

# **OST ResearchNavigator: Projekt-Tracking-Software für die Ostschweizer Fachhochschule**

## **Autoren**

Simon Hefti & Livio Mauchle

## **Semesterarbeit**

OST – Ostschweizer Fachhochschule  
Campus Rapperswil-Jona

## **Betreuung**

Laurent Metzger  
Fabio Daniel Marti  
Simon Hager

20.12.2024

# Abstract

Die OST führt im Rahmen der angewandten Forschung vielfältige und innovative Projekte durch. Derzeit erfolgt die Veröffentlichung direkt in den fachspezifischen Kanälen. Eine zentrale Übersicht, die eine schnelle Suche nach Projekten anhand von unterschiedlichen Kriterien ermöglicht, existiert jedoch noch nicht. In einer neuen Lösung, dem OST ResearchNavigator, sollen Projekte aller Organisationseinheiten der OST einheitlich mit relevanten Inhalten erfasst und publiziert werden können. Durch diese einheitliche Lösung kann die Kommunikation in einem definierten Prozess von der OST besser überprüft werden.

Ziel dieser Studienarbeit ist die Erstellung eines Prototyps der gewünschten Lösung. Basierend auf den Strukturen und Prozessen der OST sowie den relevanten Projektmetadaten wurde ein geeignetes Datenmodell erstellt. Daraus wurden die wichtigsten Such- und Filterkriterien abgeleitet. Eine Technologieanalyse mit dem Fokus auf Erweiterbarkeit führte zur Auswahl der folgenden Komponenten: Ein Nuxt.js Web-Frontend, das mit einer ASP.NET Core API kommuniziert. Eine PostgreSQL Datenbank zur Speicherung der Projektdaten, sowie ein MinIO Fileserver, der zur Ablage von Projektdateien dient. Benutzertests mit zwei Professoren des Instituts für Interaktive Informatik trugen zur Optimierung der Benutzeroberfläche bei, die sich am OST-Design orientiert. Die Systemkomponenten wurden vollständig containerisiert, um eine hohe Portabilität und ein einfaches Deployment zu gewährleisten.

Der erfolgreiche Prototyp erlaubt es Mitarbeitenden der Hochschule, ihre Projekte einheitlich zu erfassen. Ein klar definierter Veröffentlichungsprozess stellt sicher, dass Projekte umfassend geprüft werden. Such- und Filterfunktionen ermöglichen es Projekte gezielt zu finden. Zusätzlich erleichtern OST-übergreifende Statistiken den Überblick über die Forschungsaktivitäten. Somit genügt der Prototyp allen Anforderungen, die für eine erste Pilotphase essentiell sind. Nicht-funktionale Anforderungen wie Bedienbarkeit, Performance und Wartbarkeit wurden mithilfe von Usability-Tests, Load-Tests und Metriken wie Branch-Coverage überprüft. Aus sicherheitstechnischer Sicht wurde eine einfache Benutzerautorisierung implementiert, die jedoch aus Gründen der Systemintegration noch keine Authentifizierung beinhaltet. Um den Kundennutzen sicherzustellen, wurden der Projektplan und der Projektfortschritt in regelmässigen Abständen dem Auftraggeber präsentiert. Das Resultat dieses Projektes stiess auf überwiegend positives Feedback beim Auftraggeber, was als erfolgreicher Projektabschluss aufgenommen wird.

In einer anschliessenden Bachelorarbeit wird das Produkt weiterentwickelt. Dabei soll die Benutzeroberfläche, sowie auch die Suchfunktionalität weiter verbessert werden. Ausserdem soll die Anwendung um weitere Funktionen wie beispielsweise eine Administrationsoberfläche, ein Authentifizierungssystem oder eine KI-basierte Datenextraktion erweitert werden.

**Schlagwörter:** Software Engineering, Full-Stack, Forschung.

# Management Summary

Die OST ist eine angesehene Fachhochschule, die sich durch anwendungsorientierte Forschungsprojekte in verschiedenen Fachbereichen auszeichnet. Die Forschungsprojekte werden derzeit über unterschiedliche Kanäle veröffentlicht, die je nach Bedarf der einzelnen Fachbereiche gewählt werden. Eine zentrale Plattform, die alle Forschungsprojekte einheitlich erfasst, eine effiziente Suche und Filterung ermöglicht sowie Statistiken und Forschungstrends übersichtlich visualisiert, fehlte bisher. Dieser Mangel erschwerte es, sich einen vollständigen Überblick über die Forschungsaktivitäten der Hochschule zu verschaffen und Trends in der Forschung an der OST zu analysieren. Deshalb wird im Verlauf der Studienarbeit das **Minimum Viable Product (MVP)** einer zentralen Plattform für die Erfassung, Verwaltung und zielgerichtete Suche von Forschungsprojekten entwickelt. Nebst der Optimierung interner Prozesse soll diese Anwendung dazu beitragen, die Sichtbarkeit der Forschungsprojekten innerhalb wie auch ausserhalb der Hochschule zu steigern.

Die Entwicklung des Prototyps erfolgte in einem hybriden Vorgehensmodell, das den Rational **Rational Unified Process (RUP)** mit Scrum Sprints kombiniert. Nach einer detaillierten Anforderungsanalyse wurden zentrale Artefakte wie Use Cases und Mockups erstellt und mit dem Auftraggeber abgestimmt. Die darauf aufbauende Systemarchitektur wurde mit besonderem Augenmerk auf Erweiterbarkeit und Modularität entworfen. Zur Qualitätssicherung wurden automatisierte Unit Tests in den Entwicklungsprozess integriert, sowie eine Pipeline mit automatischem Linting und Formatting eingesetzt. Die Usability der Anwendung wurde zusammen mit potentiellen Benutzern getestet, um eine gute Bedienbarkeit zu gewährleisten.

Für die Frontend-Entwicklung wurde Vue.js in Kombination mit Nuxt.js eingesetzt. Das Backend basiert auf ASP.NET Core und folgt dem Prinzip der Onion-Architektur, um eine lose Kopplung zu gewährleisten. Diese Architektur unterstützt sowohl die Erweiterbarkeit als auch die Testbarkeit des Systems. Eine PostgreSQL-Datenbank dient der Persistierung der Projektdaten, während Docker für eine flexible Bereitstellung und hohe Portabilität der Anwendung verwendet wird.

Das **MVP** ermöglicht es dem Nutzer, Forschungsprojekte nicht nur nach Titel, Autor oder Abstract zu suchen, sondern diese mit Hilfe von neun verschiedenen Filtern gezielt einzugrenzen. Die kombinierbare Anwendung dieser Filter ermöglicht eine gezielte Suche nach sehr spezifischen Kriterien. Zudem werden bei der Suche und Anzeige unterschiedliche Sichtbarkeitsstufen berücksichtigt. Dadurch wird sichergestellt, dass nur Projekte, die von der OST explizit freigegeben wurden, öffentlich sichtbar sind (siehe Abbildung 1).

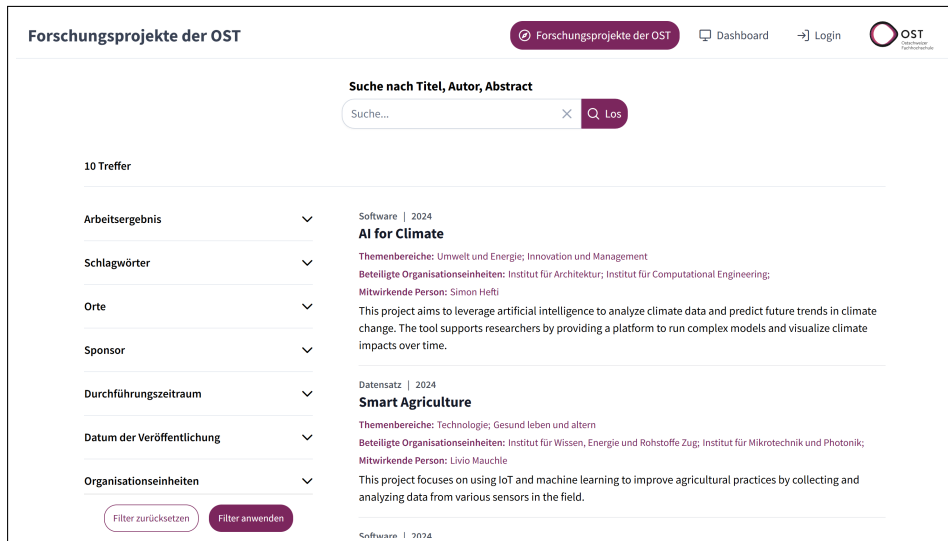


Abbildung 1: Startseite der Webapplikation

Die Detailansicht eines Projekts (siehe Abbildung 2) bietet einen vollständigen Überblick über alle verfügbaren Inhalte. Sie ermöglicht das Betrachten von Bildern und Videos sowie das Herunterladen von veröffentlichten Projektdokumenten.

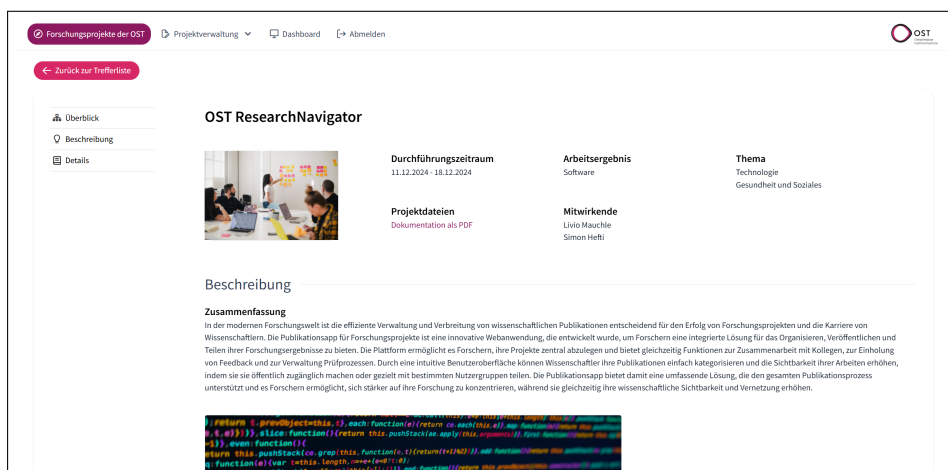


Abbildung 2: Detailansicht eines Projekts

Die Mitarbeiterinnen und Mitarbeiter der OST haben die Möglichkeit, über einen definierten Ablauf selbstständig Projekte anzulegen, zu bearbeiten und für die gewünschte Zielgruppe freizugeben. Je nach gewünschter Sichtbarkeit (nur für am Projekt beteiligte Organisationseinheiten, OST-intern oder öffentlich) gibt es unterschiedliche Prüf- und Freigabeschritte, die es zu erfüllen gilt (siehe Abbildung 3).



Abbildung 3: Seite zur Projektbearbeitung

Darüber hinaus bietet ein öffentlich zugängliches Dashboard (siehe Abbildung 4) einen grafischen Überblick über die Projekte und liefert filterbare Statistiken, um Trends und Forschungsschwerpunkte sichtbar zu machen.

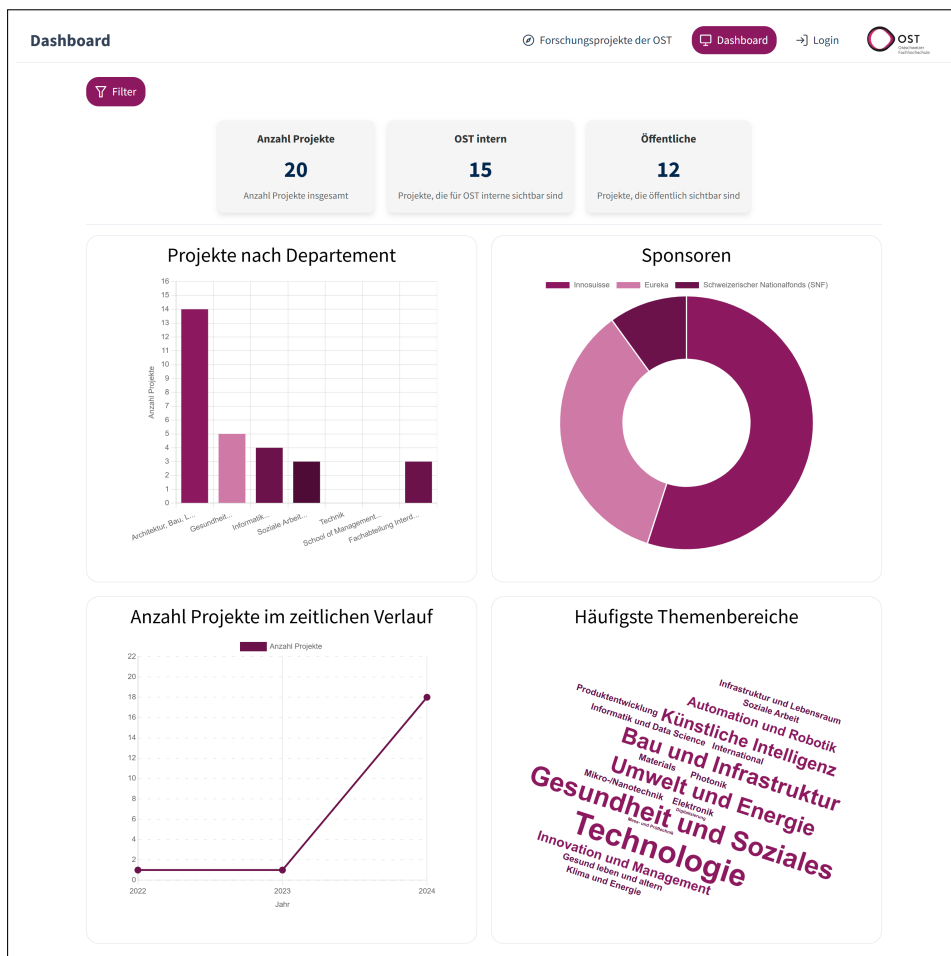


Abbildung 4: Dashboard zur Anzeige der grafischen Statistiken

Das entwickelte MVP erfüllt die vom Auftraggeber geforderten Kernfunktionen. In einer anschließenden Bachelorarbeit soll das Produkt weiter ausgebaut und um zusätzliche Funktionalitäten ergänzt werden. Geplant sind u.a. eine Administrationsoberfläche zur Erfassung und Pflege von Metadaten, die Integration eines Authentifizierungssystems sowie erweiterte Features wie die Volltextsuche in hochgeladenen PDF-Dokumenten oder die automatisierte Extraktion relevanter Projektdaten in die Projekterfassung mittels künstlicher Intelligenz. Zudem sind auch noch weitere Optimierungen des Designs sowie Verbesserungen in der Suchfunktion oder Erweiterungen des Dashboards denkbar.

# Danksagung

**Fabio Daniel Marti** danken wir für die kompetente Betreuung und die wertvolle Unterstützung, die wesentlich zum Erfolg des Projekts beigetragen haben.

**Simon Hager** danken wir für das Teilen seiner Expertise, seiner konstruktiven Vorschläge und für die durchgeführten Code Reviews, die die Qualität des Projekts verbessert haben.

**Prof. Laurent Metzger und Prof. Alex Simeon** danken wir für ihr wertvolles Feedback und die grossartigen Ideen, die das Projekt in seiner aktuellen Form ermöglicht haben.

**Prof. Dr. Frieder Loch und Prof. Dr. Markus Stolze** danken wir für ihre Teilnahme an unseren Usability-Tests. Durch ihre ausgezeichnete Expertise und das konstruktive Feedback konnte die Benutzerfreundlichkeit weiter verbessert werden.

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>X</b>
<b>Tabellenverzeichnis</b>	<b>XII</b>
<b>Literaturverzeichnis</b>	<b>XV</b>
<b>1 Einleitung</b>	<b>1</b>
1.1 Ausgangslage	1
1.2 Problemstellung	1
1.3 Stand der Technik	2
1.3.1 Stand der OST	2
1.3.2 Bestehende Lösungen	2
1.4 Ziel der Arbeit	2
1.5 Organisatorische Rahmenbedingungen	2
1.6 Abgrenzung	3
<b>2 Analyse &amp; Anforderungen</b>	<b>4</b>
2.1 Funktionale Anforderungen	5
2.1.1 Akteure	5
2.1.2 Use Case Diagramm	6
2.1.3 Use Cases	7
2.2 Nicht-Funktionale Anforderungen	17
2.3 Nicht Anforderungen	21
2.3.1 Undo- und Redo-Funktionalitäten in der Projekterfassung	21
2.3.2 Löschen von Projekten	21
2.3.3 Parallele Projekterfassung durch mehrere Personen	21
2.3.4 Benutzerdefinierte Dashboards	22
2.4 Domain Model	22
<b>3 Lösungskonzept</b>	<b>25</b>
3.1 Analyse und Evaluation	25
3.1.1 Analyse von Anwendungen mit ähnlicher Funktionalität	25
3.1.2 Sichtbarkeit von Projekten	26
3.1.3 Prüfprozess	26
3.2 Architektur	28
3.2.1 Kontext & Abgrenzung	28



3.2.2	Lösungsstrategie	29
3.2.3	Bausteinsicht Level 1: Containers	30
3.2.4	Bausteinsicht Level 2: Components	31
3.2.5	Bausteinsicht Level 3: Code	31
3.2.6	Laufzeitsicht	37
3.2.7	Deployment-Sicht	38
3.2.8	Querschnittliche Konzepte	39
3.2.9	Architekturentscheidungen	40
3.3	Technische Schulden	43
3.4	Design	44
3.4.1	Anforderungen und Allgemeines Konzept	44
3.4.2	Accessibility	44
3.4.3	Mockups	46
3.4.4	Designentscheide	51
<b>4</b>	<b>Umsetzung</b>	<b>53</b>
4.1	MinIO Konfiguration	53
4.2	Benutzer-Autorisierung	54
4.2.1	Registrierung der Authentication Services	54
4.2.2	Token-Erstellung	54
4.2.3	Autorisierung des Schreibzugriffs mit JWT Claims	55
4.2.4	Nuxt.js Middleware für die Autorisierung geschützter Routes	56
4.3	Filtern der Projekte in der Datenbank	57
4.4	Error Handling im Backend	57
4.5	Frontend Basis-Komponente	59
4.6	Weitere Frameworks und Bibliotheken	59
4.6.1	Testing im Backend	59
4.6.2	Validierung im Frontend	59
4.6.3	Integration von PrimeVue und Entwicklung eines OST-Themes	60
<b>5</b>	<b>Qualitätsmassnahmen</b>	<b>61</b>
5.1	Code Qualität	61
5.2	Versionskontrolle	62
5.2.1	Struktur	62
5.2.2	Workflow	62
5.2.3	Branch Naming	62
5.2.4	Pipeline	62
5.3	Code Reviews	63
5.4	Testing	63
5.4.1	Unit Tests	63
5.4.2	Usability Tests	64
5.4.3	System Tests	64
5.4.4	Load Tests	64
5.5	Dokumentation des Codes	65
<b>6</b>	<b>Projektplan</b>	<b>66</b>

6.1	Vorgehen im Projekt	66
6.2	Zeitplan	66
6.2.1	Meilensteine	68
6.3	Risikomanagement	68
6.3.1	Risikoauswertung vom 26.09.2024	68
6.3.2	Risikoauswertung vom 21.10.2024 - Ende der Elaboration Phase	69
6.3.3	Risikoauswertung vom 02.12.2024 - Ende der Construction Phase	69
6.3.4	Identifizierte Risiken	70
6.4	Zeiterfassung	71
<b>7</b>	<b>Ergebnisse</b>	<b>74</b>
7.1	Übersicht der Ergebnisse	74
7.1.1	Statistiken zum Code	79
7.1.2	Zielerreichung	79
<b>8</b>	<b>Ergebnisdiskussion</b>	<b>81</b>
8.1	Fazit	81
8.2	Ausblick	82
8.2.1	Erweiterungen	82
8.2.2	Verbesserungsmöglichkeiten	82
8.3	Weiterentwicklung	82
<b>9</b>	<b>Anhang</b>	<b>85</b>
9.1	Tests und Qualität	85
9.2	Ergänzende Inhalte	106

# Abbildungsverzeichnis

1	Startseite der Webapplikation	III
2	Detailansicht eines Projekts	III
3	Seite zur Projektbearbeitung	IV
4	Dashboard zur Anzeige der grafischen Statistiken	IV
2.1	Use Case Diagramm	6
2.2	Domain Model	24
3.1	Ablauf Prüfprozess	27
3.2	Kontext Diagramm	28
3.3	Container Diagramm	30
3.4	Components Diagramm	31
3.5	Onion Architektur	32
3.6	Entity Relationship Diagram	36
3.7	Sequenzdiagramm	37
3.8	Zustandsdiagramm der Projektveröffentlichung	38
3.9	C4 Deployment Diagramm	38
3.10	Mockup der Startseite	46
3.11	Mockup der Projektdetailseite	47
3.12	Mockup der Seite Meine Projekte	48
3.13	Mockup der Projekterstellungs-Seite	49
3.14	Mockup der Dashboard-Seite	50
5.1	Visualisierung der GitLab Pipeline	62
5.2	Testabdeckung der entwickelten Backend-Services	63
6.1	Projektplan	67
6.2	Aufgewendete Zeit pro Teammitglied	71
6.3	Aufgewendete Zeit nach Tätigkeitsbereich	72
6.4	Aufgewendete Zeit nach Tätigkeitsbereich - Livio Mauchle	72
6.5	Aufgewendete Zeit nach Tätigkeitsbereich - Simon Hefti	73
7.1	Screenshot Startseite	74
7.2	Screenshots von Filterfunktionen: Links die Filterung nach Schlagwörtern, rechts die Filterung nach Arbeitsergebnissen.	75
7.3	Screenshot Dashboard	75
7.4	Screenshot der Projektdetailansicht	76

7.5	Screenshot der Projekterfassungsseite . . . . .	77
7.6	Screenshots ausgewählter Elemente der Projekterfassungsseite . . . . .	77
7.7	Screenshots: Optionen der Prüfperson . . . . .	78
7.8	Screenshots: Einblick in den Prüfprozess . . . . .	78
7.9	Testabdeckung der entwickelten Backend-Services . . . . .	79
9.1	Authorization und Content Headers für alle Requests . . . . .	102
9.2	Testreport für den ersten Testplan (50 parallele Threads). Die maximale Response-Time liegt mit 350ms (filter-basierte Suche) deutlich unter dem definierten Grenzwert von 2s. Die Error Rate von 8.4% resultiert ausschliesslich aus gleichzeitigen Update-Requests, welche zu einem Concurrency Problem führen. . . . .	102
9.3	Request für die Filtersuche inklusive aller Filter ausser den Tags . . . . .	102
9.4	Request für die Tagsuche . . . . .	103
9.5	Response für die Tagsuche . . . . .	103
9.6	Request für die Textsuche . . . . .	103
9.7	Request für die Projekterstellung (Create) . . . . .	104
9.8	Request für die Projektbearbeitung (Update) mit Body, welcher die neuen Projektdaten enthält. . . . .	104
9.9	Concurrency Fehler der auftritt, wenn zwei Update-Requests zu schnell aufeinanderfolgen. Dies ist für unsere Anwendung kein Problem, da jeweils nur ein User für die Bearbeitung eines Projektes zuständig ist. . . . .	104
9.10	Testreport für den zweiten Testplan (1 Thread). Die maximale Response-Time liegt auch hier mit 1.57s unter dem Grenzwert (2s). . . . .	105
9.11	Request für das Löschen (Delete) . . . . .	105
9.12	Request für den Upload eines 10.7 MB grossen Files. Die Datei wird als multipart/form-data übermittelt. . . . .	105
9.13	Anzahl Codezeilen im Frontend . . . . .	107
9.14	Anzahl Codezeilen im Backend . . . . .	107
9.15	Anzahl Codezeilen der Unit Tests . . . . .	107

# Tabellenverzeichnis

2.1	Farbcodes der Anforderungen	4
2.2	Beschreibung der Akteure	5
2.3	UC01 - Projekt suchen	7
2.4	UC02 - Projekte filtern	8
2.5	UC03 - Projekt ansehen	9
2.6	UC04 - Dashboard einsehen	9
2.7	UC05 - Projekt CRUD	10
2.8	UC06 - Medien einbinden	11
2.9	UC07 - veröffentlichtes Projekt verwalten	12
2.10	UC08 - Projekt prüfen	13
2.11	UC09 - Authentifizieren	13
2.12	UC10 - Projekte aus SAP laden	14
2.13	UC11 - CRUD Filter	14
2.14	UC12 - CRUD Benutzer	14
2.15	UC13 - Projekt interaktiv betrachten	15
2.16	UC14 - Benachrichtigungs-E-Mail versenden	15
2.17	UC15 - automatische Erfassung der Projekteinhalte	16
2.18	UC16 - PDF Textsuche	16
2.19	NFR-1: Response Time Projektsuche	17
2.20	NFR-2: Response Time Projektbearbeitung	18
2.21	NFR-3: Browser-Kompatibilität	18
2.22	NFR-4: Affordance Projektsuche	19
2.23	NFR-5: Authorization	19
2.24	NFR-6: Test Coverage	20
2.25	NFR-7: Portable Docker Container	20
2.26	NFR-8: Backups	21
3.1	Lösungsstrategie	29
3.2	Erklärung der Ordnerstruktur	33
3.3	Erklärung der Ordnerstruktur Teil 2	34
3.4	Technische Schulden	43
3.5	Accessibility Massnahmen	45
3.6	Designentscheide	51
3.7	Designentscheide Teil 2	52

5.1	Massnahmen zur Verbesserung der Code Qualität	61
5.2	Beschreibung der Pipeline Stages	63
6.1	Meilensteine	68
6.2	Risikomatrix 26.09.2024	68
6.3	Risikomatrix 21.10.2024	69
6.4	Risikomatrix 02.12.2024	69
6.5	Risiko - Umfang	70
6.6	Risiko - Anforderungen	70
6.7	Risiko - Wissen	70
6.8	Risiko - Bugs	70
6.9	Risiko - Usability	71
6.10	Risiko - Security	71
7.1	Auflistung der Anzahl Codezeilen	79
9.1	Usability-Tests	86
9.2	Usability-Tests	87
9.3	Usability-Tests mit Dr. Frieder Loch	88
9.4	Usability-Tests mit Dr. Frieder Loch	89
9.5	Usability-Tests mit Dr. Markus Stolze	90
9.6	Usability-Tests mit Dr. Markus Stolze	91
9.7	Verbesserungen durch Usability Tests	92
9.8	1. Code Review - Feedback und Massnahmen	94
9.9	1. Code Review - Feedback und Massnahmen	95
9.10	Weitere Verbesserungsvorschläge - Feedback und Massnahmen	96

# Listings

4.1	Konfiguration des MinIO Client in Program.cs . . . . .	53
4.2	Registrierung der Authentication Services . . . . .	54
4.3	Generierung der JWTs . . . . .	55
4.4	Benutzerautorisierung mit JWT Claims . . . . .	56
4.5	Middleware zur Autorisierung geschützter Routes . . . . .	56
4.6	Projekte nach Suchkriterien filtern . . . . .	57
4.7	Registrierung der Error Handler im Backend . . . . .	58
4.8	Implementation des NotFoundExceptionHandler . . . . .	58
4.9	Code der Basis-Komponente app.vue . . . . .	59
4.10	Validierung im Frontend mit Zod . . . . .	60

# Literaturverzeichnis

- [1] C. Larman, "Use cases," in *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, 3rd ed. Upper Saddle River, NJ United States: Prentice Hall PTR, 2004.
- [2] (2024, oct) Zhaw digitalcollection. ZHAW Zürcher Hochschule für Angewandte Wissenschaften. [Online]. Available: <https://digitalcollection.zhaw.ch>
- [3] (2024, oct) Eth research collection. ETH Zürich. [Online]. Available: <https://www.research-collection.ethz.ch>
- [4] (2024, oct) Digitec online-shop. Digitec Galaxus AG. [Online]. Available: <https://www.digitec.ch>
- [5] (2024, oct) Swisscovery: Katalog und zugriff auf wissenschaftliche ressourcen. SLSP Swiss Library Service Platform. [Online]. Available: <https://swisscovery.slsp.ch>
- [6] (2024, oct) Ieee xplore digital library. IEEE. [Online]. Available: <https://ieeexplore.ieee.org>
- [7] D. G. S. Dr. Peter Hruschka. (2024) arc42 template. Abgerufen am 02.10.2024. [Online]. Available: <https://www.arc42.de/overview/>
- [8] S. Brown. (2024) C4 model. Abgerufen am 10.10.2024. [Online]. Available: <https://c4model.com/>
- [9] J. Palermo. (2013, aug) Tag archives: onion architecture. Abgerufen am 09.10.2024. [Online]. Available: <https://jeffreypalermo.com/tag/onion-architecture/>
- [10] (2024, apr) Onion architecture in asp.net core. Code Maze. Abgerufen am 09.10.2024. [Online]. Available: <https://code-maze.com/onion-architecture-in-aspnetcore/>
- [11] R. C. Martin, *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Pearson Studium, 2017.
- [12] (2024, oct) Directory structure. Abgerufen am 23.10.2024. [Online]. Available: <https://nuxt.com/docs/guide/directory-structure/nuxt>
- [13] (2024, oct) Components basics. Abgerufen am 23.10.2024. [Online]. Available: <https://vuejs.org/guide/essentials/component-basics>
- [14] (2023, nov) Logging in .net core and asp.net core. Microsoft. Abgerufen am 16.10.2024. [Online]. Available: <https://learn.microsoft.com/en-us/aspnet/core/fundamentals/logging/?view=aspnetcore-8.0#bilp>



- [15] Iexceptionhandler interface. Microsoft. Abgerufen am 16.10.2024. [Online]. Available: <https://learn.microsoft.com/en-us/dotnet/api/microsoft.aspnetcore.diagnostics.iexceptionhandler?view=aspnetcore-8.0>
- [16] (2024, oct) What is vue i18n? Abgerufen am 17.10.2024. [Online]. Available: <https://vue-i18n.intlify.dev/guide/introduction.html>
- [17] (2024, dec) Introduction to understanding wcag 2.2. W3C Web Accessibility Initiative (WAI). Abgerufen am 07.12.2024. [Online]. Available: <https://www.w3.org/WAI/WCAG22/Understanding/intro#understanding-the-four-principles-of-accessibility>
- [18] E. Scott. (2023, oct) Form field usability: Avoid extensive multicolumn layouts (16% make this form usability mistake). Abgerufen am 20.11.2024. [Online]. Available: <https://baymard.com/blog/avoid-multi-column-forms>
- [19] B. Labay. (2016, oct) Form field usability: Should you use single or multi-column forms? [original research]. Abgerufen am 20.11.2024. [Online]. Available: <https://speero.com/post/form-field-usability-should-you-use-single-or-multi-column-forms-original-research#:~:text=Conclusion-,The%20single%2Dcolumn%20form%20was%20faster%20to%20complete.,thinking%20of%2C%20don't>
- [20] P. Vaughan. (2021, june) Disproving best practices: The one- vs. two-column form test. Abgerufen am 20.11.2024. [Online]. Available: <https://blog.hubspot.com/marketing/one-vs-two-column-form-conversion-test>
- [21] (2022, june) Form ui design: 36 tips & best practices. Designlab. Abgerufen am 17.11.2024. [Online]. Available: <https://designlab.com/blog/form-ui-design-best-practices>
- [22] (2024, oct) Primevue: The next-gen ui suite for vue.js. PrimeTek. [Online]. Available: <https://primevue.org/>
- [23] (2024, march) Monorepo vs. polyrepo: How to choose. Buildkite. Abgerufen am 05.12.2024. [Online]. Available: <https://buildkite.com/resources/blog/monorepo-polyrepo-choosing/>
- [24] (2024, dec) Git feature branch workflow. Atlassian. Abgerufen am 05.12.2024. [Online]. Available: <https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow>
- [25] (2024, dec) Swagger ui. SmartBear Software. [Online]. Available: <https://swagger.io/tools/swagger-ui/>
- [26] (2024, dec) Doxygen: Code documentation. automated. doxygen. [Online]. Available: <https://www.doxygen.nl/>

# Kapitel 1

## Einleitung

### 1.1 Ausgangslage

Die OST hat exzellente Forschungsprojekte im anwendungsorientierten Bereich. Die Publikation dieser Projekte geschieht erfolgreich in den jeweiligen Fachgebieten, wobei die Veröffentlichungskanäle je nach Bereich variieren und so den spezifischen Anforderungen der jeweiligen Disziplinen gerecht werden. Eine zusätzliche zentrale, einheitliche Übersicht über alle vergangenen Forschungsprojekte, die mittels einer Suche und ausgeklügelter Filterkriterien durchforstet werden kann, wäre dennoch eine grosse Bereicherung für die OST.

Auf der Website der OST existiert bereits eine Projektsuche, die Vorzeigeprojekte präsentiert und als Aushängeschild der Hochschule für die Öffentlichkeit dient. Diese Seite bietet jedoch noch keine vollständige Übersicht über alle Projekte der OST.

### 1.2 Problemstellung

Um eine solche Übersicht zu erreichen, soll nun eine neue Lösung erstellt werden. Diese Anwendung besteht aus einem Web-Frontend, bestehend aus Projektsuche, Dashboard und dem Erfassungstool, sowie einem Backend, in welchem die Projekte mit relevanten Inhalten (auch interaktive) in einer persistenten Datenbank gespeichert und nach der vollständigen Erfassung freigegeben werden können. Jede Organisationseinheit soll eigenständig die Dateien pflegen. Die Veröffentlichung von Projekten erfolgt nach einem dafür entwickelten Workflow, der eine Prüfung der Projekte nach verschiedenen Kriterien erfordert (z.B. fachliche Prüfung oder Prüfung durch Kommunikationsverantwortliche). Projekte sollen als veröffentlicht, OST-intern oder Organisationseinheit-intern klassifiziert werden können. Dieser Sichtbarkeitsstatus kann von der veröffentlichenden Person frei gesetzt werden. Je nach Status sind wiederum höhere oder niedrigere Prüfhürden erforderlich.

## 1.3 Stand der Technik

### 1.3.1 Stand der OST

Auf der Webseite der OST existiert bereits [eine Projektübersicht](#). Diese ist (laut Auftraggeber) allerdings unvollständig. Ausserdem sind darauf nicht alle relevanten Metadaten einsehbar. Die Suchergebnisse lassen sich nur nach Titel und Beschreibung durchsuchen und nach einem einzelnen Themenbereich filtern.

### 1.3.2 Bestehende Lösungen

Diverse Hochschulen und Universitäten haben bereits eine Webseite für die Veröffentlichung und Sammlung von Projektdaten, so beispielsweise die [ETH Zürich](#). Da die neue Lösung auf die Organisation und den Bearbeitungs- bzw. Prüfprozess der OST angepasst sein soll, entsprechen diese Lösungen jedoch nicht den Anforderungen. Sie dienen aber als Quelle für Inspiration und liefern Informationen zu möglichen Metadaten. Zudem können sie dabei helfen, mögliche Herausforderungen im Verhalten der Anwendung und auch im Bereich der User-Experience zu erkennen.

## 1.4 Ziel der Arbeit

Das konkrete Ziel ist die Erstellung eines Prototyps mit den wichtigsten Funktionalitäten, welcher den tatsächlichen Nutzen einer solchen Anwendung prüfen soll und danach weiter ausgebaut werden kann.

Diese Funktionalitäten umfassen sowohl eine öffentlich einsehbare Projektseite als auch ein internes Projekterfassungs-Tool.

Auf der Projektseite können publizierte Projekte sowohl nach verschiedenen Kriterien (unter anderem Departement, Organisationseinheit, Thema, Region) als auch per Textsuche (Titel, Autoren, Abstract) gefiltert werden. Die einzelnen Projekte können in einer detaillierten Ansicht geöffnet werden, in welcher alle Metadaten (genauer Umfang noch zu definieren) und die zugehörigen Dateien aufgelistet sind. Diese Dateien können heruntergeladen werden. Im Projekterfassungs-Tool können die Organisationseinheiten neue Projekte mit allen Metadaten erfassen. Bestehende Projekte können bearbeitet oder gelöscht/archiviert werden. Zu einem Projekt können zugehörige Dateien hochgeladen werden. Projekte können zu einem beliebigen Zeitpunkt publiziert werden. Wer dafür zuständig bzw. berechtigt ist, muss noch geklärt werden.

## 1.5 Organisatorische Rahmenbedingungen

Der Prototyp wird im Rahmen einer Studienarbeit entwickelt. Diese wird mit 8 Credits pro Teammitglied vergütet, was einem Aufwand von ungefähr 240 Stunden entspricht. Bei einer Teamgrösse von zwei Personen beträgt die gesamte Arbeitszeit somit etwa 480 Stunden. Aufgeteilt auf die 14 Semesterwochen ergibt sich daraus ein Aufwand von ca. 34 Stunden pro Woche.

## 1.6 Abgrenzung

Parallel zu dieser Studienarbeit gibt es das Projekt «Publikationsdatenbank», das unabhängig von dieser Studienarbeit läuft. Die Abgrenzung der Projekte wurde wie folgt definiert:

### **Publikationsdatenbank:**

Erstellung einer einheitlichen und übersichtlichen Plattform, die speziell für die Veröffentlichung an der OST konzipiert ist. Externe können somit Projekte, Artikel und andere Veröffentlichungen der OST an einem zentralen Ort suchen und einsehen.

### **OST ResearchNavigator:**

Eine Applikation zum Tracking von Projekten an der OST mit besonderem Fokus auf die Sammlung relevanter Metadaten. Diese Applikation dient primär als interne Projektübersicht und -erfassung, soll aber auch ein öffentliches Dashboard bieten.

In dieser Arbeit wird die Projekterfassung, -suche und -anzeige inklusive aller relevanten Filterkriterien implementiert. Dazu gehören auch ein Dashboard mit Statistiken, die Möglichkeit, zugehörige Dateien zu Projekten hoch- und herunterzuladen und der Prüfprozess vor der Veröffentlichung eines Projektes. Benutzer sollen sich mit ihrer E-Mail-Adresse und Rolle autorisieren können.

Weitere Features, wie beispielsweise ein Admin-Tool zur Anpassung von Filterkriterien wie Organisationseinheiten, Sponsoren oder Suchtags oder die Benutzerauthentifizierung sind für zukünftige Arbeiten vorgesehen, sofern der Prototyp die Vision des Auftraggebers erfüllt.

## Kapitel 2

# Analyse & Anforderungen

Dieses Kapitel analysiert den Systemkontext und beschreibt die funktionalen sowie auch die nichtfunktionalen Anforderungen des ResearchNavigators. Die Anforderungen werden in die folgenden drei Stufen unterteilt und in der Dokumentation jeweils mit der abgebildeten Farbe gekennzeichnet.

