2. Komponenten-Struktur

Die Komponenten-Struktur des Frontends basiert auf dem komponentenbasierten Entwicklungsprinzip von Vue.js und bietet die Möglichkeit, bereits geschriebenen Code wiederzuverwenden. Diese Struktur gliedert sich in die zwei Kategorien Views und Components.

Views fungieren als Ausgangspunkt für die Vue-Routen und repräsentieren vollständige Seitenansichten. Das System umfasst mehrere zentrale Views: LoginView für die Authentifizierungsschnittstelle, HomeView als Dashboard mit Übersicht, RoutesView zur Auflistung verfügbarer Routen, RouteDetailView für die Detailansicht einer spezifischen Route, POIDetailView für Detailinformationen zu einem Point of Interest und ChangesView zur Übersicht über Änderungen bzw. Pull Requests.

Components hingegen sind wiederverwendbare Elemente, die in verschiedenen Views oder auch anderen Components zum Einsatz kommen. Sie sind nach Funktionsbereichen gegliedert und umfassen Komponenten für die Verwaltung von AR-Routen, die Darstellung und Bearbeitung von Points of Interest, die Handhabung von AR-Medieninhalten sowie die Verwaltung von Änderungsanfragen. Diese Struktur fördert die Wiederverwendbarkeit des Codes und erleichtert die Wartung. Ein Beispiel hierfür sind die Routen- oder Änderungslisten, die sowohl auf der HomeView als auch jeweils in der RouteDetailView oder ChangesView Verwendung finden.

Die Entwicklung der Komponenten folgt dem Prinzip der Einzelverantwortlichkeit, wobei jede Komponente eine spezifische Aufgabe übernimmt. Dies verbessert die Testbarkeit und Wartbarkeit des Systems. Das Zustandsmanagement innerhalb der Komponenten wird durch die Verwendung von Vue 3's Composition API realisiert, die eine flexiblere und effizientere Handhabung des Komponentenzustands ermöglicht.

## 5.4. Umsetzung der Kernfunktionalitäten

Die Implementierung der Kernfunktionalitäten des CMS umfasst die Verwaltung verschiedener Inhaltstypen, darunter Routen, POIs, Medien und Änderungen. Jeder dieser Bereiche erfordert spezifische Funktionen und Benutzeroberflächen, die im Folgenden beschrieben werden.

Im ersten Schritt wird die Loginseite implementiert. Diese stellt das Formular für die Eingabe der Nutzerdaten bereit (Abbildung A.7). Nach einer erfolgreichen Anmeldung werden Nutzer auf die Startseite geleitet. Hier, sowie auf jeder weiteren Seite, wird eine

## 5. Entwicklung

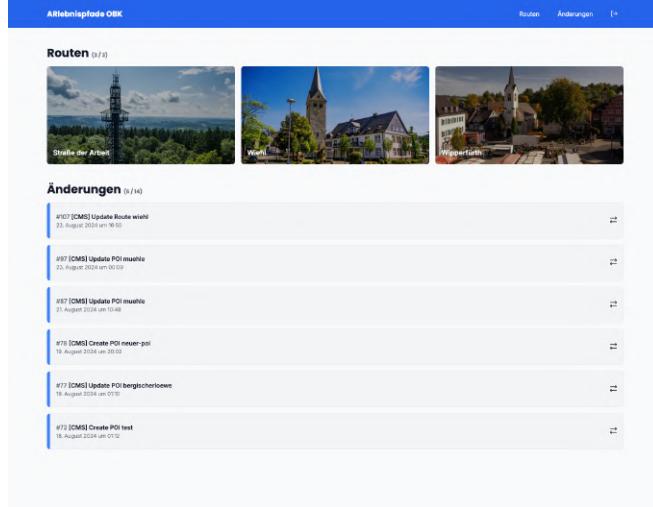


Abbildung 5.6.: Benutzeroberfläche - Startseite

Navigation im Header angezeigt. In dieser kann über das Logo zurück auf die Startseite, direkt zu den Routen oder zu den Änderungen gewechselt werden. Außerdem befindet sich hier der Logout-Button. Die Startseite bietet einen direkten Überblick über die Routen und zeigt für diese die Namen und Thumbnails an. Außerdem werden auf dieser Seite im unteren Bereich die letzten Änderungen angezeigt (Abbildung 5.6). Über einen Klick auf die „Routen“-Headline oder über die Navigation kann direkt auf die Verwaltung aller Routen gesprungen werden. Hier werden die Daten einer Routen „index.md“-Datei angezeigt und können mit einem Klick auf „Bearbeiten“ direkt angepasst werden (Abbildung A.8).

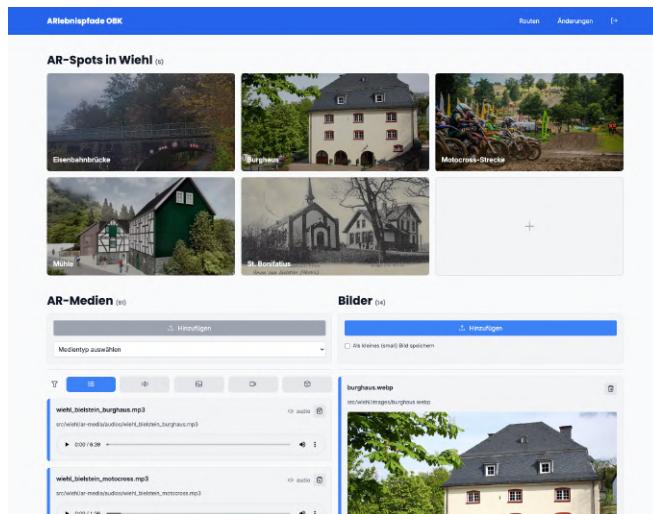


Abbildung 5.7.: Benutzeroberfläche - Routen Detailansicht

Durch einen Klick auf das Thumbnail einer Route kann in die Detailansicht der Route gewechselt werden. Hier werden alle verfügbaren POIs, alle AR-Medien sowie alle Bil-

## 5. Entwicklung

der innerhalb der Route angezeigt (Abbildung 5.7). Ein neuer POI kann über die letzte Kachel innerhalb der POI-Liste erstellt werden. Genauso können neue AR-Medien oder Bilder direkt auf dieser Seite hochgeladen werden (Abbildung A.9).

Eine POI-Kachel leitet hierbei auf die POI-Detailansicht weiter und zeigt alle verfügbaren Informationen eines AR-Spots an (Abbildung 5.8). Das Thumbnail des POI wird im oberen Bereich neben dem QR-Code für die AR-Vorschaufunktion angezeigt. Darunter befindet sich, falls für den POI ein 3D-Modell angegeben ist, das interaktive 3D-Modell, welches über den Google Model-Viewer angezeigt wird. Im unteren Bereich der Seite werden die Frontmatter-Inhalte des POIs in einem Formular geladen, welches über einen Button am Ende der Seite zur Bearbeitung freigegeben wird (Abbildung A.10).

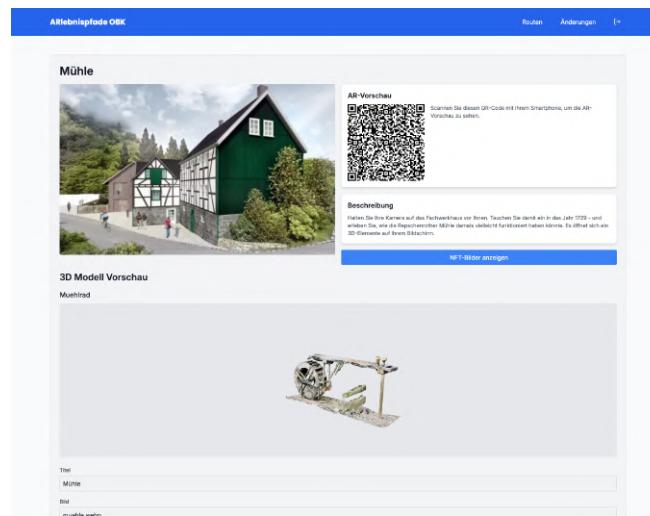


Abbildung 5.8.: Benutzeroberfläche - POI Detailansicht

Jede erfolgreiche Änderung und jeder Dateiupload muss vor der Veröffentlichung durch eine manuelle Prüfung entweder angenommen oder abgelehnt werden. Dabei werden alle Änderungen auf der gleichnamigen Seite angezeigt (Abbildung 5.9). Diese lädt alle vom CMS vorgenommenen Pull Requests über das Backend und bietet per Klick die Möglichkeit an, zu dem jeweiligen Pull Request auf GitHub zu wechseln und die genauen Änderungen dort einzusehen. Hierdurch muss keine spezifische Änderungsansicht entwickelt werden, da GitHub bereits als Versionskontrollsystem verwendet wird.

### 5.4.1. Vorschaufunktion für AR-Inhalte

Damit das CMS einen Mehrwert für das Kooperationsprojekt bieten kann, soll eine AR-Vorschaufunktion für die Inhalte implementiert werden. Diese orientiert sich technologisch an der eigentlich Anwendung und setzt auf AR.js und A-Frame für die Darstellung.