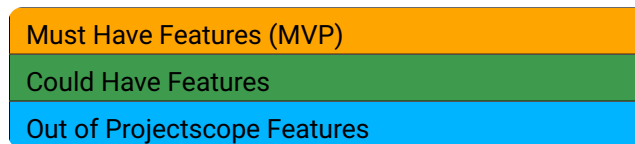


Tabelle 2.1: Farbcodes der Anforderungen

<b>Must Have Features (MVP)</b>	Anforderungen, die am Ende des Projekts erfüllt sein müssen.
<b>Could Have Features</b>	Anforderungen, die im MVP umgesetzt werden, wenn die Zeit es erlaubt.
<b>Out of Projectscope Features</b>	Anforderungen, die nach Abschluss dieser Arbeit in einer Folgearbeit umgesetzt werden können.

## 2.1 Funktionale Anforderungen

Die funktionalen Anforderungen an das System werden in diesem Abschnitt anhand von Use Cases erfasst und in einer Abbildung veranschaulicht.

### 2.1.1 Akteure

Im Folgenden werden die 5 Hauptakteure der Applikation beschrieben. Hervorzuheben ist, dass ein Benutzer der Anwendung diese aus der Perspektive verschiedener Akteure nutzen kann. So kann z.B. jemand gleichzeitig als Prüfer tätig sein und die eingegebenen Projektdaten auf fachliche Korrektheit überprüfen, gleichzeitig aber auch als Mitarbeiter die Möglichkeit haben, eigene Projekte zu erfassen.

Akteure	Beschreibung
Besucher	Bezeichnet einen Benutzer, der nicht angemeldet ist. Es kann sich dabei um eine externe Person oder einen Mitarbeiter der OST handeln. Nicht autorisierte Benutzer haben Zugang zu allen öffentlichen Projekten.
Mitarbeiter	Der Mitarbeiter bezeichnet Personen, die Forschungsprojekte veröffentlichen dürfen.
Leiter	Der Leiter ist ein mit besonderen Befugnissen ausgestatteter Mitarbeiter einer Organisationseinheit. Er kann alle Projekte seiner Organisationseinheit verwalten.
Prüfer	Der Akteur Prüfer ist eine Person, die am Prüfungsprozess eines Projekts beteiligt ist. Er überprüft und vervollständigt die Angaben zum Projekt.
Systemadministrator	Der Systemadministrator kann den Benutzern der Applikation ihre jeweiligen Berechtigungen zuweisen und ändern sowie Filterkriterien definieren und verwalten.

Tabelle 2.2: Beschreibung der Akteure

### 2.1.2 Use Case Diagramm

Das Use Case Diagramm gibt einen Überblick über alle existierenden Use Cases, die im nächsten Abschnitt näher erläutert werden. Die orange markierten Use Cases sind Teil des MVP. Um die Übersichtlichkeit zu gewähren, sind im Use Case Diagramm nicht alle Beziehungen vollständig dargestellt. So haben z.B. theoretisch alle Akteure eine Beziehung zu den Use Cases, in denen der Besucher der Hauptakteur ist. Ebenso hätten alle Akteure ausser dem Besucher eine Verbindung zum Anwendungsfall Authentifizieren.

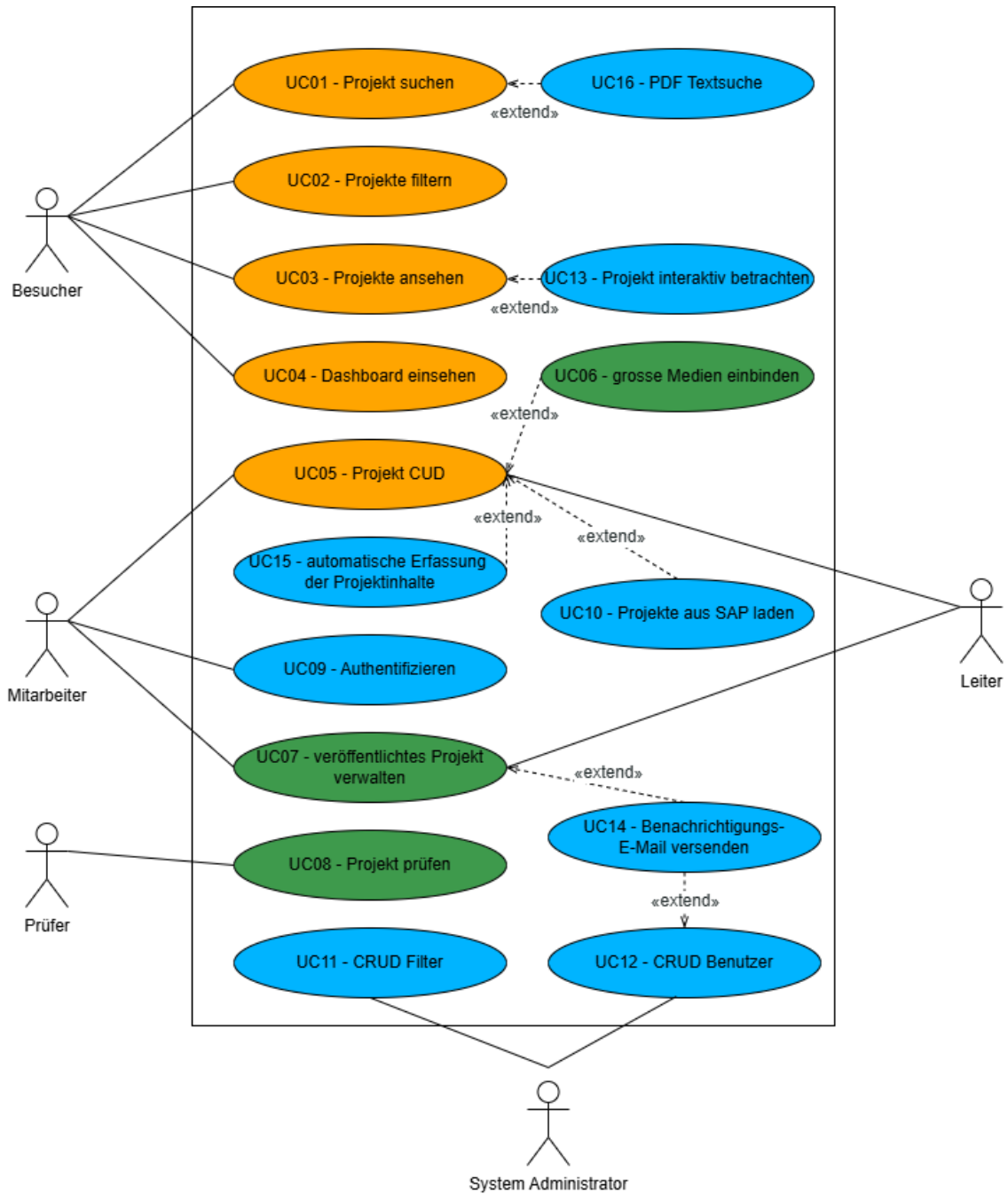


Abbildung 2.1: Use Case Diagramm

### 2.1.3 Use Cases

Die Use Cases, welche die funktionalen Anforderungen an das System bilden, werden nach dem Modell von Craig Larman erfasst [1]. Die von Larman empfohlene Vorlage im **Fully Dressed Format**, auf deren Grundlage die einzelnen Use Cases ausformuliert werden, ist jedoch aus Gründen der Übersichtlichkeit und Redundanz um einige Felder gekürzt und leicht abgeändert. Zudem werden Use Cases der Kategorie Out of Projectscope nur anhand ihres jeweiligen Haupterfolgsszenarios beschrieben. Diese werden in einer Folgearbeit ausführlicher spezifiziert.

UC01 - Projekt suchen

MVP

Ein Besucher kann im Katalog aller Projekte nach einem bestimmten Projekt suchen.

Primärer Akteur	Besucher
Weitere Akteure	Mitarbeiter, Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher öffnet die Suchleiste und gibt einen Suchbegriff ein.</li> <li>2. Besucher bestätigt die Suche.</li> <li>3. Das System sucht nach Projekten, in denen dieser Suchbegriff mit dem Autor, dem Titel oder dem Abstract übereinstimmt.</li> <li>4. Die Suchergebnisse werden in einer Liste angezeigt.</li> </ol>
Alternatives Szenario	Sind zum gesuchten Begriff keine Ergebnisse verfügbar, so erscheint ein Text, der dies dem Benutzer mitteilt.
Weitere Spezifikationen	<p>Nach folgenden Kriterien soll gesucht werden können:</p> <ul style="list-style-type: none"> <li>• Titel</li> <li>• Autor</li> <li>• Abstract</li> </ul>
Häufigkeit	Sehr Hoch

Tabelle 2.3: UC01 - Projekt suchen



## UC02 - Projekte filtern

MVP

Ein Besucher hat die Möglichkeit, im Katalog aller Projekte anhand festgelegter Kriterien nach bestimmten Projekten zu filtern.

Primärer Akteur	Besucher
Weitere Akteure	Mitarbeiter, Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher öffnet die Filteroptionen und wählt die gewünschten Filter aus.</li> <li>2. Der Besucher führt eine Suche unter Verwendung dieser Filterkriterien durch.</li> <li>3. Das System filtert die Projekte anhand der gesetzten Filterkriterien und liefert nur die Projekte, die diesen Kriterien entsprechen.</li> <li>4. Die gefundenen Projekte, deren Anzeige für den suchenden Benutzer erlaubt ist, werden in einer Liste angezeigt.</li> </ol>
Alternatives Szenario	Können anhand der gesetzten Filter keine Ergebnisse gefunden werden, so erscheint ein Text, der dies dem Benutzer mitteilt.
Weitere Spezifikationen	<p>Nach folgenden Kriterien soll gefiltert werden können:</p> <ul style="list-style-type: none"> <li>• Art des Projektergebnisses</li> <li>• Schlagwörter</li> <li>• Region der Projektdurchführung</li> <li>• Sponsor des Projekts</li> <li>• Durchführungszeitraum</li> <li>• Datum der Veröffentlichung</li> <li>• Organisationseinheit</li> <li>• Departement</li> <li>• Sichtbarkeit</li> </ul>
Häufigkeit	Hoch

Tabelle 2.4: UC02 - Projekte filtern

## UC03 - Projekt ansehen

MVP

Ein Besucher kann sich ein Projekt in einer Detailansicht ansehen.

Primärer Akteur	Besucher
Weitere Akteure	Mitarbeiter, Leiter, Prüfer
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher klickt auf ein Projekt.</li> <li>2. Das System lädt alle relevanten Daten zu diesem Projekt und zeigt diese dem Besucher an.</li> </ol>
Alternatives Szenario	Beim Laden des Projekts kommt es zu einem Fehler, z.B. aufgrund von Netzwerkproblemen. Der Benutzer erhält in diesem Fall eine Fehlermeldung.
Häufigkeit	Sehr Hoch

Tabelle 2.5: UC03 - Projekt ansehen

## UC04 - Dashboard einsehen

Ein Besucher kann sich selbstständig über Daten zu Projekten an der OST über ein Dashboard informieren.

Primärer Akteur	Besucher
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher navigiert zur Dashboard-Seite</li> <li>2. Das System zeigt alle vorhandenen Infografiken an.</li> <li>3. Der Besucher kann die Infografiken filtern, um die für ihn relevanten Informationen zu erhalten.</li> <li>4. Das System aktualisiert die Infografiken anhand der gesetzten Filterkriterien.</li> </ol>
Alternatives Szenario	Der Besucher wählt Filterkriterien, zu denen es keine Projekteinträge gibt. Dem Besucher wird in diesem Fall eine Meldung angezeigt, die ihn darauf hinweist, dass es anhand der gewählten Filterkriterien keine verfügbaren Projekte gibt.
Häufigkeit	Mittel

Tabelle 2.6: UC04 - Dashboard einsehen

UC05 - Projekt CUD

**MVP**

Ein Mitarbeiter kann neue Projekte anlegen und diese solange bearbeiten und löschen, bis diese zur Überprüfung freigegeben werden.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Mitarbeiter navigiert zur Übersicht seiner Forschungsprojekte.</li> <li>2. Das System zeigt die Forschungsprojekte an und bietet über eine Schaltfläche die Möglichkeit, ein neues Projekt zu erstellen.</li> <li>3. Der Mitarbeiter klickt auf die Schaltfläche "Neues Projekt anlegen".</li> <li>4. Das System zeigt ein Formular an, in das der Mitarbeiter die notwendigen Informationen und Dokumente zu seinem Projekt eintragen kann.</li> <li>5. Der Mitarbeiter erfasst die erforderlichen Daten und speichert sie.</li> </ol>
Alternatives Szenario	Der Mitarbeiter erstellt ein neues Projekt anhand des Navigationspunktes "Neues Projekt anlegen" in der Navigationsleiste.
Weitere Spezifikationen	<ul style="list-style-type: none"> <li>• Es können auch unvollständige Projekte gespeichert werden.</li> <li>• Erst beim Abschluss einer Erfassung (Freigabe zur Prüfung) sind alle Informationen erforderlich.</li> </ul>
Häufigkeit	Mittel

Tabelle 2.7: UC05 - Projekt CUD

UC06 - grosse Medien einbinden

MVP

Mitarbeiter können für ihr Projekt grosse Dateien hochladen.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Mitarbeiter befindet sich in der Erstellungsansicht eines Projekts.</li> <li>2. Das System zeigt dem Mitarbeiter eine Schaltfläche an, um Medien hochzuladen.</li> <li>3. Der Mitarbeiter kann Medien hochladen, die unterhalb der festgelegten maximalen Dateigrösse liegen.</li> </ol>
Alternatives Szenario	Mitarbeiter möchte Medien hochladen, die grösser als die maximal zulässige Dateigrösse sind. Das System lehnt den Upload ab und zeigt eine Fehlermeldung an, die die maximale Dateigrösse nennt (z. B. "Maximale Dateigrösse überschritten: 50 MB").
Weitere Spezifikationen	<p>Folgende Medien sollen unterstützt werden:</p> <ul style="list-style-type: none"> <li>• Titelbild des Projekts</li> <li>• Bilder und Grafiken für die Veranschaulichung von Projektergebnissen</li> <li>• Videos der Projektergebnisse</li> <li>• Dokumente, welche den Besuchern zum Download angeboten werden.</li> </ul>
Häufigkeit	Mittel

Tabelle 2.8: UC06 - Medien einbinden

## UC07 - veröffentlichtes Projekt verwalten

Ein Leiter kann Projekte, die geprüft und veröffentlicht wurden, verwalten.

Primärer Akteur	Leiter
Weitere Akteure	Mitarbeiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Leiter navigiert zur Ansicht aller Forschungsprojekte seiner Organisationseinheit.</li> <li>2. Das System zeigt alle Projekte einer Organisationseinheit an. Jedes Projekt hat einen Button, mit dem man in die Bearbeitungsansicht gelangt.</li> <li>3. Der Leiter klickt auf den Button, um ein Forschungsprojekt zu bearbeiten.</li> <li>4. Das System zeigt die Bearbeitungsansicht des Projekts mit der Möglichkeit, projektspezifische Informationen zu ändern und zu speichern.</li> </ol>
Alternatives Szenario	Handelt es sich beim Akteur um einen Mitarbeiter, so kann dieser nur seine eigenen Projekte verwalten.
Detailspezifikationen	<ul style="list-style-type: none"> <li>• Projekt kann archiviert werden.</li> <li>• Sichtbarkeit des Projekts sowie Informationen über das Projekt können bearbeitet werden.</li> <li>• Projekte werden an Prüfstellen übergeben, sofern die Sichtbarkeit des Projekts erhöht werden soll.</li> </ul>
Häufigkeit	Selten

Tabelle 2.9: UC07 - veröffentlichtes Projekt verwalten

## UC08 - Projekt prüfen

Ein Prüfer kann ein erstelltes Projekt hinsichtlich Fehlern und kritischer Inhalte überprüfen.

Primärer Akteur	Prüfer
Vorbedingungen	Das Projekt wurde erfasst und zur Prüfung durch den entsprechenden Prüfer freigegeben.
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Prüfer navigiert zur Inbox der zu prüfenden Projekte.</li> <li>2. Das System zeigt alle Projekte an, die eine Prüfung durch diesen Prüfer erfordern (hinsichtlich Kommunikations- oder fachlicher Prüfung).</li> <li>3. Der Prüfer wählt ein Projekt aus und kann dessen Informationen bearbeiten.</li> <li>4. Der Prüfer kann das Projekt veröffentlichen oder zur Prüfung durch die nächste Prüfstelle freigeben.</li> </ol>
Alternatives Szenario	Ist der Prüfer mit den erfassten Informationen nicht zufrieden und der Meinung, dass dieses Projekt erneut überarbeitet werden muss, weist er dieses an den Ersteller zurück.
Häufigkeit	Mittel

Tabelle 2.10: UC08 - Projekt prüfen

## UC09 - Authentifizieren

Ein Benutzer kann sich mit seinem Benutzerkonto am System anmelden.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter, Prüfer
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Mitarbeiter navigiert zur Loginseite.</li> <li>2. Das System zeigt ein Formular an, in dem der Benutzer seine E-Mail-Adresse und sein Passwort eingeben kann.</li> <li>3. Der Benutzer gibt seine Anmeldedaten ein.</li> <li>4. Das System überprüft die Eingaben.</li> <li>5. Der Mitarbeiter wird in seinen Account eingeloggt.</li> </ol>

Tabelle 2.11: UC09 - Authentifizieren

**UC10 - Projekte aus SAP laden**

In SAP erfasste Projekte werden automatisch in die Datenbank der Applikation eingetragen. Ein Mitarbeiter des Projekts kann diesen Eintrag anschliessend vervollständigen.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Mitarbeiter schliesst ein Projekt im SAP ab.</li> <li>2. Das System trägt erfasst dieses Projekt automatisch in der Datenbank der Anwendung.</li> <li>3. Der Mitarbeiter kann die zur Veröffentlichung benötigten Angaben ergänzen.</li> </ol>

Tabelle 2.12: UC10 - Projekte aus SAP laden

**UC11 - CRUD Filter**

Ein Administrator kann neue Filter erstellen und bestehende Filter verwalten.

Primärer Akteur	Systemadministrator
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Administrator navigiert zur Übersichtsseite der Filter.</li> <li>2. Das System zeigt alle vorhandenen Filter an.</li> <li>3. Der Administrator kann bestehende Filter verwalten.</li> </ol>

Tabelle 2.13: UC11 - CRUD Filter

**UC12 - CRUD Benutzer**

Ein Administrator kann neue Benutzer erfassen, diese bearbeiten und löschen. Er kann Berechtigungen erteilen und auch wieder entziehen.

Primärer Akteur	Systemadministrator
Haupterfolgsszenario	<p>Das folgende Szenario ist noch von der konkreten Implementation der Authentifizierung abhängig und kann sich daher noch leicht ändern.</p> <ol style="list-style-type: none"> <li>1. Der Administrator navigiert zur Übersichtsseite der Benutzerverwaltung.</li> <li>2. Das System zeigt Optionen zur Verwaltung der Rechte von bestehenden Benutzern sowie zur Erfassung neuer Benutzer.</li> <li>3. Der Administrator kann Benutzerrechte ändern und neue Benutzer erfassen.</li> </ol>

Tabelle 2.14: UC12 - CRUD Benutzer

**UC13 - Projekt interaktiv betrachten**

Ein Besucher kann ein Projekt oder die dazugehörigen Medien interaktiv betrachten.

Primärer Akteur	Besucher
Weitere Akteure	Mitarbeiter, Leiter, Prüfer
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher klickt auf ein Projekt.</li> <li>2. Das System lädt alle relevanten Daten zu diesem Projekt und zeigt diese dem Besucher an.</li> <li>3. Der Besucher kann mit den dargestellten Inhalten interagieren.</li> </ol>

Tabelle 2.15: UC13 - Projekt interaktiv betrachten

**UC14 - Benachrichtigungs-E-Mail versenden**

Ein Benutzer kann über wichtige Änderungen per E-Mail benachrichtigt werden.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter, Prüfer
Haupterfolgsszenario	<p>Für diesen Use Case sind diverse Szenarien denkbar, die in den Abnahmekriterien genauer beschrieben werden.</p> <ol style="list-style-type: none"> <li>1. Der Mitarbeiter überweist ein Projekt an einen bestimmten Prüfer, für die Prüfung nach fachlichen Kriterien.</li> <li>2. Das System sendet eine Benachrichtigungs-E-Mail an den Prüfer, um ihn auf das zu prüfende Projekt aufmerksam zu machen.</li> <li>3. Der Prüfer erhält die E-Mail und kann sofort mit der Überprüfung beginnen.</li> </ol>

Tabelle 2.16: UC14 - Benachrichtigungs-E-Mail versenden



### UC15 - automatische Erfassung der Projektinhalte

Ein Benutzer kann die für die Projekterfassung nötigen Projekteinhalte automatisch anhand eines PDF-Dokuments generieren lassen.

Primärer Akteur	Mitarbeiter
Weitere Akteure	Leiter
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Mitarbeiter navigiert zur Seite der Projekterfassung.</li> <li>2. Hier kann er die Auswahl treffen, den Inhalt automatisch erfassen zu lassen.</li> <li>3. Das System fordert den Mitarbeiter dazu auf ein PDF-Dokument hochzuladen.</li> <li>4. Der Mitarbeiter lädt ein PDF seiner Arbeit hoch.</li> <li>5. Das System füllt die Felder, die für eine Projekterfassung nötig sind, automatisch anhand des PDF-Inhalts aus.</li> </ol>

Tabelle 2.17: UC15 - automatische Erfassung der Projekteinhalte

### UC16 - PDF Textsuche

Projekte können anhand von Inhalten, die nur in zum Projekt zugehörigen PDF-Dokumenten vorhanden sind, gefunden werden.

Primärer Akteur	Besucher
Weitere Akteure	Mitarbeiter, Leiter, Prüfer
Haupterfolgsszenario	<ol style="list-style-type: none"> <li>1. Der Besucher navigiert zur Seite der Projektsuche.</li> <li>2. In einem dafür vorgesehenen Textfeld sucht er nach einem bestimmten Projekt.</li> <li>3. Sofern sich der von ihm gesuchte Text in einem der PDF-Dokumente des gesuchten Projekts befindet, wird dieses Projekt in der Liste aller Suchresultate aufgelistet.</li> </ol>

Tabelle 2.18: UC16 - PDF Textsuche

## 2.2 Nicht-Funktionale Anforderungen

Die **Non-functional Requirements (NFR)** sind nach der Norm ISO/IEC 25010 gestaltet, welches die folgenden acht Qualitätscharakteristiken enthält:

- Functional Suitability
- Performance Efficiency
- Compatibility
- Operability/Usability
- Reliability
- Security
- Maintainability
- Transferability

Im Folgenden sind Anforderungen für sieben der acht Charakteristiken in Tabellenform definiert. Für "Functional Suitability" wurden bereits im vorhergehenden Kapitel die **Functional Requirements (FR)** als Use Cases formuliert. Die **NFR** erhalten eine ID, bestehend aus dem Prefix "NFR-" und einer Nummerierung. Für die Priorisierung werden die drei Level "Niedrig", "Medium", und "Hoch" verwendet.

Die Anforderungen müssen bis zum angegebenen Fälligkeitsdatum mit den beschriebenen Verifikationstools überprüft werden. Für das Erfüllungslevel werden die Zustände "Nicht erfüllt", "Teilweise erfüllt" sowie "Vollständig erfüllt" eingetragen und mit einer kurzen Beschreibung ergänzt.

<b>ID</b>	NFR-1
<b>Titel</b>	Response Time Projektsuche
<b>Anforderung</b>	Performance Efficiency
<b>Beschreibung</b>	Die Response Time ist <3s für eine einfache filterbasierte Suche, <4s für eine Textsuche eine Tag-Suche
<b>Verifikationstools</b>	Apache JMeter
<b>Priorität</b>	Hoch
<b>Definiert am</b>	01.10.2024
<b>Fälligkeitsdatum</b>	22.11.2024
<b>Verifikationsdatum</b>	20.11.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	In Apache JMeter wurden gemäss <a href="#">Unterabschnitt 5.4.4</a> Tests für filter-, tag- und textbasierte Suche ausgeführt. Jeder Durchlauf blieb unter dem vorgegebenen Grenzwert. Diese Auswertung bietet einen ersten Anhaltspunkt, sollte aber zu einem späteren Zeitpunkt wiederholt werden, wenn mehr Projekte mit umfangreichem Inhalt vorhanden sind. Screenshots dazu befinden sich im Anhang ( <a href="#">Unterabschnitt 9.1.4</a> ).

Tabelle 2.19: NFR-1: Response Time Projektsuche

<b>ID</b>	NFR-2
<b>Titel</b>	Response Time Projektbearbeitung
<b>Anforderung</b>	Performance Efficiency
<b>Beschreibung</b>	Die Response Time für CUD Operationen der Projektmetadaten bzw. einzelner Projektdateien (bis zu einer Grösse von 10 MB) ist <2s.
<b>Verifikationstools</b>	Apache JMeter
<b>Priorität</b>	Medium
<b>Definiert am</b>	01.10.2024
<b>Fälligkeitsdatum</b>	06.12.2024
<b>Verifikationsdatum</b>	13.12.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	<p>06.12.2024: In Apache JMeter wurden gemäss <a href="#">Unterabschnitt 5.4.4</a> Tests für CUD-Operationen durchgeführt. Jeder Durchlauf blieb unter dem vorgegebenen Grenzwert. Der File-Upload ist noch nicht vollständig implementiert.</p> <p>13.12.2024: In Apache JMeter wurden gemäss <a href="#">Unterabschnitt 5.4.4</a> Tests für den File-Upload durchgeführt. Bis zur maximalen getesteten Grösse von 10.7 MB blieb jeder Durchlauf unter dem vorgegebenen Grenzwert. Die Upload-Zeiten sind allerdings stark abhängig von der Verbindungsgeschwindigkeit des Anwenders. Screenshots dazu befinden sich im Anhang (<a href="#">Unterabschnitt 9.1.4</a>).</p>

Tabelle 2.20: NFR-2: Response Time Projektbearbeitung

<b>ID</b>	NFR-3
<b>Titel</b>	Browser-Kompatibilität
<b>Anforderung</b>	Compatibility
<b>Beschreibung</b>	Die Projektsuche und -anzeige sowie die Projektbearbeitung ist kompatibel mit den neusten Versionen der Webbrowser Google Chrome, Microsoft Edge, Safari und Mozilla Firefox.
<b>Verifikationstools</b>	Cross Browser Testing Online Tools, diverse Web-Browser
<b>Priorität</b>	Medium
<b>Definiert am</b>	01.10.2024
<b>Fälligkeitsdatum</b>	29.11.2024
<b>Verifikationsdatum</b>	29.11.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	Alle Seiten der Web-App wurden in den vorgegebenen Browsern geöffnet und die zentralen Funktionalitäten wurden überprüft. Wird fortlaufend erneut getestet.

Tabelle 2.21: NFR-3: Browser-Kompatibilität

<b>ID</b>	NFR-4
<b>Titel</b>	Affordance Projektsuche
<b>Anforderung</b>	Usability
<b>Beschreibung</b>	Benutzer können die Projektsuche, insbesondere die Filterfunktion, ohne vorhergehendes Training verwenden und finden die benötigten Filter-/Suchkriterien in <6s. Pro (erweitertem) Filterkriterium sind nicht mehr als 3 Klicks notwendig.
<b>Verifikationstools</b>	Usability Testing mit mindestens 3 potenziellen Benutzern
<b>Priorität</b>	Hoch
<b>Definiert am</b>	01.10.2024
<b>Fälligkeitsdatum</b>	15.11.2024
<b>Verifikationsdatum</b>	11.11.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	<p>11.11.2024: Usability Testing (siehe <a href="#">Unterabschnitt 5.4.2</a>) mit Frieder Loch. Die meisten Filter- bzw. Suchkriterien wurden in der gegebenen Zeit gefunden, es sind aber auch diverse Verbesserungsmöglichkeiten aufgefallen.</p> <p>09.12.2024: Usability Testing mit Markus Stolze und Frieder Loch. Die Filterkriterien wurden gefunden. Alle Kriterien können innerhalb von drei Klicks gesetzt werden. (Anmerkung: Bei den hierarchischen Filtern (Region, Tags) und den Datumsfiltern sind je nach Abstufung bzw. nach Datum mehr Klicks nötig.)</p> <p>Das vollständige Protokoll befindet sich im Anhang (<a href="#">Unterabschnitt 9.1.1</a>).</p>

Tabelle 2.22: NFR-4: Affordance Projektsuche

<b>ID</b>	NFR-5
<b>Titel</b>	Authorization
<b>Anforderung</b>	Security
<b>Beschreibung</b>	Das Projekt-Editor-Tool kann nur mit der entsprechenden Autorisierung (Institute/Fachpersonen) verwendet werden. Die Berechtigungen werden mit jeder Backend-Request geprüft. Die Authentifizierung ist in diesem Prototyp explizit ausgeschlossen.
<b>Verifikationstools</b>	Swagger UI
<b>Priorität</b>	Hoch
<b>Definiert am</b>	08.10.2024
<b>Fälligkeitsdatum</b>	06.12.2024
<b>Verifikationsdatum</b>	07.12.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	Alle Endpoints für die CRUD-Operationen wurden mittels SwaggerUI mit und ohne Autorisierung getestet und verhalten sich gemäss den Anforderungen. Im Frontend lassen sich die Bearbeitungsseiten nicht ohne Autorisierung öffnen und führen den Benutzer zur Login-Page.

Tabelle 2.23: NFR-5: Authorization

<b>ID</b>	NFR-6
<b>Titel</b>	Test Coverage
<b>Anforderung</b>	Maintainability
<b>Beschreibung</b>	Die Unit Tests weisen eine <b>Branch Coverage</b> von >80% für die Service-Methoden im Backend auf.
<b>Verifikationstools</b>	XUnit
<b>Priorität</b>	Medium
<b>Definiert am</b>	08.10.2024
<b>Fälligkeitsdatum</b>	06.12.2024
<b>Verifikationsdatum</b>	18.12.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	Die <b>Branch Coverage</b> in den Service-Methoden liegt bei 87% und somit über dem vorgegebenen Grenzwert (siehe <b>Coverage Metrics auf GitLab</b> und <b>Unterabschnitt 5.4.1</b> ). Muss nach der Implementation neuer Features erneut verifiziert werden.

Tabelle 2.24: NFR-6: Test Coverage

<b>ID</b>	NFR-7
<b>Titel</b>	Portable Docker Container
<b>Anforderung</b>	Transferability
<b>Beschreibung</b>	Damit die Anwendungen einfach auf neuen Systemen aufgesetzt und gestartet werden können, müssen alle Teilsysteme als Docker-Container ausführbar sein.
<b>Verifikationstools</b>	Docker Compose, verschiedene OS
<b>Priorität</b>	Medium
<b>Definiert am</b>	08.10.2024
<b>Fälligkeitsdatum</b>	22.11.2024
<b>Verifikationsdatum</b>	15.11.2024
<b>Erfüllungslevel</b>	Vollständig erfüllt
<b>Verifikationsdetails</b>	Setup und Start der Applikation mit Docker Compose funktionieren mit einem einzigen Befehl (sofern die benötigten Umgebungsvariablen gemäss README gesetzt sind.)

Tabelle 2.25: NFR-7: Portable Docker Container

<b>ID</b>	NFR-8 (Optional)
<b>Titel</b>	Backups
<b>Anforderung</b>	Reliability
<b>Beschreibung</b>	Mindestens alle 24h wird ein Backup der gesamten Datenbank angelegt, welches alle aktuell gespeicherten Projekt-Metadaten enthält.
<b>Verifikationstools</b>	Manuelle Überprüfung in regelmässigen Abständen
<b>Priorität</b>	Niedrig
<b>Definiert am</b>	08.10.2024
<b>Fälligkeitsdatum</b>	13.12.2024
<b>Verifikationsdatum</b>	-
<b>Erfüllungslevel</b>	Nicht erfüllt
<b>Verifikationsdetails</b>	Diese Anforderung wurde aufgrund der geringen Bedeutung im aktuellen Produktstand nicht umgesetzt. PostgreSQL erstellt allerdings automatisch Snapshots in gewissen Abständen.

Tabelle 2.26: NFR-8: Backups

## 2.3 Nicht Anforderungen

Nicht Anforderungen sind Anforderungen, die für diese Art der Anwendung denkbar wären, aber bewusst weggelassen werden. Es sind Anforderungen, die vom Kunden nicht explizit gewünscht werden und durch das Entwicklerteam als nicht sinnvoll angesehen werden. Dies kann unterschiedliche Gründe haben. Zum Beispiel, weil sie den Benutzer einschränken oder deren Nutzen den zur Umsetzung benötigten Aufwand nicht rechtfertigen würde.

### 2.3.1 Undo- und Redo-Funktionalitäten in der Projekterfassung

Bei der Erfassung von Projekten wird kein eigens implementiertes Undo oder Redo angeboten. Diese Funktionalität wird von modernen Browsern für HTML Text-Elemente automatisch angeboten. Ein eigenes Undo/Redo könnte daher im Formular nur für Aktionen ohne Textfeld einen Mehrwert bieten, wie z.B. das Ändern der gewünschten Sichtbarkeit oder das Hinzufügen von Personen. Da es sich hierbei aber um Aktionen handelt, die jeweils mit einem Klick rückgängig gemacht werden können, ist der Mehrwert dieser Funktionalität sehr begrenzt.

### 2.3.2 Löschen von Projekten

Das Löschen von veröffentlichten Projekten wird bewusst nicht angeboten, da dadurch wichtige Informationen verloren gehen könnten. Stattdessen können veröffentlichte Projekte nur archiviert werden, um Informationen zu möglichen Archivzwecken zu behalten. Die einzige Ausnahme sind Projekte, die sich im Status "Draft" befinden. Diese können beliebig gelöscht werden.

### 2.3.3 Parallele Projekterfassung durch mehrere Personen

Eine parallele Erfassung von Projekten wird nicht angeboten, da dies die Komplexität des Systems erheblich erhöhen und zu Inkonsistenzen oder Fehlern bei zeitgleicher Bearbeitung führen könnte. Die

Projekterfassung ist nicht besonders umfangreich und kann von einer Person effizient durchgeführt werden. Zudem wird das Vier-Augen-Prinzip durch den Prüfprozess sichergestellt.

### 2.3.4 Benutzerdefinierte Dashboards

Das Dashboard ist im aktuellen Stand der Anwendung für alle Benutzer gleich. Individuelle Dashboards sind keine aktuelle Anforderung. Dies könnte in Zukunft jedoch in Betracht gezogen werden. Sollte es deutlich mehr Vorschläge und Wünsche für zusätzliche Grafiken geben, wäre es denkbar, dass die Nutzer ihr Dashboard selbst anpassen können, z.B. durch die Auswahl der angezeigten Module oder deren Reihenfolge. Für die aktuell überschaubare Anzahl an Grafiken ist dies jedoch nicht verhältnismässig.

## 2.4 Domain Model

Die Domänenanforderungen wurden in zwei Meetings den Projektwochen 2 und 5 definiert und nach einem weiteren Meeting in Projektwoche 11 überarbeitet. Die Protokolle der Besprechungen befinden sich im Anhang (??).

Ein nicht angemeldeter User kann eine Suchanfrage starten und dabei nach Titel, Autor und Abstract suchen sowie nach Organisationseinheiten, Departementen, Tags, Sponsoren, Regionen, Sichtbarkeit, Veröffentlichungs- und Enddatum filtern. Diese Anfrage liefert passende Projekte als Suchergebnisse zurück.

Angemeldete Benutzer können Projekte zusätzlich erstellen und bearbeiten, sofern sie Mitarbeiter einer Organisationseinheit sind, welche am Projekt beteiligt ist. Projekte werden erst nach ihrer Fertigstellung veröffentlicht.

Nach Vervollständigung der benötigten Projektdaten muss jedes Projekt zur Prüfung eingereicht werden, um eine höhere Sichtbarkeitsstufe zu erreichen. Für interne Projekte der Organisationseinheit ist keine Prüfung notwendig, für OST-interne Projekte nur eine fachliche und für öffentliche Projekte zusätzlich eine kommunikative Prüfung.

Bei der Bearbeitung eines bereits veröffentlichten Projektes wird eine neue Version dieses angelegt, welche erneut den Prüfprozess durchlaufen muss. Die Vorgänger-Version wird archiviert.

Der Status des Projektes ergibt sich aus dem Bearbeitungszustand und den Prüfstadien. Dabei gibt es folgende Abstufungen:

- Entwurf
- Fachliche Prüfung
- Kommunikative Prüfung
- Freigegeben
- Archiviert

Benutzer, welche für die fachliche oder die kommunikative Prüfung ausgewählt wurden, können das Projekt ebenfalls überarbeiten, wobei für letztere nur bestimmte kommunikative Prüfpersonen in Frage kommen. Die Prüfung kann die Zustände ausstehend, abgeschlossen und abgelehnt einnehmen. Letzteres bedingt eine erneute Prüfung nach Überarbeitung des Projektes.

In der Realität kann ein Projekt von mehreren (oder gar keinen) Sponsoren gefördert werden. Die Sponsoren-Auswahl ist zudem nicht fest definiert, sondern sowohl erweiter- als auch veränderbar (z.B. Namensänderungen oder Nachfolger). Dies widerspricht den vorherigen Anforderungen aus Woche 5, welche genau einen Sponsor pro Projekt aus einer begrenzten Auswahl verlangen. Aus zeitlichen Gründen wurde von den neuen Anforderungen (definiert im Meeting in Woche 11) nur die Option "Kein Sponsor" hinzugefügt.

Auf die folgenden rekursiven Beziehungen wurde aus Gründen der Komplexität in diesem Diagramm verzichtet:

- Projekt: Vorgänger/Nachfolger
- Region: Hierarchie (z.B. Schweiz -> Kantone)
- Suchtags: Hierarchie (z.B. Technologie -> KI)
- Organisationseinheiten: Organigramm

Da die Erweiterung der Anforderungen an die Organisationseinheiten (zuvor "Institute") erst im letzten Meeting erfolgte, wurde diese rekursive Beziehung aus Zeitgründen und aufgrund der Priorisierung anderer Features und Optimierungen nicht umgesetzt.





# Kapitel 3

## Lösungskonzept

### 3.1 Analyse und Evaluation

#### 3.1.1 Analyse von Anwendungen mit ähnlicher Funktionalität

Vor der Entwicklung der Anwendung werden zunächst bestehende Lösungen mit ähnlicher Funktionalität betrachtet. Ziel ist es, Inspirationen für die Gestaltung zu sammeln, bewährte Ansätze zu identifizieren, und auch zu erkennen, wo bestehende Systeme ihre Schwächen haben.

Zu den Anwendungen, die näher betrachtet wurden, gehören die *ZHAW digitalcollection* [2] und die *ETH Research Collection* [3]. Obwohl sie inhaltlich primär auf Publikationen ausgerichtet sind, ist ihre Gestaltung und Struktur sowie ihre Such- und Filterfunktionalität auch für den OST ResearchNavigator relevant.

Die *ZHAW digitalcollection* zeigt sich modern und bietet nebst einer robusten Suche auch vielfältige Filterkriterien. Was hier auffällt, ist, dass die Arbeiten nicht beliebig nach allen Filterkriterien filterbar sind. Nach gewissen Kriterien, die einen grossen Umfang an Filteroptionen hätten, wie "Publikationstyp" oder "Organisationseinheit" kann stattdessen gebrowsed werden. Eine nützliche Funktion, die jedoch bei einem grossen Umfang an Optionen auch unübersichtlich werden kann. Zudem schränkt das "browsen" auch die Filterfunktionalität ein.

Die *ETH Research Collection* weist einen besonders grossen Umfang an Projekten auf. Besonders ist die baumartige Struktur, die es ermöglicht, Publikationen nach Organisationseinheiten zu durchsuchen. Diese stark verschachtelte Struktur ermöglicht die Navigation durch die verschiedenen Organisationseinheiten. Darüber hinaus bietet die Research Collection eine Vielzahl von Filtermöglichkeiten, die entweder direkt von der Startseite aus verlinkt sind oder nach einer Suchanfrage am Seitenrand angezeigt werden. Zudem können Filter über einen Filter-Builder selbst zusammengestellt werden. Dies ermöglicht eine hohe Flexibilität, bringt aber auch eine gewisse Komplexität für den Anwender mit sich.

Mit besonderem Augenmerk auf moderne Such- und Filterfunktionalitäten werden weitere Anwendungen analysiert, die für solche Funktionen bekannt sind. Dazu gehören z.B. Online-Shops und Bibliotheksplattformen wie *Digitec* [4], *Swisscovery* [5] [5], *SpringerLink* oder *IEEE Xplore* [6]. Diese Plattformen setzen Filter meist über oder neben der Ergebnisliste ein, wobei die meisten eine Kombination aus beiden Positionierungen verwenden. Problematisch erscheinen dabei Seiten, die keine klare Struktur bieten und deren Filteroptionen verstreut wirken.

Um eine leicht bedienbare Suche zu bieten, wird darauf geachtet, dass diese einheitlich ist. Für die Art des Filters auf der Suchseite werden 3 Varianten in Betracht gezogen:

1. Filter auf der Seite vertikal angeordnet.
2. Filter oberhalb der Suchergebnisse horizontal als eine Reihe oder Fläche von Dropdowns angeordnet.
3. Filter oberhalb der Suchergebnisse als interaktiver Filter-Builder: Der Benutzer wählt durch Klicken auf einen +-Button nacheinander ein Filterkriterium, einen Vergleichsoperator (z. B. <, >, =) und einen Suchwert aus.

Variante 3 bietet den Vorteil, dass sie auch bei einer grossen Anzahl an Filtern übersichtlich bleibt, da die Filter im Builder auch gesucht werden könnten. Allerdings fehlt ihr eine schnelle Übersicht über alle Filterkriterien. Die Bedienung gestaltet sich zudem im Vergleich zu den anderen Optionen umständlicher. Der derzeit geplante Umfang an Filterkriterien rechtfertigt die Vorteile dieser Option nicht, daher wird Variante 3 ausgeschlossen. Für Variante 1 und Variante 2 wurden jeweils Mockups erstellt. Die Entscheidung fiel zugunsten von Variante 1 aufgrund subjektiver Bewertungskriterien. Die vertikale Anordnung der Filter neben den Suchergebnissen nutzt den vorhandenen Platz auf grossen Bildschirmen besser und wurde als intuitiver und übersichtlicher empfunden.

### 3.1.2 Sichtbarkeit von Projekten

Der ResearchNavigator soll einerseits als internes Tool zur Ablage, Suche und Auswertung von Projekten dienen. Er soll aber auch für die Öffentlichkeit zur Suche von Forschungsprojekten genutzt werden können. Eine klare Trennung zwischen internen und öffentlich zugänglichen Inhalten ist daher besonders wichtig. In Absprache mit dem Auftraggeber wurden drei Sichtbarkeitsstufen definiert:

- **Organisationseinheit-intern:** Das Projekt ist nur für Zugehörige derjenigen Organisationseinheiten sichtbar, die am Projekt beteiligt sind.
- **OST-intern:** Das Projekt ist sichtbar für alle Personen der OST.
- **Öffentlich:** Das Projekt ist uneingeschränkt sichtbar.

### 3.1.3 Prüfprozess

Die von der OST veröffentlichten Projekte müssen hohen Qualitätsansprüchen gerecht werden. Dies betrifft sowohl die fachliche Korrektheit der Daten, als auch bei öffentlich zugänglichen Projekten die Einhaltung spezifischer Kommunikationsrichtlinien. Um diesen Ansprüchen gerecht zu werden, wurde ein Prüfprozess entwickelt, bei dem ein Projekt je nach gewünschter Sichtbarkeit von jeweils keinem, einem oder zwei Prüfern begutachtet werden muss. Dabei wird zwischen zwei Arten von Prüfern unterschieden:

- **Fachlicher Prüfer:** Prüfen das Projekt hinsichtlich fachlicher Kriterien
- **Kommunikations-Prüfer:** Prüfen Projekte, die auch für externe veröffentlicht werden, hinsichtlich der Kommunikationsrichtlinien der Hochschule

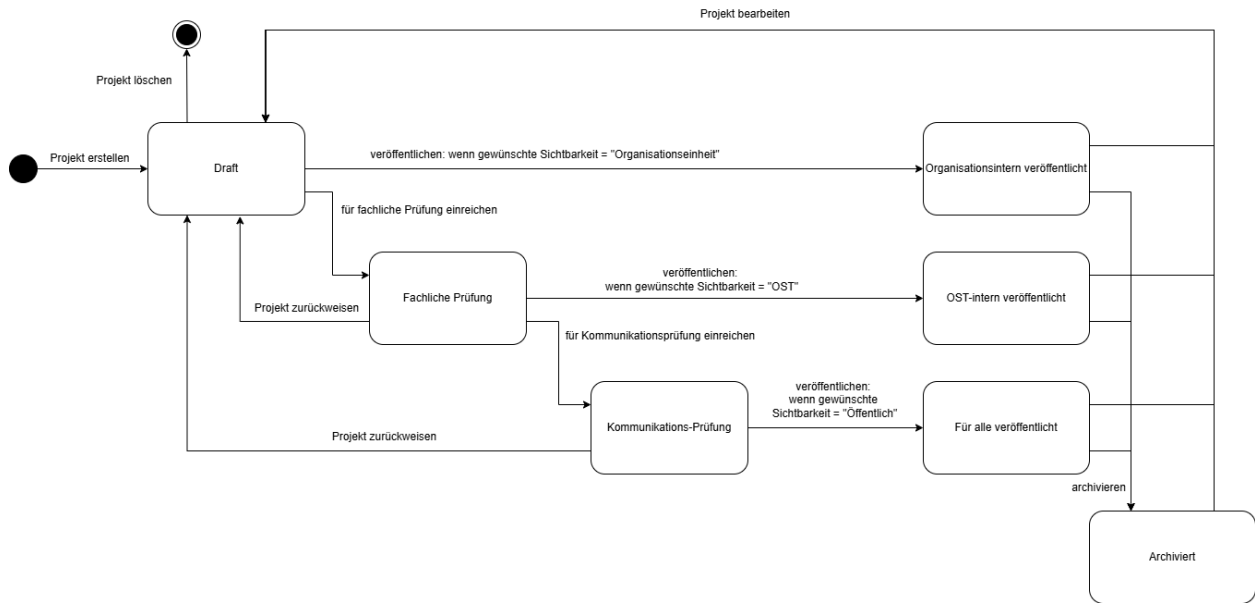


Abbildung 3.1: Ablauf Prüfprozess

Der in Abbildung 3.1 dargestellte Prüfprozess wurde in Zusammenarbeit mit dem Auftraggeber erarbeitet und durch das Entwicklungsteam finalisiert. Jedes Projekt startet im Zustand "Draft". Soll es nur für die eigene Organisationseinheit sichtbar sein, kann es ohne Prüfung veröffentlicht werden. Ab Sichtbarkeitsstufe OST-intern ist eine fachliche Prüfung erforderlich. Für eine Veröffentlichung, die auch für die allgemeine Öffentlichkeit sichtbar ist, wird zusätzlich eine Kommunikations-Prüfung benötigt.

## 3.2 Architektur

Dieser Abschnitt dient der Beschreibung der Architektur des Systems. Dafür wird nach dem Arc42-Template vorgegangen [7]. Zusätzlich werden passende Diagramme aus dem C4-Modell verwendet [8].

### 3.2.1 Kontext & Abgrenzung

Für diese Arbeit sind keine festen Vorgaben bezüglich der zu verwendenden Technologien definiert. Das Entwicklerteam kann sinnvolle und zielführende Entscheidungen selbstständig treffen. Der zentrale Design-Constraint liegt in der zeitlichen Begrenzung dieser Arbeit. Um die geplanten Ziele innerhalb der verfügbaren Zeit zu erreichen, wird daher viel Wert auf die Verwendung von bewährten Best Practices gelgt. Zudem werden Ansätze und Technologien bevorzugt, die dem Entwicklerteam bereits bekannt sind.

Abbildung 3.2 zeigt den Kontext des Systems anhand eines C4-Kontextdiagramms. Dargestellt sind das System als Blackbox sowie die Akteure, die mit dem System interagieren. Diese wurden etwas zusammengefasst, um die notwendige Übersichtlichkeit zu gewährleisten.

Andere Systeme wie beispielsweise die bestehende Projektseite der OST oder das Projekt "Publikationsdatenbank" sind hier nicht dargestellt, da zu diesem Zeitpunkt keine Interaktionen zwischen diesen Systemen geplant sind.

[System Context] OST ResearchNavigator  
Business Kontext Diagramm des OST ResearchNavigator

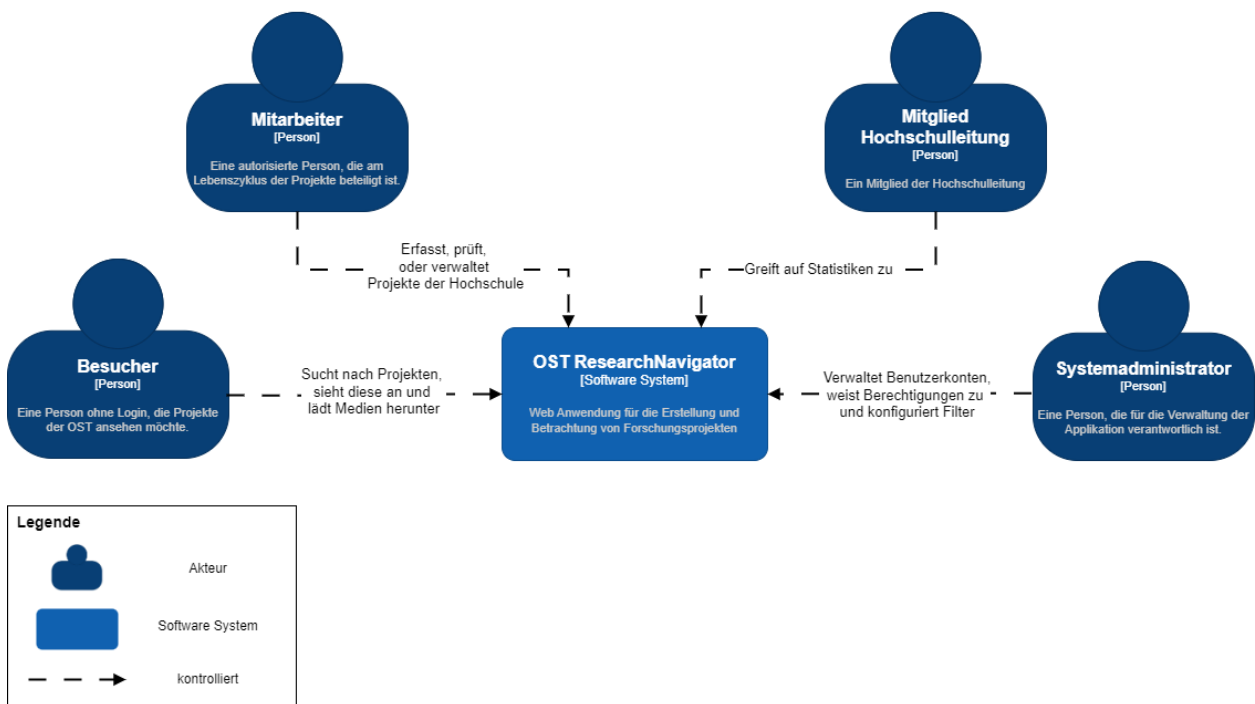


Abbildung 3.2: Kontext Diagramm

### 3.2.2 Lösungsstrategie

Ziel	Lösungsansatz
Benutzerberechtigung	Für den Zugriff auf das Projekterfassungs-Tool muss der Benutzer erst identifiziert werden. In einem Login-Formular kann der Benutzer seine E-Mail-Adresse angeben. Damit wird ein JWT generiert, mit welchem die API verwendet werden kann. Im JWT werden neben Claims für die E-Mail und UserId auch die OrganisationalUnitId und Rolle des Benutzers gespeichert, da diese für die Autorisierung bei bestimmten Requests nötig sind.
Portabilität	Für die Vereinfachung der Installation und Ausführung auf diversen Systemen wird für alle Systemkomponenten ein Dockerfile und zusätzlich ein Docker-Compose-File angelegt. Für die Entwicklung wurde in einem weiteren Docker-Compose-File zusätzlich Hot-Reloading für das Frontend aktiviert.
Trennung Client-Server	Um die Web-Page vollständig vom Backend zu trennen, wird als Schnittstelle eine Level 2 RESTful HTTP API mit dem Web-Framework ASP.NET Core entwickelt.
Erweiterbarkeit	Für die API werden die dem Entwicklerteam bereits bekannte, weit verbreitete Sprache C# sowie eine Adaption des Onion Architecture Pattern angewendet.  Neben den Attributen in der Projekt-Tabelle gibt es eine Tabelle für weitere Metadaten, welche eine beliebig erweiterbare Sammlung an weiteren Projektattributen enthält.
Austauschbarkeit	Für die automatische Generierung der Datenbank beim Startup der Applikation werden EF Core Migrations eingesetzt. Der Zugriff auf die Datenbank erfolgt mit dem Repository Pattern.
Wiederverwendbarkeit	Im Nuxt.js-Frontend werden wiederverwendbare Vue.js Components, Composables und selbst definierte Typen erstellt, die auf mehreren Seiten eingesetzt werden können (z.B. Filterkomponente)
Usability	Bei der Projekterfassung gibt es eine Auswahl an Schlagwörtern, nach welchen per Filter gesucht werden kann. Die beteiligten Personen können in der Benutzerliste gesucht oder als einfacher Text eingetragen werden.
Performance	Zur Reduktion der Serverlast wird beim Laden der Filterkriterien für die Projektsuche Caching eingesetzt.

Tabelle 3.1: Lösungsstrategie

### 3.2.3 Bausteinsicht Level 1: Containers

Die Containerebene, dargestellt in Abbildung 3.3, zeigt die einzelnen Bestandteile des Systems. Sie ist als eine Art Whitebox-Beschreibung des Gesamtsystems aus Abbildung 3.2 zu verstehen, wobei die einzelnen Bestandteile als Blackboxen dargestellt werden.

#### [Container] OST ResearchNavigator

Container Diagramm für die Webanwendung OST ResearchNavigator

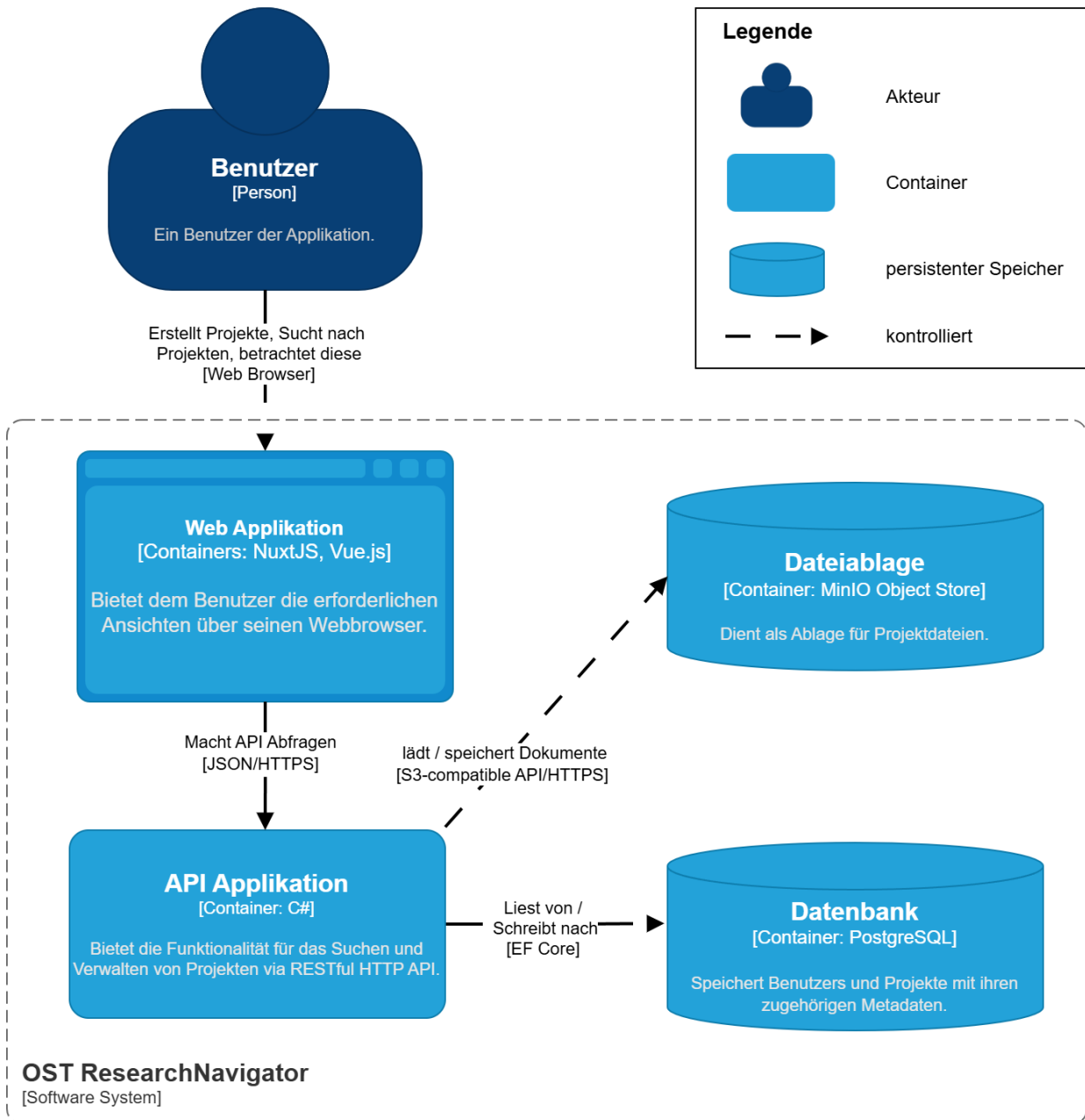


Abbildung 3.3: Container Diagramm

### 3.2.4 Bausteinsicht Level 2: Components

In Abbildung 3.4 wird das Komponentendiagramm für den Container "API Applikation" visualisiert. Für das Diagramm wird dieser Container gewählt, da er für das System von zentraler Bedeutung ist. Das Diagramm enthält nur die wichtigsten Komponenten und schliesst somit Schnittstellen zwischen den Komponenten sowie Exception-Klassen und spezifische Entitäts- oder Modellklassen aus.

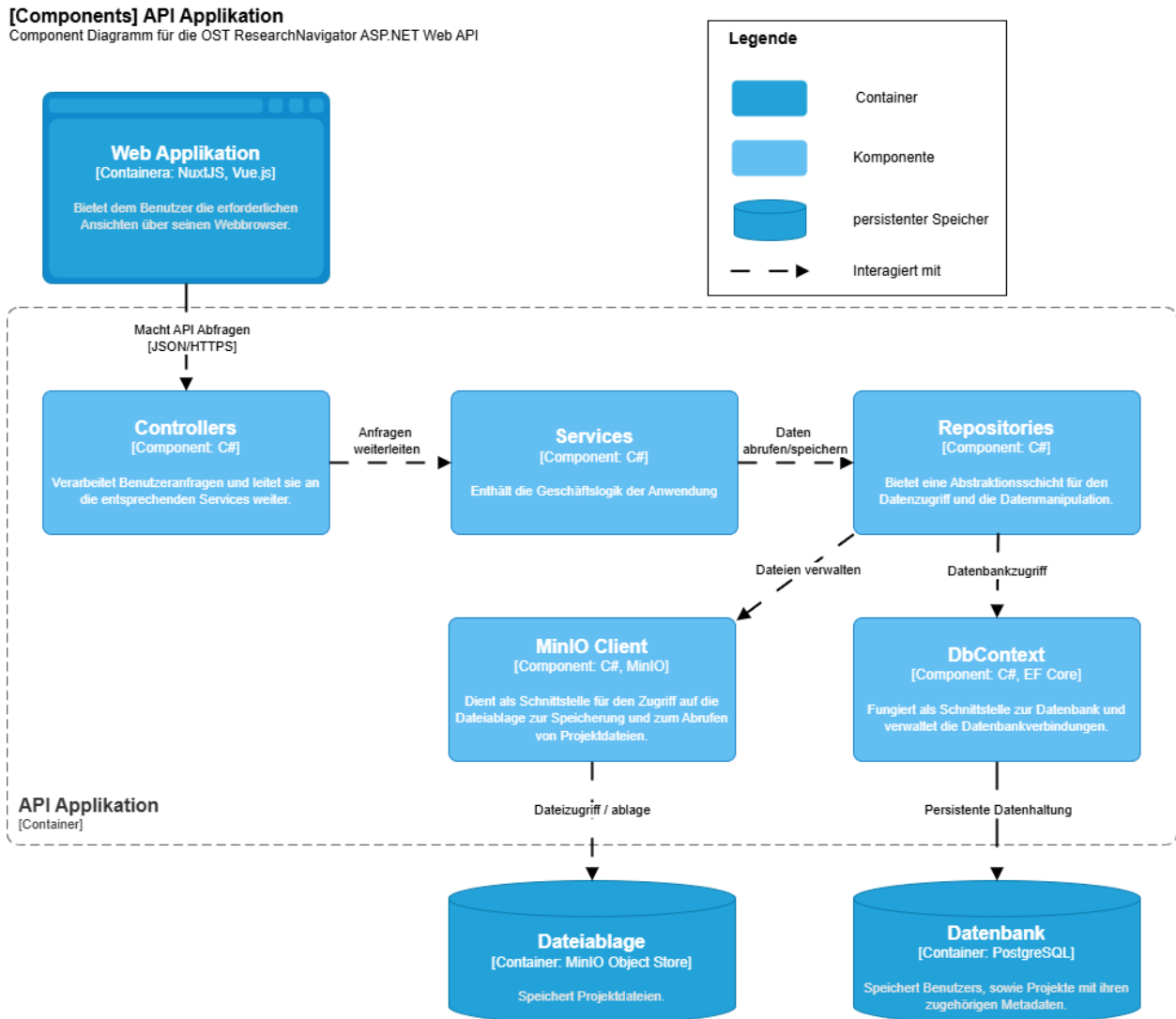


Abbildung 3.4: Components Diagramm

### 3.2.5 Bausteinsicht Level 3: Code

#### Backend - Onion Architektur

Das Backend unserer Anwendung wird mit ASP.NET Core entwickelt und folgt der von Jeffrey Palermo konzipierten Onion-Architektur, wie in der folgenden Abbildung dargestellt [9] [10]. Diese ähnelt stark der bekannten Clean Architecture von Robert C. Martin [11], legt den Fokus aber noch stärker auf das zentrale Domain Model und die Trennung der Verantwortlichkeiten (Separation of Concerns). Ein zentrales Prinzip dabei ist, dass sich Abhängigkeiten nur gegen innere Schichten der Architektur richten, aber nie gegen aussen. Diese Abhängigkeiten werden jeweils durch die Verwendung von Interfaces abstrahiert. Je tiefer



eine Schicht in der Onion-Architektur liegt, desto weniger Abhängigkeiten hat sie. Dieses Design gewährleistet eine hohe Testbarkeit, da die Abhängigkeiten auf Abstraktionen basieren, die sich leicht durch Tools wie Moq mocken lassen. In unserem Ansatz stehen die Präsentations- und Infrastrukturschicht auf derselben hierarchischen Ebene.

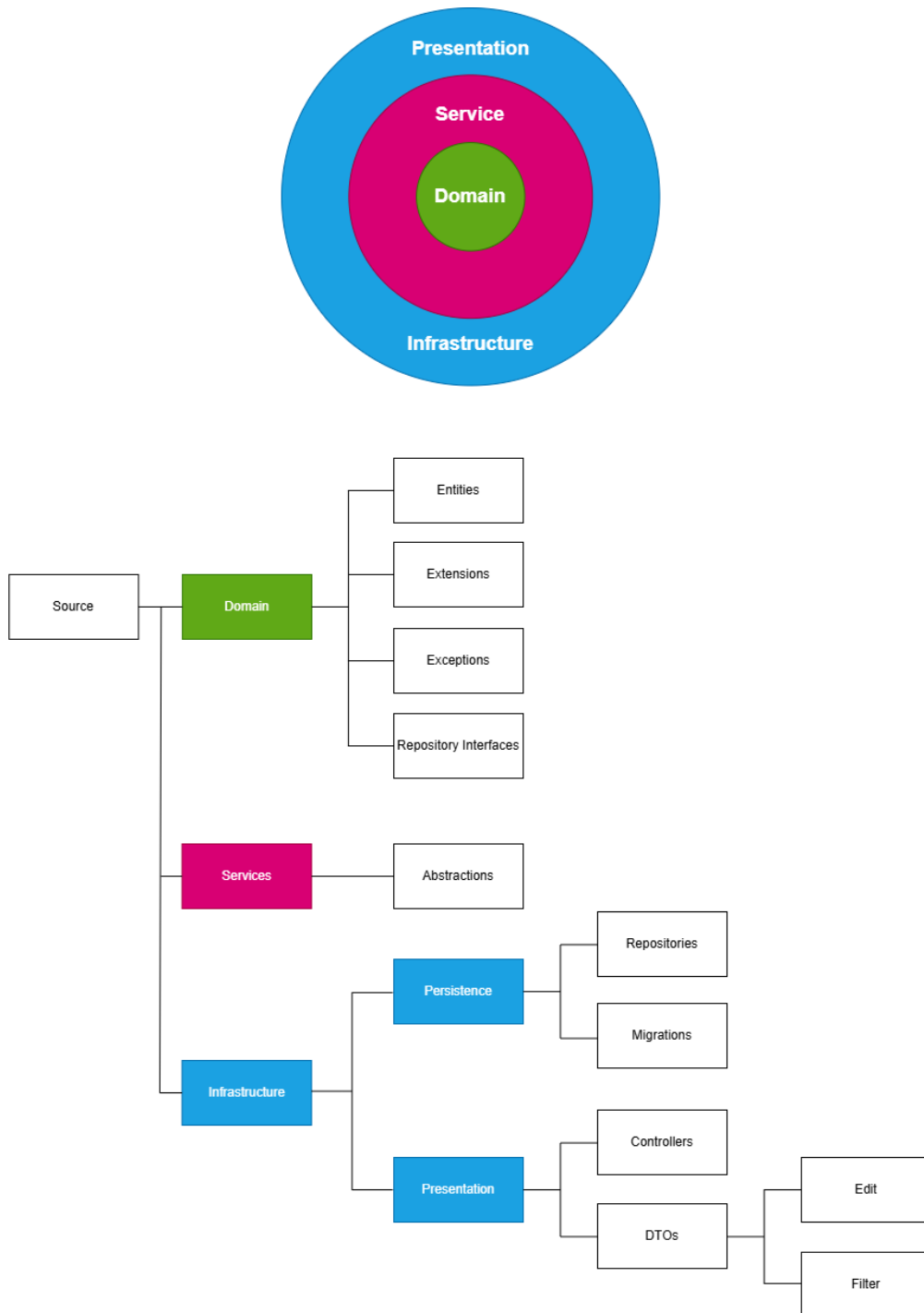


Abbildung 3.5: Onion Architektur

## Frontend - Nuxt.js Ordnerstruktur

Um sich an Best Practices zu halten, ist der Aufbau der Ordnerstruktur des Frontends stark an die von Nuxt empfohlene Ordnerstruktur [12] angelehnt. In folgender Tabelle wird die verwendete Ordnerstruktur konkret beschrieben, sodass sich allfällige neue Entwickler schnell einen Überblick verschaffen können.

Ordner	Verwendungszweck
assets	Dient zur Ablage von Ressourcen wie Stylesheets. Hier wird auch das anwendungsübergreifende Theme definiert, welches eine konsistente Gestaltung der Benutzeroberfläche ermöglicht.
components	<p>Ablage von Vue-Komponenten [13]. Diese Komponenten sollen in unserer Applikation zu folgenden Zwecken erstellt/verwendet werden:</p> <ol style="list-style-type: none"> <li>1. Zur Wiederverwendbarkeit von Code, um Redundanzen zu vermeiden.</li> <li>2. Zur Reduktion der Komplexität einzelner Seiten, indem sie in überschaubare Elemente aufgeteilt wird.</li> </ol> <p>Die Komponenten können, ähnlich wie HTML-Elemente, durch die Verwendung von Tags in einer Seite eingebunden werden.</p>
composables	Im composables-Ordner werden wiederverwendbare Funktionen und Logiken abgelegt, die mittels der Composition API von Vue 3 erstellt wurden. So können Funktionalitäten in der gesamten Anwendung geteilt werden, was die Wartbarkeit des Codes verbessert.
layouts	Der Ordner layouts enthält die allgemeinen Layouts der Anwendung. Layouts ermöglichen es, gemeinsame Strukturen wie (in unserem Fall den Header) über mehrere Seiten hinweg zu verwenden, ohne den Code zu duplizieren. Dadurch ein konsistentes Design erleichtert.
middleware	Im middleware-Ordner befinden sich Middleware-Funktionen, die vor dem Rendering einer Seite ausgeführt werden. Mittels Middleware können beispielsweise Authentifizierungsprüfungen und Umleitungen implementiert werden, was dabei helfen kann, den Zugriff auf bestimmte Bereiche der Anwendung zu regulieren.
pages	Der Ordner pages enthält die Seiten der Anwendung. Nuxt generiert automatisch Routen basierend auf der Verzeichnisstruktur dieses Ordners.
plugins	Im plugins-Ordner werden Plugins integriert. Durch die Registrierung von Plugins können zusätzliche Funktionalitäten global verfügbar gemacht werden. Diese werden von Nuxt automatisch zur Erstellungszeit der Anwendung geladen.
public	Der Ordner public enthält statische Dateien, die direkt vom Server ausgeliefert werden sollen. In unserem Fall sind dies Bilder und Fonts. Dateien in diesem Ordner sind über den Root-Pfad der Anwendung zugänglich, was die Verwaltung von öffentlichen Ressourcen vereinfacht.

Tabelle 3.2: Erklärung der Ordnerstruktur

types	Der Ordner types enthält die TypeScript Typdefinitionen der Anwendung. Durch die zentrale Ablage der Typen kann die Typisierung konsistent gehalten werden. Es ist jedoch nicht zwingend erforderlich, alle Typdefinitionen in diesem Ordner zu speichern. Falls klar ist, dass bestimmte Typen ausschliesslich lokal in einem spezifischen Modul verwendet werden und nicht für andere Teile der Anwendung relevant sind, können diese direkt in der entsprechenden Datei definiert werden. Dies gilt vor allem für weniger komplexe Typen, da die Definition direkt im File die Lesbarkeit und Verständlichkeit des Codes verbessern kann.
utils	Im Ordner utils werden allgemeine Hilfsfunktionen und Utilities abgelegt, die in verschiedenen Teilen der Anwendung verwendet werden. Dies fördert die Wiederverwendbarkeit von Code und hilft, Redundanzen zu vermeiden. Im Gegensatz zu den composables, die Vue-spezifische Logik und Zustände mit der Composition API bereitstellen, enthalten utils reine JavaScript- oder TypeScript-Funktionen, die unabhängig von Vue sind.

Tabelle 3.3: Erklärung der Ordnerstruktur Teil 2

### Datenbank - Datenmodell

Die PostgreSQL-Datenbank wird mit EF Core Migrations automatisch aus den annotierten Entitäten im Backend generiert, wobei sich das untenstehende relationale Datenschema ergibt.

Die Users-Tabelle enthält im Unterschied zum Domain Model kein Passwort mehr, da dies aufgrund der fehlenden Authentifizierung nicht benötigt wird. Ein User wird abgesehen von seiner Id über die eindeutige E-Mail-Adresse identifiziert. Die Autorisierung erfolgt über das Role-Attribut, das folgende Werte einnehmen kann (Enum Role):

- InstituteEmployee (0): Kann Projekte erstellen und die eigenen Projekte bearbeiten. Sieht zusätzlich zu öffentlichen und OST-internen auch die Projekte seines Instituts
- HeadOfInstitute (1): Hat alle Rechte des InstituteEmployee und kann zusätzlich alle Projekte seines Instituts bearbeiten.
- UniversityManager (2): Sieht zusätzlich zu öffentlichen auch OST-interne Projekte.
- Administrator (3): Soll in einer zukünftigen Version die Bearbeitung von Filterdaten (Regionen, Suchtags, Organisationseinheiten, Sponsoren etc.) vornehmen können.

Das Attribut is\_comm\_validator sagt aus, ob ein User für kommunikative Prüfungen ausgewählt werden kann. Für die fachlichen Prüfungen ist jeder User berechtigt.

Für die zukünftige Implementierung von Refresh Tokens wird ein neues Attribut in der User-Tabelle benötigt. Diese Erweiterung ist dank den Migrations aber wenig umständlich.

Die Institutes-Tabelle (sowie die interne Bezeichnung 'Institut' allgemein) wurde im aktuellen Stand noch nicht umbenannt. Die Änderung in 'Organisationseinheiten' fand nur oberflächlich statt, da die exakte Bezeichnung im Hintergrund wenig Einfluss hat. Andere Optimierungen wurden in der begrenzten Zeit nach dem Meeting in Woche 11 (Änderung der Bezeichnung, siehe ??) höher priorisiert.

Der Projekttyp (im Vordergrund: Arbeitsergebnisse) bildet ab, was bei einem Projekt geschaffen wurde. Die möglichen Werte sind im Enum ProjectType definiert. Die Visibility ist die gewünschte Sichtbarkeit, welche ein Projekt am Ende einnehmen soll. Die möglichen Werte InstituteOnly (0), OstOnly (1) sowie Publicly (2)

sind im Enum ProjectVisibility definiert. Die effektive Sichtbarkeit wird daraus und aus dem Status des Projektes bestimmt. Die Status-Werte entsprechen denjenigen, welche im Domain Model definiert wurden und sind im Enum Status festgehalten. Für die Suche und Berechnung der Statistiken wird das Feld DateTo verwendet, also das Abschlussdatum des Projektes. Somit muss für ein Projekt ein Enddatum bestimmt sein, bevor es erfasst werden kann. In diesem Diagramm ist auch die rekursive Beziehung der Projekte zu ihren Vorgängerversionen ersichtlich.

Die Sponsoren sind im aktuellen Stand ebenfalls als Enum (Sponsor) definiert, benötigen in einer zukünftigen Version aber eine eigene Tabelle (siehe [Abschnitt 2.4](#)). Aktuell sind folgende Werte möglich:

- Innosuisse
- Eureka
- Snf
- NoSponsor

Contributors sind Personen, die am Projekt selbst beteiligt waren und/oder dieses im ResearchNavigator erfasst haben. In letzterem Fall wird das Flag is\_editor und der contributor\_type auf 'Publisher' gesetzt. Die weiteren möglichen contributor\_type's sind im Enum ContributorType definiert. War der erfassende User auch anderweitig am Projekt beteiligt, werden für ihn zwei Einträge angelegt. Diese Redundanz ist auf einen früheren Entwicklungsstand zurückzuführen, wurde aber aus Zeitgründen und aufgrund der geringen Relevanz und Fehleranfälligkeit nicht aufgelöst. Ein Contributor hat entweder eine Referenz auf einen User oder einen einfachen Beschreibungstext. Dies ermöglicht die Erfassung von Contributors, welche keinen Account in der Web-Applikation haben.

Die Zwischentabellen ProjectSearchtag und InstituteProject wurden aufgrund der n:m-Beziehung zwischen Projekt und der jeweils anderen Tabelle von EF Core automatisch generiert, weshalb sie unsere Naming Conventions nicht vollständig einhalten. Dies hat auf die Entwicklung aber keinen Einfluss. Die gleichnamigen Klassen sind nicht annotiert und dienen nur zur Generierung von Testdaten.

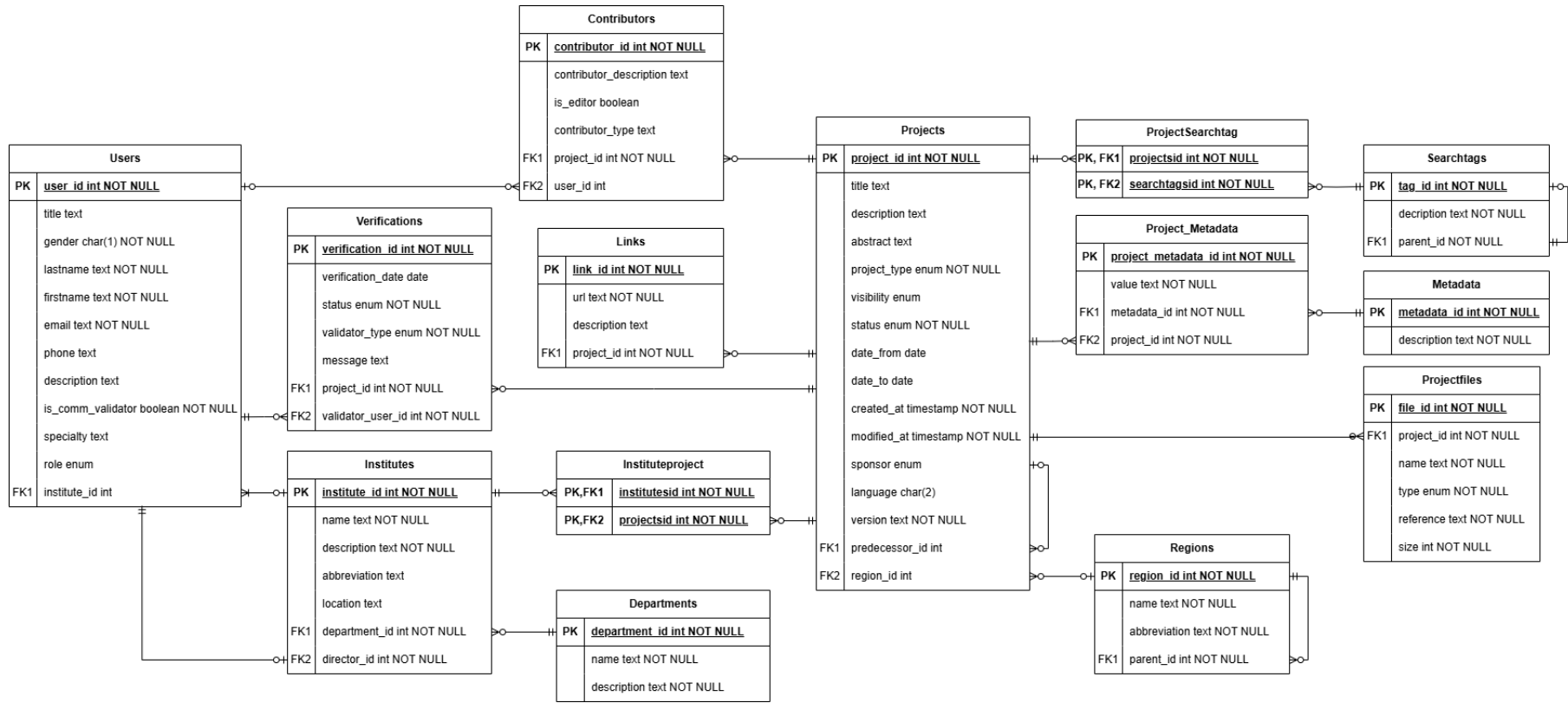


Abbildung 3.6: Entity Relationship Diagram

### 3.2.6 Laufzeitsicht

Das in Abbildung 3.7 dargestellte Sequenzdiagramm zeigt den Ablauf des Zurückweisens eines Projekts durch eine Prüfperson. Der Benutzer kann im Frontend unter Angabe einer Begründung ein Projekt an den Ersteller zurückweisen. Dies wird als **Hypertext Transfer Protocol (HTTP)** POST-Anfrage an das Backend geschickt. Dort wird der Prozess für das Zurückweisen gestartet. Unter anderem wird geprüft, ob der Benutzer, der die Anfrage gesendet hat, im aktuellen Projektstatus überhaupt berechtigt ist, diese Änderung durchzuführen. Die aktuelle Prüfung wird in den Status abgelehnt versetzt, dabei wird auch die Begründung der Zurückweisung gespeichert. Kommt es in diesem Ablauf zu Fehlern, werden diese vom Errorhandler (nicht im Diagramm dargestellt) abgefangen und mit Angabe einer Fehlermeldung dem Frontend übermittelt.