Im ersten Schritt wird der Versuch unternommen, die Vorschaufunktionalität vollständig im Frontend zu realisieren. Hierfür werden die benötigten Dependencies der Bibliotheken installiert. Auf der POI-Detailansichtsseite wird ein QR-Code generiert,

## 5. Entwicklung

Abbildung 5.9.: Benutzeroberfläche - Änderungen

der auf eine neue Route für die Vorschaufunktion zeigen soll. Jedoch erweist sich dieser Ansatz als problematisch, da Vue.js ein virtuelles DOM verwendet, während AR.js und A-Frame direkte DOM-Manipulationen vornehmen und dadurch keine Initialisierung stattfinden kann. Diese Inkompatibilität führt zu Konflikten in der Renderinglogik und erfordert eine Neukonzeption der Funktion.

Um diese Herausforderung zu bewältigen, wird die Entscheidung getroffen, die AR-Vorschau in das Backend zu verlagern. Diese Strategie ermöglicht eine saubere Trennung der Verantwortlichkeiten und umgeht die Kompatibilitätsprobleme im Frontend. Die neue Implementierung sieht vor, dass im Frontend weiterhin der QR-Code generiert wird, der jetzt aber auf eine spezifische URL im Backend verweist. Beim Scannen des QR-Codes mit einem Smartphone generiert das Backend dynamisch eine EJS-View, die AR.js und A-Frame integriert und das entsprechende 3D-Modell rendernt.

Damit auch diese Vorschaufunktion geschützt und nicht unbefugt aufgerufen werden kann, wird hier wie bisher auch auf eine tokenbasierte Authentifizierung gesetzt. Da der JWT-Token zur Authentifizierung üblicherweise im localStorage des Browsers gespeichert wird, ist er für die Backend-generierte Vorschau nicht direkt verfügbar. Um dennoch eine geschützte Umgebung zu gewährleisten, wird der JWT-Token als Parameter in der URL übermittelt. Im Backend erfolgt dann eine Extraktion und Verifizierung des Tokens, bevor die AR-Inhalte freigegeben werden. Dieser Mechanismus stellt sicher, dass nur autorisierte Nutzer Zugriff auf die Vorschaufunktion erhalten.

Beim Öffnen der Vorschaufunktion auf dem Smartphone werden die von AR.js bekannten Berechtigungen abgefragt. Die NFT-Bilder können anschließend im CMS geöffnet und mit dem Smartphone aufgenommen werden (Abbildung A.11). Der AR-Inhalt sollte im Template geladen werden und auf dem Smartphone zu sehen sein (Abbildung 5.10). In der aktuellen Version der Vorschaufunktion wurde aus zeitlichen Gründen lediglich die Vorschau von 3D-Modellen implementiert. Hier sollte für zukünftige Iterationen ein System entwickelt werden, welche die Implementierungen der

## 5. Entwicklung

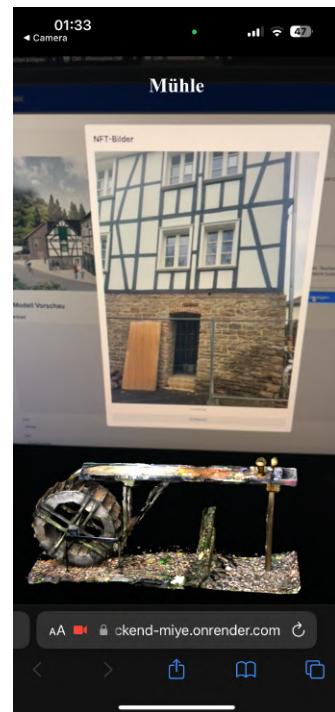


Abbildung 5.10.: AR-Vorschaufunktion - 3D-Modell

AR-Funktionen aus der AR-Anwendung übernehmen kann und damit eine Vorschau für alle Inhaltstypen darstellen kann.

# 6. Evaluation und Diskussion

In diesem Kapitel werden die Ergebnisse der Entwicklung evaluiert, die Erfüllung der Anforderungen überprüft und die Chancen sowie Risiken des Systems analysiert. Zudem werden die Limitationen des aktuellen Entwicklungsstands aufgezeigt und ein Ausblick auf zukünftige Entwicklungsmöglichkeiten gegeben.

## 6.1. Erfüllung der Anforderungen

Die Evaluation des entwickelten CMS fokussiert sich auf die Analyse der Erfüllung der in Abschnitt 3.5 definierten funktionalen und non-funktionalen Anforderungen.

Bei der Betrachtung der funktionalen Anforderungen zeigt sich, dass das CMS die zentralen Funktionen weitgehend erfüllt. Die Fähigkeit, verschiedene Medienformate zu verarbeiten und zu verwalten [F10], wurde erfolgreich implementiert. Das System kann Inhalte in Form von Texten, Bildern, Audio, Video und 3D-Modellen verwalten und bereitstellen.

Die GitHub-Integration, eine Kernanforderung des Projekts, wurde erfolgreich umgesetzt. Das System interagiert mit den bestehenden Datenstrukturen des Projekts [F160], was eine effiziente Nutzung vorhandener Ressourcen und eine reibungslose Integration in die Strukturen des Kooperationsprojekts gewährleistet. Die Implementierung von Pull Requests für Inhaltsänderungen ermöglicht zudem eine nachverfolgbare Versionskontrolle.

Die Verwaltung von AR-Inhalten, einschließlich der Organisation verschiedener Routen eines ARlebnispfades [F170] und der AR-Spots einer Route [F180], wurde implementiert. Diese Funktionen ermöglichen die Strukturierung und Organisation der AR-Erlebnisse im Kontext des Projekts. Die Benutzeroberfläche ermöglicht dabei intuitive Navigation durch die verschiedenen Routen und POIs.

Eine Herausforderung stellte die Implementierung der AR-Vorschaufunktion [F30] dar. Während die grundlegende Funktionalität zur Visualisierung von AR-Inhalten implementiert wurde, zeigt sich hier noch Optimierungspotenzial. Die aktuelle Version ermöglicht zwar die Vorschau von 3D-Modellen, ist jedoch in der Darstellung komplexerer AR-Szenarien noch eingeschränkt. Dies ist ein Bereich, der in zukünftigen Iterationen weiter ausgebaut werden sollte, um den vollen Umfang der Anforderung zu erfüllen.

Die Implementierung der Funktion zur Verfolgung und Anzeige des Bearbeitungsstatus von Inhalten [F80] konnte in einer Vereinfachten Version erfolgreich umgesetzt. Dies unterstützt die Zusammenarbeit im Team und ermöglicht es den redaktionellen Anwendern, den Fortschritt der Inhaltsproduktion effektiv zu überwachen.

Bei der Betrachtung der definierten non-funktionalen Anforderungen wurde Wert auf die Darstellung einer intuitiven Benutzeroberfläche [O10] gelegt. Hier sollte zusammen

## 6. Evaluation und Diskussion

Nr	Anforderung	Status
<b>Funktionale Muss-Anforderungen</b>		
F10	Verwaltung verschiedener Medienformate (Video, Audio, Bilder, 3D-Modelle)	Erfüllt
F30	AR-Vorschau für hochgeladene Inhalte	Teilweise erfüllt
F80	Verfolgung und Anzeige des Bearbeitungsstatus von Inhalten	Teilweise erfüllt
F160	Interaktion mit bestehenden Datenstrukturen des Projektes	Erfüllt
F170	Verwaltung verschiedener Routen eines ARlebnispfades	Erfüllt
F180	Verwaltung der AR-Spots einer Route	Erfüllt
<b>Nicht-funktionale Muss-Anforderungen</b>		
O10	Intuitive Benutzeroberfläche für Nutzer mit begrenzten technischen Kenntnissen	Erfüllt
O30	Skalierbarkeit bei wachsenden Datenmengen und Nutzerzahlen	Teilweise erfüllt
O40	Einhaltung relevanter Datenschutzbestimmungen (DSGVO)	Erfüllt
O50	Kompatibilität mit gängigen Webbrowsersn und mobilen Geräten	Teilweise erfüllt

Tabelle 6.1.: Übersicht der Erfüllung der Muss-Anforderungen

mit Testnutzern Feedback gesammelt und in einer Iteration umgesetzt werden. Die Skalierbarkeit des Systems [O30] wurde durch die Verwendung moderner Technologien und einer durchdachten Architektur adressiert. Die Verwendung des Systems zeigt, dass das System in der Lage ist, mit den Bereitgestellten Daten umzugehen. Langzeittests unter realen Bedingungen stehen jedoch noch aus, um die volle Skalierbarkeit zu validieren.

In Bezug auf Sicherheits- und Datenschutzaspekte wurden grundlegende Maßnahmen implementiert, um die Einhaltung relevanter Datenschutzbestimmungen, insbesondere der DSGVO [O40], zu gewährleisten. Die Implementierung von JWT-basierter Authentifizierung und sicheren Passwort-Hashing-Verfahren bietet einen grundlegenden Schutz gegen unbefugten Zugriff. Die Übertragung sensibler Daten erfolgt verschlüsselt, was den gängigen Sicherheitsstandards entspricht. Ein potenzielles Sicherheitsrisiko stellt jedoch die Übermittlung des JWT-Tokens als URL-Parameter bei der AR-Vorschau dar, was in zukünftigen Versionen adressiert werden sollte.

Die Kompatibilität mit gängigen Webbrowsersn und mobilen Geräten [O50] wurde weitgehend erreicht, wobei die native Integratioion von Audio- und Videoinhalten in weiteren Tests auf verschiedenen Plattformen untersucht werden sollte.