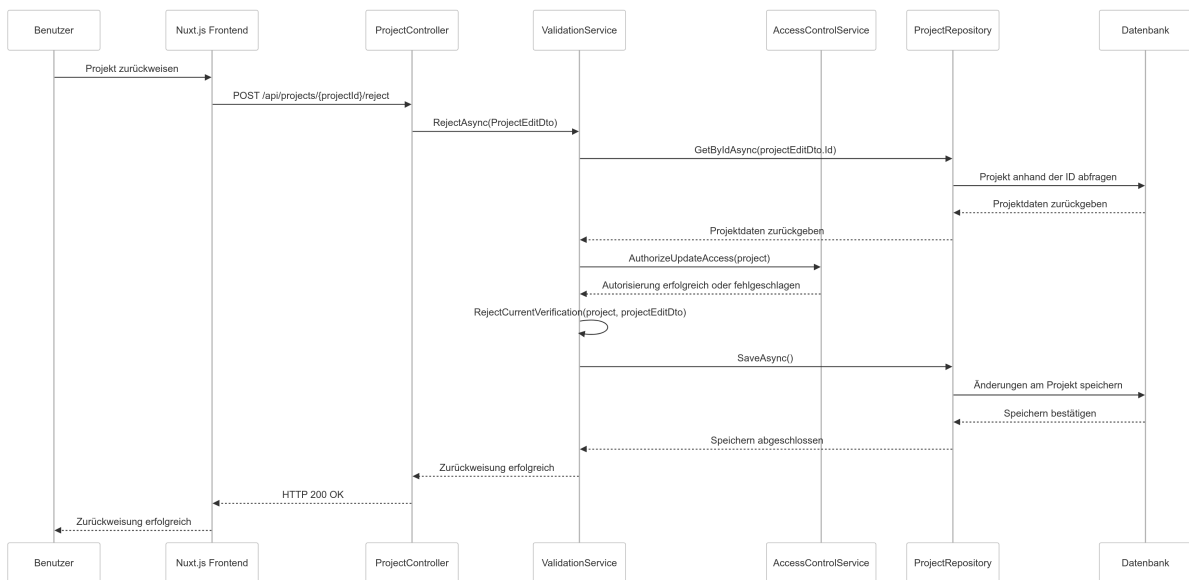


Abbildung 3.7: Sequenzdiagramm

In Abbildung 3.8 wird der bereits in Abschnitt 3.1.3 beschriebene Prüfprozess nochmals detaillierter abgebildet. Zu jedem möglichen Projektzustand werden die jeweils möglichen Zustände der gewünschten Sichtbarkeit aufgezeigt.

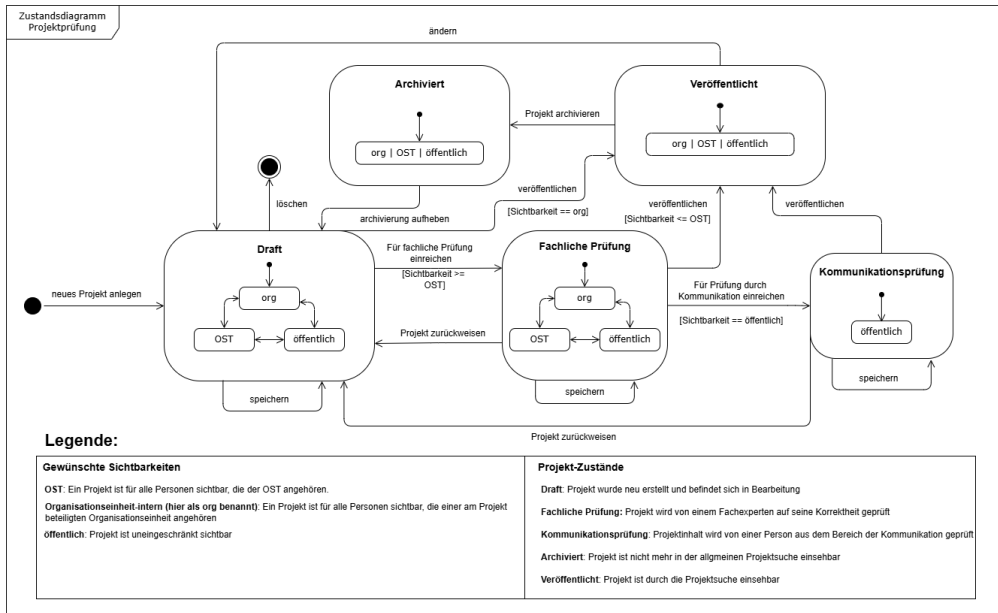


Abbildung 3.8: Zustandsdiagramm der Projektveröffentlichung

### 3.2.7 Deployment-Sicht

In Abbildung 3.9 ist eine Übersicht des Deployments der Anwendung dargestellt. Das Deployment erfolgt mittels Docker-Containern. Diese werden auf einem virtuellen Server im internen Netzwerk der OST gehostet.

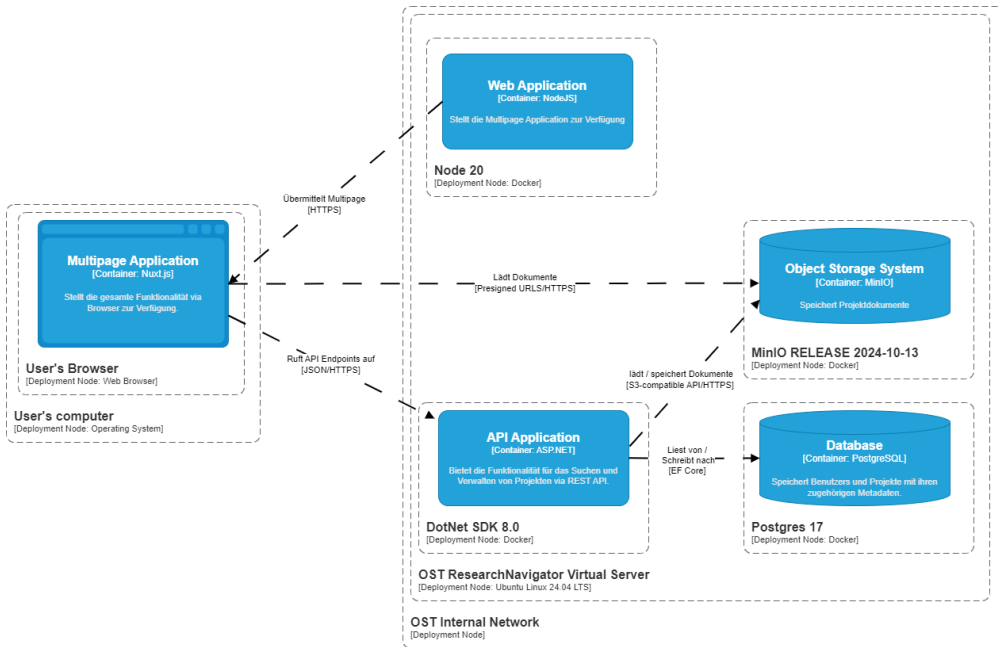


Abbildung 3.9: C4 Deployment Diagramm

Konkrete Informationen zum Live-System:

- Link zur Webanwendung: [OST ResearchNavigator](#)
- Nur innerhalb von OST Netzwerk erreichbar
- OS: Linux Ubuntu 6.8.0-47-generic
- Hostname: srbsci-127.ost.ch
- IP-Adresse: 10.8.36.127
- Port für Webanwendung: 3000
- Port für API: 5050
- Port für Minio GUI: 9001
- VM läuft bis: 31.08.2025

Hardware:

- 2 vCPU (empfohlen)
- 4 GB RAM (empfohlen)
- 50 GB Disk Drive (muss je nach Anzahl Projekten, Dateien pro Projekt sowie durchschnittlicher/maximaler Dateigrösse skaliert werden)
- Eine Verfügbarkeit von >95% insgesamt und >99% während den Arbeitszeiten der OST sollte gewährleistet sein.

### 3.2.8 Querschnittliche Konzepte

#### Logging

In unserer Anwendung ist Logging ein wichtiger Bestandteil. Aus Sicherheitsgründen werden alle erfolgreichen und fehlerhaften Anfragen an die Controller gelogged. Zusätzlich werden alle auftretenden Exceptions im Code gelogged, um eine effiziente Fehleranalyse zu ermöglichen. Um ein einheitliches Logging zu ermöglichen, werden die von .NET zur Verfügung gestellten Logging-Provider verwendet [14].

#### Exception- und Error handling

In Backend-Projekt unserer Anwendung sollen alle Exceptions abgefangen und gelogged werden. Anschliessend soll der Benutzer eine geeignete Fehlermeldung erhalten. Dazu wird ein globaler Exception-Handler verwendet, der auf dem .NET `IExceptionHandler` Interface basiert [15]. Dieser kümmert sich um alle nicht unbehandelten Fehler. Dort, wo eine detailliertere Fehlerbehandlung nötig ist, werden spezifische Fehler innerhalb der Applikation abgefangen und separat behandelt. Die jeweilige Fehlermeldung, ohne jegliche sensible Daten, wird anschliessend dem Benutzer im Frontend angezeigt.

#### Internationalization I18n

Damit die Anwendung zukunftssicher und erweiterbar ist, wird im Frontend das Vue I18n Internationalisierungs-Plugin verwendet [16]. Die Anwendung wird zwar anfänglich nur auf deutsch gehalten, kann aber so zu einem späteren Zeitpunkt problemlos auf mehrere Sprachen erweitert werden.



### 3.2.9 Architekturentscheidungen

#### Datenbank Volltextsuche

**Datum:** 01.10.2024

Für die Textsuche in Titel, Abstract und den Autoren sollen auch mehrere Suchbegriffe möglich sein, welche nicht genau in der Reihenfolge im Text vorkommen. Die folgenden Optionen wurden analysiert:

1. PostgreSQL mit Plugin pg\_tgrm (Trigram), welches Fuzzy Matching und Partial String Search unterstützt.
2. PostgreSQL mit tsvector / tsquery für Volltextsuche
3. Secondary Store für die Texte in ElasticSearch

Die Entscheidung fiel auf tsvector / tsquery, da die Dokumentation in Verbindung mit EF Core sehr viel besser ist als bei pg\_tgrm und da für ElasticSearch eine weitere Datenbank nötig wäre. Zu Beginn wird aber nur eine normale Textsuche implementiert, welche später einfach mit der gewählten Lösung erweitert werden kann, indem ein neues tsvector-Attribut zur Projekttable hinzugefügt wird. Dies ist mit EF Core über ein NpgsqlTsVector Property möglich.

Die fehlende Erfahrung des Entwicklerteams im Bereich der PostgreSQL-Volltextsuche muss mit einem höheren Lernaufwand ausgeglichen werden. Dies gilt aber für alle Lösungen.

**Status:** Angenommen

#### Frontend Framework

**Datum:** 01.10.2024

Für die Entwicklung des Frontends muss ein passendes Framework ausgewählt werden, welches nicht zu komplex ist und einen verhältnismässigen Lernaufwand bedeutet, aber auch die Anforderungen erfüllt. Dies betrifft vor allem die Bereiche Performance und SEO.

Als Frontend-Framework wurde Nuxt.js gewählt, da sich das Entwicklerteam bereits mit dem Framework Vue.js auskennt, auf dem Nuxt.js basiert. Nuxt bietet die Möglichkeit, die Renderingstrategie speziell anzupassen, wodurch Nuxt besser für Performance-Optimierungen und SEO geeignet ist.

Es wird ein geringer Lernaufwand für die Umstellung von Vue.js auf Nuxt.js erwartet, ebenso für die SEO.

**Status:** Angenommen

#### Speicherung von Dokumenten

**Datum:** 25.10.2024

Pro Projekt sollen beliebige Dokumente hoch- und heruntergeladen werden können. Diese müssen entweder in der Datenbank in einem bytea-Attribut oder auf einem Filesystem gespeichert werden.

Aufgrund der schlechten Handhabung und Performance von Dokumenten in Datenbankattributen fiel die Entscheidung auf ein eigenes Filesystem. Dafür wurde das MinIO Object Storage System verwendet. Dieses bietet ein .NET Package an, was die Integration in unsere ASP.NET API stark erleichtert.

Als Konsequenz steigt die Komplexität aufgrund einer weiteren Systemkomponente. Dies steht jedoch in keinem Verhältnis zu einer Umstellung auf ein Filesystem zu einem späteren Zeitpunkt. Durch die simple MinIO .NET API wird ausserdem die Entwicklung stark erleichtert.

**Status:** Angenommen

### Rendering-Strategie

**Datum:** 12.10.2024

Suchmaschinenoptimierung spielt für die OST eine wichtige Rolle, insbesondere für die Projektdetailseite. Gleichzeitig müssen Performance-Anforderungen und Betriebskosten optimiert werden. Nuxt bietet 4 mögliche Rendering-Strategien an (Universal Rendering, Client-Side Rendering, Hybrid Rendering und Edge-Side Rendering), die näher betrachtet wurden.

Es wurde entschieden, die Hybrid-Rendering-Strategie zu verwenden. Diese bietet die grösste Flexibilität und erlaubt es, die Renderingstrategie für jede Route individuell zu wählen. So kann die Projektdetailseite gezielt serverseitig gerendert werden, während andere Routen clientseitig gerendert werden. Dadurch können Serverkosten gespart werden, da serverseitiges Rendering nur dort eingesetzt wird, wo es notwendig ist.

Diese Entscheidung führt zu etwas mehr Komplexität und könnte auch einen etwas höheren Entwicklungsaufwand bedeuten. Im Gegenzug bietet sie jedoch umfangreiche Möglichkeiten für aktuelle und zukünftige Performance-Optimierungen. Durch den hybriden Rendering-Ansatz sind vielfältige Rendering-Optionen wie **Prerendering**, **Server Side Rendering (SSR)** oder Caching-Konfigurationen möglich.

**Status:** Angenommen

### Authentifizierung

**Datum:** 16.10.2024

Die Mitarbeiter in den Instituten sollen sich anmelden müssen, um Projekte zu erfassen. Die Authentifizierung ist allerdings sehr komplex und sollte im besten Fall mit einer bereits bestehenden Lösung vorgenommen werden, beispielsweise Switch edu-ID oder Microsoft.

Aufgrund der fehlenden Erfahrung mit den vorhandenen Authentifizierungsmöglichkeiten und dem grossen Mehraufwand einer eigenen Implementation wird in diesem Projekt vorerst auf die Authentifizierung verzichtet. Benutzer können für die Projekterfassung eine einfache Identifizierung per E-Mail vornehmen. Die Überprüfung der Identität soll zu einem späteren Zeitpunkt implementiert werden.

Das Weglassen der Authentifizierung spart enorm viel Aufwand und Unsicherheit aufgrund unbekannter Technologien ein. Es führt aber zu neuen technischen Schulden, die später beglichen werden müssen. Die Struktur soll somit bereits jetzt so aufgebaut werden, dass beim Einbau der Authentifizierung keine grossen Änderungen am vorhandenen Code nötig sind.

**Status:** Angenommen

### Frontend-State

**Datum:** 22.10.2024 Um wiederholte Requests für dieselben konstanten Daten zu vermeiden, sollen die möglichen Werte für die Filterkriterien (Institute, Suchtags etc.) im Frontend zwischengespeichert werden.

Ausserdem sollen die vom Benutzer ausgewählten Werte zwischen den Suchanfragen erhalten bleiben, auch bei zwischenzeitlicher Navigation auf andere Seiten.

In Nuxt lässt sich für das globale State-Handling ein Store mithilfe des Pinia Modules definieren. Eine Alternative dazu bietet der Shared Runtime Context über das Composable useNuxtApp. Damit können bereits geladene Daten aus dem Cache über einen Key wiederhergestellt werden.

Die Entscheidung fiel auf das Caching mit useNuxtApp, da damit keine weitere Komponente (der Pinia Store) hinzugefügt werden musste. Die ausgewählten Filterwerte werden in der URL-Query zwischengespeichert.

Das Weglassen des Pinia Stores sorgt für eine etwas geringere Komplexität. Falls zu einem späteren Zeitpunkt mehr State benötigt wird, würde dies zu einem Mehraufwand führen. Dies ist aber zum Zeitpunkt der Entscheidung nicht vorgesehen.

**Status:** Angenommen

### **Automapper**

**Datum:** 17.10.2024

Die Objekte, welche mittels EF Core geladen werden, sollen für die Übermittlung an das Frontend auf ein der API Request entsprechendes **Data Transfer Object (DTO)** projiziert werden. Das DTO enthält nur die benötigten Informationen und zum Teil auch berechnete Werte oder Listen von anderen DTOs.

Dies kann mit selbst geschriebenen Mapping Funktionen gemacht werden. Gut geeignet hierfür sind Extension Methods in C#. Es kann auch ein Automapper eingesetzt werden, welcher automatisch alle gleichnamigen bzw. die konfigurierten Attribute eines Objektes auf ein neues DTO (und umgekehrt) mapped.

Das Projektteam entschied sich gegen einen Automapper, da dafür viel Konfiguration sowie ein zusätzlicher Lernaufwand nötig wäre und das manuelle Schreiben der Extension Methods keinen besonders grossen oder schwierigen Arbeitsschritt darstellt.

Die Konsequenz für diese Entscheidung ist die Implementierung der Extension Methods in einer neuen Klasse MappingExtensions. Dafür muss jedoch kein weiteres Package installiert werden.

**Status:** Angenommen

### 3.3 Technische Schulden

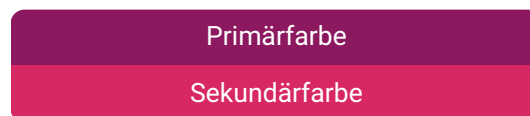
Titel	Beschreibung
Redundante Enums	Alle Enums sind sowohl im Frontend als auch im Backend mit den (nahezu) gleichen Werten definiert. In einer zukünftigen Version sollten die Werte dynamisch aus dem Backend geladen werden, um die Datenkonsistenz zu erhöhen.
Authentifizierung	Wie bereits beschrieben fehlt die Identitätsprüfung der Benutzer noch komplett. Es kann weder ein Passwort noch ein anderer Authentication Factor festgelegt werden.
Refresh-Token	Es gibt bis her noch keinen Refresh-Token, was eine längere Gültigkeit der Access Tokens erzwingt und somit die Sicherheit reduziert. Der Code ist bereits auf die Einführung eines Refresh Tokens ausgelegt, in der Datenbank fehlt noch das User-Attribut dafür.
Data Seeding	Aktuell findet beim Einfügen der Testdaten in die Datenbank keine Unterscheidung zwischen Entwicklungs- und produktiver Umgebung statt. Dies ist bewusst so, da die Produktionsumgebung noch mit den Testdaten getestet wird, solange sie nicht in Betrieb ist.
Textsuche	Projekte sollen auch angezeigt werden, wenn Suchbegriffe im Text nicht direkt aufeinanderfolgen oder in unterschiedlichen Feldern (Autor, Abstract, Titel) vorkommen. Dafür soll wie in <a href="#">Abschnitt 3.2.9</a> beschrieben ein Attribut vom Typ tsvector auf dem Projekt hinzugefügt werden.
Projektsuch-Query	Die Projektsuche wird auf der Datenbank durchgeführt. Dafür wird im ProjectRepository eine SearchQuery mit allen Filtern aus dem DTO erstellt. Die Methode CreateProjectsSearchQuery, welche dazu verwendet wird, ist sehr lang und sollte aufgeteilt werden. Ausserdem ist im gleichen Repository noch etwas Authorization-Logik enthalten.
Dashboard Page	Die Konfigurationen der Diagramme befinden sich alle im gleichen File (dashboard.vue). Um Diagramme zu einem späteren Zeitpunkt dynamisch (z.B. unterschiedlich nach Rolle) laden zu können, sollten diese ausgelagert werden.
Anpassung Datenmodell	Im Datenmodell fehlen neben den hier bereits beschriebenen Änderungen (tsvector/refresh-token) auch die letzten Änderungen an den Anforderungen, also die Umbenennung und Umstrukturierung von Instituten zu Organisationseinheiten und die Auslagerung der Sponsoren in eine eigene Tabelle (siehe <a href="#">Abschnitt 3.2.5</a> ).

Tabelle 3.4: Technische Schulden

## 3.4 Design

### 3.4.1 Anforderungen und Allgemeines Konzept

Das Design der Anwendung ist von zentraler Bedeutung, da sie öffentlich zugänglich sein wird und den Standard einer modernen technischen Hochschule widerspiegelt. Die Anwendung soll einen professionellen und positiven Eindruck vermitteln. Der Schwerpunkt liegt auf einem klaren und übersichtlichen Design, um die Benutzerfreundlichkeit zu maximieren. Farblich orientiert sich das Design an den corporate-Farben der OST, um eine einheitliche visuelle Identität zu gewährleisten. Dies entspricht einem Design unter Verwendung der folgenden Primär- und Sekundärfarben.



Zusätzlich wird ein responsives Design umgesetzt, um die Anwendung auf unterschiedlichen Geräten und Bildschirmgrößen optimal nutzbar zu machen. Wichtig ist es, eine intuitive und benutzerorientierte Bedienung zu gewährleisten. Die Gestaltung folgt aktuellen Designtrends, um ein modernes und zeitgemäßes Erscheinungsbild zu schaffen.

### 3.4.2 Accessibility

Barrierefreiheit, auch als Accessibility bezeichnet, ist ein zentraler Aspekt moderner Softwareentwicklung. Ziel ist es, digitale Anwendungen und Inhalte so zu gestalten, dass sie von allen Menschen unabhängig von ihren individuellen Fähigkeiten oder Einschränkungen genutzt werden können. Die wichtigsten Leitlinien für barrierefreie Webanwendungen werden durch die Web Content Accessibility Guidelines (WCAG) des World Wide Web Consortium (W3C) definiert. Die Leitlinien basieren auf vier Prinzipien [17]. Diese haben wir in nachfolgender Tabelle festgehalten, sowie auch Massnahmen für die Entwicklung dokumentiert, um diese so weit wie möglich zu erfüllen:

Prinzip	Massnahme
<p><b>Wahrnehmbarkeit:</b> Informationen und Bestandteile der Benutzerschnittstelle müssen den Benutzern so präsentiert werden, dass diese sie wahrnehmen können.</p>	<ul style="list-style-type: none"> <li>• Für Bilder werden Alternativtexte angeboten</li> <li>• Schaltflächen beinhalten klare Beschreibungen, Informationen werden nie nur über Farben oder Icons dargestellt</li> <li>• Wichtige Schaltflächen haben einen ausreichend hohen Farbkontrast</li> </ul>
<p><b>Bedienbarkeit:</b> Bestandteile der Benutzerschnittstelle und Navigation müssen bedienbar sein.</p>	<ul style="list-style-type: none"> <li>• Webseite ist auch rein über die Tastatur bedienbar</li> <li>• Fokus-Zustand interaktiver Elementen ist sichtbar</li> </ul>
<p><b>Verständlichkeit:</b> Informationen und die Bedienung der Benutzerschnittstelle müssen verständlich sein.</p>	<ul style="list-style-type: none"> <li>• Nutzen des i18n-Internationalisierungspakets, um Inhalte einfach auf weitere Sprachen erweitern zu können</li> <li>• Verständliche und konkrete Eingabefehlermeldungen in Projekterfassungsformular</li> </ul>
<p><b>Robustheit:</b> Inhalte müssen robust genug sein, damit sie von einer grossen Auswahl an Benutzeragenten interpretiert werden können.</p>	<ul style="list-style-type: none"> <li>• Responsives Design (Mobile Fähigkeit)</li> <li>• Kompatibilität mit unterschiedlichen Browsern (siehe NFR-3, Browser-Kompatibilität 2.21)</li> </ul>

Tabelle 3.5: Accessibility Massnahmen

### 3.4.3 Mockups

Um das Layout und die Gestaltung der Benutzeroberfläche zu planen, werden zunächst einfache Skizzen angefertigt. Diese werden anschliessend in einem weiteren Schritt in Form von Mockups ausgearbeitet. Nachfolgend sind Screenshots der Mockups dargestellt. Sie bieten eine High-Level-Übersicht und dienen dazu, das geplante Design dem Kunden zu präsentieren und zur Diskussion zu stellen. Die Mockups sind hinsichtlich der dargestellten Elemente bewusst nicht vollständig und ihr Design ist nicht als final zu betrachten. Die finale Applikation soll sich jedoch hinsichtlich der Gestaltung an diesen Mockups orientieren, kann jedoch aufgrund von Verbesserungen, neuen Anforderungen oder Kundenfeedback davon abweichen.

Auf der Startseite (siehe Abbildung 3.10) können Projekte gesucht und gefiltert werden. Ziel ist es, dass alle Filtermöglichkeiten auf einen Blick sichtbar sind. Zudem sollen die wichtigsten Projektinformationen der gefundenen Projekte sofort ersichtlich sein.

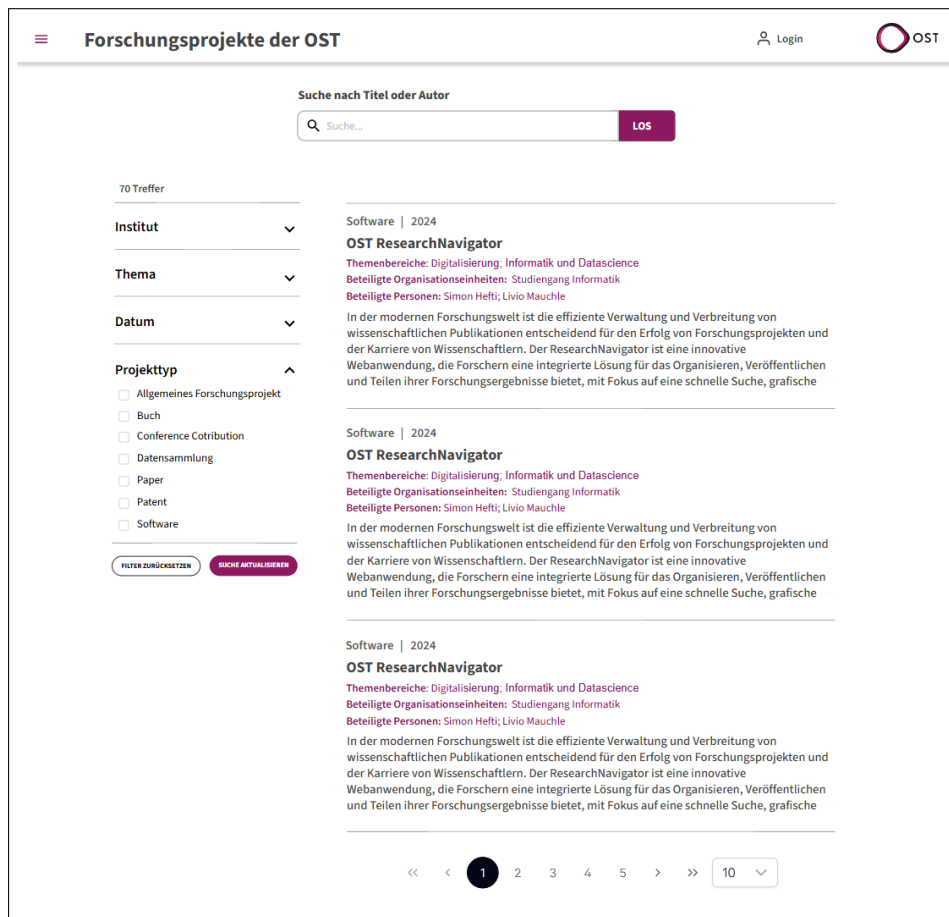


Abbildung 3.10: Mockup der Startseite

Diese Projektdetailseite, welche in Abbildung 3.11 zu sehen ist, bildet alle verfügbaren Informationen über ein Projekt ab. Ziel ist es, alle Informationen möglichst übersichtlich zu veranschaulichen.

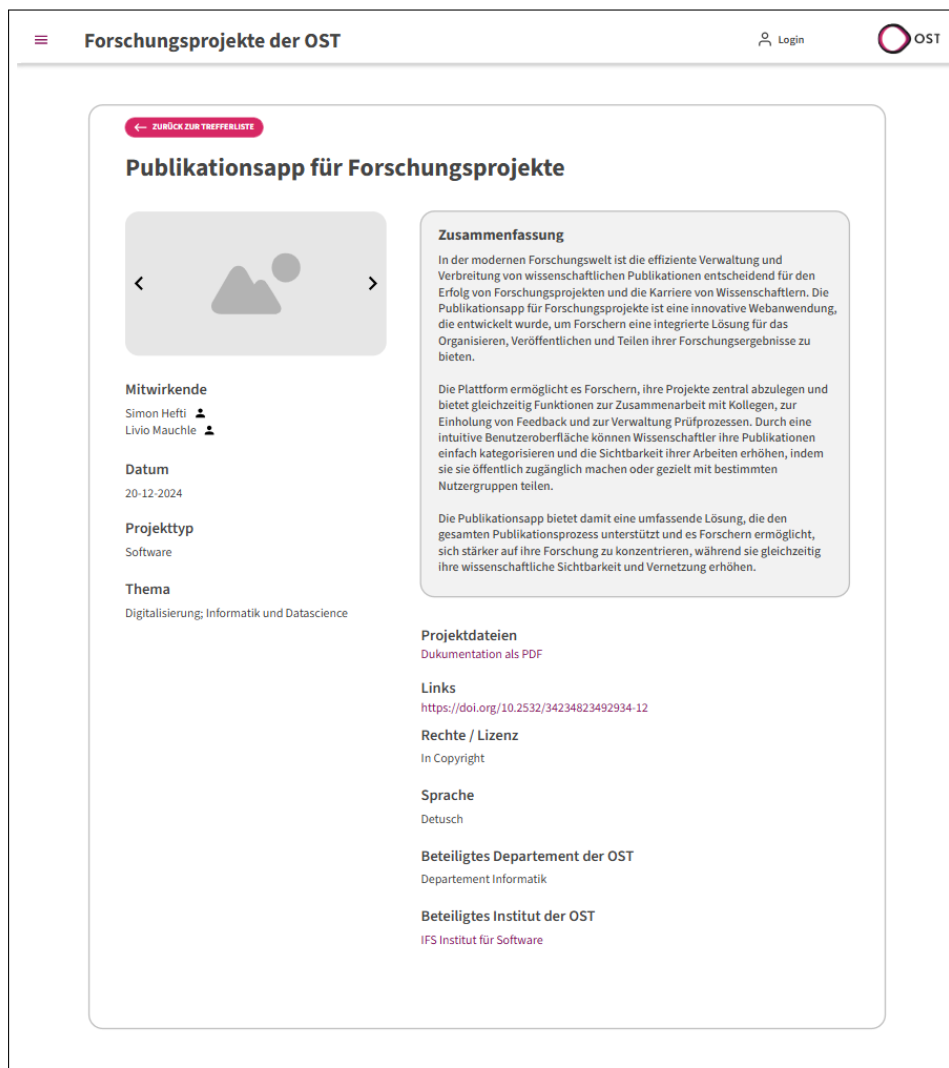


Abbildung 3.11: Mockup der Projektdetailseite

In der finalen Version liegt dieses Design jedoch in einer leicht abgeänderten Version vor. Um die gesamte Breite für allfällige Bilder und Videos zu nutzen, wurde der Inhalt der Seite in 3 untereinander angeordnete Bereiche unterteilt. Diese können jeweils die gesamte Breite nutzen. Zuerst sind die wichtigsten Projektinformationen, gefolgt von einer Zusammenfassung inklusive Bildern und Videos. Zuunterst befinden sich die restlichen Metadaten. Auf dieser Seite kann mittels einer seitlich angebrachten Navigationsleiste navigiert werden.



Auf der Seite "Meine Projekte" (siehe Abbildung 3.12) können berechtigte Personen diejenigen Projekte einsehen, die von ihnen erfasst wurden. Das Design ist dem der Startseite sehr ähnlich. Hier werden aber zusätzlich noch wichtige Statusinformationen angezeigt und farbig hervorgehoben, damit diese sofort ersichtlich sind.

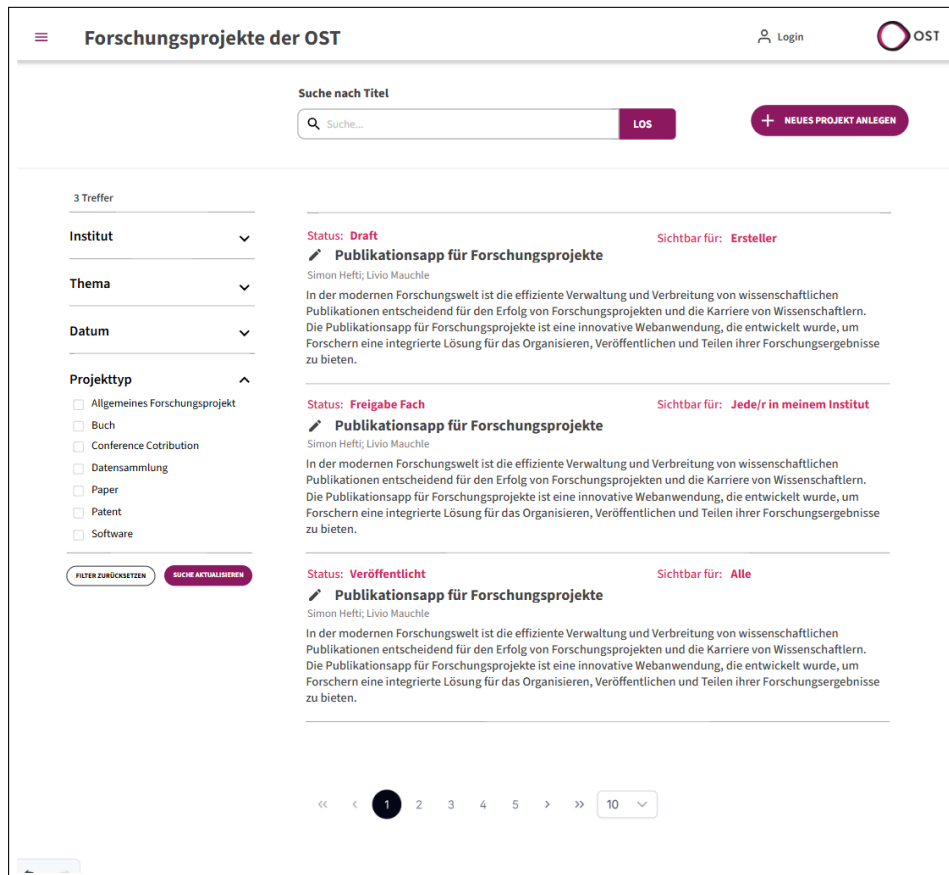


Abbildung 3.12: Mockup der Seite Meine Projekte

In der Benutzeroberfläche der Projekterstellung können Projektinformationen erfasst werden (siehe Abbildung 3.13). Zur Unterstützung der Benutzerfreundlichkeit sind zentrale Interaktionselemente, wie beispielsweise die Schaltflächen "Speichern" oder "Löschen", in einer stets sichtbaren Aktionsleiste oberhalb der Eingabeformulare positioniert. Diese Anordnung gewährleistet eine schnelle und intuitive Bedienbarkeit.

Abhängig vom Veröffentlichungsstatus des Projekts werden nur die jeweils relevanten Schaltflächen angezeigt, wodurch die Auswahlmöglichkeiten auf das notwendige Minimum reduziert werden.

Die einzugebenden Projektinformationen sind in logisch strukturierte Gruppen unterteilt, um den Erfassungsprozess zu vereinfachen und eine klare Übersichtlichkeit zu gewährleisten.

In der finalen Version wurden noch Anpassungen an der Gestaltung des Eingabeformulars vorgenommen. Diese sind im Abschnitt 3.6 dokumentiert.

**Forschungsprojekte der OST** Login

PROJEKT ARCHIVIEREN
VERÖFFENTLICHEN
LÖSCHEN
FÜR FACHLICHE PRÜFUNG EINREICHEN
SPEICHERN

---

← zurück Status: Draft

## Publikationsapp für Forschungsprojekte

### Titel & Beschreibung

**Titel** **Gewünschte Sichtbarkeit *i***

Publikationsapp für Forschungsprojekte Intern  89/300

**Kurzbeschreibung *i***

In der modernen Forschungswelt ist die effiziente Verwaltung und Verbreitung von wissenschaftlichen Publikationen entscheidend für den Erfolg von Forschungsprojekten und die Karriere von Wissenschaftlern. Die Publikationsapp für Forschungsprojekte ist eine innovative Webanwendung, die entwickelt wurde, um Forschern eine integrierte Lösung für das Organisieren, Veröffentlichen und Teilen ihrer Forschungsergebnisse zu bieten. 324/500

**Zusammenfassung *i***

In der modernen Forschungswelt ist die effiziente Verwaltung und Verbreitung von wissenschaftlichen Publikationen entscheidend für den Erfolg von Forschungsprojekten und die Karriere von Wissenschaftlern. Die Publikationsapp für Forschungsprojekte ist eine innovative Webanwendung, die entwickelt wurde, um Forschern eine integrierte Lösung für das Organisieren, Veröffentlichen und Teilen ihrer Forschungsergebnisse zu bieten.

Die Plattform ermöglicht es Forschern, ihre Projekte zentral abzulegen und bietet gleichzeitig Funktionen zur Zusammenarbeit mit Kollegen, zur Einholung von Feedback und zur Verwaltung Prüfprozessen. Durch eine intuitive Benutzeroberfläche können Wissenschaftler ihre Publikationen einfach kategorisieren und die Sichtbarkeit ihrer Arbeiten erhöhen, indem sie sie öffentlich zugänglich machen oder gezielt mit bestimmten Nutzergruppen teilen.

Die Publikationsapp bietet damit eine umfassende Lösung, die den gesamten Publikationsprozess unterstützt und es Forschern ermöglicht, sich stärker auf ihre Forschung zu konzentrieren, während sie gleichzeitig ihre wissenschaftliche Sichtbarkeit und Vernetzung erhöhen. 324/2500

### Medien & Dokumente

### Metadaten

**Projekttyp *i*** **Mitwirkende**

Software 

+ NEU ERFASSEN

**Thema** 
 Simon Hefti; simon.hefti@ost.ch 
  
 Livio Mauchle; livio.mauchle@ost.ch

**Beteiligtes Departement der OST**

Departement Informatik

**Beteiligtes Institut der OST**

Institut für Software

Abbildung 3.13: Mockup der Projekterstellungs-Seite

Das Dashboard (siehe Abbildung 3.14) bietet eine grafische Übersicht über die von der OST durchgeführten Forschungsprojekte. Ziel ist es, die dargestellten Grafiken so zu gestalten, dass sie intuitiv lesbar sind und Benutzer diese entsprechend ihrer individuellen Interessen und Bedürfnisse anpassen können.

Zur Umsetzung der Filterfunktionalität wurden zwei Ansätze in Betracht gezogen: ein individueller Filter pro Grafik und ein globaler Filter für alle Grafiken. Beide Ansätze wurden im Mockup visualisiert, um diese dem Kunden verständlich zu präsentieren. Eine detaillierte Analyse dieser Alternativen findet sich in Abschnitt 3.6.

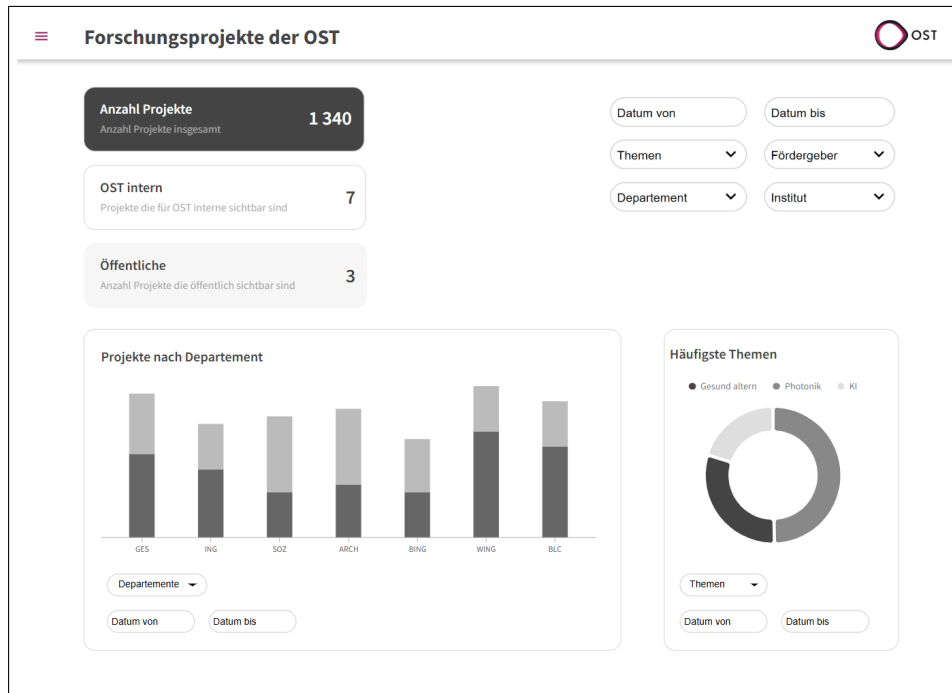


Abbildung 3.14: Mockup der Dashboard-Seite

### 3.4.4 Designentscheide

Die während der Konzeption des User-Interfaces getroffenen Designentscheide sind in den nachfolgenden Tabellen (3.6 und 3.7) festgehalten.

Entscheid	Begründung
Wahl des Farbschemas	Da seitens des Kunden keine spezifischen Farben vorgegeben wurden, wird das Farbschema an das Corporate Design der OST angelehnt. So wird ein einheitliches Erscheinungsbild gewährleistet, wenn die Anwendung auf Seiten der OST verlinkt wird. Es werden die Primärfarbe #8C195F und die Sekundärfarbe #D72764 gewählt. Zusätzlich werden 10 Abstufungen der Primärfarbe erstellt, um ein harmonisches Design zu gewährleisten. Eine Visualisierung der Farben findet sich in Abschnitt 3.4.1.
Projekt-Detailansicht Seitennavigation	Da die Projektdetailansicht bereits zum Stand des MVP Inhaltlich umfangreich ist und als Erweiterung auch interaktive Funktionen integriert werden sollen, wurde entschieden, die Seite in mehrere logische Blöcke zu unterteilen. Diese Blöcke können über eine Seitennavigation direkt erreicht werden. Dies soll dem Benutzer eine schnellere Orientierung ermöglichen und somit den Zugriff auf die gewünschten Inhalte erleichtern.
Filtermöglichkeiten der Dashboard Seite	Für die Art wie die Daten der Grafiken der Dashboard-Seite gefiltert werden wurden zwei mögliche Ansätze evaluiert. <ol style="list-style-type: none"> <li>1. Individuelle Filteroptionen pro Grafik.</li> <li>2. Globale Filteroptionen für alle Grafiken.</li> </ol> Option 2 wurde als vorteilhafter bewertet. Sie reduziert den Aufwand, mehrfach identische Filter setzen zu müssen, was insbesondere bei einer vielen Filtermöglichkeiten als umständlich empfunden werden könnte. Zudem erleichtert die globale Lösung die Implementierung, da die bereits von der Startseite bekannte Filterkomponente mit angepassten Einschränkungen der Filteroptionen wiederverwendet werden kann. Dies verbessert auch die Benutzerfreundlichkeit, da die Anwender bereits mit der Filterkomponente vertraut sein sollten.

Tabelle 3.6: Designentscheide

Entscheid	Begründung
Projekterstellung: Anzeige auf einer oder mehreren Spalten	<p>Möglichkeiten für die Gestaltung der Spalten im Formular, die evaluiert wurden:</p> <ol style="list-style-type: none"> <li>1. Erfassung in einer einzigen Spalte</li> <li>2. Erfassung in 2 oder mehr Spalten</li> </ol> <p>Das optimale Layout für Formulare ist ein umstrittenes Thema. Formulare mit einer einzigen Spalte fördern die Nutzerführung und reduzieren die Wahrscheinlichkeit, dass Felder übersehen werden, während mehrspaltige Layouts kürzer wirken, jedoch anfälliger für Eingabefehler sind [18]. Studien und von Firmen veröffentlichte A/B-Tests liefern teils unterschiedliche Ergebnisse: Viele bestätigen, die These, dass einspaltige Layouts übersichtlicher und weniger fehleranfällig sind [19], andere messen schnellere Zeiten bei Formularen mit mehreren Spalten, lange Formulare abschreckend wirken können [20].</p> <p>Die Entscheidung fiel auf ein einspaltiges Layout, da die Beweislage für dessen Effektivität überwiegt. Um Überforderung zu vermeiden, wird das Formular in logisch gruppierte, ausklappbare Abschnitte unterteilt. Dies gewährleistet eine klare Struktur und reduziert den Eindruck von Informationsüberfluss.</p>
Projekterstellung: Erweiterte Informationen anzeigen oder verbergen	<p>Zur Darstellung erweiterter Informationen wurden folgende Optionen geprüft:</p> <ol style="list-style-type: none"> <li>1. Anzeige der Informationen bei Hover über ein Info-Icon (analog Mockup)</li> <li>2. Dauerhafte Anzeige der Informationen</li> </ol> <p>Die Anzeige bei Hover spart Platz und reduziert die visuelle Länge des Formulars. Gleichzeitig erhöht diese Lösung jedoch den Interaktionsaufwand, da Benutzer aktiv auf das Icon zeigen müssen, um die Informationen einzusehen. Wir haben uns für die permanente Anzeige entschieden, da der Zusatztext in unserem Formular nicht als übermässig verbos empfunden wird und die Bedienbarkeit somit verbessert wird.</p>
Projekterstellung: markieren von optionalen Inhalten	<p>Optionale Felder werden nicht nur mit einem Stern (*) gekennzeichnet, sondern zusätzlich mit dem Tag „(optional)“ versehen. In den Usability Tests hat sich gezeigt, dass diese doppelte Kennzeichnung vorteilhaft ist. Die optionalen Felder (abgesehen von den Medien) sind ausserdem in einem logischen Block am Ende des Formulars zusammengefasst. Diese Gestaltung unterstützt die Benutzer dabei, sich zunächst auf die wichtigsten Inhalte zu konzentrieren. Diese Vorgehensweise wird auch als Best Practice empfohlen [21].</p>

Tabelle 3.7: Designentscheide Teil 2

# Kapitel 4

## Umsetzung

In diesem Kapitel werden ausgewählte Aspekte der Umsetzung aufgezeigt. Im Unterschied zum Architektur-Kapitel beziehen sich die hier gezeigten Ausschnitte auf konkrete Implementierungslogik oder auf die verwendeten Frameworks.

### 4.1 MinIO Konfiguration

In Program.cs wird der MinIO Client zur Applikation hinzugefügt und mit dem Endpoint und den Keys aus der Konfiguration konfiguriert. Ist eine dieser Konfigurationsvariablen nicht vorhanden oder leer, führt dies zu einem Absturz. Der Proxy wird verwendet, um den Filezugriff per Server oder IP auf den Docker Container umzuleiten.

In den Zeilen 8-12 wird die maximale Grösse der Requests auf 64 MB gesetzt, da mit den .NET Standardeinstellungen die API Anfragen bei zu grossen Files (ca. 25 MB) abgebrochen wurden. Der Wert muss an die maximale Dateigrösse angepasst werden (aktuell 50 MB) und noch etwas Spielraum für weitere Daten in der Request bieten.

```
1 builder.Services.AddMinio(configureClient => configureClient
2     .WithEndpoint(configuration["Minio:Endpoint"])
3     .WithCredentials(configuration["Minio:AccessKey"], configuration["Minio:SecretKey"])
4     .WithProxy(new WebProxy("minio", 9000))
5     .WithSSL(false)
6     .Build());
7
8 app.Use(async (context, next) =>
9 {
10     context.Features.Get<IHttpRequestBodySizeFeature>().MaxRequestBodySize = 64 * 1024 * 1024;
11     await next();
12 });
```

Listing 4.1: Konfiguration des MinIO Client in Program.cs

## 4.2 Benutzer-Autorisierung

In diesem Abschnitt werden verschiedene Codeauszüge im Zusammenhang mit der Benutzerautorisierung dargestellt. Dies beinhaltet die Registrierung der Authentication Services, die Erstellung sowie das Auslesen der [Json Web Tokens \(JWTs\)](#).

### 4.2.1 Registrierung der Authentication Services

Dem [Dependency Injection \(DI\)](#) Container im Program.cs wird ein [JWT Bearer Authentication Schema](#) hinzugefügt. Hierfür wird das Package [JwtBearer](#) benötigt. Die Validierungsparameter (Aussteller, Zielgruppe, Key zur Generierung der Signing Credentials) werden aus der Konfiguration (also aus den AppSettings bzw. den Environment-Variablen) ausgelesen.

Mit [UseAuthorization](#) wird die Autorisierung mit den registrierten Authentication Schemas für die API Endpunkte aktiviert, welche mit der Annotation [\[Authorize\]](#) gekennzeichnet sind.

```
1 builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
2     .AddJwtBearer(options =>
3     {
4         options.TokenValidationParameters = new TokenValidationParameters
5         {
6             ValidateIssuer = true,
7             ValidateAudience = true,
8             ValidateLifetime = true,
9             ValidateIssuerSigningKey = true,
10
11             ValidIssuer = configuration["Jwt:Issuer"],
12             ValidAudience = configuration["Jwt:Audience"],
13             IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(configuration["
14             Jwt:Key"]));
15         });
16     });
17 app.UseAuthorization();
```

Listing 4.2: Registrierung der Authentication Services

### 4.2.2 Token-Erstellung

Nachdem sich ein Benutzer erfolgreich registriert oder eingeloggt hat, erhält er einen Access Token zurück, welchen er danach im Authorization Header als Bearer Token mitsenden kann. Dieser Token wird im AccountService mit der Methode [CreateToken](#) erstellt. Dafür werden SigningCredentials mit dem [JWT Key](#) in der Konfiguration und der Hashfunktion [SHA-256](#) erstellt und damit die UserClaims (Informationen zum Benutzer) signiert. Die UserClaims beinhalten die E-Mail, die Id und die Rolle des Benutzers sowie die Id der Organisationseinheit, bei welcher der Benutzer angestellt ist. Relevant für die Autorisierung sind davon alle ausser die E-Mail. Die Gültigkeitsdauer des Tokens ist noch hard-codiert auf 800 Minuten, da noch kein Refresh Token erstellt wird. Dies ist für den produktiven Einsatz deutlich zu lange. Die Einstellung soll in einer zukünftigen Version ebenfalls aus der Konfiguration entnommen werden.

Für diesen Prozess wird das Package [JWT](#) benötigt.

```
1 private TokenDto CreateToken(User user)
2 {
3     var signingCredentials = GetSigningCredentials();
4     var claims = GetClaims(user);
5     var tokenOptions = GenerateTokenOptions(signingCredentials, claims);
6
7     var refreshToken = "";
8     var accessToken = new JwtSecurityTokenHandler().WriteToken(tokenOptions);
9
10    return new TokenDto(accessToken, refreshToken);
11 }
12
13 private SigningCredentials GetSigningCredentials()
14 {
15     var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_configuration["Jwt:Key"
16     ]));
17     return new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);
18 }
19 private List<Claim> GetClaims(User user)
20 {
21     return
22     [
23         new Claim("id", user.Id.ToString()),
24         new Claim("email", user.Email),
25         new Claim("role", user.Role.GetDescription()),
26         new Claim("institute", user.InstituteId.ToString() ?? "")
27     ];
28 }
29
30 private JwtSecurityToken GenerateTokenOptions(SigningCredentials signingCredentials, List<
31 Claim> claims)
32 {
33     return new JwtSecurityToken(
34         issuer: _configuration["Jwt:Issuer"],
35         audience: _configuration["Jwt:Audience"],
36         claims: claims,
37         expires: DateTime.UtcNow.AddMinutes(800),
38         signingCredentials: signingCredentials);
39 }
```

Listing 4.3: Generierung der JWTs

### 4.2.3 Autorisierung des Schreibzugriffs mit JWT Claims

Ein Benutzer darf Projekte nur in folgenden Fällen bearbeiten:

- Der Benutzer ist der Ersteller des Projektes
- Der Benutzer ist Leiter einer am Projekt beteiligten Organisationseinheit
- Der Benutzer ist fachlicher oder kommunikativer Prüfer und das Projekt ist in der entsprechenden Prüfphase

Hierfür werden in `AccessControlService.AuthorizeUpdateAccess` nacheinander die Claims `Id`, `Role` und `Institute` in der privaten Methode `GetUserClaim` über ein `HttpContextAccessor`-Objekt ausgelesen und die



drei bzw. vier oben genannten Varianten geprüft (die Prüfer-Autorisierung ist nicht im Code-Snippet enthalten). Trifft kein Fall zu, wird eine Exception geworfen.

```
1 var userId = GetUserClaim("id") ?? string.Empty;
2 if (project.Contributors.Any(c => c.IsEditor && c.UserId.ToString() == userId)) return;
3 var userRole = GetUserClaim("role") ?? string.Empty;
4 if (userRole == Role.HeadOfInstitute.GetDescription())
5 {
6     var userInstituteId = GetUserClaim("institute") ?? string.Empty;
7     if (project.Institutes.Any(i => i.Id.ToString() == userInstituteId))
8         return;
9 }
10
11 public string? GetUserClaim(string claimType)
12 {
13     return _HttpContextAccessor.HttpContext?.User.FindFirst(claimType)?.Value;
14 }
```

Listing 4.4: Benutzerautorisierung mit JWT Claims

#### 4.2.4 Nuxt.js Middleware für die Autorisierung geschützter Routes

Benutzer sollten daran gehindert werden, eine Seite zu öffnen, auf welche sie keinen Zugriff haben. Dafür wird die folgende Nuxt Middleware zur Autorisierung eingesetzt. Sie wird automatisch ausgeführt, wenn ein Client (Browser) eine Seite der Anwendung öffnet, welche die Middleware in `definePageMeta` verwendet.

```
1 // middleware/auth.js
2 export default defineNuxtRouteMiddleware((to) => {
3     if (import.meta.server) return
4     const token = localStorage.getItem('jwt')
5     if (!token || isTokenExpired(token)) {
6         return navigateTo('/login')
7     }
8     const requiresAnyRoleOf = to.meta.requiresAnyRoleOf || []
9     if (requiresAnyRoleOf.length > 0 && !hasAnyRole(requiresAnyRoleOf)) {
10        return abortNavigation('Insufficient permissions.')
11    }
12 })
13
14 // Verwendung der Middleware in einer Page (Hier: my-projects/[id].vue)
15 definePageMeta({
16     middleware: ['auth'],
17     requiresAnyRoleOf: [Role.InstituteEmployee, Role.HeadOfInstitute, Role.UniversityManager],
18     title: 'header.editProject',
19 })
```

Listing 4.5: Middleware zur Autorisierung geschützter Routes

### 4.3 Filtern der Projekte in der Datenbank

Die Projekte sollen direkt in der Datenbank nach den Suchkriterien gefiltert werden. Dafür kann im Repository ein `IQueryable` mit allen Suchbedingungen erstellt und am Ende ausgeführt werden. Dieses `IQueryable` wird in der Methode `CreateProjectsSearchQuery` mit allen angewendeten Filtern aufgebaut. Bei der Projektsuche muss neben den übergebenen Filterkriterien auch der Projektstatus in Verbindung mit den Benutzerberechtigungen mit einbezogen werden. Dies ist beim Dashboard nicht der Fall, weshalb dort nur Projekte mit Status `Draft` oder `Archiviert` herausgefiltert werden. Die weiteren Berechnungen für die Dashboard-Statistiken befinden sich im `ProjectService`.

Ausserdem wird hier das Paging angewendet. Hier wird für den Offset die aktuelle `PageNumber` um 1 reduziert und von da aus die nächsten `PageSize` Projekte geladen. Neben den geladenen Projekten wird auch die Gesamtanzahl der passenden Projekte zurückgegeben, welche für die korrekte Anzeige des Paging im Frontend nötig ist.

```
1 IQueryable<Project> projectsQuery = CreateProjectsSearchQuery(searchRequestDto);
2
3 if (authorizedOnly)
4 {
5     projectsQuery = AddAuthorizationFilters(projectsQuery, searchRequestDto);
6 }
7 else
8 {
9     projectsQuery = projectsQuery.Where(p => p.Status != Status.Draft && p.Status != Status.Archived);
10 }
11
12 var page = Math.Max(searchRequestDto.PageNumber, 1);
13 var size = Math.Max(searchRequestDto.PageSize, 1);
14
15 var totalCount = projectsQuery.Count();
16 return new ProjectSearchResponseDto<SearchResponseItemDto>
17 {
18     Items = (await projectsQuery.Include(p => p.Contributors)
19         .ThenInclude(c => c.User)
20         .Include(p => p.Institutes)
21         .Include(p => p.SearchTags)
22         .Skip((page - 1) * size)
23         .Take(size)
24         .ToListAsync(cancellationToken: cancellationToken)).ToSearchResponseDtoList(),
25     TotalCount = totalCount
26 };
```

Listing 4.6: Projekte nach Suchkriterien filtern

### 4.4 Error Handling im Backend

Durch die Methode "AddExceptionHandler" werden Exception-Handler als Services im Dependency Injection Container des Backends registriert. Jeder Exception Handler ist für die Verarbeitung eines spezifischen Exceptiontyps verantwortlich. Bei der Registrierung (siehe Listing 4.7) spielt die Reihenfolge eine wichtige Rolle. Kommt es zu einem Fehler, werden die Handler nämlich nacheinander in der Reihenfolge der Registrierung aufgerufen, bis einer die Exception erfolgreich behandelt. Der letzte

Exception-Handler "GlobalExceptionHandler" kümmert sich somit um all diejenigen Exceptions, um die sich kein Handler explizit kümmert.

```
1 builder.Services.AddExceptionHandler<BadRequestExceptionHandler>();
2 builder.Services.AddExceptionHandler<NotFoundExceptionHandler>();
3 builder.Services.AddExceptionHandler<UnauthorizedExceptionHandler>();
4 builder.Services.AddExceptionHandler<GlobalExceptionHandler>();
```

Listing 4.7: Registrierung der Error Handler im Backend

Im Listing 4.8 wird exemplarisch ein Ausschnitt aus dem NotFoundExceptionHandler gezeigt. Zuerst wird geprüft, ob es sich um eine Fehlermeldung handelt, die von diesem Handler bearbeitet werden muss, in diesem Fall eine NotFoundException. Ist dies nicht der Fall, wird "false" zurückgegeben, wodurch alle später registrierten Handler die Möglichkeit haben, die Exception zu behandeln. Handelt es sich um eine NotFoundException, wird der Fehler anschliessend gelogged (hier nicht zu sehen). Um die Fehlermeldung zu standardisieren, wird anhand des Fehlers ein ProblemDetails-Objekt erstellt. Dieses Objekt wird als JSON mittels einer HTTP-Response dem Client zurückgegeben.

```
1 if (exception is not NotFoundException notFoundException)
2 {
3     return false;
4 }
5 ...
6 var problemDetails = new ProblemDetails
7 {
8     Status = StatusCodes.Status404NotFound,
9     Title = "Not Found",
10    Detail = notFoundException.Message
11 };
12
13 httpContext.Response.StatusCode = problemDetails.Status.Value;
14
15 await httpContext.Response
16     .WriteAsJsonAsync(problemDetails, cancellationToken);
17 ...
```

Listing 4.8: Implementation des NotFoundExceptionHandler

## 4.5 Frontend Basis-Komponente

Im Listing 4.9 ist die Datei "app.vue" zu sehen. Diese wird beim Start der Anwendung geladen und bildet die Grundlage für die Darstellung von allen Seiten. Sie beinhaltet die Layout-spezifische Komponente "NuxtLayout". Über diese wird das aktuell gesetzte Layout (in unserem Fall der Header) integriert. Mit der Komponente "NuxtPage" werden die Seiteninhalte dynamisch geladen. Zudem wird "Toast" hier für die Anzeige von Benachrichtigungen eingebunden.

```
1 <script setup lang="ts">
2 </script>
3
4 <template>
5   <div class="h-screen">
6     <NuxtLayout>
7       <Toast />
8     <NuxtPage />
9   </NuxtLayout>
10 </div>
11 </template>
12
13 <style>
14 </style>
```

Listing 4.9: Code der Basis-Komponente app.vue

## 4.6 Weitere Frameworks und Bibliotheken

In diesem Abschnitt werden weitere wichtige Frameworks und Bibliotheken erläutert, welche für die Umsetzung der Anwendung verwendet werden.

### 4.6.1 Testing im Backend

Im Backend-Projekt wird das Open-Source-Testframework xUnit verwendet. Dieses speziell für.NET entwickelte Framework setzt auf voneinander isolierte Testinstanzen, was eine parallele Ausführung aller Tests ermöglicht. Darüber hinaus können auch datengetriebene Tests geschrieben werden, bei denen anhand bestimmter Attribute mehrere Eingabewerte für denselben Testfall definiert werden können. Dies ermöglicht ein effizientes Testen von verschiedenen Szenarien mit nur minimalem zusätzlichem Codeaufwand.

Zusätzlich wird die Bibliothek Moq verwendet, die den Prozess des Unit-Testings durch die Erstellung von Mock-Objekten vereinfacht. Mit Hilfe dieser Mock-Objekte können Abhängigkeiten zwischen Klassen oder Methoden simuliert werden, wodurch Unit-Tests unabhängig von externen Systemen (wie beispielsweise Datenbanken) durchgeführt werden können.

### 4.6.2 Validierung im Frontend

Für die Validierungslogik der Projektbearbeitungsseite im Frontend wird die Bibliothek Zod verwendet. Zod bietet eine deklarative Möglichkeit, um Validierungsfunktionalitäten zu definieren. Im gezeigten Codeausschnitt (Listing 4.10) wird ein Schema für die Validierung eines Projekts definiert. Zudem sind die Validierungsfunktionen des Felds "description" zu sehen. So können spezifische Anforderungen an die

Inhalte jedes einzelnen Feldes geprüft werden. Sind diese nicht erfüllt, werden für das Feld passende Fehlermeldungen angewendet.

```
1  const projectEditSchema = toTypedSchema(  
2    z.object({  
3      description:  
4        z.preprocess(val => (val === null ? '' : val),  
5          z.string()  
6            .min(1, { message: t('edit.message.descriptionRequired') })  
7            .max(maxDescriptionLength, {  
8              message: t('edit.message.descriptionMax', { max: maxDescriptionLength }),  
9            })  
10     })  
11  ),
```

Listing 4.10: Validierung im Frontend mit Zod

### 4.6.3 Integration von PrimeVue und Entwicklung eines OST-Themes

Für die Erstellung der Benutzeroberfläche wird unter anderem das Framework PrimeVue [22] verwendet. PrimeVue ist ein Open-Source-Framework, das eine Vielzahl von UI-Komponenten und Icons zur Verfügung stellt. Darüber hinaus bietet das Framework mit dem sogenannten "Styled Mode" die Möglichkeit, auf einfache Art und Weise ein eigenes, einheitliches UI-Theme für eine Anwendung zu erstellen. Für den OST ResearchNavigator wird ein durch PrimeVue erstelltes Standard-Theme den farblichen Ansprüchen der geplanten Anwendung angepasst.

#### Diskussion zur Verwendung von UI-Komponenten

Die Verwendung von UI-Komponenten erleichtert die Entwicklung und sorgt für ein einheitliches Design, da die Komponenten aufeinander abgestimmt sind. Ausserdem kann durch die Verwendung eines selbst definierten PrimeVue-Themes der Aufwand für das Styling der einzelnen Komponenten reduziert werden. Jedoch kann es bei UI-Bibliotheken umständlich sein, diese für einen bestimmten Anwendungszweck zu optimieren. PrimeVue bietet mit dem sogenannten "Pass Through" eine API an, um auf die interne DOM-Struktur der Komponenten zuzugreifen. Dies erleichtert die individuelle Anpassung einzelner Komponenten, benötigt aber wiederum Kenntnisse der entsprechenden API, was den Entwicklungsprozess verlangsamen kann.

# Kapitel 5

## Qualitätsmassnahmen

Dieses Kapitel beschreibt Konzepte und Strategien, die von uns zur Sicherung der Qualität angewendet werden.

### 5.1 Code Qualität

Damit eine hohe Code-Qualität sichergestellt werden kann, werden folgende Richtlinien und Massnahmen definiert:

Massnahme	Beschreibung
Backend	Im Backend werden die von Microsoft vorgegebenen Coding Guidelines für C# verwendet. Anhand einer eingerichteten run-config kann der Code so formatiert werden, dass er den von Microsoft empfohlenen Regeln zur Code-Formatierung entspricht.
Frontend	Im Frontend werden die von Nuxt ESLint vorgegebenen Standardrichtlinien verwendet. Auch diese coding-Richtlinien können durch ESLint automatisiert geprüft und Probleme in vielen Fällen direkt behoben werden. Der Code wird zudem mittels einer ESLint Konfiguration automatisch formatiert.
Test Coverage	Für die Unit Tests im Frontend kann ein Test coverage Report generiert werden. Der Report beinhaltet: <ul style="list-style-type: none"><li>• Line coverage insgesamt</li><li>• Branch coverage insgesamt</li><li>• Totale Anzahl Code-Zeilen</li><li>• Die line und branch coverage von jedem File</li></ul> Dadurch kann eingesehen werden ob die Testabdeckung ausreichend ist und herausgefunden werden, an welchen Stellen im Code potentiell noch Tests fehlen.
Automation	Diese Massnahmen werden zudem in der Pipeline automatisiert ausgeführt. Dies wird im Abschnitt 5.2.4 genauer beschrieben. Sind die oben definierten Massnahmen zur Einhaltung der Code Qualität nicht erfüllt, so kann dieser Code auch nicht in den main-branch eingeführt werden.

Tabelle 5.1: Massnahmen zur Verbesserung der Code Qualität

## 5.2 Versionskontrolle

Zur Versionskontrolle wird die GitLab-Instanz der OST genutzt. Diese bietet eine sichere Umgebung für die Verwaltung des Quellcodes und die Zusammenarbeit im Team.

### 5.2.1 Struktur

Für das Projekt wird ein Monorepo-Ansatz verwendet, bei dem Frontend und Backend in einem einzigen Repository verwaltet werden. Dieser Ansatz ermöglicht es, Änderungen, die sowohl das Frontend als auch das Backend betreffen, in einem einzigen Commit/Branch durchzuführen, wodurch Inkonsistenzen vermieden werden. Die überschaubare Grösse des Projekts und des Teams verhindert typische Nachteile eines Monorepos wie Performance-Probleme oder häufige Merge-Konflikte [23].

### 5.2.2 Workflow

Als Entwicklungsworkflow wird der Git Feature Branch Workflow verwendet. Jeder Feature-Branch wird direkt vom main-Branch aus erstellt. Nach Abschluss der Arbeit wird der Feature-Branch per Merge-Request in den main-Branch integriert [24]. Dieser Prozess eignet sich, da es sich bei dieser Entwicklung um einen MVP handelt, der nur einen simplen Deployment-Prozess erfordert.

### 5.2.3 Branch Naming

Branchennamen werden wie folgt aufgebaut: <Präfix>/<Jira Ticketnummer>-<Branchname>  
Mögliche Präfixe sind:

- **feature:** für die Entwicklung neuer Features verwendet
- **bugfix:** für das Beheben von Bugs
- **experimental:** für das probieren von Ideen, die nicht teil eines Releases sein sollten

Ein Beispiel wäre: feature/SPFF-95-projekte-erfassen

### 5.2.4 Pipeline

Anhand der GitLab Pipeline werden Massnahmen zur Einhaltung der Codequalität automatisiert durchgesetzt. Dies erfolgt in den in Abbildung 5.1 dargestellten Stages. Tritt in einer Stage ein Fehler auf, wird die Pipeline abgebrochen. Dadurch wird sichergestellt, dass alle identifizierten Probleme behoben werden müssen, bevor die entsprechenden Änderungen in den main-Branch übernommen werden können.

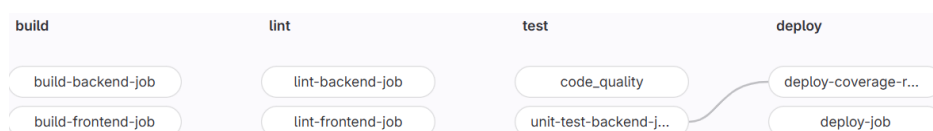


Abbildung 5.1: Visualisierung der GitLab Pipeline

Stage	Beschreibung
Build	Die Build Stage stellt sicher, dass der Code kompiliert und die Anwendung erfolgreich gebildet werden kann.
Lint	Überprüft den Code auf Verstösse gegen definierte Coding-Standards.
Test	Führt alle Unit Tests des Backends durch und generiert einen Bericht über die Code Qualität.
Deploy	Wird nur ausgeführt, wenn Änderungen in den main-branch übernommen werden. Veröffentlicht einen coverage Report für die Testabdeckung des Backends in einem GitLab Environment. In dieser Stage wird kein automatisiertes Deployment der Anwendung durchgeführt.

Tabelle 5.2: Beschreibung der Pipeline Stages

### 5.3 Code Reviews

Anhand von Code Reviews zusammen mit einem Fachexperten der OST erhält das Entwicklerteam wertvolles Feedback. Dadurch kann es die Qualität des Codes sowie auch der Architektur, wo nötig, verbessern. Die aus den Code Reviews resultierenden Fragen, Mängel und deren entsprechenden Verbesserungen sind im Anhang in der Tabelle 9.8 aufgelistet.

### 5.4 Testing

In diesem Abschnitt werden die durchgeführten Tests beschrieben.

#### 5.4.1 Unit Tests

Unit Tests werden gemacht, um sicherzustellen, dass einzelne Komponenten und Funktionen unabhängig voneinander korrekt arbeiten. Sie werden während der Entwicklung kontinuierlich geschrieben. Ziel nach NFR-6 "Test Coverage" (siehe Tabelle 2.24) ist eine Branch Coverage von >80% für die Service-Methoden im Backend. Dadurch wird derjenige Teil der Anwendung getestet, der die Businesslogik enthält. Die Metrik der Branch Coverage wird gewählt, da sie einen Einblick in die Abdeckung aller Entscheidungswege im Code liefert und somit die Abdeckung von Sonderfällen besser erfasst als eine Line Coverage. Die erreichte Testabdeckung in den Service-Klassen ist in der Abbildung 5.2 unten dargestellt.

Name	Covered	Uncovered	Coverable	Total	Percentage	Covered	Total	Percentage
Source.Services.Abstractions.SearchService	52	10	62	94	83.8%	12	14	85.7%
Source.Services.AccessControlService	36	0	36	64	100%	36	36	100%
Source.Services.AccountService	72	0	72	116	100%	13	14	92.8%
Source.Services.FileService	60	4	64	104	93.7%	19	22	86.3%
Source.Services.InstituteService	8	0	8	21	100%	0	0	
Source.Services.ProjectService	337	21	358	504	94.1%	69	82	84.1%
Source.Services.ValidationService	137	5	142	212	96.4%	44	54	81.4%

Abbildung 5.2: Testabdeckung der entwickelten Backend-Services



### 5.4.2 Usability Tests

Um zu prüfen, wie gut ein Benutzer tatsächlich mit dem System interagieren kann, werden Usability-Tests durchgeführt. Anhand dieser Tests können konkrete Probleme der Benutzerfreundlichkeit identifiziert und behoben werden. Um früh auf Probleme reagieren zu können, werden Usability-Tests auch frühzeitig durchgeführt und später wiederholt. Das dazu verwendete Testprotokoll befindet sich im Anhang (Tabelle 9.1).

Die Tests werden mit 2 Professoren der OST durchgeführt, die zum Zielpublikum der Anwendung passen. Beide gehören zur potenziellen Nutzergruppe für die Projekterfassung und die Projektsuche. Prof. Dr. Frieder Loch ist am I3 Institut für Interaktive Informatik tätig und ist Professor für User-Centered Design. Seine fachlichen Schwerpunkte liegen unter anderem in den Bereichen Mensch-Maschine-Interaktion, Webentwicklung und Barrierefreiheit. Prof. Dr. Markus Stolze ist ebenfalls am I3 Institut tätig und konzentriert sich auf den Bereich User Interface Technologien. Durch ihre hohe Fachkompetenz konnten die Testpersonen während der Durchführung der Usability Tests detailliert aufzeigen, wo es noch mögliche Usability-Probleme in der Anwendung gibt.

Die im Rahmen der Usability Tests durchgeführten Verbesserungen sind im Anhang in der Tabelle 9.7 aufgelistet. Alle Ideen und Verbesserungsvorschläge, die noch nicht realisiert wurden, sind im Product Backlog des Entwicklerteams festgehalten und auch im Anhang unter 9.1.1.

### 5.4.3 System Tests

Für die Prüfung aller wichtigen Abläufe und Funktionen der Anwendung wurde ein Systemtestprotokoll erstellt. Das Protokoll wurde in Form einer Checkliste verfasst und ist direkt im Github Repository abgelegt, um die wiederholte Durchführung zu erleichtern.

Das vollständig ausgefüllte Protokoll der letzten Durchführung der Systemtests befindet sich im Anhang 9.1.3.

### 5.4.4 Load Tests

Um die Performance des Systems unter extremen Bedingungen zu testen, wurde das Lasttest-Tool Apache JMeter verwendet. Zur Verifizierung der NFR 1 und 2 (Abschnitt 2.2) wurde ein Testplan angelegt. Der Testplan enthält die Definitionen der Header (Content-Type und Authorization), die Request-Defaults (Servername und Ports) sowie zwei Thread Groups.

In der ersten Thread Group werden jeweils 50 Threads in sehr kurzer Zeit ausgeführt. Dieser dient zur Verifizierung der Such-, Filter-, Create- und Update-Operationen.

Die zweite startet nur einen Thread auf einmal, was nötig war, um die nicht mit den gleichen Einstellungen wiederholbaren Lösch- und File-Upload-Operationen zu testen. Diese Group wurde manuell wiederholt mit verschiedenen Einstellungen gestartet.

Screenshots der Apache JMeter Anwendung mit Einstellungen und Ergebnissen befinden sich im Anhang (9.1.4).

## 5.5 Dokumentation des Codes

Gut dokumentierter Code hilft es Entwicklern, den Sourcecode schneller zu verstehen und effizient mit der Codebasis zu arbeiten. Eine übermässige Dokumentation kann jedoch auch zu grossem Mehraufwand führen, da bei jeder Änderung des Codes auch die Dokumentation aktualisiert werden muss. Geht die Aktualisierung der Dokumentation vergessen, besteht die Gefahr, dass die Dokumentation veraltet und damit unzuverlässig wird.

Um die Dokumentation wartbar zu gestalten, werden Tools verwendet, die aus Code und den dazu geschriebenen Code-Kommentaren automatisch übersichtliche Dokumentationen generieren.

Zum einen wird Swagger UI verwendet [25], das aus den [Application Programming Interface \(API\)](#)-Definitionen eine interaktive [Hypertext Markup Language \(HTML\)](#)-Seite generiert. Diese Seite zeigt alle verfügbaren Endpunkte und deren Schnittstellendefinitionen an. Entwickler können die Endpunkte direkt im Browser testen, was den Entwicklungsprozess erleichtert. Die Swagger-Dokumentation wird beim lokalen Start des Backends automatisch generiert und geöffnet. Eine Abbildung von dieser interaktiven Seite ist im Anhang unter [9.2.5](#) zu finden.

Zum anderen werden alle Service-Schnittstellen im Backend mit [Extensible Markup Language \(XML\)](#)-Dokumentationskommentaren versehen. Diese Kommentare ermöglichen es, zusammen mit dem Quellcode, eine durchsuchbare Dokumentation der Anwendung mit dem Tool Doxygen zu generieren [26]. Diese Dokumentation kann während der Entwicklung über eine Run-Configuration mit dem Namen "Open Docs" geöffnet werden. Beispiele der Doxygen-Dokumentation finden sich im Anhang unter [9.2.6](#) und [9.2.7](#).