Zusammenfassend lässt sich festhalten, dass das entwickelte CMS die Mehrheit der definierten funktionalen und non-funktionalen Anforderungen erfüllt (Tabelle 6.1). Es bietet eine solide Grundlage für die Verwaltung von AR-Inhalten im Kontext des Kooperationsprojekts. Gleichzeitig wurden Bereiche identifiziert, in denen Verbesserungen

## 6. Evaluation und Diskussion

und Erweiterungen sinnvoll sind, um die Funktionalität und Benutzerfreundlichkeit weiter zu optimieren.

### 6.2. Chancen und Risiken

Zur Evaluierung der Chancen und Risiken des entwickelten CMS wird eine SWOT-Analyse durchgeführt, welche die Stärken, Schwächen, Chancen und Risiken des Systems aufzeigt.

SWOT-Analyse CMS für das CMS	
Stärken	Schwächen
Chancen	Risiken
Effiziente Verwaltung von AR-Inhalten GitHub-Integration Intuitive Benutzeroberfläche Skalierbare Architektur	Begrenzte AR-Vorschau Internetabhängigkeit Potenzielle Sicherheitsrisiken Eingeschränkte Offline-Funktionalität
Erweiterung AR-Funktionalitäten KI-Integration Anwendung in ähnlichen Projekten Open-Source-Community-Entwicklung	Schnelle technologische Veralterung Datenschutzbedenken GitHub-Abhängigkeit Skalierungsherausforderungen

Tabelle 6.2.: SWOT-Analyse des CMS für das CMS

Zu den Stärken des Systems zählt die Integration in die Datenstrukturen des Kooperationsprojekts, die effiziente Verwaltung verschiedener AR-Inhalte und eine erste AR-Vorschaufunktion.

Als Schwächen können die begrenzte AR-Vorschaufunktion und die teilweise vorhandene Kollaborations- bzw. Statusmöglichkeiten eingestuft werden.

Die Chancen des Systems liegen in der möglichen Erweiterung der AR-Funktionalitäten für komplexere Szenarien und der Integration einer automatisierten Inhaltsoptimierung. Das Potenzial zur Anwendung in ähnlichen Projekten und der mögliche Ausbau in eine Open-Source-Lösung könnten die Reichweite und den Einfluss des Projekts erheblich vergrößern.

Zu den Risiken zählen die schnelle technologische Entwicklung im AR-Bereich, die das System ohne kontinuierliche Weiterentwicklung obsolet machen könnten. Die Abhängigkeit von GitHub als zentralem Repository und potenzielle Herausforderungen bei der Skalierung auf große Datenmengen stellen weitere Risikofaktoren dar. Während der Entwicklung des Systems gab es einen weltweiten Ausfall von GitHub, welcher die Funktionalität des Systems eingeschränkt hat (Abbildung A.12).

Für das Kooperationsprojekt bedeutet die Einführung des CMS eine signifikante Verbesserung der Verwaltung von AR-Inhalten, da dieses durch eine verbesserte Benutzerfreundlichkeit zu einer effizienteren Zusammenarbeit zwischen den Projektbeteiligten führt.

Langfristig hat das CMS das Potenzial, als Modell für ähnliche AR-basierte Kulturerbe-Projekte zu dienen und könnte zur Positionierung des Oberbergischen Kreises als In-

## *6. Evaluation und Diskussion*

novator in der digitalen Kulturvermittlung beitragen. Dies könnte neue Kooperationen und Fördermöglichkeiten erschließen.

### **6.3. Limitationen und Ausblick**

Das entwickelte System weist trotz seiner erfolgreichen Implementierung einige Limitationen auf.

Die aktuell begrenzte Funktionalität der AR-Vorschau und fehlende kleinere Funktionen können Herausforderungen darstellen. Zudem ist durch die enge Kopplung an GitHub mit einer eingeschränkten Flexibilität zu rechnen. Verbesserungspotenziale liegen primär in der Erweiterung der AR-Vorschaufunktion und der Integration von Bearbeitungs- bzw. Optimierungsfunktionalitäten von AR-Inhalten. Langfristig könnte die Integration von KI-Funktionen relevant werden, mit welcher Inhalte automatisch kategorisiert oder optimiert werden könnten.

Trotz der vorhandenen Einschränkungen bietet das CMS eine solide Grundlage für die Verwaltung von AR-Inhalten im Projekt. Mit kontinuierlicher Anpassung an technologische Entwicklungen und Nutzerbedürfnisse hat es das Potenzial, sich zu einer passenden Anwendung zur Verwaltung von AR-Erlebnissen zu entwickeln.

## 7. Fazit und Ausblick

Diese Arbeit hat sich mit der Konzeption und Entwicklung eines spezialisierten Content-Management-Systems für die Verwaltung von Augmented Reality Inhalten befasst. Für das Kooperationsprojekt „ARlebnispfade OBK“ sollte eine Lösung erarbeitet werden, die den spezifischen Anforderungen der AR-Inhaltserstellung und -verwaltung gerecht wird.

Die Hauptergebnisse dieser Arbeit umfassen die erfolgreiche Implementierung eines webbasierten CMS, welches eine intuitive Verwaltung verschiedener AR-Inhaltstypen ermöglicht. Die Integration mit dem GitHub-Repository des Projekts stellt eine erfolgreiche Integration in vorhandene Datenstrukturen dar. Die Implementierung einer AR-Vorschaufunktion, wenn auch in begrenztem Umfang, markiert einen wichtigen Schritt in Richtung einer verbesserten Qualitätskontrolle der AR-Erlebnisse.

Für das Kooperationsprojekt bedeutet die Einführung des CMS eine signifikante Verbesserung der Effizienz bei der Verwaltung von Inhalten. Die Benutzeroberfläche und die strukturierte Organisation der Inhalte ermöglichen es auch technisch weniger versierten Personen, effektiv an der Inhaltserstellung mitzuwirken.

Der Entwicklungsprozess hat einige Herausforderungen mit sich gebracht. Der Fokus des Systems sollte in den ersten Überlegungen auf der AR-Vorschau und der Verwaltung der Inhalte liegen. Da die Inhalte aus dem GitHub-Repository geladen werden, musste ein unerwarteter Großteil der Entwicklungszeit in die Datenbeschaffung des Backends gesteckt werden. Zudem hätte die Entwicklung des Frontends durch das Designen von Mockups effizienter gestaltet werden können.

Bei der kritischen Betrachtung der eingesetzten Methoden und Technologien zeigt sich, dass der gewählte Ansatz einer eigenständigen CMS-Entwicklung sowohl Vor- als auch Nachteile mit sich bringt. Die hohe Anpassungsfähigkeit an die spezifischen Projektanforderungen steht dem erhöhten Entwicklungsaufwand gegenüber. Die Verwendung moderner Webtechnologien erwies sich als vorteilhaft für die Entwicklungsgeschwindigkeit und Flexibilität, stellte jedoch auch Herausforderungen in Bezug auf die Integration komplexer AR-Funktionalitäten dar.

Durch die Bearbeitung des Projektes konnte vor allem ein verstärktes persönliches Verständnis für Content-Management-Systeme und deren Aufbau geschaffen werden. Zudem konnten erste Erfahrungen mit dem produktiven Umgang des Frontend-Frameworks Vue.js und Typescript gesammelt werden.

Der wissenschaftliche und praktische Beitrag dieser Arbeit liegt in der Exploration und praktischen Umsetzung eines spezialisierten CMS im Kontext der digitalen Kulturvermittlung. Die Erkenntnisse aus der Entwicklung und Integration von AR-Technologien in Webanwendungen können als Grundlage für zukünftige Forschungen und Entwicklungen in diesem Bereich dienen. Praktisch bietet das entwickelte System eine Basis für die effiziente Verwaltung von AR-Inhalten in kulturellen und edukativen Kontexten.

## *7. Fazit und Ausblick*

Für zukünftige Forschungs- und Entwicklungsrichtungen ergeben sich mehrere mögliche Pfade. Die Weiterentwicklung und Optimierung der AR-Vorschaufunktion stellt einen wichtigen Aspekt dar, um die Qualitätskontrolle von AR-Inhalten weiter zu verbessern. Die Integration fortgeschritten Kollaborationswerkzeuge könnte die teamübergreifende Zusammenarbeit in AR-Projekten weiter fördern. Zudem bietet die Exploration von Möglichkeiten zur automatisierten Inhaltserstellung und -optimierung mittels künstlicher Intelligenz ein spannendes Forschungsfeld.

Als größtes persönliches Learning aus dieser Arbeit stellt sich die Wichtigkeit von definierten Datenstrukturen heraus. Dabei ist nicht nur die Definition an sich, sondern insbesondere auch die Implementierung dieser von großer Bedeutung. Wenn Daten und Inhalte nicht konsistent gespeichert werden, müssen in der Entwicklung unnötige Fehlerquellen analysiert und umgangen werden.

Abschließend lässt sich festhalten, dass das entwickelte System einen wichtigen Beitrag zur Verwaltung von AR-Inhalten im Kontext kultureller Bildungsprojekte leistet. Sie bietet eine solide Grundlage für die weitere Entwicklung und Erforschung von AR-basierten Projekten und öffnet neue Möglichkeiten für die interaktive Präsentation von kulturellem Erbe. Die gewonnenen Erkenntnisse und entwickelten Technologien können als Ausgangspunkt für zukünftige Innovationen in diesem dynamischen Forschungsfeld dienen.