# Kapitel 6

## Projektplan

### 6.1 Vorgehen im Projekt

Das Projektvorgehen orientiert sich am **Rational Unified Process (RUP)**, welcher die langfristige Planung und Phasenstruktur vorgibt. Für die kurzfristige Planung wird hingegen das Scrum-Framework angewendet, um mehr Flexibilität und iterative Anpassungen zu ermöglichen. Diese Kombination schafft eine ausgewogene Balance zwischen Struktur und Agilität, ein Ansatz, der zumindest OST-intern auch als Scrum+ bezeichnet wird.

Während der gesamten Projektlaufzeit finden wöchentliche Besprechungen mit den Betreuungspersonen statt. Interne Teambesprechungen finden mindestens einmal pro Woche statt, werden aber flexibel und spontan nach Bedarf geplant und durchgeführt.

### 6.2 Zeitplan

Die einzelnen Phasen nach RUP sind im Zeitplan durch die dunkelblauen Balken dargestellt, während die gelben Balken die von uns definierten Epics darstellen. Die grünen Balken bezeichnen Anforderungen, die als "could have" eingestuft wurden. Bei sehr gutem Vorankommen während der Entwicklung, wird an den grünen Epics weitergearbeitet. Einzelne Elemente befinden sich nicht immer exakt innerhalb ihrer primär übergeordneten Phase. Dies ist gewollt und gehört zum üblichen Vorgehen nach RUP.

Der Zeitplan wurde weitgehend eingehalten, wobei es gegen Ende des Zeitplans Abweichungen ersichtlich sind. Es wurde entschieden, auch über die Construction-Phase hinaus noch an "could have" Features zu arbeiten, da bereits vereinbart wurde, dass das Projekt vom Entwicklerteam auch als Bachelorarbeit fortgeführt wird. Somit konnte auch das Risiko eingegangen werden, dass Features nicht mehr vollständig fertiggestellt werden, da der Entwicklungsstand an diesen Features zu einem späteren Zeitpunkt fortgesetzt werden könnte.

# Publikationsapp für Forschungsprojekte

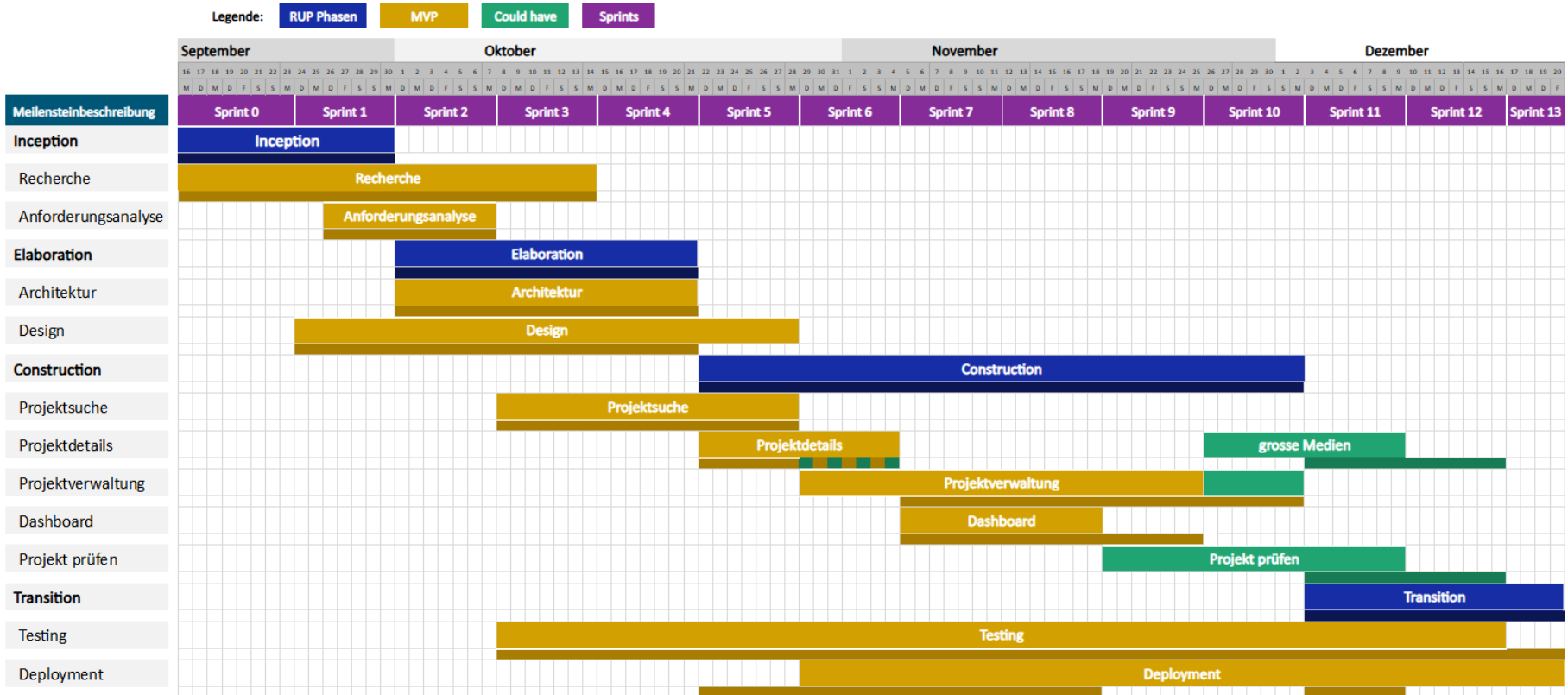


Abbildung 6.1: Projektplan

### 6.2.1 Meilensteine

Das Ende von jedem violetten Epic markiert gleichzeitig einen wichtigen Meilenstein, der erreicht werden muss. In folgender Tabelle sind die genauen Daten der einzelnen Meilensteine aufgelistet.

Datum	Meilenstein
<b>30.09.2024</b>	<b>Ende der Inception Phase</b>
07.10.2024	Anforderungsanalyse abgeschlossen
14.10.2024	Initiale Rechercharbeiten abgeschlossen
21.10.2024	Architektur ist fertig ausgearbeitet
<b>21.10.2024</b>	<b>Ende der Elaboration Phase</b>
28.10.2024	Design ist ausgearbeitet
28.10.2024	Epic: Projektsuche abgeschlossen
04.11.2024	Epic: Projektdetails (MVP) abgeschlossen
18.11.2024	Epic: Dashboard abgeschlossen
25.11.2024	Epic: Projektverwaltung (MVP) abgeschlossen
<b>02.12.2024</b>	<b>Ende der Construction Phase</b>
17.12.2024	Usability Testing abgeschlossen
20.12.2024	Projekt ist OST intern veröffentlicht
<b>20.12.2024</b>	<b>Ende der Transition Phase (Abgabe)</b>

Tabelle 6.1: Meilensteine

## 6.3 Risikomanagement

In diesem Abschnitt sind die Projektrisiken (6.3.4) dokumentiert. Da sich Risiken über den Zeitraum des Projekts verändern, werden hier auch mehrere Risikoauswertungen festgehalten.

### 6.3.1 Risikoauswertung vom 26.09.2024

W/I <sup>1</sup>	Vernachlässigbar	Geringfügig	Kritisch	Katastrophal
Sicher	-	-	-	-
Wahrscheinlich	-	Bugs	-	-
Möglich	-	-	Umfang	Anforderungen
Unwahrscheinlich	-	-	Wissen	Security
Selten	-	-	Usability	-
Eliminiert	Eliminiert			

<sup>1</sup> Wahrscheinlichkeit/Impact

Tabelle 6.2: Risikomatrix 26.09.2024

### 6.3.2 Risikoauswertung vom 21.10.2024 - Ende der Elaboration Phase

W/I <sup>1</sup>	Vernachlässigbar	Geringfügig	Kritisch	Katastrophal
Sicher	-	-	-	-
Wahrscheinlich	-	-	-	-
Möglich	-	Bugs	-	-
Unwahrscheinlich	-	Security	-	-
Selten	-	-	Usability	Anforderungen
Eliminiert	Umfang, Wissen			

<sup>1</sup> Wahrscheinlichkeit/Impact

Tabelle 6.3: Risikomatrix 21.10.2024

#### Anmerkungen

Das Risiko „**Umfang**“ wurde eliminiert, indem dem Kunden ein Minimum Viable Product (MVP) vorgeschlagen wurde. Der Umfang des MVP wurde ausführlich mit dem Kunden besprochen und von diesem genehmigt. Darüber hinaus konnte das Risiko „**Wissen**“ ausgeschlossen werden, da das erforderliche Know-how im Bereich Nuxt aufgebaut wurde. Das Risiko bezüglich der **Anforderungen** wurde auf die Wahrscheinlichkeit „Selten“ herabgestuft. Dies wurde durch das Vorführen eines Mockups und der vom Projektteam ausgearbeiteten Use Cases erreicht, welche mit der Vision des Kunden übereinstimmen. Dieses Risiko konnte jedoch noch nicht eliminiert werden, da kleinere Missverständnisse noch immer möglich und nicht auszuschliessen sind. Das Risiko „**Security**“ wurde zudem auf „Geringfügig“ heruntergestuft, da die Implementierung einer Authentifizierung für den MVP nicht vorgesehen ist.

### 6.3.3 Risikoauswertung vom 02.12.2024 - Ende der Construction Phase

W/I <sup>1</sup>	Vernachlässigbar	Geringfügig	Kritisch	Katastrophal
Sicher	-	-	-	-
Wahrscheinlich	-	-	-	-
Möglich	-	-	-	-
Unwahrscheinlich	-	Bugs	-	-
Selten	-	Security	Usability	Anforderungen
Eliminiert	Umfang, Wissen			

<sup>1</sup> Wahrscheinlichkeit/Impact

Tabelle 6.4: Risikomatrix 02.12.2024

#### Anmerkungen

Das Risiko **Bugs** konnte von „Möglich“ auf „Unwahrscheinlich“ heruntergestuft werden, da das Produkt mehrfach manuell getestet wurde und auch die Businesslogik in den Serviceklassen eine hohe Testabdeckung aufweist. Zudem konnte das Risiko **Security** von „Unwahrscheinlich“ auf „Selten“ heruntergestuft werden, da das NFR betreffend Security geprüft wurde und als erfüllt gilt (siehe 2.23).

### 6.3.4 Identifizierte Risiken

Nr	1
Titel	Umfang
Beschreibung	Der Kunde wünscht ein Gesamtsystem, dass vom Umfang her nicht im vorgesehenen Zeitrahmen vollständig umgesetzt werden kann. Eine solches System würde unter anderem auch eine umfassende Benutzerverwaltung sowie weitere Administrative Eingriffsmöglichkeiten verlangen.
Vorbeugung	Mit dem Kunden eine Priorisierung der Anforderungen durchführen, um die wichtigsten Features für den MVP (Minimum Viable Product) festzulegen. So kann der Fokus auf grundlegende Funktionen wie das Anzeigen und erstellen von Projekten gelegt werden.

Tabelle 6.5: Risiko - Umfang

Nr	2
Titel	Anforderungen
Beschreibung	Missverständnisse oder Änderungen bei den Anforderungen können zu Mehraufwand sowie zu unzureichender oder fehlerhafter Funktionalität führen. Dieses Risiko wird berücksichtigt, da das Projektteam die konkreten Anforderungen ausarbeitet und der Kunde aufgrund seiner begrenzten Verfügbarkeit nicht immer unmittelbar eingebunden werden kann.
Vorbeugung	Die Sitzungen zur Ausarbeitung der Anforderungen sollten klar geplant werden, um die zur Verfügung stehende Zeit optimal zu nutzen. Im weiteren Projektverlauf soll so oft und so schnell wie möglich Feedback vom Kunden eingeholt werden.

Tabelle 6.6: Risiko - Anforderungen

Nr	3
Titel	Wissen (Nuxt.js)
Beschreibung	Mangelndes technologisches Wissen über Nuxt.js führt zu Fehlentscheidungen beim Entwurf der Softwarearchitektur oder zu erheblichem Mehraufwand bei der Implementierung.
Vorbeugung	Frühzeitige Auseinandersetzung mit nuxt.js, Information über dessen Funktionsweise sammeln, und diese in kommende Entscheide einfließen lassen.

Tabelle 6.7: Risiko - Wissen

Nr	4
Titel	Bugs
Beschreibung	Die Implementierung von komplexen Funktionen, sowie beispielsweise der Bearbeitung eines Projekts durch verschiedene Personen mit unterschiedlichen Rollen und Verantwortlichkeiten führt zu Bugs.
Vorbeugung	Genügend Zeit für das Ausarbeiten der Architektur einplanen. Eine hohe Robustheit der Geschäftslogik durch automatisierte Unit-Tests sicherstellen.

Tabelle 6.8: Risiko - Bugs

Nr	5
Titel	Usability
Beschreibung	Die Kernfunktionalität der Software (Suchen und Filtern nach Projekten) ist nicht intuitiv und mühsam zu bedienen.
Vorbeugung	Einholen von Feedback zur geplanten Benutzeroberfläche Testen der Benutzerfreundlichkeit des Endprodukts durch Usability-Tests

Tabelle 6.9: Risiko - Usability

Nr	6
Titel	Security
Beschreibung	Sicherheitslücken wie beispielsweise eine unzureichende Eingabevalidierung oder Probleme in der Benutzerverwaltung können dazu führen, dass die Anwendung anfällig für Angriffe (z.B. SQL-Injections) ist.
Vorbeugung	Durchführung manueller Sicherheitschecks für alle Endpoints mit SwaggerUI und für alle Routes im Webbrowser. Anwenden von Best Practices der sicheren Programmierung.

Tabelle 6.10: Risiko - Security

## 6.4 Zeiterfassung

Die Zeiterfassung erfolgt direkt in Jira anhand des Tools "Time Tracker". So kann die Arbeitszeit für jede in Jira erfasste Aufgabe direkt erfasst werden. Bei der Erfassung wird die getätigte Arbeit anhand unterschiedlicher Tags kategorisiert. Anhand dieser können Grafiken generiert werden. Die unten stehenden Grafiken visualisieren die erfassten Arbeitszeiten nach Teammitglied 6.2, nach Tätigkeitsbereich 6.3, sowie nach Teammitglied und Tätigkeitsbereich (siehe 6.4 und 6.5).

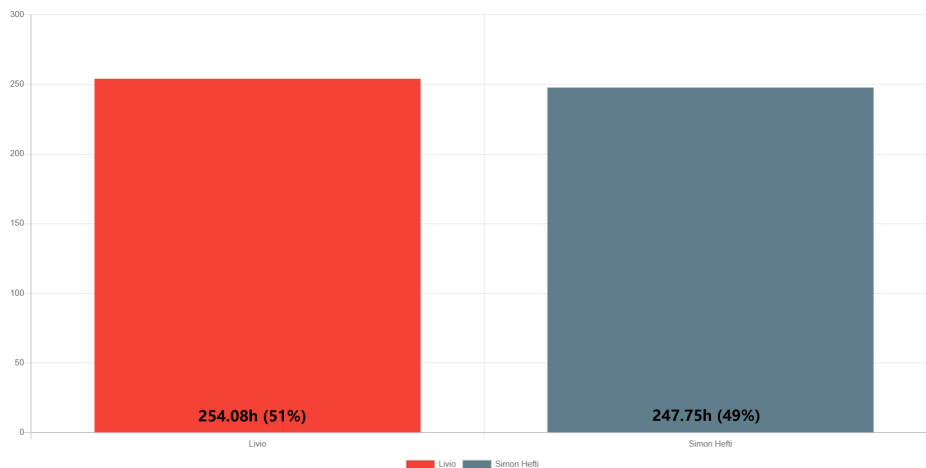


Abbildung 6.2: Aufgewendete Zeit pro Teammitglied



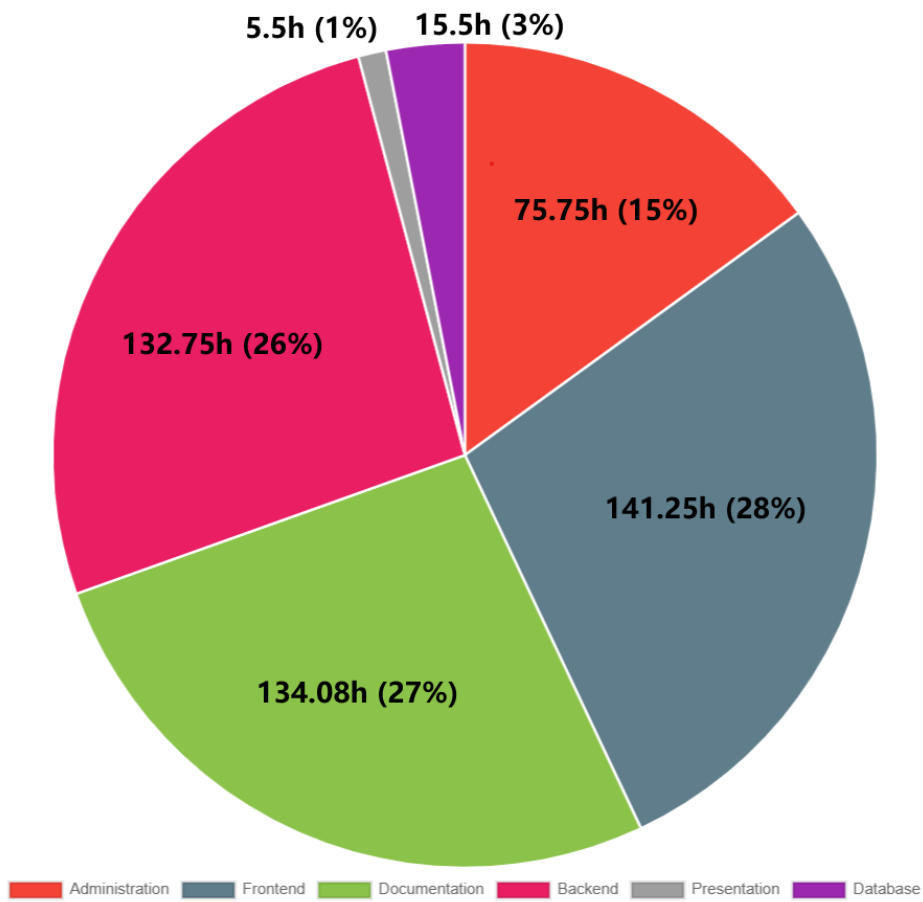


Abbildung 6.3: Aufgewendete Zeit nach Tätigkeitsbereich

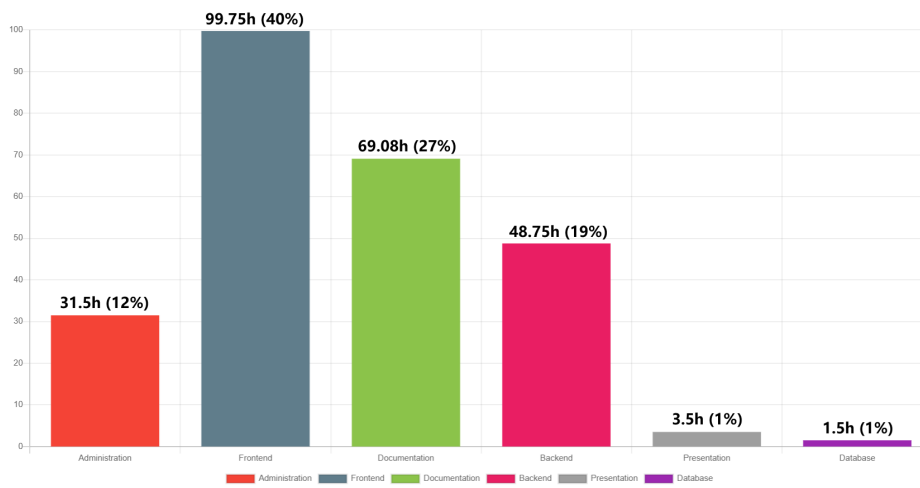


Abbildung 6.4: Aufgewendete Zeit nach Tätigkeitsbereich - Livio Mauchle

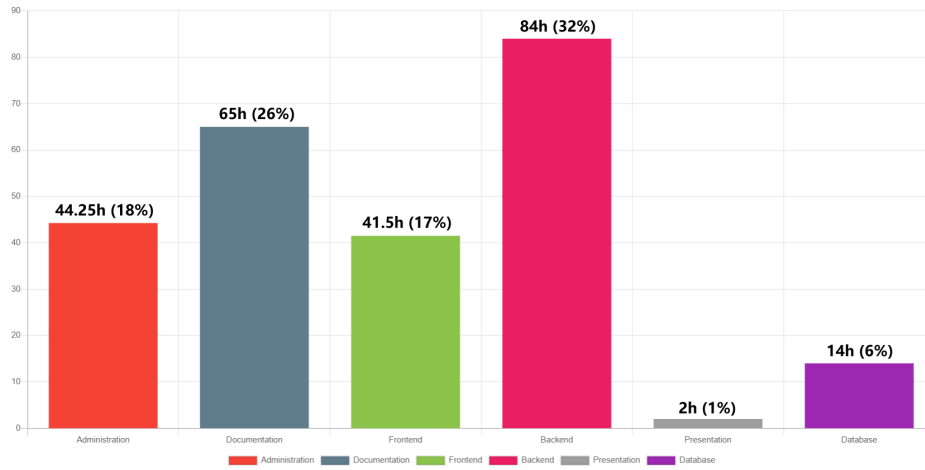


Abbildung 6.5: Aufgewendete Zeit nach Tätigkeitsbereich - Simon Hefti

# Kapitel 7

## Ergebnisse

### 7.1 Übersicht der Ergebnisse

Auf den nachfolgenden Seiten wird der finale Zustand der Anwendung anhand von Screenshots der Benutzeroberfläche aufgezeigt. Diese zeigen die wichtigsten Oberflächen und Funktionen die im Rahmend der Studienarbeit umgesetzt wurden.

Auf der Startseite bietet eine Suchfunktion mit der Benutzer anhand von Titel, Autor oder Abstract gezielt nach Forschungsprojekten suchen können (siehe Abbildung 7.1). Zusätzlich können Projekte anhand festgelegter Kriterien gefiltert werden. Diese Kombination unterstützt die Benutzer dabei, relevante Forschungsprojekte schnell und effizient zu finden.

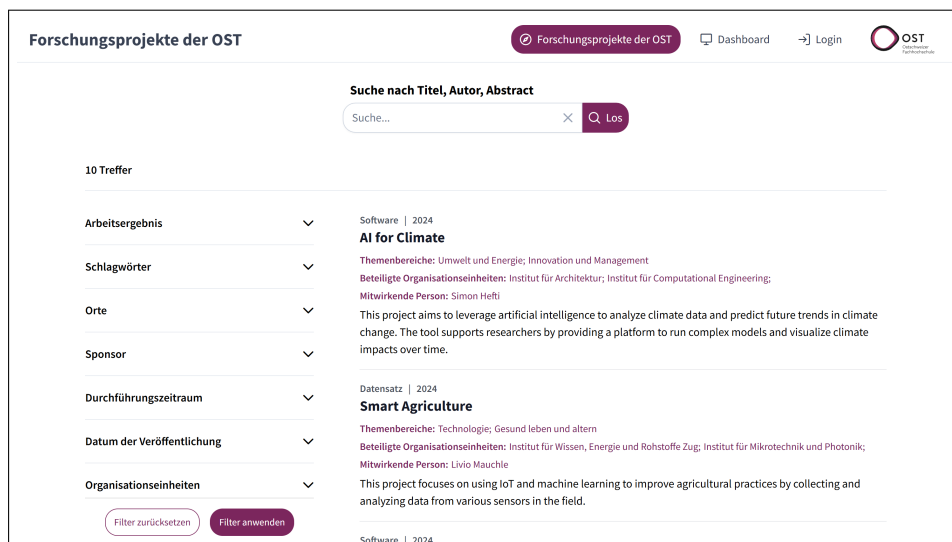


Abbildung 7.1: Screenshot Startseite

Die Abbildung 7.2 zeigt zwei Filterfunktionen dargestellt. Links zu sehen ist die Filterung nach Schlagwörtern. Diese ermöglicht eine thematische Eingrenzung der Projekte. Da eine grosse Anzahl an unterschiedlichen Schlagwörtern möglich ist, sind diese thematisch sortiert und werden als Baumstruktur dargestellt. Zudem sind diese durchsuchbar. Rechts ist die Filterung nach Arbeitsergebnissen zu sehen.

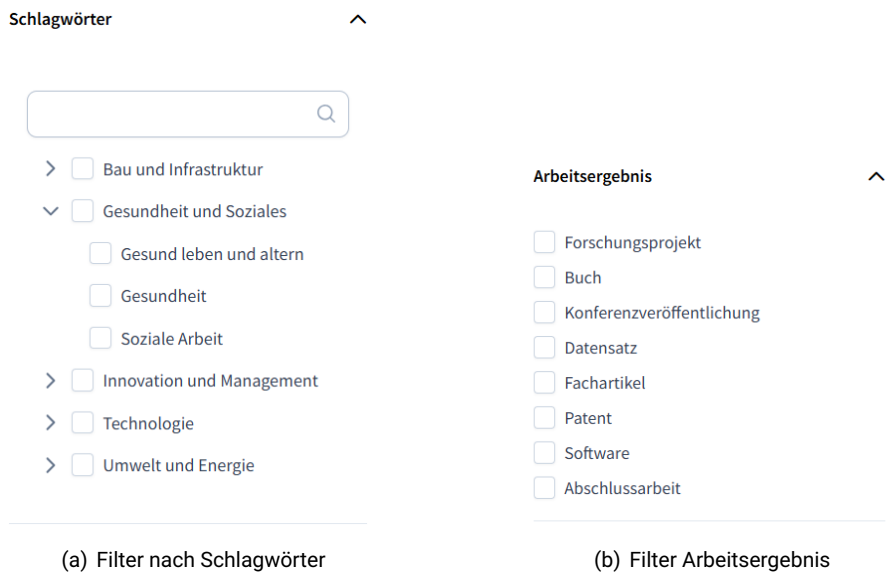


Abbildung 7.2: Screenshots von Filterfunktionen: Links die Filterung nach Schlagwörtern, rechts die Filterung nach Arbeitsergebnissen.

In Abbildung 7.3 ist Dashboard dargestellt, das einen Überblick über die wichtigsten Kennzahlen und Statistiken der Forschungsprojekte liefert.

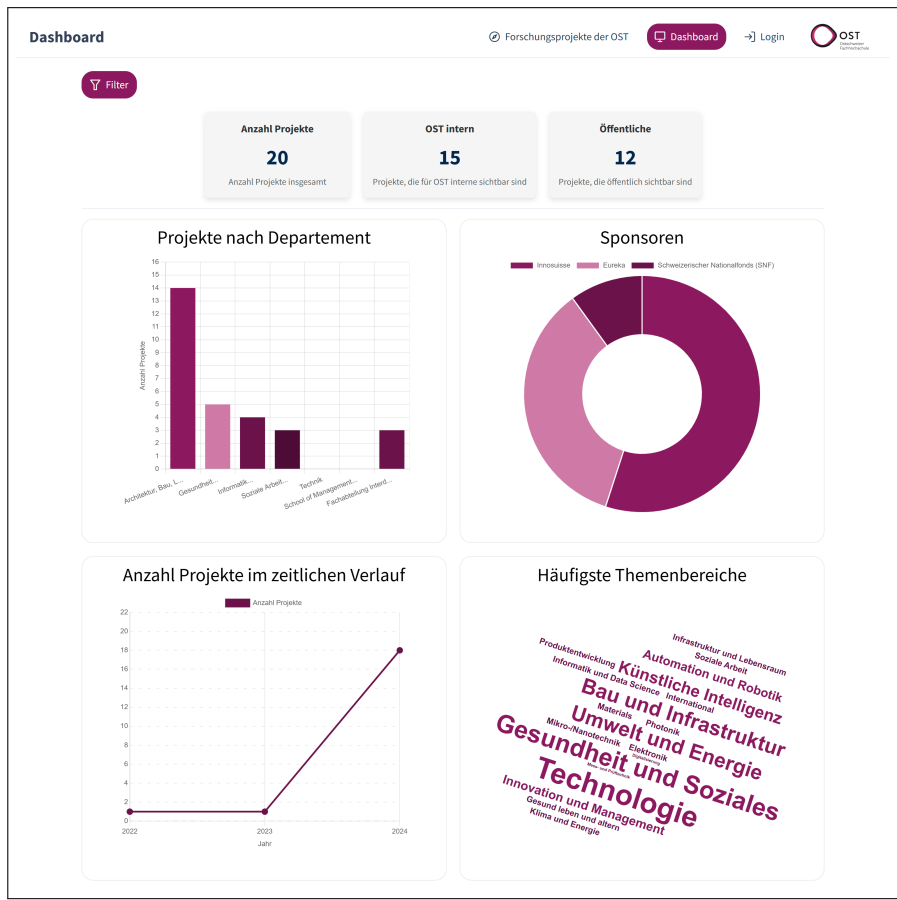


Abbildung 7.3: Screenshot Dashboard

Die in Abbildung 7.4 dargestellte Projektdetailseite zeigt alle relevanten Informationen eines Projekts. Um eine bessere Übersicht zu gewährleisten, ist die Seite in drei Bereiche unterteilt:

- **Oben:** Anzeige der wichtigsten Projektinformationen auf einen Blick. Falls vorhanden, wird auch das Titelbild des Projekts angezeigt. Dokumente, die vom Ersteller zur Verfügung gestellt wurden, können hier heruntergeladen werden.
- **Mitte:** Ein Bereich für die Zusammenfassung, Bilder (in einer Diashow) und ein Videoplayer für allfällige Videos.
- **Unten:** Auflistung aller verbleibenden Metadaten.

Zusätzlich bietet die Seite eine Seitennavigation, die bei umfangreichen Projekten (viel Text und Medien) hilfreich ist.

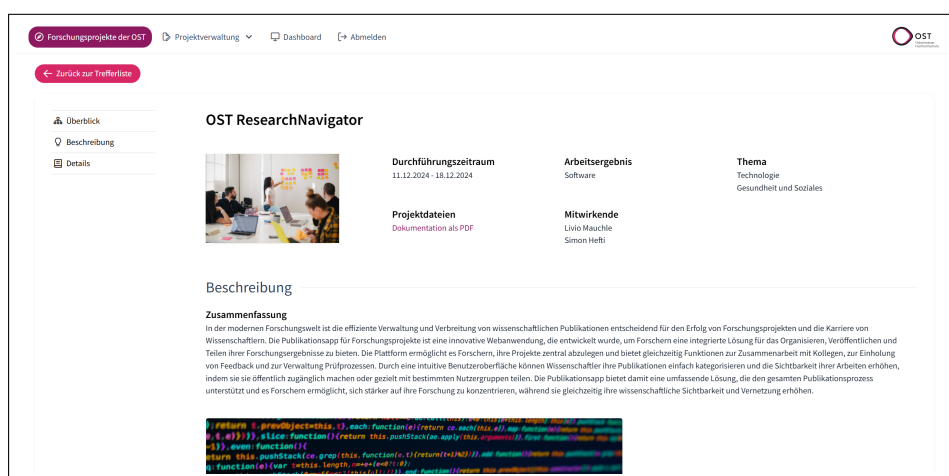


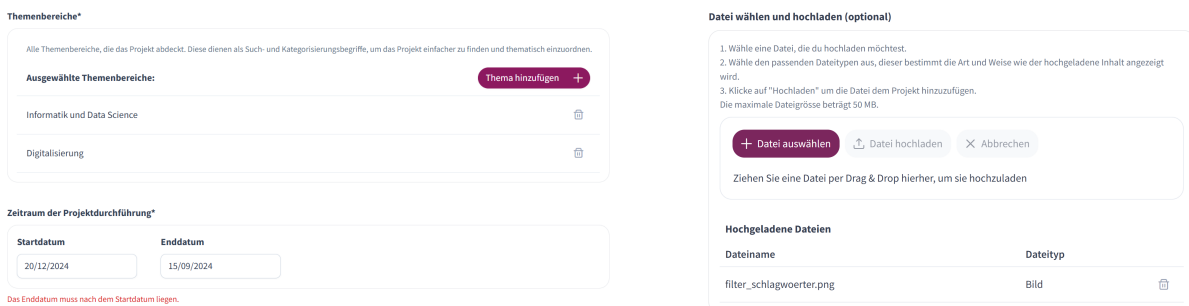
Abbildung 7.4: Screenshot der Projektdetailansicht

Auf der Seite "Meine Projekte" können berechtigte Benutzer Projekte anlegen und ihre erfassten Projekte bearbeiten und löschen, solange sich diese noch im Status "Draft" befinden. Auf der Projekterstellungsseite, wie in der Abbildung 7.5 dargestellt, können Projektdaten einheitlich erfasst werden. Die Seite ist in vier logische Abschnitte unterteilt. Während der Erfassung besteht jederzeit die Möglichkeit, den aktuellen Fortschritt zu speichern. Um das Formular zur Prüfung oder Veröffentlichung freizugeben, müssen alle erforderlichen Felder vollständig ausgefüllt sein.



Abbildung 7.5: Screenshot der Projekterfassungsseite

Auf Fehleingaben wird während der Erfassung sofort mit einer entsprechenden Meldung direkt unterhalb des betroffenen Feldes hingewiesen (siehe Abbildung 7.6). Darüber hinaus ermöglicht das Formular den Upload von Dateien mit einer maximalen Grösse von bis zu 50 Megabyte. Diese Begrenzung wird später an die tatsächlichen Speicherkapazitäten der Produktivumgebung angepasst.



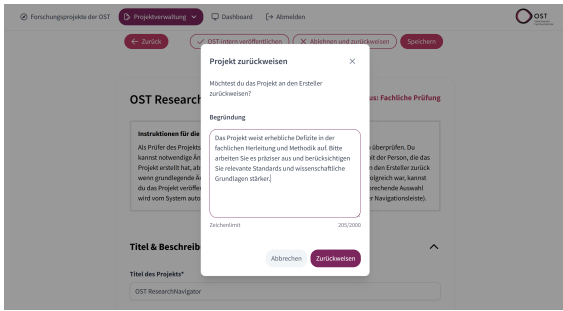
(a) Hervorhebung einer ungültigen Eingabe

(b) Möglichkeit für den Upload von Videos, Bildern und anderen Dokumenten

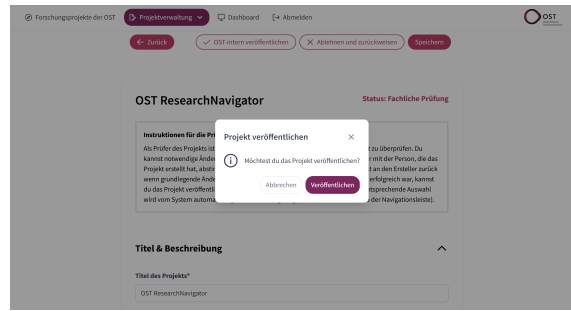
Abbildung 7.6: Screenshots ausgewählter Elemente der Projekterfassungsseite

Erfasste Projekte können anschliessend, je nach gewünschter Sichtbarkeit, wie unter 3.1.3 beschrieben, zur Prüfung eingereicht oder veröffentlicht werden. Die angezeigten Elemente in der Menüleiste passen sich den möglichen Optionen im Prozess an. Wenn ein Projekt zur Prüfung eingereicht wird, erscheint es in der Inbox der zuständigen Prüfperson. Diese kann das Projekt in der Bearbeitungsansicht öffnen und prüfen. Dort wird sie auch über ihre Aufgabe als Prüfperson informiert. In dieser Ansicht können Prüfer selbst Änderungen vornehmen und können darüber entscheiden, ob das Projekt den Anforderungen der Prüfung genügt, oder mit einer Begründung zurückzuweisen ist.

Die Abbildung 7.7 zeigt links, wie ein Projekt mit Angabe einer Nachricht abgelehnt werden kann. Rechts wird der Dialog gezeigt, mit dem die endgültige Veröffentlichung bestätigt werden muss.



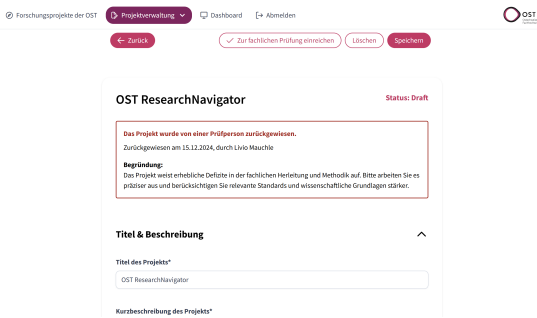
(a) Die Prüfperson kann ein Projekt mit Angabe einer Begründung zurückweisen



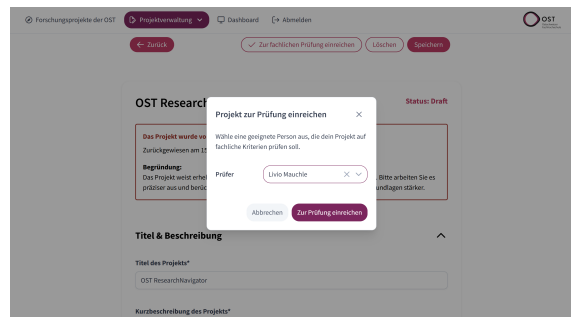
(b) Ist die letzte Prüfperson im Prozess mit den Inhalten zufrieden, kann Sie das Projekt veröffentlichen

Abbildung 7.7: Screenshots: Optionen der Prüfperson

In der Abbildung 7.8 sieht man links ein Projekt, das die Prüfung nicht bestanden hat und zurückgewiesen wurde. Es wird der Grund für die Ablehnung, sowie auch die Person angezeigt, die das Projekt abgelehnt hat. Rechts zu sehen ist die Wahl des fachlichen Prüfers. Der Ersteller kann auswählen welche Person am besten für diese Rolle geeignet ist.



(a) Ein von der Prüfperson zurückgewiesenes Projekt



(b) Wahl des fachlichen Prüfers

Abbildung 7.8: Screenshots: Einblick in den Prüfprozess

### 7.1.1 Statistiken zum Code

Um den Umfang des Projekts ungefähr zu ermitteln, wurden die Anzahl Files und Codezeilen ausgelesen. Die Methode zur Herleitung der Zeilen wird in Abschnitt 9.2.2 gezeigt. In Tabelle 7.1 ist die Anzahl der Codezeilen ohne Leerzeilen oder Kommentare aufgelistet:

Projekt	Sprache/Technologie	Files	Anzahl Codezeilen
Frontend	Vuejs Component	33	4621
Frontend	TypeScript	28	1047
<b>Frontend</b>	<b>Alle</b>	<b>61</b>	<b>5668</b>
Backend	C#	98	4033
Unit Tests	C#	13	2073
<b>Projekt Total</b>	<b>Alle</b>	<b>170</b>	<b>11774</b>

Tabelle 7.1: Auflistung der Anzahl Codezeilen

Die **Branch Coverage** der Backend-Klassen, welche die Geschäftslogik der Anwendung implementieren, liegt bei über 80%, wie in Abbildung 7.9 zu sehen ist. Dabei wird auch auf die Relevanz der einzelnen Funktionalitäten geachtet. So wird für den AccessControlService, der sicherstellt, dass ein Benutzer im aktuellen Zustand des Projekts berechtigt ist, um Änderungen vorzunehmen, eine Testabdeckung von 100% erreicht.