# Literaturverzeichnis

- [Arena u. a. 2022] ARENA, Fabio ; COLLOTTA, Mario ; PAU, Giovanni ; TERMINE, Francesco: An Overview of Augmented Reality. In: *Computers* 11 (2022), Nr. 2. <http://dx.doi.org/10.3390/computers11020028>. – DOI 10.3390/computers11020028. – ISSN 2073–431X
- [ARtillery Intelligence 2024] ARTILLERY INTELLIGENCE: <https://artilleryiq.com/reports/mobile-ar-global-revenue-forecast-2023-2028/>. <https://artilleryiq.com/reports/mobile-ar-global-revenue-forecast-2023-2028/>. Version: 2024. – Letzter Zugriff am 01. September 2024
- [Azuma 1997] AZUMA, Ronald T.: A Survey of Augmented Reality. In: *Presence: Teleoperators and Virtual Environments* 6 (1997), 08, Nr. 4, 355–385. <http://dx.doi.org/10.1162/pres.1997.6.4.355>. – DOI 10.1162/pres.1997.6.4.355
- [Barker 2016] BARKER, D.: *Web Content Management: Systems, Features, and Best Practices*. O'Reilly Media, 2016 [https://books.google.de/books?id=x6\\_NCwAAQBAJ](https://books.google.de/books?id=x6_NCwAAQBAJ). – ISBN 9781491908105
- [Bierman u. a. 2014] BIERMAN, Gavin ; ABADI, Martín ; TORGERSEN, Mads: Understanding TypeScript. In: JONES, Richard (Hrsg.): *ECOOP 2014 – Object-Oriented Programming*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2014. – ISBN 978–3–662–44202–9, S. 257–281
- [Billinghurst u. a. 2015] BILLINGHURST, Mark ; CLARK, Adrian ; LEE, Gun: A Survey of Augmented Reality. In: *Foundations and Trends® in Human-Computer Interaction* 8 (2015), 01, S. 73–272. <http://dx.doi.org/10.1561/1100000049>. – DOI 10.1561/1100000049
- [Bizjak u. a. 2017] BIZJAK, Igor ; KLINC, Robert ; TURK, Žiga: A framework for open and participatory designing of built environments. In: *Computers, Environment and Urban Systems* 66 (2017), 11, S. 65–82. <http://dx.doi.org/10.1016/j.comenvurbssys.2017.08.002>. – DOI 10.1016/j.comenvurbssys.2017.08.002
- [Boiko 2005] BOIKO, B.: *Content Management Bible*. Wiley, 2005 (Bible). <https://books.google.de/books?id=p6nUDn3ZaBoC>. – ISBN 9780764583643
- [Borisov u. a. 2018] BORISOV, Andrey ; SIECK, Jürgen ; ASHIKOTO, Leonard ; KAMENYE, Gabriel ; MWENYO, Jason ; LIKANDO, Nawa: Development of an efficient, cost-reducing content management system for augmented reality applications. In: *Proceedings of the Second African Conference for Human Computer Interaction*:

## Literaturverzeichnis

- Thriving Communities.* New York, NY, USA : Association for Computing Machinery, 2018 (AfriCHI '18). – ISBN 9781450365581
- [Bourhis u. a. 2017] BOURHIS, Pierre ; REUTTER, Juan L. ; SUÁREZ, Fernando ; VRGOČ, Domagoj: JSON: Data model, Query languages and Schema specification. In: *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. New York, NY, USA : Association for Computing Machinery, 2017 (PODS '17). – ISBN 9781450341981, 123–135
- [Can I Use 2024] CAN I USE: *WebXR Device API*. <https://caniuse.com/webxr>. Version: 2024. – Letzter Zugriff am 01. September 2024
- [Carroll u. Olson 1988] CARROLL, John M. ; OLSON, Judith R.: Chapter 2 - Mental Models in Human-Computer Interaction11This chapter appeared in its entirety and is reprinted from Mental Models in Human Computer Interaction: Research Issues about What the User of Software Knows, J.M. Carroll and J.R. Olson, Editors,- The report of the workshop on software human factors: Users mental models, Nancy Anderson, chair, sponsored by the Committee on Human Factors, Commission on Behavioral and Social Sciences and Education, National Research Council, published by the National Academy Press, 1987. Version: 1988. <http://dx.doi.org/https://doi.org/10.1016/B978-0-444-70536-5.50007-5>. In: HELANDER, MARTIN (Hrsg.): *Handbook of Human-Computer Interaction*. Amsterdam : North-Holland, 1988. – DOI <https://doi.org/10.1016/B978-0-444-70536-5.50007-5>. – ISBN 978-0-444-70536-5, 45-65
- [DIN EN ISO 9241-210:2020-03 2020] *Ergonomie der Mensch-System-Interaktion - Teil 210: Menschzentrierte Gestaltung interaktiver Systeme (ISO 9241-210:2019); Deutsche Fassung EN ISO 9241-210:2019*
- [Fielding 2000] FIELDING, Roy T.: *REST: Architectural Styles and the Design of Network-based Software Architectures*, University of California, Irvine, Doctoral dissertation, 2000. <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>
- [GitHub 2022] GITHUB: *Rate limits for the REST API - GitHub Docs*. <https://docs.github.com/en/rest/using-the-rest-api/rate-limits-for-the-rest-api?apiVersion=2022-11-28>. Version: 11 2022. – Letzter Zugriff am 06. September 2024
- [GitHub 2024] GITHUB: *Octokit*. <https://github.com/octokit>. Version: 2024. – Letzter Zugriff am 06. September 2024
- [Jones u. a. 2015] JONES, Michael B. ; BRADLEY, John ; SAKIMURA, Nat: *JSON Web Token (JWT)*. RFC 7519. <http://dx.doi.org/10.17487/RFC7519>. Version: Mai 2015 (Request for Comments)
- [Loeliger u. McCullough 2012] LOELIGER, J. ; MCCULLOUGH, M.: *Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development*. O'Reilly Media, Incorporated, 2012 (Oreilly and Associate Series). <https://books.google.de/books?id=ZkXELyQWf4UC>. – ISBN 9781449316389

## Literaturverzeichnis

- [Massee 2011] MASSE, M.: *REST API Design Rulebook*. O'Reilly Media, 2011 (O'Reilly and Associate Series). <https://books.google.de/books?id=4lZcsRwXo6MC>. – ISBN 9781449310509
- [Milgram u. Kishino 1994] MILGRAM, Paul ; KISHINO, Fumio: A Taxonomy of Mixed Reality Visual Displays. In: *IEICE Trans. Information Systems* vol. E77-D, no. 12 (1994), 12, S. 1321–1329
- [Murphy 2023] MURPHY, Tim: *The top 4 content management trends in 2024*. <https://www.techtarget.com/searchcontentmanagement/feature/The-top-5-content-management-trends>. Version: 12 2023. – Letzter Zugriff am 01. September 2024
- [Provost u. Mazieres 1999] PROVOST, Niels ; MAZIERES, David: Bcrypt algorithm. In: *USENIX*, 1999
- [Qiao u. a. 2019] QIAO, Xiuquan ; REN, Pei ; DUSTDAR, Schahram ; LIU, Ling ; MA, Huadong ; CHEN, Junliang: Web AR: A Promising Future for Mobile Augmented Reality—State of the Art, Challenges, and Insights. In: *Proceedings of the IEEE* 107 (2019), Nr. 4, S. 651–666. <http://dx.doi.org/10.1109/JPROC.2019.2895105>. – DOI 10.1109/JPROC.2019.2895105
- [Rock Paper Reality 2022] ROCK PAPER REALITY: Is WebAR better than AR apps? In: *Rock Paper Reality* (2022), 9. <https://rockpaperreality.com/insights/web-ar/is-webar-better-than-ar-apps/>. – Letzter Zugriff am 01. September 2024
- [Shepiliev u. a. 2021] SHEPILIEV, Dmytro S. ; MODLO, Ye.O. ; YECHKALO, Yu.V. ; TKACHUK, V.V. ; MINTII, Mykhailo ; MINTII, I.S. ; MARKOVA, Oksana ; SELIVANOVA, T.V. ; DRASHKO, Olena M. ; KALINICHENKO, Olga O. ; VAKALIUK, Tetiana ; OSADCHYI, Viacheslav V. ; SEMERIKOV, Serhiy O.: WebAR development tools: An overview. In: *Proceedings of the 3rd Workshop for Young Scientists in Computer Science & Software Engineering (CS&SE@SW 2020)* Bd. 2832. Kryvyyi Rih, Ukraine : CEUR Workshop Proceedings, November 2021. – ISSN 1613–0073, 84–93. – CEUR Workshop Proceedings (CEUR-WS.org), ISSN 1613-0073
- [Van Krevelen u. Poelman 2010] VAN KREVELEN, Rick ; POELMAN, Ronald: A Survey of Augmented Reality Technologies, Applications and Limitations. In: *International Journal of Virtual Reality (ISSN 1081-1451)* 9 (2010), 06, S. 1. <http://dx.doi.org/10.20870/IJVR.2010.9.2.2767>. – DOI 10.20870/IJVR.2010.9.2.2767
- [Zaitsev 2024a] ZAITSEV, Anton: *Backend API*. <https://github.com/antonztsv/arlebnispfade-cms/wiki/Backend-API>. Version: 2024. – Letzter Zugriff am 10. September 2024
- [Zaitsev 2024b] ZAITSEV, Anton: *GitHub - antonztsv/arlebnispfade-cms: Entwicklungsrepository zur Bachelorarbeit mit dem Titel "Konzeption und Entwicklung einer Webanwendung zur einfachen Integration und Verwaltung von Augmented Reality Inhalten, am Beispiel des Kooperationsprojektes "ARlebnispfade OBK"*. <https://github.com/antonztsv/arlebnispfade-cms>. Version: 2024. – Letzter Zugriff am 10. September 2024

# A. Anhang

## A.1. Analyse

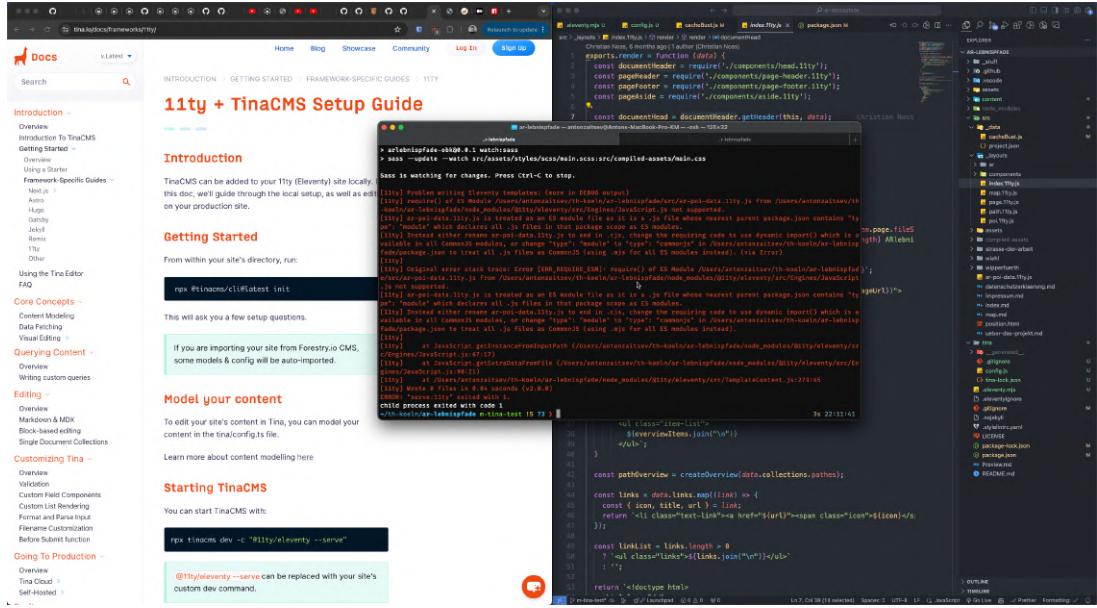


Abbildung A.1.: Versuch TinaCMS zu integrieren

## A. Anhang

Tabelle A.1.: Stakeholderanalyse

Primär / Sekundär / Tertiär	Stakeholder	Bezeichnung	Bezug zum System	Objektbereich	Erfordernis/Erwartung
Primär	AR-Content-Ersteller	Einzelperson	Interesse: Erstellung und Verwaltung von AR-Inhalten	CMS, AR-Inhalte	Erfordernis: Intuitive Benutzeroberfläche zur Content-Erstellung - Erwartung: Effiziente Arbeitsabläufe, einfache Integration verschiedener Medientypen
Primär	Redaktionelle Anwender	Einzelperson	Anteil: Verwaltung und Pflege von AR-Inhalten	CMS, AR-Inhalte	Erfordernis: Umfassende Content-Management-Funktionen - Erwartung: Übersichtliche Verwaltung, effektive Versionskontrolle
Sekundär	Entwickler	Einzelperson	Anspruch: Integration und Wartung des Systems	Gesamtsystem, API	Erfordernis: Zugang zu technischer Dokumentation - Erwartung: Gut strukturierte API, Erweiterbarkeit des Systems
Sekundär	Projektmanager	Einzelperson	Anspruch: Erfolgreiche Projektumsetzung	Gesamtsystem	Erfordernis: Übersicht über Projektfortschritt - Erwartung: Einhaltung von Zeitplan und Budget
Sekundär	Qualitätssicherung	Einzelperson/ Organisation	Anspruch: Sicherstellung der Systemqualität	Gesamtsystem	Erfordernis: Zugang zu Testumgebungen - Erwartung: Testbare Funktionen, Stabilität
Sekundär	UX/UI-Designer	Einzelperson	Anteil: Gestaltung der Benutzeroberfläche	CMS, AR-Anwendung	Erfordernis: Flexibilität bei der UI-Gestaltung - Erwartung: Konsistentes Design, positive Nutzererfahrung
Sekundär	Kooperationspartner	Organisation	Interesse: Nutzung für eigene AR-Projekte	Gesamtsystem	Erfordernis: Anpassbarkeit an spezifische Bedürfnisse - Erwartung: Flexibilität, Skalierbarkeit
Tertiär	Nutzerinnen der AR-Anwendung	Einzelperson	Interesse: Nutzung der AR-Erlebnisse	AR-Anwendung	Erfordernis: Zugänglichkeit der AR-Inhalte - Erwartung: Qualitativ hochwertige, interaktive AR-Erlebnisse
Tertiär	Datenschutzbeauftragte	Einzelperson	Anrecht: Einhaltung von Datenschutzbestimmungen	Gesamtsystem, Datenverarbeitung	Erfordernis: Transparenz in der Datenverarbeitung - Erwartung: DSGVO-Konformität
Tertiär	Marketing	Organisation	Interesse: Nutzung der AR-Inhalte für Marketing	AR-Inhalte, AR-Anwendung	Erfordernis: Zugang zu fertigen AR-Inhalten - Erwartung: Attraktive, teilbare Inhalte
Tertiär	Externe Technologiepartner	Organisation	Interesse: Kompatibilität und Optimierung	AR-Anwendung, Hardware-Integration	Erfordernis: Technische Spezifikationen - Erwartung: Zusammenarbeit, Optimierung

### **A.1.1. User Profile - Redaktioneller Anwender**

#### **Demografische Daten:**

- Alter: 30-55 Jahre
- Bildung: Hochschulabschluss in Journalismus, Kommunikation oder verwandten Bereichen
- Technische Affinität / Kenntnisse: Niedrig bis mittel

#### **Beruflicher Hintergrund:**

- 5-15 Jahre Erfahrung in der redaktionellen Arbeit
- Vertraut mit traditionellen und digitalen Publikationsprozessen
- Grundlegende Kenntnisse in Content-Management-Systemen

#### **Fähigkeiten:**

- Starke redaktionelle und organisatorische Fähigkeiten
- Erfahrung in der Verwaltung und Pflege von digitalen Inhalten

#### **Ziele:**

- Effiziente Verwaltung und Organisation von AR-Inhalten
- Sicherstellung der inhaltlichen Qualität und Konsistenz
- Optimierung des Workflows für Content-Erstellung und -Veröffentlichung

#### **Herausforderungen:**

- Umgang mit der Komplexität von AR-Inhalten
- Koordination zwischen Content-Erstellern und technischem Team
- Einhaltung von gemeinsamen redaktionellen Standards

#### **Erwartungen an das CMS:**

- Übersichtliche und strukturierte Content-Verwaltung
- Vorschaufunktion für AR-Inhalte
- Effektive Versionskontrolle und Änderungsnachverfolgung
- Einfache Überprüfungs- und Freigabeprozesse

### **A.1.2. User Profile - AR-Content-Erststeller**

#### **Demografische Daten:**

- Alter: 25-45 Jahre
- Bildung: Hochschulabschluss in Mediendesign, digitale Medien oder verwandten Bereichen
- Technische Affinität / Kenntnisse: Hoch

#### **Beruflicher Hintergrund:**

- 3-10 Jahre Erfahrung in der Erstellung digitaler Inhalte
- Vertraut mit verschiedenen Medienformaten (Text, Bild, Audio, Video, 3D-Modelle)
- Grundkenntnisse in AR-Technologien

#### **Fähigkeiten:**

- Kreatives Denken und visuelle Gestaltung
- Erfahrung mit Content-Creation-Tools

#### **Ziele:**

- Erstellung innovativer und ansprechender AR-Erlebnisse
- Effiziente Arbeitsabläufe bei der Content-Erstellung
- Kontinuierliche Verbesserung der AR-Inhaltsqualität

#### **Herausforderungen:**

- Komplexität der AR-Technologie
- Zeitdruck bei der Inhaltserstellung
- Anpassung an sich ändernde AR-Standards und -Plattformen

#### **Erwartungen an das CMS:**

- Intuitive Benutzeroberfläche
- Einfache Integration verschiedener Medientypen
- Vorschaufunktion für AR-Inhalte

## A. Anhang

### A.1.3. Persona - Redaktioneller Anwender

**Alter:** 45

**Beruf:** Kulturbeauftragter der Stadt Gummersbach

**Wohnort:** Bergneustadt, Oberbergischer Kreis

**Hintergrund:**

Thomas hat einen Abschluss in Kulturmanagement und arbeitet seit 10 Jahren für die Stadtverwaltung Gummersbach. Er ist verantwortlich für die Koordination kultureller Projekte und die Pflege des digitalen Kulturangebots der Stadt.