Name	Covered	Uncovered	Coverable	Total	Percentage	Covered	Total	Percentage
Source.Services.Abstractions.SearchService	52	10	62	94	83.8%	12	14	85.7%
Source.Services.AccessControlService	36	0	36	64	100%	36	36	100%
Source.Services.AccountService	72	0	72	116	100%	13	14	92.8%
Source.Services.FileService	60	4	64	104	93.7%	19	22	86.3%
Source.Services.InstituteService	8	0	8	21	100%	0	0	
Source.Services.ProjectService	337	21	358	504	94.1%	69	82	84.1%
Source.Services.ValidationService	137	5	142	212	96.4%	44	54	81.4%

Abbildung 7.9: Testabdeckung der entwickelten Backend-Services

### 7.1.2 Zielerreichung

Von den insgesamt 16 definierten Use Cases gehören 5 zur Kategorie **MVP**. Alle Use Cases dieser Kategorie konnten erfüllt werden, womit die funktionalen Anforderungen an die Anwendung erfüllt sind. Darüber hinaus wurde die Zeit genutzt, um zwei der drei optional definierten Features der Kategorie **Could Have** zu implementieren.

- UC01 - Projekt suchen (**MVP**)
- UC02 - Projekte filtern (**MVP**)
- UC03 - Projekt ansehen (**MVP**)
- UC04 - Dashboard einsehen (**MVP**)
- UC05 - Projekt CUD (**MVP**)
- UC06 - grosse Medien einbinden (**MVP**)



- UC08 - Projekt prüfen (MVP)

In gewissen Bereichen sieht das Entwicklerteam aber auch noch Verbesserungspotential. Beispielsweise im Use Case 08 - Projekte prüfen 2.3. Derzeit fehlt ein visueller Indikator, mit dem eine Prüfperson bereits von der Startseite aus sehen kann, dass Projekte zur Prüfung anstehen.

Für UC01 - Projekte suchen 2.3 sind noch diverse Suchoptimierungen denkbar, die aber den Rahmen dieser Studienarbeit überstiegen hätten. So wäre eine Erweiterung der Suche um diejenigen Attribute denkbar, nach denen aktuell nur gefiltert werden kann. Interessant wäre zudem auch eine gewichtete Suche. Man könnte Inhalte nach gewissen Kriterien priorisieren, beispielsweise Projekte die einen Suchbegriff häufig beinhalten in der Resultatliste weiter vorne darstellen.

Auch für den Use Case UC04 - Dashboard einsehen 2.6 sind Erweiterungen denkbar. Beispielsweise könnten personalisierte Dashboards je nach Rolle oder Benutzer angezeigt werden. Darüber hinaus wäre eine Erweiterung um zusätzliche Statistiken denkbar.

Die erforderlichen NFR 1-7 wurden vollständig erfüllt. Sie beinhalten Anforderungen zu den folgenden Qualitätscharakteristiken:

- Performance Efficiency (Response Time)
- Compatibility (Browser)
- Usability (Affordance)
- Security (Autorisierung)
- Maintainability (Branch Coverage)
- Transferability (Docker Container)

Teilweise verzögerte sich die Verifizierung aufgrund fehlender Features um einige Tage (z.B. Performance File-Upload, Tabelle 2.20) oder musste nach Verbesserungen erneut verifiziert werden, um den Status 'Vollständig erfüllt' zu erreichen (z.B. Usability, Tabelle 2.22).

Nicht umgesetzt wurde das optionale NFR-8 (Backups, Tabelle 2.26), welches dem Qualitätskriterium Reliability angehört.

Detaillierte Beschreibungen zu den NFR und deren Verifizierung befinden sich im Abschnitt 2.2.

# Kapitel 8

## Ergebnisdiskussion

### 8.1 Fazit

Das Ziel dieser Arbeit war die Erstellung eines MVP für eine neue Lösung zur einheitlichen Erfassung und Übersicht aller Forschungsprojekte der OST.

Der entwickelte MVP erfüllt die funktionalen und die nicht-funktionalen Anforderungen. Alle Must-Have Features sowie auch zwei der drei Could-Have Features konnten implementiert werden. Alle erforderlichen Qualitätskriterien wurden vollständig erfüllt.

Die Projektsuche funktioniert wie gewünscht und ist gemäss Usability Tests intuitiv bedienbar. Mittels Textsuche oder den diversen Filtereinstellungen lassen sich die angezeigten Projekte nach allen vom Auftraggeber gewünschten Kriterien eingrenzen.

Die Projekterstellung mit Erfassung aller Metadaten und der anschliessende Prüfprozess erfüllt ebenfalls die Anforderungen und wurde durch Usability Tests sowie auch System Tests verifiziert.

Das Dashboard ermöglicht nicht nur externen Nutzern einen transparenten Einblick in die Forschungsaktivitäten der Hochschule, sondern dient auch der Hochschulleitung als wertvolles Instrument zur Analyse aktueller Trends. Durch die integrierten Filteroptionen können die dargestellten Statistiken gezielt angepasst werden, was den Nutzen deutlich erhöht.

In der abschliessenden Produktpräsentation zeigten sich zwei der wichtigsten Stakeholders, Prof. Alex Simeon und Prof. Laurent Metzger, sehr beeindruckt von dem vorgestellten Ergebnis. Die Arbeit habe die Erwartungen übertroffen, welche Prof. Laurent Metzger vor Projektbeginn hatte. Die Vision sei verwirklicht worden und auf dieser Arbeit könne aufgebaut und weitergearbeitet werden.

Aufgrund dieser Auswertung und insbesondere dem positiven Feedback des Auftraggebers sowie dem Interesse an einer Folgearbeit lässt sich die Arbeit als Erfolg werten. Die Ziele wurden erreicht und teilweise übertroffen, besonders im funktionalen Bereich und die Vision wurde umgesetzt.

## 8.2 Ausblick

### 8.2.1 Erweiterungen

Zu Beginn des Projektes wurden mit den Use Cases 9-16 einige Features definiert, welche nicht Teil dieser Arbeit waren. Ausserdem verbleibt noch ein nicht implementiertes Could-Have-Feature. Während der Entwicklung kamen zudem diverse weitere Ideen für zusätzliche Features auf.

Ein wichtiger Aspekt sowohl in funktionaler als auch in nicht-funktionaler Sichtweise ist die Authentifizierung der Benutzer. Diese soll wenn möglich über die Microsoft Accounts abgewickelt werden.

Besonders hervorgehoben wurde auch die Wichtigkeit eines Administrator-Tools, in welchem die Filterkriterien wie z.B. Suchtags und Regionen, die Organisationseinheiten sowie auch die Sponsoren erfasst und bearbeitet werden können.

Auch eine Anbindung an Umsysteme wie dem SAP ist denkbar. Damit könnten Projektdaten direkt abgefüllt statt manuell eingetragen werden. Dieses Ziel wäre auch mithilfe eines LLM realisierbar, welches Dokumente einliest und daraus die relevanten Projektdaten extrahiert.

Für die Optimierung der Projektsuche stehen ebenfalls noch Möglichkeiten zur Verbesserung oder Erweiterung im Raum. Für die Suche im Text kann beispielsweise mit ElasticSearch und Fuzzy-Matching eine gewisse Toleranz für Tippfehler erlaubt werden. Auch die Volltext-Suche in den angehängten Dokumenten wäre eine interessante Option.

### 8.2.2 Verbesserungsmöglichkeiten

Dank Usability Tests, Code Reviews und dem kontinuierlichen Lernfortschritt des Entwicklungsteams konnten im Verlauf des Projektes einige Verbesserungsmöglichkeiten im UI und Code festgestellt werden. Viele davon konnten bereits umgesetzt werden, einige waren jedoch aus zeitlichen Gründen nicht mehr realisierbar.

Die technischen Schulden sind weiter beschrieben unter [3.3](#). Offene Verbesserungsmöglichkeiten im Bereich der Usability sind im Anhang unter [9.1.1](#) zu finden.

## 8.3 Weiterentwicklung

Der OST ResearchNavigator wird in einer anschliessenden Bachelorarbeit vom gleichen Projektteam weiterentwickelt. Dabei sollen sowohl technische Mängel behoben als auch neue Features implementiert werden. Zu den neuen Features gehören mit Sicherheit die Authentifizierung und die Administrationsoberfläche. Die genauen Projektspezifikationen müssen jedoch noch ausgearbeitet werden.

# Glossar

**Branch Coverage** Ein Mass für die Qualität von Softwaretests, welches angibt, wie viele der möglichen Verzweigungen (z.B. in If-Else-Abzweigungen) innerhalb des Codes durch Tests ausgeführt wurden. Eine hohe Branch Coverage deutet darauf hin, dass die Tests viele verschiedene Entscheidungspfade im Code abgedeckt haben, was die Wahrscheinlichkeit von Fehlerhafter Logik verringert. . 20, 63, 79

**Fully Dressed Format** Das Fully Dressed Format ist eine detaillierte und strukturierte Vorlage zur Beschreibung von Use Cases in der Softwareentwicklung. Es enthält spezifische Abschnitte wie Akteure, Vorbedingungen, Hauptszenario und Erweiterungen, um ein umfassendes Verständnis des Use Cases zu ermöglichen. Im Vergleich zu den kürzeren Formaten "Brief" und "Casual", die weniger ausführlich sind, bietet das Fully Dressed Format eine tiefere Analyse und Struktur. 7

**Prerendering** Prerendering ist ein Prozess, bei dem Seiteninhalte statisch vorgeneriert werden, damit sie dem Benutzer schneller als vollständig gerenderte HTML-Dateien zur Verfügung gestellt werden können. . 41

**Rational Unified Process (RUP)** Ein iteratives und inkrementelles Vorgehensmodell für Softwareentwicklung, das aus den Phasen Inception, Elaboration, Construction und Transition besteht, mit besonderem Fokus auf eine langfristige Planung und die schrittweise Fertigstellung des Projekts. II, 66

# Akronyme

**API** Application Programming Interface. 65

**CSS** Cascading Style Sheets. 107

**DI** Dependency Injection. 54

**DTO** Data Transfer Object. 42

**FR** Functional Requirements. 17

**HTML** Hypertext Markup Language. 65

**HTTP** Hypertext Transfer Protocol. 37

**JWT** Json Web Token. 54

**MVP** Minimum Viable Product. II, V, 79, 80, 81

**NFR** Non-functional Requirements. 17, 80

**SSR** Server Side Rendering. 41

**XML** Extensible Markup Language. 65

# Kapitel 9

## Anhang

### 9.1 Tests und Qualität

#### 9.1.1 Usability Tests

In diesem Abschnitt werden die erstellten Usability-Tests, sowie das jeweilig erwartete Resultat beschrieben. Zudem werden weiter unten auch Ergebnisse der durchgeführten Tests dokumentiert. Für alle Aufgaben gilt als Vorbedingung, dass der Benutzer eingeloggt ist und die für den Test erforderlichen Berechtigungen hat.

Nr	Aufgabe	Erwartetes Resultat
1	Ein Kollege "Simon Hefti" hat Ihnen einen Link geteilt, der Sie zur Startseite der zu testenden Webapplikation weiterleitet. Er meinte, dass hier auch von ihm Projekte zu finden sind. Finden Sie Projekte ihres Kollegen "Simon Hefti".	<ol style="list-style-type: none"> <li>1. Der Benutzer findet das Suchfeld</li> <li>2. Der Benutzer gibt den gewünschten Suchbegriff ein</li> <li>3. Der Benutzer löst eine Suche aus durch das drücken der Enter Taste oder einem click auf den Button "Los"</li> </ol>
2	Als begeisterter Softwareentwickler fragen Sie sich, ob auch interessante Forschungsprojekte in diesem Bereich veröffentlicht wurden. Finden Sie heraus welche Projekte es im Bereich der Softwareentwicklung gibt.	<ol style="list-style-type: none"> <li>1. Der Benutzer bemerkt die Filterleiste</li> <li>2. Der Benutzer öffnet den Filter "Werke"</li> <li>3. Der Benutzer wählt das Kriterium "Software"</li> <li>4. Der Benutzer löst eine Suche aus (klickt auf "Suche aktualisieren")</li> </ol>
3	Auf der Reise im Zug sehen Sie eine Nau.ch Werbeanzeige zum Thema Künstliche Intelligenz. Sie fragen sich welche Projekte die OST wohl zu diesem Thema hat. Finden Sie auf dem Handybildschirm (Fenster verkleinert) Projekte im Bereich der Künstlichen Intelligenz.	<ol style="list-style-type: none"> <li>1. Der Benutzer merkt das er die Filter anhand des Buttons "Filter" öffnen kann.</li> <li>2. Der Benutzer öffnet den Filter "Schlagwörter"</li> <li>3. Der Benutzer findet und wählt das Kriterium "Künstliche Intelligenz"</li> <li>4. Der Benutzer löst eine Suche aus (klickt auf "Suche aktualisieren")</li> </ol>
4	Das Projekt "Research Navigator" hat ihr Interesse geweckt. Öffnen Sie das Projekt und laden Sie die Projektdokumentation herunter.	<ol style="list-style-type: none"> <li>1. Der Benutzer findet das Projekt anhand einer Suchabfrage</li> <li>2. Der Benutzer öffnet das Projekt anhand eines Klicks auf die entsprechende Kachel</li> <li>3. Auf der Projektseite angekommen findet er den Downloadlink der Projektdokumentation</li> <li>4. Der Benutzer clickt auf den Link um die Dokumentation herunterzuladen</li> </ol>

Tabelle 9.1: Usability-Tests

Nr	Aufgabe	Erwartetes Resultat
5	Finden Sie heraus an welchem Datum das Projekt veröffentlicht wurde.	<p>Vorbedingung: Der Benutzer befindet sich auf der Detailseite eines Projekts.</p> <ol style="list-style-type: none"> <li>1. Der Benutzer navigiert zum Menübereich "Details" entweder durch Herunterscrollen oder durch das eingebaute Navigationsmenue</li> <li>2. Der Benutzer findet die gesuchte Information und "Veröffentlichungsdatum"</li> </ol>
6	Soeben haben Sie und ihr Team ein wichtiges Projekt abgeschlossen. Nun ist es Ihre Aufgabe dieses im Katalog der Forschungsprojekte der OST zu ergänzen. Legen Sie ein neues Projekt an.	<ol style="list-style-type: none"> <li>1. Der Benutzer navigiert anhand des Menus zum Menüpunkt "Meine Projekte"</li> <li>2. Auf der geöffneten Seite findet der Besucher den Button "Neues Projekt anlegen" und wählt diesen an</li> </ol>
7	Für die Erfassung haben Sie aktuell leider nur wenig Zeit. Erfassen Sie nur den Titel und die Sichtbarkeit, die ihr Forschungsprojekt haben soll und kehren Sie anschliessend zur Übersicht ihrer Projekte zurück.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Der Benutzer findet die Felder zur Erfassung des Titels und der Sichtbarkeit und setzt diese.</li> <li>2. Der Benutzer drückt auf Speichern oder navigiert mit direkt mit dem "Zurück"-Button zu seine Inbox zurück.</li> </ol>
8	Über den Mittag haben Sie kurz Zeit um die wichtigsten Informationen über ihr Forschungsprojekt zu erfassen. Befüllen Sie alle Pflichtfelder und sorgen Sie anschliessend dafür dass das Projekt auch veröffentlicht wird.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Der Benutzer bemerkt, dass Pflichtfelder mit "*" markiert sind und befüllt diese Felder mit Informationen.</li> <li>2. Der Benutzer findet den korrekten Button um die Projekterfassung abzuschliessen und dieses für die fachliche Prüfung einzureichen.</li> </ol>
9	Es wundert Sie zu welchen Themenbereichen es wohl die meisten Forschungsprojekte gibt. Finden Sie heraus welches die 3 beliebtesten Forschungsthemen sind.	<ol style="list-style-type: none"> <li>1. Der Benutzer findet die Dashboardseite, welche Grafiken und Statistiken über die erfassten Forschungsprojekte enthält.</li> <li>2. Der Benutze findet die Grafik der häufigsten Themen und kann daraus die 3 häufigsten Themen ablesen</li> </ol>

Tabelle 9.2: Usability-Tests



**Usability Tests mit Dr. Frieder Loch**

Nr	Aufgabe	Resultat
1	Ein Kollege "Simon Hefti" hat Ihnen einen Link geteilt, der Sie zur Startseite der zu testenden Webapplikation weiterleitet. Er meinte, dass hier auch von Ihm Projekte zu finden sind. Finden Sie Projekte ihres Kollegen "Simon Hefti".	<ol style="list-style-type: none"> <li>1. Benutzer versucht Browsersuche</li> <li>2. Benutzer verwendet das Suchfeld der Webseite und findet die Projekte</li> </ol>
2	Als begeisterter Softwareentwickler fragen Sie sich ob auch Interessante Forschungsprojekte in diesem Bereich veröffentlicht wurden. Finden Sie heraus welche Projekte es im Bereich der Softwareentwicklung gibt.	<ol style="list-style-type: none"> <li>1. Im ersten Versuch wurde in den Tags gesucht</li> <li>2. Suche in der Suchleiste nach "Software"</li> <li>3. Die Filterkategorie "Werke" wurde vom Benutzer nicht wie gewünscht interpretiert</li> </ol>
3	Auf der Reise im Zug sehen Sie eine Nau.ch Werbeanzeige zum Thema Künstliche Intelligenz. Sie fragen sich welche Projekte die OST wohl zu diesem Thema hat. Finden Sie auf dem Handybildschirm (Fenster verkleinert) Projekte im Bereich der Künstlichen Intelligenz.	<ol style="list-style-type: none"> <li>1. Der Benutzer findet das Filterkriterium "Tags" (bereits Bekannt aus vorheriger Aufgabenstellung)</li> <li>2. Der Benutzer kann den Filter erfolgreich anwenden</li> </ol>
4	Das Projekt "Research Navigator" hat ihr Interesse geweckt. Öffnen Sie das Projekt und laden Sie die Projektdokumentation herunter.	<ol style="list-style-type: none"> <li>1. Der Benutzer findet das Projekt</li> <li>2. Der Benutzer klickt auf das Projekt, es öffnet sich</li> </ol>
5	Finden Sie heraus an welchem Datum das Projekt veröffentlicht wurde.	<p>Vorbedingung: Der Benutzer befindet sich auf der Detailseite eines Projekts.</p> <ol style="list-style-type: none"> <li>1. Der Benutzer schaut sich erst kurz im oberen Bereich um</li> <li>2. Der Benutzer wechselt nach kurzer Zeit in den unteren Bereich der Projektdetailseite und findet dort in kürze die gesuchte Information</li> </ol>

Tabelle 9.3: Usability-Tests mit Dr. Frieder Loch

Nr	Aufgabe	Resultat
6	Soeben haben Sie und ihr Team ein wichtiges Projekt abgeschlossen. Nun ist es Ihre Aufgabe dieses im Katalog der Forschungsprojekte der OST zu ergänzen. Legen Sie ein neues Projekt an.	<ol style="list-style-type: none"> <li>1. Benutzer sucht in der Navigationsleiste nach Schaltfläche für Projekterfassung -&gt; findet dort nichts</li> <li>2. Benutzer navigiert anschliessend auf Seite "Meine Projekte"</li> <li>3. Auf dieser Seite wird die gesuchte Schaltfläche für die Projekterstellung sofort gefunden.</li> </ol>
7	Für die Erfassung haben Sie aktuell leider nur wenig Zeit. Erfassen Sie nur den Titel und die Sichtbarkeit, die ihr Forschungsprojekt haben soll, und kehren Sie anschliessend zur Übersicht ihrer Projekte zurück.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Felder werden intuitiv korrekt bedient</li> <li>2. Die Schaltfläche Speichern ist schnell gefunden</li> </ol>
8	Über den Mittag haben Sie kurz Zeit um die wichtigsten Informationen über ihr Forschungsprojekt zu erfassen. Befüllen Sie alle Pflichtfelder und sorgen Sie anschliessend dafür dass das Projekt auch veröffentlicht wird.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Die verbleibenden Elemente zur Erfassung der Projektdaten werden auf Anhieb korrekt bedient.</li> </ol>
9	Es wundert Sie zu welchen Themenbereichen es wohl die meisten Forschungsprojekte gibt. Finden Sie heraus welches die 3 beliebtesten Forschungsthemen sind.	<ol style="list-style-type: none"> <li>1. Wechselt erst zu Projektsuche und versucht dort Anhand einer Suche das Ergebnis herauszufinden</li> <li>2. Mit Hinweis auf Navigationsleiste wird das Dashboard gefunden</li> <li>3. Die 3 beliebtesten Forschungsthemen werden in der Grafik sofort erkannt</li> </ol>

Tabelle 9.4: Usability-Tests mit Dr. Frieder Loch

**Usability Tests mit Dr. Markus Stolze**

Nr.	Aufgabe	Resultat
1	Ein Kollege "Simon Hefti" hat Ihnen einen Link geteilt, der Sie zur Startseite der zu testenden Webapplikation weiterleitet. Er meinte, dass hier auch von Ihm Projekte zu finden sind. Finden Sie Projekte ihres Kollegen "Simon Hefti".	<ol style="list-style-type: none"> <li>1. Direkter Versuch über die Suchleiste</li> <li>2. Mit richtigem Case wird das Projekt gefunden</li> </ol>
2	Als begeisterter Softwareentwickler fragen Sie sich ob auch Interessante Forschungsprojekte in diesem Bereich veröffentlicht wurden. Finden Sie heraus welche Projekte es im Bereich der Softwareentwicklung gibt.	<ol style="list-style-type: none"> <li>1. Sucht erst über "Tags"</li> <li>2. Nach weiterer Suche wird korrektes Filterkriterium gefunden</li> <li>3. Sieht den Typ "Softwareentwicklung" und wendet den Filter erfolgreich an</li> </ol>
3	Auf der Reise im Zug sehen Sie eine Werbung Werbeanzeige zum Thema Künstliche Intelligenz. Sie fragen sich welche Projekte die OST wohl zu diesem Thema hat. Finden Sie auf dem Handybildschirm (Fenster verkleinert) Projekte im Bereich der Künstlichen Intelligenz.	<ol style="list-style-type: none"> <li>1. Findet den Filter im Mobile Modus</li> <li>2. Setzt die aktuellen Filterkriterien erfolgreich zurück</li> <li>3. Sucht in den "Tags"</li> <li>4. Findet dort das Korrekte Schlagwort und wendet den Filter an</li> </ol>
4	Das Projekt "Research Navigator" hat ihr Interesse geweckt. Öffnen Sie das Projekt und laden Sie die Projektdokumentation herunter.	<ol style="list-style-type: none"> <li>1. Findet Projekt über die Suchleiste</li> <li>2. Klick auf das Projekt</li> <li>3. Findet und öffnet das PDF-Dokument erfolgreich</li> </ol>
5	Finden Sie heraus an welchem Datum das Projekt veröffentlicht wurde.	<p>Vorbedingung: Der Benutzer befindet sich auf der Detailseite eines Projekts.</p> <ol style="list-style-type: none"> <li>1. Schaut erst auf das Ausführungsdatum</li> <li>2. Findet im unteren Bereich das Veröffentlichungsdatum</li> </ol>

Tabelle 9.5: Usability-Tests mit Dr. Markus Stolze

Nr	Aufgabe	Resultat
6	Soeben haben Sie und ihr Team ein wichtiges Projekt abgeschlossen. Nun ist es Ihre Aufgabe, dieses im Katalog der Forschungsprojekte der OST zu ergänzen. Legen Sie ein neues Projekt an.	<ol style="list-style-type: none"> <li>1. Sucht im Header nach passendem Navigationsselement</li> <li>2. Navigiert mittels dem Menüpunkt "Meine Projekte" zur Projektseite</li> <li>3. Findet die Schaltfläche "Neues Projekt anlegen"</li> </ol>
7	Für die Erfassung haben Sie aktuell leider nur wenig Zeit. Erfassen Sie nur den Titel und die Sichtbarkeit, die ihr Forschungsprojekt haben soll und kehren Sie anschliessend zur Übersicht ihrer Projekte zurück.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Erfasst die Informationen in den gewünschten Felder</li> <li>2. Verwendet für die Navigation zurück zur Seite "Meine Projekte" die Browsernavigation (Zurückbutton wird intuitiv nicht verwendet)</li> </ol>
8	Über den Mittag haben Sie kurz Zeit um die wichtigsten Informationen über ihr Forschungsprojekt zu erfassen. Befüllen Sie alle Pflichtfelder und sorgen Sie anschliessend dafür dass das Projekt auch veröffentlicht wird.	<p>Vorbedingung: Benutzer befindet sich auf der Projekterfassungs-Seite.</p> <ol style="list-style-type: none"> <li>1. Dem Benutzer ist nicht sofort klar welche Felder Pflichtfelder sind.</li> <li>2. Felder werden korrekt erfasst</li> </ol>
9	Es wundert Sie zu welchen Themenbereichen es wohl die meisten Forschungsprojekte gibt. Finden Sie heraus welches die 3 beliebtesten Forschungsthemen sind.	<ol style="list-style-type: none"> <li>1. Benutzer findet den Eintrag "Dashboard" in der Navigationsleiste</li> <li>2. Der Benutzer sieht sich gleich die Wordcloud an und kann die Frage beantworten</li> </ol>

Tabelle 9.6: Usability-Tests mit Dr. Markus Stolze

### Aus Usability Tests resultierende Verbesserungen

Mangel	Verbesserung
Auf der Seite der Projektsuche ist der Hover-Effekt zu schwach und wird daher zu wenig wahrgenommen.	Anstelle des Hover-Effekts, wird der Titel des Projekts durch eine markante, schwarze Linie unterstrichen, wenn sich der Mauszeiger über das Projekt bewegt.
Es fehlt die Möglichkeit, den in der Suche eingegebenen Text leicht zurückzusetzen.	Die Suchleiste beinhaltet neu eine Löschen-Icon, mit dem der Suchtext gelöscht werden kann.
Bei der Suche nach Projekten des Typs Software wird der dafür benötigte Filter von den Benutzern nicht gefunden.	Umbenennung des Filtes von "Werke" zu "Arbeitsergebnis".
Der Name der Schaltfläche "Suche aktualisieren", welche die gesetzten Filter auf die aktuelle Suche anwendet, wird als unpassend wahrgenommen.	Umbenennung der Schaltfläche zu "Filter anwenden"
Die Breite der Filterleiste ändert sich beim Aufklappen von Filteroptionen, die mehr Platz benötigen als die Standardbreite der Filterleiste. Dies wirkt unruhig.	Fixierung der Breite der Filterleiste. Platzsparende Einbindung der Filteroptionen.
Das Titelbild auf der Projektdetailseite, wirkt in seiner derzeitigen Position direkt überhalb der Navigationsleiste unpassend.	Das Bild wird neu einheitlich mit den restlichen Inhalten (weiter rechts) dargestellt. Die Navigationsleiste ist nach oben gewandert.
Die Elemente der Navigationsleiste haben zu wenig Struktur.	Zwischen Navigationselementen wird eine Horizontale Linie eingefügt. Zudem erhalten die Navigationselemente je ein Icon.
Die Navigationsfelder im Header sind zu wenig deutlich hervorgehoben.	Die Navigationsfelder werden durch die Primärfarbe hervorgehoben und sind dadurch deutlich sichtbar.
Die Schaltfläche "Neue Projekte anlegen" wird intuitiv in der Navigationsleiste gesucht, kann dort aber nicht gefunden werden.	Zusätzlich zu ihrer aktuellen Position auf der Seite "Meine Projekte" wird die Schaltfläche auch direkt in der Navigationsleiste angezeigt.
Das verwendete Löschsymbol "-" ist unüblich. Besser ist es, das Mülleimersymbol zu verwenden.	Für das Löschen wird das Mülleimersymbol verwendet.
Auf der Projektdetailseite kann die Bezeichnung des Zurück-Buttons verwirrend sein.	Statt des Namens "Zurück zur Trefferliste" wird nur noch "Zurück" verwendet, da der Begriff Trefferliste leicht mit der Startseite assoziiert werden kann.
Dem Benutzer ist nicht klar, welche Felder Pflichtfelder sind.	Statt nur die optionalen Felder mit (optional) zu kennzeichnen, sind nun auch die obligatorischen Felder mit einem Stern gekennzeichnet.

Tabelle 9.7: Verbesserungen durch Usability Tests

### **Offene Punkte aus Usability Tests**

Hier werden alle in den Usability Tests erwähnten Vorschläge aufgelistet, die nicht umgesetzt sind. Diese sind auch im Backlog des Entwicklungsteams notiert und sollen im Rahmen der Weiterentwicklung umgesetzt werden.

- Aktive Filter sind im zugeklappten Zustand des Filters nicht sichtbar. Diese könnte man als Tags unterhalb des Suchfeldes oder über Badges anzeigen.
- Feedback nach Suche ist zu wenig präsent.
- Beim Klick auf " Filter zurücksetzen " ist der resultierende Effekt für den Benutzer nicht klar sichtbar.
- Radiobuttons wirken unschön.
- Die Regionen in der Projekterfassungsseite sind ungeordnet.
- Die Titel der einzelnen Abschnitte in der Erfassungsseite haben bei der Anwendung des Hover-Effekts zu wenig Abstand nach links.
- Organisationseinheiten sind bei der Erfassung in beliebiger Reihenfolge sortiert.
- Der Hover-Effekt der Buttons dürfte dunkler sein.

### 9.1.2 Massnahmen aus Code Review

Nr	Frage / Mangel	Lösungsansatz / Verbesserung
1	Filterkriterien werden aktuell nicht in URL übertragen. Dadurch kann ein Link nicht geteilt werden.	Dies wurde verbessert. Filter werden in der URL abgebildet und beim Öffnen der Seite aus dieser ausgelesen. Dadurch wird automatisch die korrekte Suche ausgeführt. Auch beim Zurücknavigieren aus der Detailansicht werden die Filter korrekt beibehalten. Dies gilt für alle Seiten, auf denen gesucht und gefiltert werden kann (Dashboard, Startseite, Meine Projekte und Projekte prüfen).
2	Bei einem Filter kann nicht auf den Namen geklickt werden, sondern nur auf die Checkbox.	Dies wurde korrigiert.
3	In der CheckboxList sollte statt mit v-model + computed-property besser mit change-event gearbeitet werden.	Dies wurde korrigiert.
4	Warum wird State nicht persistiert, z.B. mittels PiniaStore?	Dazu wird ein caching-Mechanismus von Nuxt verwendet. Dieser wird für Abfragen auf gleichbleibende Daten wie die Filter-Metadate oder die Institute angewendet.
5	Inkonsistenzen in der Verwendung von reactive, shallowRef und ref.	Inkonsistenzen in der Verwendung von reactive, shallowRef und ref, wurden überprüft und korrigiert. ShallowRef wird weiterhin im Videoplayer verwendet, da hier nur eine Referenz auf den Player benötigt wird, ohne dass dessen interner Zustand verändert werden muss.
6	Mittels Object.assign wird aktuell für das Zuweisen eines Werts auf eine reactive Property verwendet.	Dies wurde korrigiert.
7	Die PhotoGallery enthält sehr umfassenden und komplex wirkenden Code.	Der Code ist die Implementation einer Komponente aus einer Bibliothek. Die Implementation erfolgte anhand der Dokumentation dieser Bibliothek. Auf diese wird nun mittels eines Kommentars verwiesen.
8	Die Formatierung ist nicht überall einheitlich.	Mit dem Lint-Kommando wird neu auch die Formatierung automatisiert geprüft und vereinheitlicht.
9	Die Wiederverwendbarkeit von ProjectEditListItem gestaltet sich möglicherweise schwierig, da "path" hartkodiert ist.	Path wird der Komponente neu als property übergeben.

Tabelle 9.8: 1. Code Review - Feedback und Massnahmen

Nr	Frage / Mangel	Lösungsansatz / Verbesserung
10	In ProjectSearchAndFilter wird ein Typ mehrfach verwendet. Diesen könnte man definieren und wiederverwenden.	Diese Komponente wurde überarbeitet. Der angesprochene Typ wird nicht mehr benötigt.
11	In der Anwendung gibt es 2 Header. Anhand der PageMetadata wird entschieden, welcher Header verwendet wird. Ist das Best-Practice?	Diese Logik wurde überarbeitet. Neu gibt es nur noch einen einzigen Header, der die Logik zur Anzeige von Feldern für autorisierte und autorisierte Benutzer handhabt.
12	Wofür wird das server/tsconfig.json File verwendet?	Hierbei handelt es sich um ein File, welches von Nuxt beim Erstellen des Projekts automatisch angelegt wird. Als Verbesserung wurde der gesamte /server Ordner entfernt, da dieser in der Anwendung nicht benötigt wird.
13	Mit Tailwind werden aktuell gewisse Styles mehrfach definiert.	Lösungsansatz: Diese Styles in eine CSS Klasse auslagern mithilfe von @apply. Hier gibt es noch Verbesserungspotenzial. Das Problem ist noch nicht überall behoben worden und wird erst nach und nach angewendet.
14	CORS ist nicht auf die spezifische Domain angepasst.	Dies wurde korrigiert.
15	Das Backend enthält einige nicht verwendete usings.	Diese wurden entfernt.
16	ServiceManger sowie das dort verwendete Lazy-Loading scheinen nicht nötig zu sein.	Der ServiceManager wurde ursprünglich aus einem Guide übernommen, der als Grundlage für die Implementierung der Onion-Architektur diente. Für unsere Anwendung ist er jedoch nicht erforderlich. Neu werden die Services stattdessen mittels der .NET Dependency Injection registriert.
17	RepositoryWrapper scheint nicht nötig zu sein.	Der RepositoryWrapper vereinfacht und vereinheitlicht den Zugriff auf die Repositories in der Service Layer Schicht. Ein einziger Wrapper kann verwendet werden, anstatt dass viele Repositories einzeln injiziert werden müssen. Zudem lässt sich dadurch der ProjectContext leicht teilen, wodurch Operationen leicht im selben Transaktionskontext ausgeführt werden können.
18	Dataseeding wird aktuell in allen Umgebungen gemacht. Besser wäre es, dieses nur in der Development-Umgebung zu machen.	Dies ist bei uns als Technical Debt vermerkt (siehe 3.4).

Tabelle 9.9: 1. Code Review - Feedback und Massnahmen



Nr	Frage / Mangel	Lösungsansatz / Verbesserung
1	Die Erstellung eines .env Files könnte durch ein .env-Example file in der Dokumentation vereinfacht werden.	Dies Vorschlag wurde umgesetzt.
2	Die Environmentvariablen im Docker-Compose-File definieren.	Dies wurde umgesetzt.
3	In den Docker Images sollte nicht Latest verwendet werden.	Dies wurde korrigiert.
4	Variablen für das OST-Theme sind aktuell an 2 Orten definiert.	Dies wurde vereinheitlicht: Alle Variablen sind neu in osttheme.js definiert. Der Grund warum das zuvor nicht so war, ist, dass die Entwicklungsumgebung WebStorm diese Variablen im osttheme.js standardmässig nicht erkennt und Fehlermeldungen anzeigt. Wie sich herausstellte, ist dies allerdings nur ein Problem der Entwicklungsumgebung.
5	Gewisse for attribute verweisen nicht auf ein existierendes ID-Feld.	Dies wurde in allen Feldern des EditDialogs geprüft und korrigiert.

Tabelle 9.10: Weitere Verbesserungsvorschläge - Feedback und Massnahmen

### 9.1.3 System Tests - Finale Durchführung

## System Test Protokoll

### Vorbereitung

- Aktuellste Version des Systems ist deployed.
- Benutzeraccounts mit Berechtigungsstufe Mitarbeiter, Institutsleiter, Kommunikationsprüfer sind erfasst.

### Navigationsleiste

- OST-Logo sowie die Buttons für die Navigation werden korrekt angezeigt.
- Ohne Anmeldung wird Schaltfläche "Login" angezeigt, mit Anmeldung wird Schaltfläche "Abmelden" angezeigt.
- Navigationsschaltflächen reagieren bei Klick erwartungsgemäss.

### Erfassen

Vorbedingung: Angemeldet als Mitarbeiter oder Leiter

- Über die Seite "Projektverwaltung" kann mittels der Schaltfläche "Neues Projekt anlegen" ein neues Projekt erfasst werden.
- Über die Navigationsleiste kann mittels "Neues Projekt anlegen" ein neues Projekt erfasst werden.
- Erfassungsformular wird korrekt dargestellt.
- Ein angelegtes Projekt ist persistiert und wird nach dem zurücknavigieren zu "Meine Projekte" in der Resultatliste angezeigt.
- In der Resultatliste der Suche auf der Seite "Projektverwaltung" werden für Projekte ohne Titel oder Beschreibung entsprechende Platzhalter angezeigt.
- Das Erstellungsdatum wird korrekt angezeigt.
- Durch einen Klick auf ein Projekt in der Projektverwaltung wird die Bearbeitungsansicht angezeigt.
- Es werden die Schaltflächen "Zurück", "Speichern" und "Löschen" korrekt dargestellt.
- Durch einen Klick auf "Löschen" öffnet sich ein Dialog zur Bestätigung des Löschvorgangs.
- Wird das Löschen bestätigt, wird das Projekt gelöscht. Der Benutzer erhält daraufhin eine visuelle Rückmeldung.
- Durch einen Klick auf "Speichern" wird das Projekt gespeichert und der Benutzer erhält eine visuelle Rückmeldung.
- Informationen können in allen Feldern erfasst werden.
- Werden Informationen in Pflichtfeldern erfasst und anschliessend wieder gelöscht, so wird dies als Fehlermeldung angezeigt.
- Felder mit einer maximalen Zeichenlänge (Kurzbeschreibung und Beschreibung des Projekts) zeigen eine entsprechende Fehlermeldung an, wenn die maximale Zeichenlänge überschritten wird.
- Personen können über die Schaltfläche "Person hinzufügen" gesucht werden.

- Wird keine Person gefunden, kann eigenständig eine Person erfasst werden.
- Der Dialog zur Personenerfassung lässt keine Erfassung ohne Namen oder Rolle zu.
- Erfasste Personen werden in der Tabelle korrekt angezeigt und können über diese bearbeitet oder gelöscht werden.
- Organisationseinheiten können gesucht und hinzugefügt werden.
- Organisationseinheiten können über die Tabelle entfernt werden.
- Themen können gesucht und hinzugefügt werden.
- Themen können über die Tabelle entfernt werden.
- Erfassen von Start und Enddatum funktioniert. Ein Enddatum kann nicht vor einem Startdatum liegen.
- Erfassen von "Art des Projektergebnisses", "Sponsor des Projekts", "Sprache" und "Region" funktioniert.
- Links können über einen Dialog erfasst werden.
- Entspricht ein Link nicht einem korrekten URL-Schema, so wird dies dem Benutzer unterhalb der Tabelle angezeigt.
- Durch die Tabelle können Links bearbeitet und gelöscht werden.
- Im Abschnitt "Diverses" können weitere Metadaten optional erfasst werden.