**Ziele:**

- Effiziente Organisation und Verwaltung von digitalen Inhalten für kulturelle Projekte
- Förderung des kulturellen Erbes des Oberbergischen Kreises durch moderne Medien
- Verbesserung der Zusammenarbeit zwischen verschiedenen lokalen Akteuren (Museen, Vereine, Schulen)



**Frustrationen:**

- Begrenzte finanzielle und personelle Ressourcen für digitale Projekte
- Schwierigkeiten bei der Integration von historischen Informationen in moderne Medienformate
- Zeitaufwändige Prozesse bei der Abstimmung von Inhalten mit verschiedenen Interessengruppen

**Typischer Tag:**

Thomas beginnt seinen Tag mit der Durchsicht von E-Mails und der Planung anstehender kultureller Veranstaltungen. Er verbringt viel Zeit damit, eingereichte Inhalte von lokalen Künstlern und Historikern zu sichten und für digitale Plattformen aufzubereiten. Regelmäßig führt er Gespräche mit Vertretern lokaler Kultureinrichtungen und koordiniert die Zusammenarbeit für übergreifende Projekte.

**Technische Fähigkeiten:**

- Grundlegende Kenntnisse in Content-Management-Systemen
- Erfahrung mit Social-Media-Plattformen für kulturelle Kommunikation
- Basiswissen in Bildbearbeitung und einfacher Videobearbeitung

**Erwartungen an das CMS:**

- Benutzerfreundliche Oberfläche, die auch für weniger technikaffine Mitarbeiter zugänglich ist
- Möglichkeit zur einfachen Integration von historischen Daten und Geodaten
- Werkzeuge zur Zusammenarbeit und Abstimmung mit verschiedenen lokalen Partnern
- Flexible Möglichkeiten zur Kategorisierung von Inhalten nach kulturellen und historischen Aspekten

**Zitat:**

*"Unser kulturelles Erbe ist ein Schatz, den wir durch moderne Technologien für alle Generationen erlebbar machen wollen."*

Abbildung A.2.: Persona - Thomas Weber (Redaktioneller Anwender)

## A. Anhang

**Alter:** 28

**Beruf:** Mediengestalterin bei einer lokalen Produktionsfirma

**Wohnort:** Gummersbach, Oberbergischer Kreis

**Hintergrund:**

Lisa hat eine Ausbildung zur Mediengestalterin Bild und Ton absolviert und arbeitet seit 4 Jahren bei einer Produktionsfirma in Gummersbach. Sie ist in der Region verwurzelt und engagiert sich in lokalen Kulturprojekten.

**Ziele:**

- Hochwertige visuelle Inhalte produzieren, die die Geschichte und Kultur des Oberbergischen Kreises hervorheben
- Ihre Fähigkeiten in der Erstellung verschiedener Medienformate (Video, Foto, Audio) weiterentwickeln
- Zur Attraktivität der Region durch innovative Medienprojekte beitragen



**Frustrationen:**

- Begrenzte technische Ressourcen im Vergleich zu größeren Städten
- Herausforderungen bei der Darstellung komplexer historischer oder kultureller Inhalte in modernen Medienformaten
- Zeitaufwändige Prozesse beim Organisieren und Teilen von Inhalten mit verschiedenen Projektpartnern

**Typischer Tag:**

Lisa beginnt ihren Tag oft mit Außenaufnahmen an historischen Stätten oder bei lokalen Veranstaltungen. Nachmittags bearbeitet sie das Material in der Firma, schneidet Videos oder bearbeitet Fotos. Sie steht in regelmäßiger Austausch mit lokalen Partnern und Kultureinrichtungen.

**Technische Fähigkeiten:**

- Versiert in Video- und Bildbearbeitungssoftware
- Erfahren im Umgang mit professionellem Kamera- und Audioequipment
- Grundkenntnisse in Content-Management-Systemen und Social-Media-Plattformen

**Erwartungen an das CMS:**

- Benutzerfreundliche Oberfläche für das einfache Hochladen und Organisieren verschiedener Medienformate
- Möglichkeit, Inhalte mit Geodaten und historischen Informationen zu verknüpfen
- Kollaborationsfunktionen für die Zusammenarbeit mit anderen lokalen Akteuren
- Flexibles Tagging-System zur Kategorisierung von Inhalten nach Themen, Orten und historischen Epochen

**Zitat:**

*"Ich möchte die reiche Geschichte und Kultur unserer Region durch moderne Medien zum Leben erwecken und für alle zugänglich machen."*

Abbildung A.3.: Persona - Lisa Müller (AR-Content-Erststellerin)

#### A.1.4. Benutzererfordernisse

- Als Content-Erstellerin muss man verschiedene Medienformate (Video, Audio, Bilder, 3D-Modelle) einfach hochladen und organisieren können, um effizient Inhalte für AR-Erlebnisse bereitzustellen.
- Als redaktioneller Anwender muss man Inhalte mit historischen und geografischen Daten verknüpfen können, um kontextrelevante AR-Erlebnisse zu schaffen.
- Als Content-Erstellerin muss man eine Vorschaufunktion für AR-Inhalte verfügbar haben, um die Qualität und Wirkung der Arbeit überprüfen zu können.
- Als redaktioneller Anwender muss man Inhalte kategorisieren und taggen können, um sie effizient zu verwalten und wiederzufinden.
- Als Content-Erstellerin muss man Kollaborationswerkzeuge verfügbar haben, um effektiv mit anderen Teammitgliedern zusammenarbeiten zu können.
- Als redaktioneller Anwender muss man Zugriffsrechte verwalten können, um die Sicherheit und Integrität der Inhalte zu gewährleisten.
- Als Content-Erstellerin muss man ortsbezogene AR-Trigger definieren können, um standortspezifische Erlebnisse zu erstellen.
- Als redaktioneller Anwender muss man Analyse-Tools verfügbar haben, um die Nutzung und Wirkung von AR-Inhalten zu messen und daraus Erkenntnisse für zukünftige Projekte zu gewinnen.
- Als Content-Erstellerin muss man verschiedene Versionen von Inhalten verwalten können, um Änderungen nachzuverfolgen und bei Bedarf auf frühere Versionen zurückgreifen zu können.
- Als redaktioneller Anwender muss man eine Übersicht über den Bearbeitungsstatus von Inhalten haben, um den Fortschritt von Projekten effektiv verfolgen zu können.
- Als Content-Erstellerin muss man 3D-Modelle importieren, bearbeiten und in AR-Szenen platzieren können, um realistische und interaktive AR-Erlebnisse zu schaffen.
- Als redaktioneller Anwender muss man AR-Inhalte für verschiedene Zielgruppen (z.B. Kinder, Erwachsene, Experten) kategorisieren und anpassen können, um zielgruppengerechte Erlebnisse zu ermöglichen.
- Als Content-Erstellerin muss man AR-Inhalte für verschiedene Gerätetypen (Smartphones, Tablets, AR-Brillen) optimieren können, um eine breite Zugänglichkeit zu gewährleisten.
- Als redaktioneller Anwender muss man Werkzeuge zur Erstellung und Verwaltung von AR-gestützten Touren oder Lernpfaden verfügbar haben, um zusammenhängende Erlebnisse zu gestalten.

## A. Anhang

- Als Content-Erststellerin muss man Interaktionen und Animationen für AR-Objekte definieren können, um engagierende und lehrreiche Erlebnisse zu schaffen.

### A.1.5. Funktionale Anforderungen

[F10] Das System muss einem Benutzer die Möglichkeit bieten, verschiedene Medienformate (Text, Audio, Bilder, Videos, 3D-Modelle) hochladen und verwalten zu können.

[F20] Das System sollte einem Benutzer die Möglichkeit bieten, Inhalte mit geografischen Koordinaten und historischen Zeiträumen verknüpfen zu können.

[F30] Das System muss fähig sein, eine AR-Vorschau für hochgeladene Inhalte anzubieten.

[F40] Das System sollte einem Benutzer die Möglichkeit bieten, Inhalte mit Tags und Kategorien zu versehen.

[F50] Das System kann Kollaborationswerkzeuge wie Kommentarfunktionen und Aufgabenzuweisungen bereitstellen.

[F60] Das System sollte fähig sein, Benutzer mit verschiedenen Zugriffsebenen und Rollen zu verwalten.

[F70] Das System sollte einem Benutzer die Möglichkeit bieten, benutzerdefinierte Metadaten zu Inhalten hinzuzufügen.

[F80] Das System muss fähig sein, den Bearbeitungsstatus von Inhalten zu verfolgen und anzuzeigen.

[F90] Das System sollte fähig sein, eine Versionskontrolle für alle Inhalte zu implementieren.

[F100] Das System sollte einem Benutzer die Möglichkeit bieten, Inhalte exportieren zu können.

[F110] Das System kann fähig sein, ein integriertes Feedback-System für Inhalte bereitzustellen.

[F120] Das System kann fähig sein, Analyse-Tools zur Messung der Nutzung und Wirkung von AR-Inhalten zu integrieren.

[F130] Das System sollte eine zentrale Asset-Bibliothek mit Suchfunktion bereitstellen.

[F140] Das System kann einem Benutzer die Möglichkeit bieten, AR-Inhalte mit externen Kalendersystemen für lokale Events zu verknüpfen.

[F150] Das System sollte einem Benutzer die Möglichkeit bieten, ortsbzogene AR-Trigger zu definieren und zu verwalten.

[F160] Das System muss fähig sein, mit den bestehenden Datenstrukturen des Projektes interagieren zu können.

[F170] Das System muss fähig sein, verschiedene Routen eines ARlebnispfades verwalten zu können.

[F180] Das System muss fähig sein, die AR-Spots einer Route verwalten zu können.

[F190] Das System sollte fähig sein, 3D-Modelle zu importieren, zu bearbeiten und in AR-Szenen platzieren zu können.

[F200] Das System sollte einem Benutzer die Möglichkeit bieten, Interaktionen und Animationen für AR-Objekte definieren zu können.

[F210] Das System sollte fähig sein, AR-Inhalte für verschiedene Gerätetypen (Smartphones, Tablets, AR-Brillen) zu optimieren.

#### **A.1.6. Non-Funktionale Anforderungen**

- [O10] Das System muss eine intuitive Benutzeroberfläche bieten, die auch von Nutzern mit begrenzten technischen Kenntnissen bedient werden kann.
- [O20] Das System sollte fähig sein, große Medienformate effizient zu verarbeiten und eine Ladezeit von maximal 5 Sekunden für Standardoperationen zu gewährleisten.
- [O30] Das System muss fähig sein, mit wachsenden Datenmengen und Nutzerzahlen umzugehen, ohne signifikant an Performance zu verlieren.
- [O40] Das System muss alle relevanten Datenschutzbestimmungen (DSGVO) einhalten.
- [O50] Das System muss mit gängigen Webbrowsersn und mobilen Geräten kompatibel sein.
- [O60] Das System kann eine definierte Verfügbarkeit gewährleisten.
- [O70] Das System sollte modular aufgebaut sein, um einfache Wartung und zukünftige Erweiterungen zu ermöglichen.
- [O80] Das System kann mehrsprachig sein, mit initialem Fokus auf Deutsch.
- [O90] Das System sollte eine grundlegende Barrierefreiheit gewährleisten.
- [O100] Das System kann fähig sein, grundlegende Funktionen auch bei temporären Netzwerkausfällen zu gewährleisten.

## A.2. Wireframes

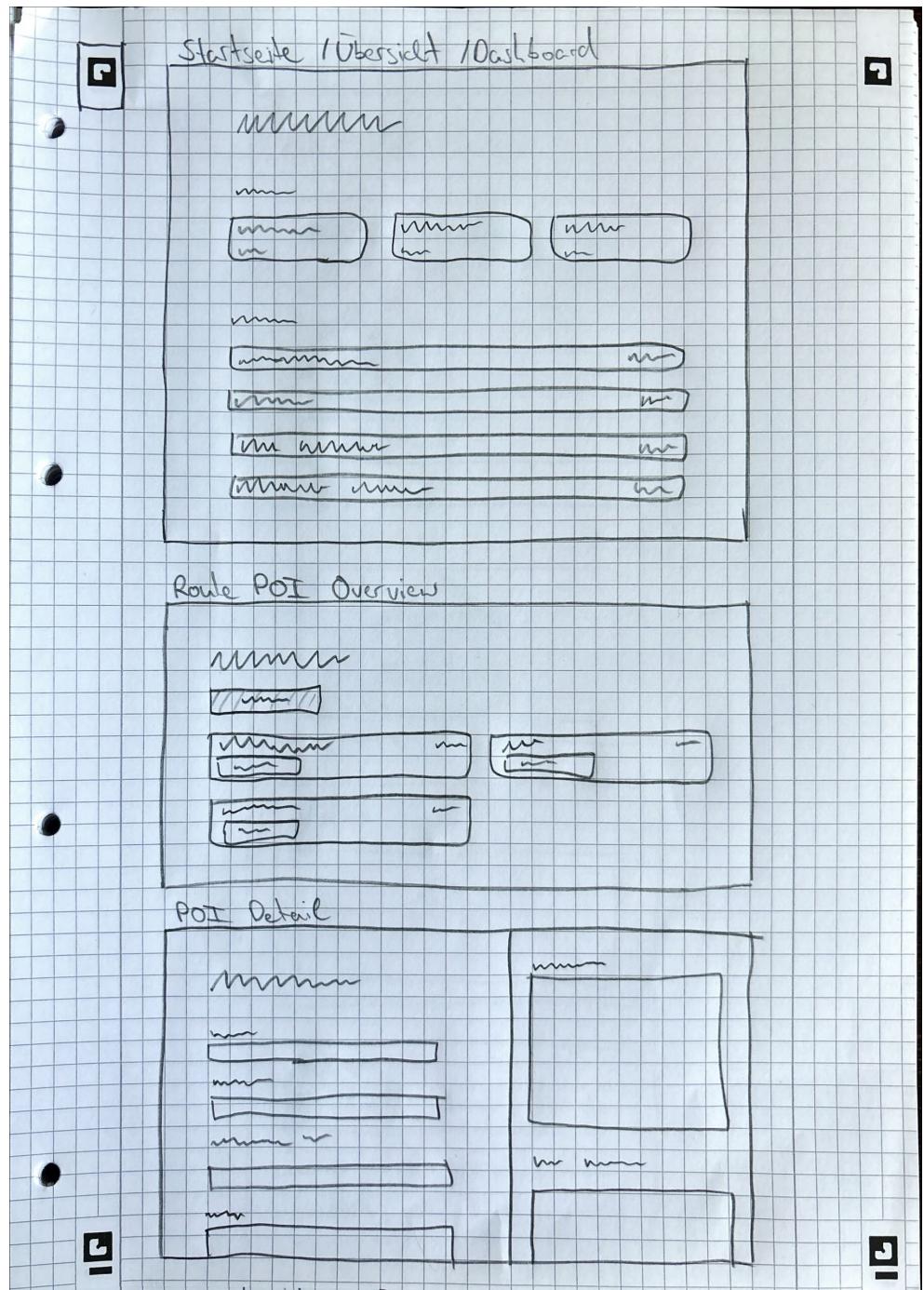


Abbildung A.4.: Wireframes Seiten

## A. Anhang

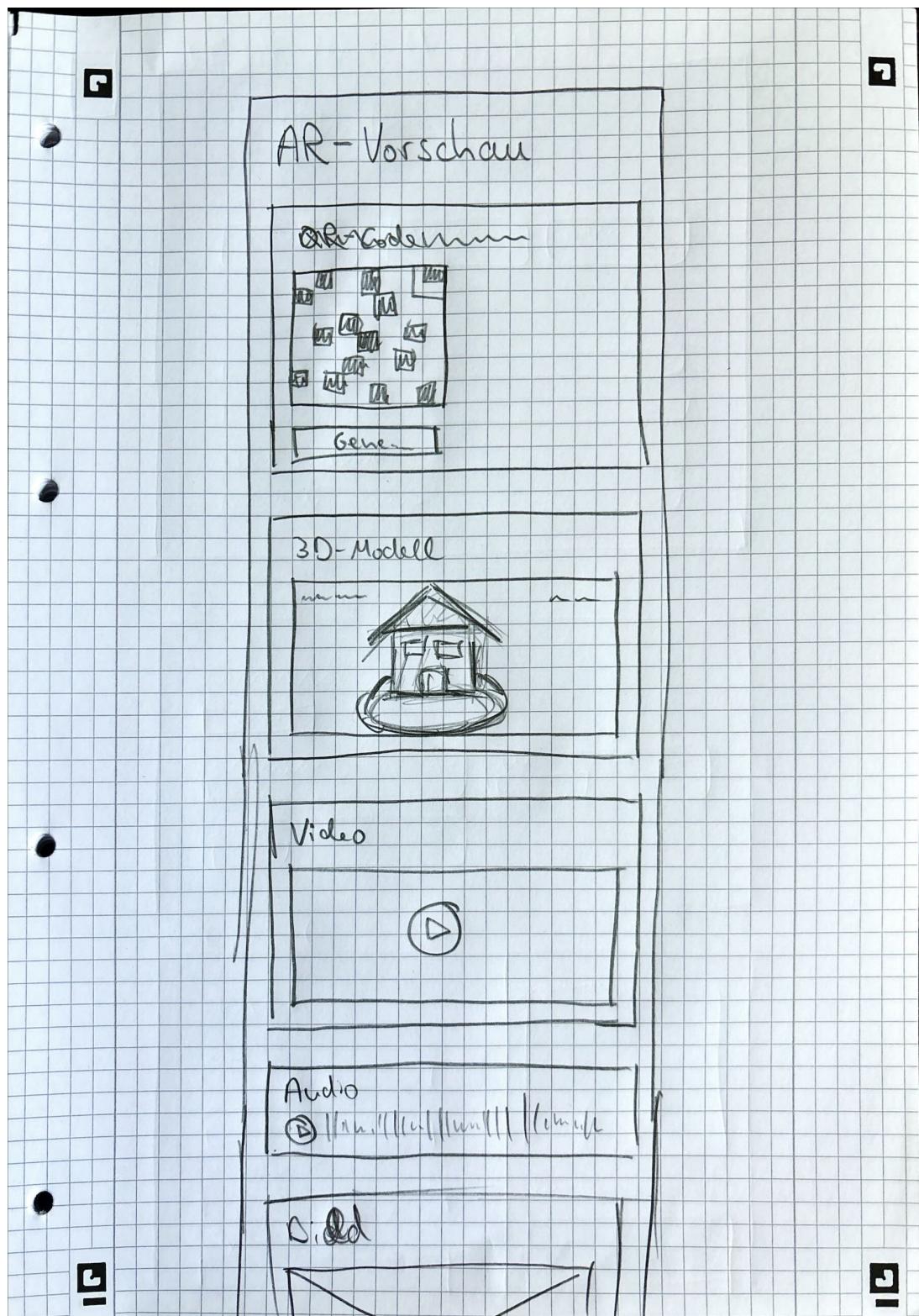


Abbildung A.5.: Wireframes AR-Vorschaufunktion

### A.3. Entwicklung

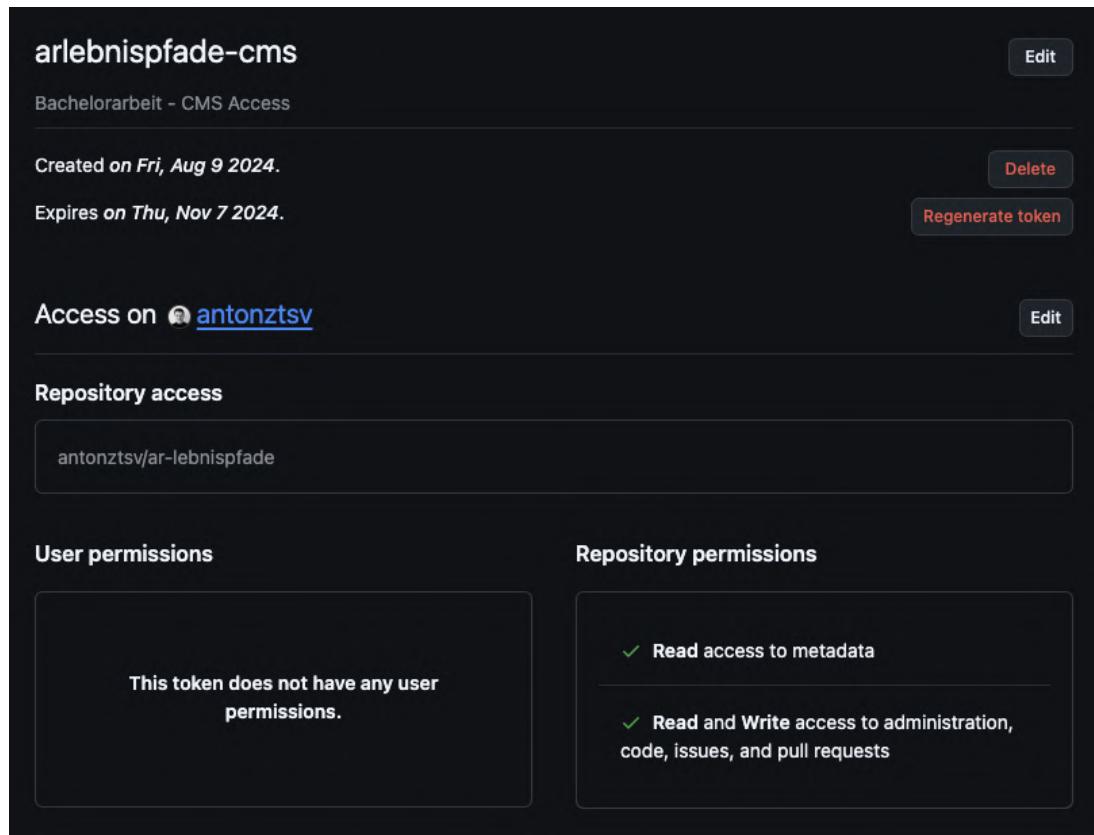


Abbildung A.6.: GitHub Access Token Generierung



Abbildung A.7.: Benutzeroberfläche - Login

## A. Anhang

The screenshot shows the 'Routen' section of the ARLebnispfade OBK software. It displays three entries, each with a thumbnail image and configuration fields:

- Straße der Arbeit**: Shows a tall metal tower against a cloudy sky. Configuration fields include:
  - Titel: Straße der Arbeit
  - Layout: path.11ty.js
  - Bild: strasse-der-arbeit.webp
  - Typ: metadata
- Wiehl**: Shows a church with a tall spire and surrounding buildings. Configuration fields include:
  - Titel: Wiehl
  - Layout: path.11ty.js
  - Bild: wiehl.webp
  - Typ: metadata
- Wipperfürth**: Shows a view of a town with a church and houses. Configuration fields include:
  - Titel: Wipperfürth
  - Layout: path.11ty.js
  - Bild: wipperfuerth.webp
  - Typ: metadata

Abbildung A.8.: Benutzeroberfläche - Routen

The screenshot shows the 'AR-Medien' and 'Bilder' sections of the ARLebnispfade OBK software.

- AR-Medien (81)**: A list of media files:
  - wiehl\_bielstein\_burghaus.mp3: audio file, preview 0:00 / 6:39
  - wiehl\_bielstein\_motocross.mp3: audio file, preview 0:00 / 1:26
  - wiehl\_repschenrothermuehle.mp3: audio file, preview 0:00 / 5:17
  - wiehl\_stbonifatius.mp3: audio file, preview 0:00 / 5:15
  - bergischerloewe.fset: image file
  - bergischerloewe.fset3: image file
  - bergischerloewe.iset: image file
- Bilder (14)**: A list of image files:
  - burghaus.webp**: image file, preview of a yellow building with a grey roof.
  - eisenbahnbruecke.webp**: image file, preview of a bridge over water with trees in the background.

Abbildung A.9.: Benutzeroberfläche - Routen Medien und Inhalte

## A. Anhang

The screenshot shows a form for a POI (Point of Interest). The title is "Mühle". The form includes fields for "Titel" (Title), "Bild" (Image), "Typ" (Type), "Layout" (Layout), and "Info" (Information). The "Info" section contains a detailed description of the Repschenrother Mühle. Below this is an "AR" section with fields for "AR Type" (Image Tracking), "AR Inhalt" (3D Model), "AR Standort" (Location), and "Audio" (MP3 file). There is also a "Video" section with an "NFT" field. At the bottom is a blue "Bearbeiten" (Edit) button.

Abbildung A.10.: Benutzeroberfläche - POI Detailansicht Formular

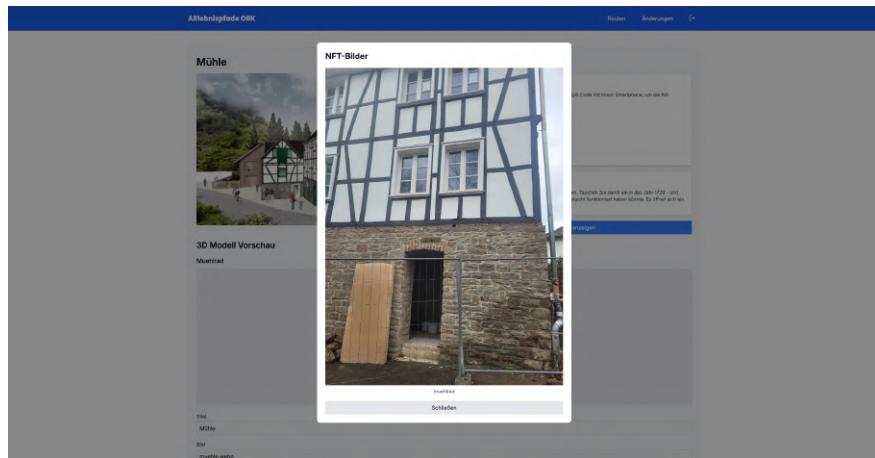


Abbildung A.11.: Benutzeroberfläche - NFT Bilder Vorschau

## A. Anhang

### A.4. Evaluation

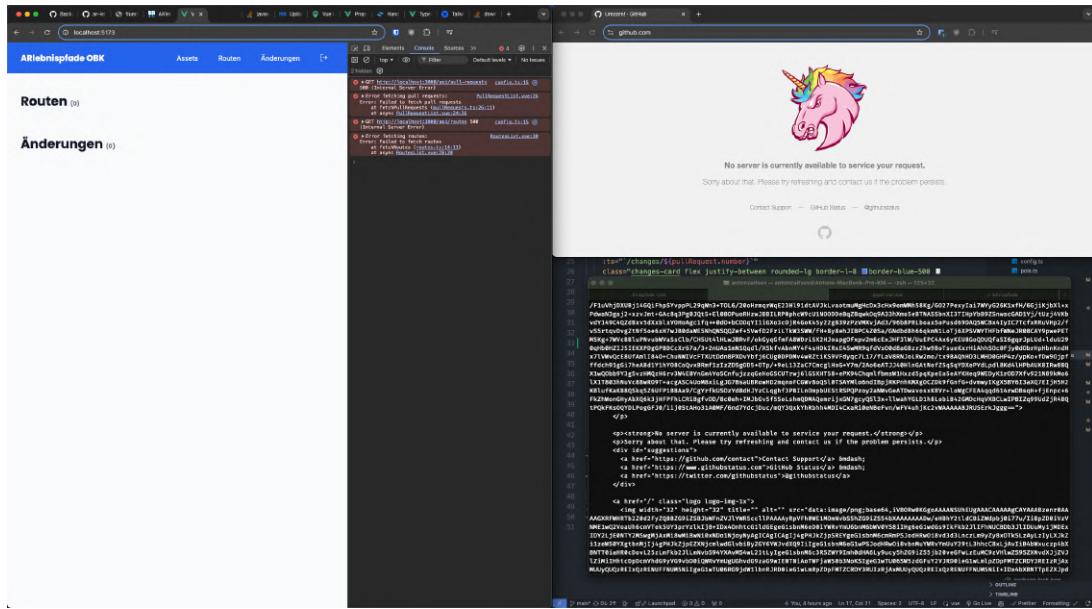


Abbildung A.12.: Weltweiter GitHub Ausfall

# **Eidesstattliche Erklärung**

Ich versichere, die von mir vorgelegte Arbeit selbständig verfasst zu haben.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben.

Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Gummersbach, 12. September 2024



Anton Zaitsev