#### **Erfassen von Medien**

- Hochladen von Bild funktioniert korrekt.
- Hochladen von Videos funktioniert korrekt.
- Hochladen von Dokument funktioniert korrekt.
- Hochladen von Titelbild funktioniert korrekt.
- Hochgeladene Medien können mittels der Tabelle gelöscht werden.

### **Validierungsprozess**

#### **Organisationsinterne Veröffentlichung und Einreichung zur fachlichen Prüfung**

- Ohne Eingabe einer Sichtbarkeit werden Buttons für die Veröffentlichung als deaktiviert angezeigt.
- Mit Setzen von Titel und Sichtbarkeit = "Organisationseinheit" wird der Button "Organisationseinheit-intern veröffentlichen" aktiviert.
- Sind noch nicht alle Pflichtfelder erfasst oder Eingaben ungültig, erscheint beim Klick auf die angezeigten Buttons für die Veröffentlichung eine Meldung, die auf die ungültigen Eingaben hinweist.
- Durch Klick auf den Button "Organisationseinheit-intern veröffentlichen" wird das Projekt als "Organisationseinheit-intern" veröffentlicht.
- Ein vollständig erfasstes Projekt mit Sichtbarkeit = "OST" kann durch Klick auf "Zur fachlichen Prüfung einreichen" eingereicht werden.
  - Es öffnet sich ein Dialog zur Bestätigung des Einreichvorgangs.
  - Die Einreichung kann nur dann geschehen, wenn ein Prüfer gewählt wurde.

- Der Ersteller darf sich nicht selbst als Prüfer wählen können.

### Fachliche Prüfung

- Das zur fachlichen Prüfung eingereichte Projekt wird bei der Prüfperson unter "Projekte prüfen" angezeigt.
- Die Prüfperson kann das Projekt bearbeiten und speichern.
- Die Prüfperson kann das Projekt nicht löschen.
- Der Prüfer wird über seine Aufgabe als Prüfer informiert.
- Der angezeigte Projektstatus lautet "Fachliche Prüfung"
- Die Prüfperson kann das Projekt ablehnen und zurückweisen.
  - Beim Ablehnen öffnet sich ein Dialog in dem die Ablehnung begründet werden muss. Ohne Begründung kann das Projekt nicht abgelehnt werden.
  - Nach dem Zurückweisen sieht der Ersteller das Projekt wieder in der Projektverwaltung.
  - Der Ersteller sieht die Begründung der Ablehnung.
- Mit der gewünschten Sichtbarkeit = "OST" kann der Prüfer das Projekt OST-intern veröffentlichen.
  - Die OST-interne Veröffentlichung muss mittels Dialog bestätigt werden.
- Mit der gewünschten Sichtbarkeit = "öffentlich" wird die Schaltfläche für das Einreichen zur Kommunikationsprüfung angezeigt.
  - Das Einreichen zur Kommunikationsprüfung muss mittels Dialog bestätigt werden.

### Kommunikationsprüfung

- Das zur Kommunikationsprüfung eingereichte Projekt wird bei Benutzern, die als Kommunikationsprüfer eingetragen sind unter "Projekte prüfen" angezeigt.
- Der Prüfer wird über seine Aufgabe als Prüfer informiert.
- Der angezeigte Projektstatus lautet "Kommunikationsprüfung"
- Der Prüfer kann das Projekt bearbeiten und speichern.
- Der Prüfer kann das Projekt nicht löschen.
- Der Prüfer kann das Projekt ablehnen und zurückweisen.
  - Beim Ablehnen öffnet sich ein Dialog, in dem die Ablehnung begründet werden muss. Ohne Begründung kann das Projekt nicht abgelehnt werden.
  - Nach dem Zurückweisen sieht der Ersteller das Projekt wieder in der Projektverwaltung.
  - Der Ersteller sieht die Begründung der Ablehnung.
- Der Prüfer kann das Projekt veröffentlichen.
  - Die Veröffentlichung muss mittels Dialog bestätigt werden.

### Startseite: Forschungsprojekte der OST

- Anzahl Treffer wird korrekt angezeigt und entspricht der Menge an gefundenen Projekten.
- Kacheln zur Darstellung der Ergebnisse werden korrekt angezeigt.

## Filtern

### Unautorisierter Benutzer (Startseite)

- Filter nach Sichtbarkeit wird nicht angezeigt.
- Projekte mit Sichtbarkeitsstatus OST und intern werden nicht gelistet, es werden nur Projekte mit Status Öffentlich gefunden.
- Filterkombinationen ohne Treffer zeigt Text "Keine Treffer gefunden." an.

### Mit Login

- Filtern nach Arbeitsergebnis liefert korrekte Resultate.
- Filtern nach Schlagwörtern liefert korrekte Resultate.
- Filtern nach Orte liefert korrekte Resultate.
- Filtern nach Sponsor liefert korrekte Resultate.
- Filtern nach Durchführungszeitraum liefert korrekte Resultate.
- Filtern nach Datum der Veröffentlichung liefert korrekte Resultate.
- Filtern nach Organisationseinheiten liefert korrekte Resultate.
- Filtern nach Departemente liefert korrekte Resultate.
- Filtern nach Sichtbarkeit liefert korrekte Resultate.
- Gesetzte Filter lassen sich über den Button "Filter zurücksetzen" zurücksetzen.

## Suchen

### Unautorisierter Benutzer (Startseite)

- Suche nach Titel, Autor und Abstract liefert korrekte Resultate.
- Suche funktioniert auch in Kombination mit setzen von Filtern.
- Suche nach nicht existierendem Begriff zeigt Text "Keine Treffer gefunden" an.
- Suche ist nicht case sensitiv.
- Suche mit Teilwörtern funktioniert korrekt (z.B. "Simo" findet "Simon").

## Projekt-Detailansicht

- Die Seitennavigation wird korrekt dargestellt.
- Die Seitennavigation bringt den Benutzer zu den entsprechenden Abschnitten.
- Im Bereich "Überblick" werden Daten zu Durchführungszeitraum, Arbeitsergebnis, Thema und Mitwirkenden korrekt angezeigt.
- Das Thumbnail (sofern vorhanden) wird korrekt dargestellt.
- Projektdateien (sofern vorhanden) lassen sich über einen Klick herunterladen.

- Im Bereich "Beschreibung" wird die Beschreibung korrekt dargestellt.
- Hochgeladene Medien werden auf der Projektdetailseite korrekt angezeigt.
- Bilder (sofern vorhanden) werden als Slideshow dargestellt.
- Videos (sofern vorhanden) können durch einen Videoplayer abgespielt werden.
- Im Bereich "Details" werden Beteiligte Organisationseinheiten, Sponsor, Sprache und Veröffentlichungsdatum dargestellt.
- Sofern das Projekt über weitere optionale Inhalte verfügt, werden auch diese korrekt dargestellt.
- Nach Klick auf einen Link (optionales Element) wird der Benutzer auf die entsprechende Seite weitergeleitet.

## Dashboard

- Jedes Diagramm im Dashboard wird korrekt dargestellt.
- Das Anwenden der Filter führt zu einer korrekten Aktualisierung der Diagramme.
- Sind durch das Anwenden eines Filters keine Daten zur Anzeige vorhanden, wird dies dem Benutzer entsprechend mitgeteilt.

## Registrieren

- Das Formular wird inklusive der Felder Email, Anrede, Titel, Vorname und Nachname korrekt dargestellt.
- Das Feld Organisationseinheiten listet die Auswahl aller erfassten Organisationseinheiten auf.
- Das Feld Rolle listet alle möglichen Benutzerrollen auf.
- Checkbox auswahl für Prüfer Kommunikation wird korrekt dargestellt.
- Durch Click auf Registrieren wird Benutzerkonto angelegt, Benutzer wird auf Startseite geleitet.

## Login

- Link um sich zu registrieren wird dargestellt und funktioniert.
- Login Feld wird korrekt dargestellt.
- Durch die Eingabe einer existierenden Mail führt der Anmelde-Button zur Startseite.





Abbildung 9.4: Request für die Tagsuche

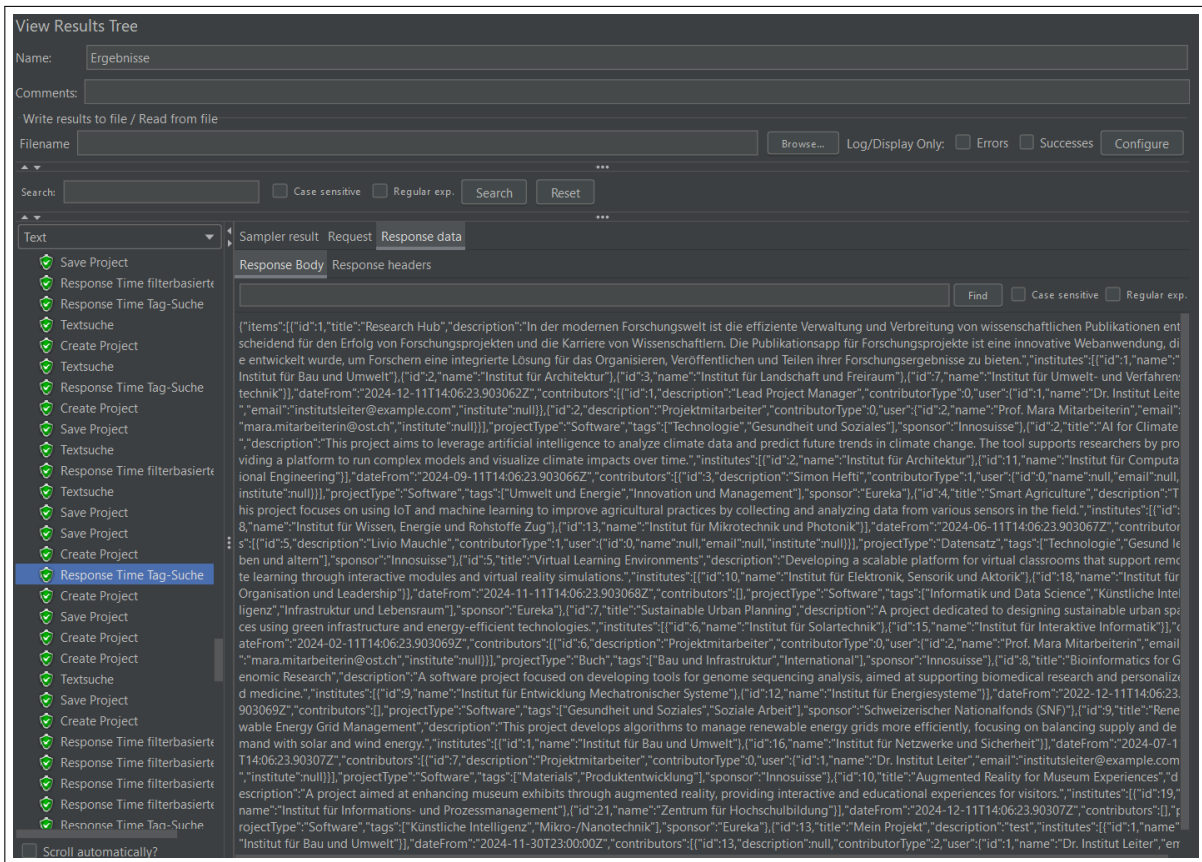


Abbildung 9.5: Response für die Tagsuche

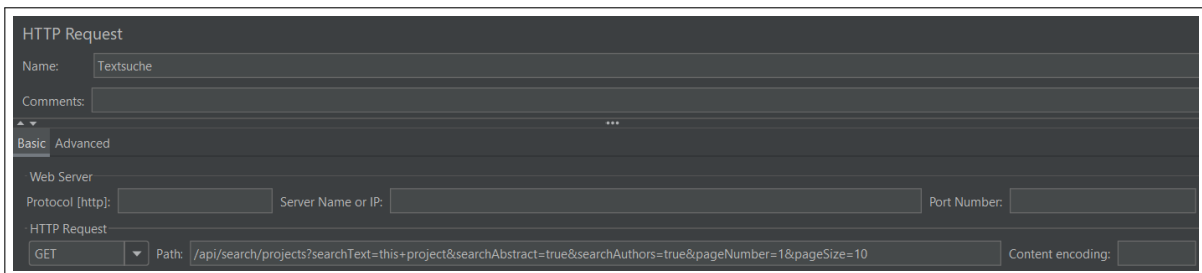


Abbildung 9.6: Request für die Textsuche





Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Delete Project	4	92	25	196	64.56	25.00%	4/hour	0.00	0.00	114.2
File Upload Small	4	712	161	1567	526.54	25.00%	4/hour	0.00	0.55	301.0
TOTAL	8	402	25	1567	486.86	25.00%	7/hour	0.00	0.55	207.6

Abbildung 9.10: Testreport für den zweiten Testplan (1 Thread). Die maximale Response-Time liegt auch hier mit 1.57s unter dem Grenzwert (2s).

Abbildung 9.11: Request für das Löschen (Delete)

Abbildung 9.12: Request für den Upload eines 10.7 MB grossen Files. Die Datei wird als multipart/form-data übermittelt.

## 9.2 Ergänzende Inhalte

### 9.2.1 Eigenständigkeitserklärung




#### Eigenständigkeitserklärung für die Studienarbeit OST ResearchNavigator

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selbst und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit der Betreuerin / dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe,
- dass ich keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt habe.

St. Gallen, 10.12.2024

Livio Mauchle, 

Simon Hefti, 

## 9.2.2 Auswertung der Anzahl Codezeilen

Um herauszufinden, wie viele Zeilen Code das Projekt enthält, wird das Tool "cloc" verwendet. Dies kann mit dem folgenden Kommando gemacht werden:

```
cloc --list-file=folders_to_check.txt
```

Dabei ist "folders\_to\_check.txt" ein File, welches diejenigen Ordner auflistet, die den von uns geschriebenen Code enthalten. In den Auswertungen in Kapitel 7 wurden vom Umfang her vernachlässigbare Sprachen wie JavaScript und [Cascading Style Sheets \(CSS\)](#) weggelassen. Die folgenden Abbildungen zeigen die von cloc generierte Auswertung der Anzahl an Codezeilen für das Frontend, Backend und Testing-Projekt.

Language	files	blank	comment	code
Vuejs Component	33	450	35	4621
TypeScript	28	110	4	1047
JavaScript	1	8	1	62
CSS	2	8	1	39
SUM:	64	576	41	5769

Abbildung 9.13: Anzahl Codezeilen im Frontend

Language	files	blank	comment	code
C#	98	631	240	4033
SUM:	98	631	240	4033

Abbildung 9.14: Anzahl Codezeilen im Backend

Language	files	blank	comment	code
C#	16	461	203	2073
C# Generated	1	4	9	9
SUM:	17	465	212	2082

Abbildung 9.15: Anzahl Codezeilen der Unit Tests

### 9.2.3 Setup Anleitung für Entwickler

## OST ResearchNavigator

---

A website for searching and managing information about the projects of all institutes of the OST University of Applied Sciences Eastern Switzerland.

This project was initiated during the Studienarbeit OST ResearchNavigator (formerly "Publikationsapp für Forschungsprojekte") by Livio Mauchle and Simon Hefti.

### Overview

The software consists of the following parts:

- ASP.NET Core Web API
- Nuxt.js Frontend
- Postgres Database
- Minio Filesystem

The frontend and backend can be run individually as described in their respective README-Files

The whole system can also be run with docker compose (see below)

### Setup

Most of the configuration is included in the compose.yaml and the Dockerfiles. Some environment variables have to be provided. The root directory must contain a file ".env" with the following variables required by the compose-file. A sample file named ".env.example" has been provided and (for local setup) needs only little reconfiguration (e.g. the Minio Keys).

#### Environment Variables

##### Database

Variables used for the database setup and to build the connection string for EF Core

- Server: Name of the postgres container.
- Port: Default 5432
- Database: Created when starting the postgres container the first time.
- Username & Password: Admin user for the postgres database

```
POSTGRES_SERVER=...
POSTGRES_PORT=5432
POSTGRES_DB=...
POSTGRES_USER=...
POSTGRES_PASSWORD=...
```

##### JWT

- Key: some unique key for jwt generation (at least 256 Bits)
- Issuer: Your backend address
- Audience: Your frontend address

```
JWT_KEY=...  
JWT_ISSUER=... # e.g. http://localhost for local development  
JWT_AUDIENCE=... # e.g. http://localhost for local development
```

## Minio

The password, endpoint and keys of the minio fileserver

- Password: The password for the admin user (Default username ROOTNAME). Required for logging into the Minio Console and creating the access keys.
- Bucket Name: Default bucket that contains all project files. Created when starting Minio the first time.
- Endpoint: Host:Port of the Fileserver. The port is set in the docker compose minio configuration (Default localhost:9000).
- Access & Secret Key: Credentials used by the backend to connect to MinIO. They have to be created in the Minio Console first. See below for more information

```
MINIO_ROOT_PASSWORD=...  
MINIO_BUCKET_NAME=...  
MINIO_ENDPOINT=<host>:9000  
MINIO_ACCESS_KEY=...  
MINIO_SECRET_KEY=...
```

## CORS

The allowed Cross-Origin-Hosts (without port), separated by comma

```
ALLOWED_ORIGINS=localhost,127.0.0.1
```

## Frontend

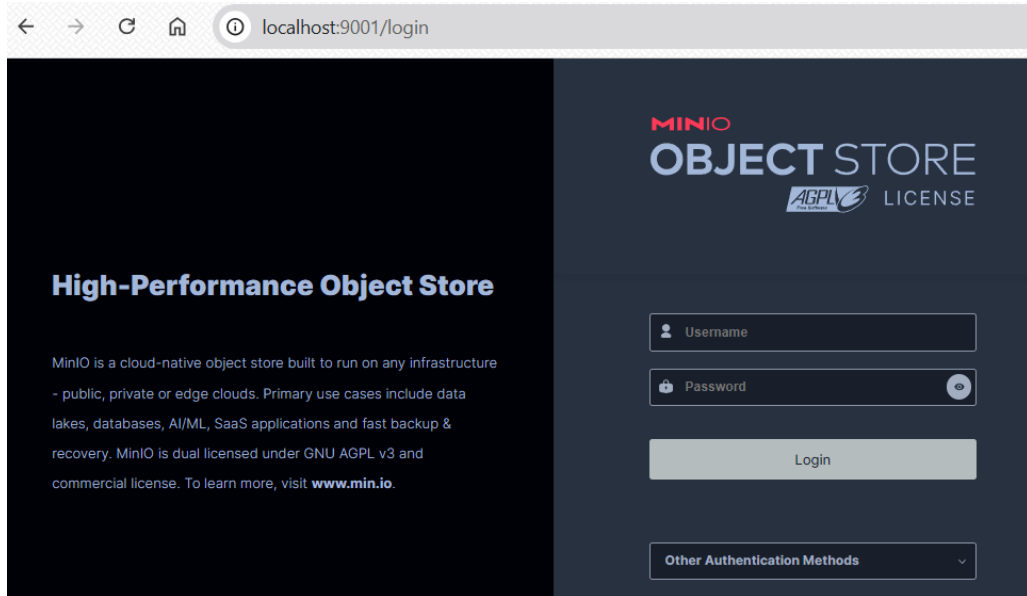
The API base URL for communication with the backend

```
API_BASE_URL=http(s)://<host>:5050
```

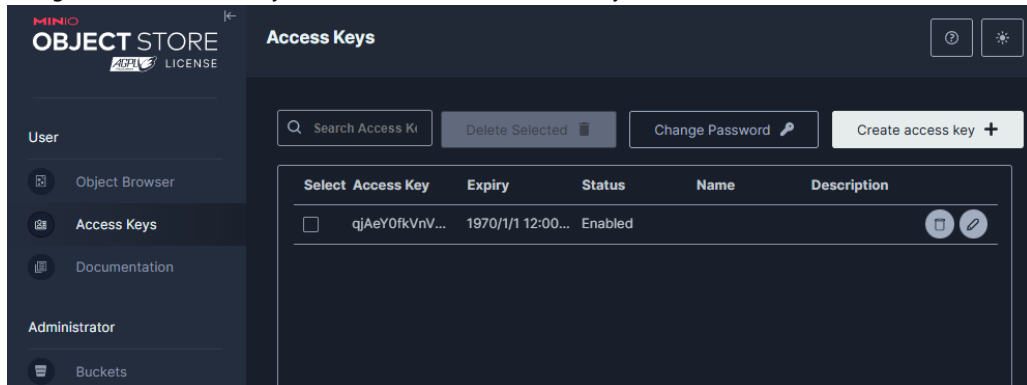
## Minio

### Creating the Minio Keys

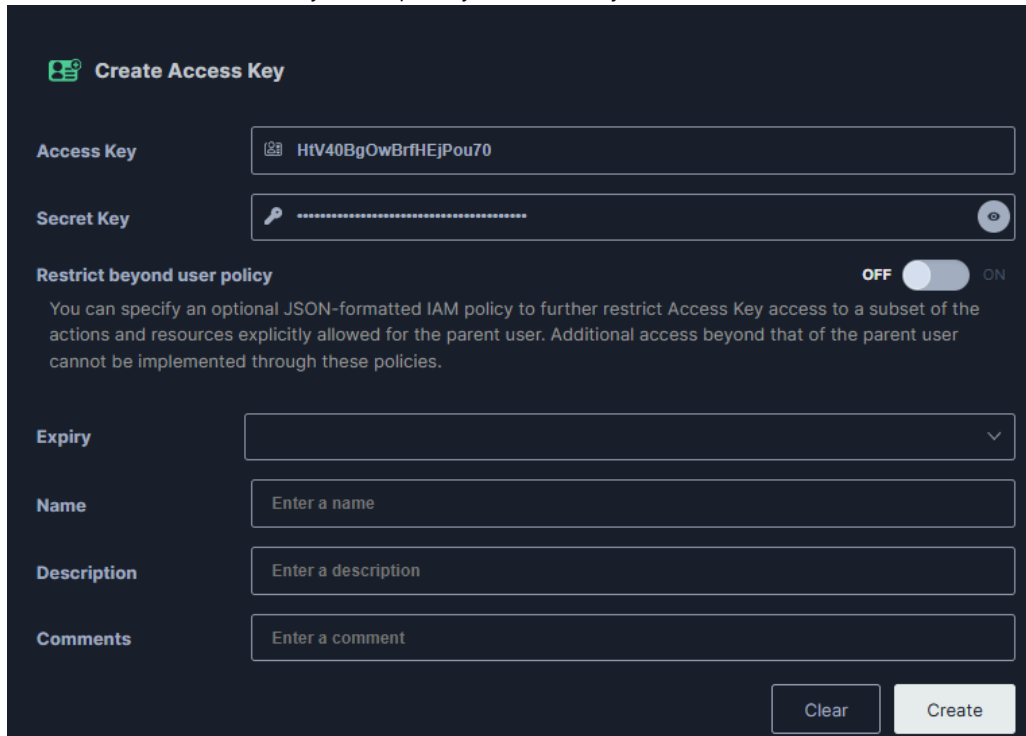
1. Open the console in a web browser. The host is the same as for the file server, the port is the second one in the docker compose configuration (Default localhost:9001).
2. Sign in: The username is also set in the docker compose file (Default ROOTNAME), the password inside the .env file in the root dir (MINIO\_ROOT\_PASSWORD).



3. Navigate to the "Access Keys"-Tab and click "Create access key"



4. Write down both the access key and (especially) the secret key



**Create Access Key**

**Access Key**

**Secret Key**

**Restrict beyond user policy**  OFF  ON

You can specify an optional JSON-formatted IAM policy to further restrict Access Key access to a subset of the actions and resources explicitly allowed for the parent user. Additional access beyond that of the parent user cannot be implemented through these policies.

**Expiry**

**Name**

**Description**

**Comments**

5. Click the "Create"-Button and set the keys in your configuration

## Starting the application

All containers can be built and run with the following command

```
docker compose up --build
```

To enable Hot Reloading for the frontend during development, use the following command:

```
docker compose -f compose.dev.yaml up --build
```



# Backend

---

Our backend is an ASP.NET Core Web API built with .Net 8 and C#

It can be run locally (without Minio filesystem) or with docker (with Minio filesystem) as described in the [root README](#). Alternatively, you can also run Minio in Docker and the Backend locally as described below.

## Local Setup

### 1. Installations

The following programs and installations are required / recommended:

- [.NET 8.0 SDK](#)
- Postgres: postgres server (Required) & pgadmin (Recommended)
- JetBrains Rider (Recommended)

### 2. Clone the Project

Clone the project onto your device and open it in your IDE (preferably Rider)

### 3. Install Dependencies

Run the following command in your cli to install all required packages and tools:

```
dotnet restore
```

### 4. Configuration

The default configuration is set inside the appsettings.json file. For local development, those settings can be overwritten in a new file named appsettings.Development.json, which is ignored by git.

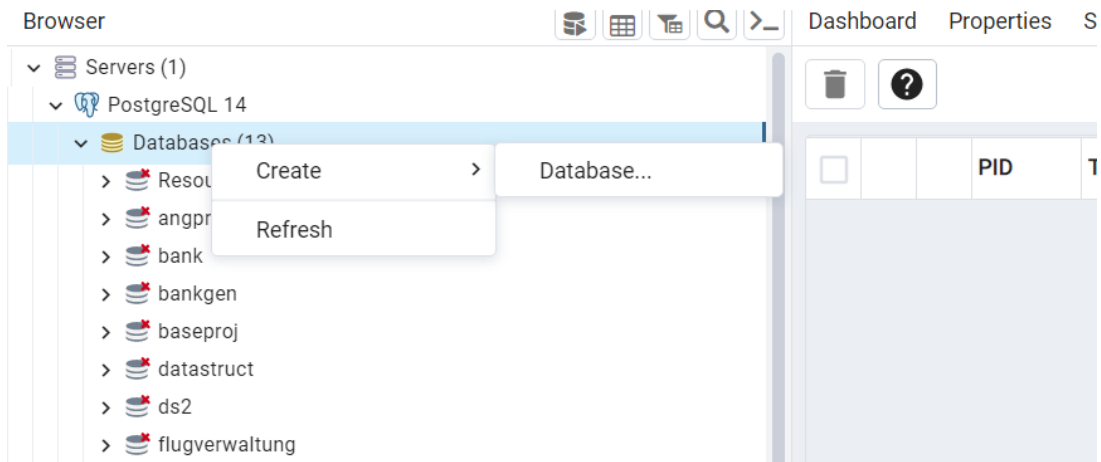
### 5. Setup your Postgres Database locally

During setup, it is important that the DB connection matches the connection string in your appsettings.\*.json. The default user in postgres has the username "postgres" with the password "admin". You are free to create a new postgres user and use it for the database connection, however keep in mind that you have to adjust the configuration accordingly.

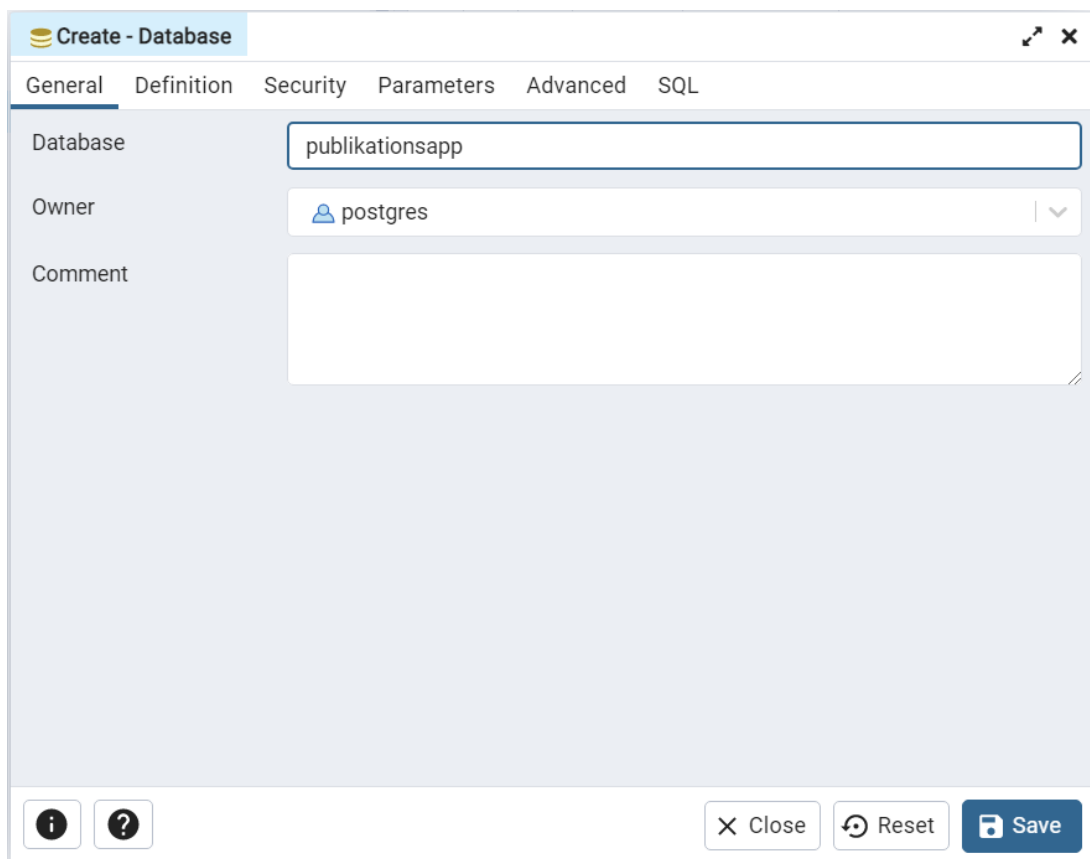
More details about the configuration are described below.

To run the project locally and not in docker, a Postgres database must be created locally. This can be done with PG Admin 4 as follows:

1. navigate to databases -> rightclick -> create -> database



2. Give it the name "publikationsapp"
3. Select the user "postgres" (or whatever user you have set in your configuration)
4. save it



5. Execute Database Migrations

This command executes all DB migrations, thereby creating all database tables locally. This is not required, since the migrations are executed when starting the application

```
cd Source
dotnet ef database update
```

## 6. Starting the application

```
cd Source
dotnet run
```

## Tests

Use the following command to run all unit tests:

```
cd UnitTests
dotnet test
```

To generate a coverage report, run the following commands:

```
cd UnitTests

dotnet test -c Debug --results-directory ./TestResults/cobertura --collect:"XPlat
Code Coverage" --test-adapter-path:. --
logger:"junit;LogFilePath=./TestResults/junit/junit-test-
result.xml;MethodFormat=Class;FailureBodyFormat=Verbose"

dotnet reportgenerator -reports:TestResults/cobertura/*/coverage.cobertura.xml -
targetdir:TestResults/coveragereport -reporttypes:Html

Invoke-Item .\TestResults\coveragereport\index.html
```

## Formatting

Commands to format the Source-Project:

```
cd Source
dotnet format Source.csproj
```

Commands to format the UnitTest-Project:

```
cd UnitTests  
dotnet format UnitTests.csproj
```

## Frontend

---

Built with [Nuxt 3](#) to learn more.

### Setup

Installing the dependencies:

```
# npm
npm install
```

For local development with API communication and without docker, you also need to specify the API Endpoint in a `.env`-File located in this directory

```
API_BASE_URL=... # e.g. http://localhost:5050 for local development
```

If the frontend service is started with docker compose, this is not required.

### Development Server

Start the development server on <http://localhost:3000>:

```
# npm
npm run dev
```

### Production

Build the application for production:

```
# npm
npm run build
```

Locally preview production build:

```
# npm
npm run preview
```

### Lint

Display lint errors and warnings:

```
npm run lint
```

Fix lint errors and warnings (where possible):

```
npm run lint:fix
```

## 9.2.4 Benutzeranleitung

# Benutzeranleitung

In diesem Dokument sind die Schritte zur Bedienung der wichtigsten Funktionen der Anwendung beschrieben.

## Projektsuche mit Filtern

Die Startseite ist die Übersicht der veröffentlichten Projekte. Führen Sie die folgenden Schritte durch, um diese zu filtern.

1. Um nach dem Namen eines Autors oder bestimmten Wörtern im Titel und Abstract zu suchen, können Sie diese in die Textleiste oben in der Mitte eingeben und auf "Los" klicken oder die Enter-Taste drücken.
2. Um nach Kriterien wie den Instituten oder nach bestimmten Schlagwörtern (Themen) zu filtern, verwenden Sie die Filterleiste auf der linken Seite. Die einzelnen Filterkriterien können aufgeklappt werden. Falls Sie die Website auf einem kleinen Gerät wie einem Handy aufrufen, müssen Sie die Filter erst mit einem Klick auf "Filter" anzeigen.
3. Ist bei einem Filterkriterium keine Auswahl getroffen, wird das Kriterium nicht verwendet und die Projekte somit nicht danach gefiltert. Falls mehrere Werte ausgewählt sind, werden alle Projekte angezeigt, die einem der Werte entsprechen (z.B. falls drei Institute ausgewählt sind, werden alle Projekte angezeigt, an denen mindestens eines davon beteiligt war).
4. Um die ausgewählten Filtern anzuwenden, klicken Sie auf "Filter anwenden" (oder auf "Los").
5. Mit "Filter zurücksetzen" können Sie die Filterauswahl leeren.

**Forschungsprojekte der OST** Forschungsprojekte der OST Dashboard → Login OST

**Suche nach Titel, Autor, Abstract**

Suche... × Q Los

10 Treffer

Arbeitsergebnis	Software   2024
Schlagwörter	<b>AI for Climate</b>
Orte	Themenbereiche: Umwelt und Energie; Innovation und Management
Sponsor	Beteiligte Organisationseinheiten: Institut für Architektur; Institut für Computational Engineering;
Durchführungszeitraum	Mitwirkende Person: Simon Hefti
Datum der Veröffentlichung	This project aims to leverage artificial intelligence to analyze climate data and predict future trends in climate change. The tool supports researchers by providing a platform to run complex models and visualize climate impacts over time.
Organisationseinheiten	Datensatz   2024
	<b>Smart Agriculture</b>
	Themenbereiche: Technologie; Gesund leben und altern
	Beteiligte Organisationseinheiten: Institut für Wissen, Energie und Rohstoffe Zug; Institut für Mikrotechnik und Photonik;
	Mitwirkende Person: Livio Mauchle
	This project focuses on using IoT and machine learning to improve agricultural practices by collecting and analyzing data from various sensors in the field.
	Software   2024

Filter zurücksetzen Filter anwenden

6. Möchten Sie zu einem Projekt weitere Informationen erhalten, klicken Sie in der Liste auf das Projekt. Damit werden Sie zur Detailseite weitergeleitet.
7. Auf der Detailseite können Sie mit der Navigation auf der rechten Seite (Überblick, Beschreibung und Details) zu einem bestimmten Abschnitt springen. Alternativ können Sie auch herunterscrollen, um alle

Details, Beschreibungen und Dateien zum Projekt zu sehen.

8. Mit einem Klick auf "Zurück zur Startseite" (oben links) können Sie zur Projektsuche zurückkehren

The screenshot shows the OST ResearchNavigator project page. At the top, there is a navigation bar with 'Forschungsprojekte der OST', 'Projektverwaltung', 'Dashboard', and 'Abmelden'. Below this is a red button labeled 'Zurück zur Trefferliste'. The main content area is titled 'OST ResearchNavigator' and features a grid of project details:

Durchführungszeitraum	Arbeitsergebnis	Thema
11.12.2024 - 18.12.2024	Software	Technologie Gesundheit und Soziales
Projektdateien	Mitwirkende	
Dokumentation als PDF	Livio Mauchle Simon Helft	

Below the grid is a 'Beschreibung' section with a 'Zusammenfassung' sub-section. The summary text describes the platform's purpose in supporting research projects and publication management. At the bottom of the screenshot, there is a small image of a code editor showing JavaScript code.

## Dashboard

1. Das Dashboard erreichen Sie mit einem Klick auf "Dashboard" in der Titelleiste
2. Hier sehen Sie Statistiken zu den Projekten nach verschiedenen Kriterien
3. Um Filter auf die Projekte anzuwenden, aus denen die Statistiken berechnet werden, klicken Sie auf "Filter" (oben links).
4. Die einzelnen Filter und die Anwendung dieser funktioniert wie bei der Projektsuche. Nur die Filter "Schlagwörter" und "Sichtbarkeit" gibt es hier nicht.



Filter

Anzahl Projekte

**20**

Anzahl Projekte insgesamt

OST intern

**15**

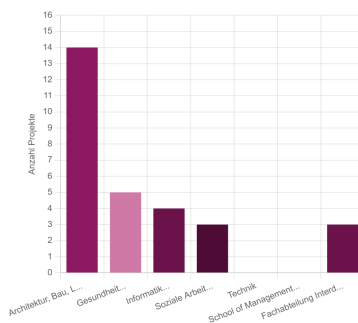
Projekte, die für OST interne sichtbar sind

Öffentliche

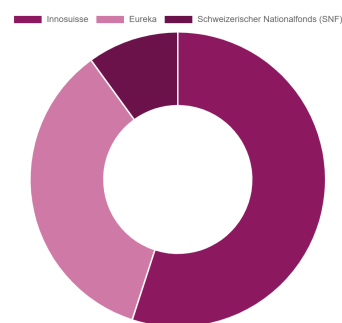
**12**

Projekte, die öffentlich sichtbar sind

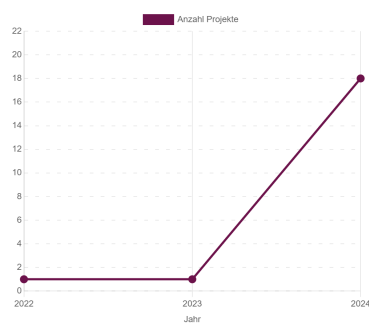
Projekte nach Departement



Sponsoren



Anzahl Projekte im zeitlichen Verlauf



Häufigste Themenbereiche



## Login

Um Projekte zu erfassen und um OST- oder Instituts-interne Projekte anzusehen, müssen Sie sich zuerst anmelden

1. Navigieren Sie mit einem Klick auf "Login" in der Titelleiste auf die Login-Seite.
2. Falls Sie noch keinen Account haben, klicken Sie auf "Registriere dich Hier!". Fahren Sie dann beim dem nächsten Abschnitt "Account erstellen" fort.
3. Wenn Sie schon einen Account haben, geben Sie die E-Mail-Adresse (OST-Email) in das Textfeld ein und klicken dann auf "Einloggen".
4. Sie werden auf die Startseite weitergeleitet. Dort wird Ihnen nun auch der Filter "Sichtbarkeit" angezeigt.

**OST**  
Ostschweizer  
Fachhochschule

## Willkommen Zurück


Du hast noch keinen Account? [Registriere dich Hier!](#)

E-Mail-Adresse

## Account erstellen

1. Falls Sie doch schon ein Login haben, kehren Sie mit "Zum Login" auf die Login-Seite zurück.
2. Ansonsten füllen Sie alle Informationen zu Ihrer Person aus. Verwenden Sie als E-Mail-Adresse Ihre OST-E-Mail.
3. Falls Sie berechtigt sind, kommunikative Prüfungen durchzuführen, wählen Sie die Box "Prüfer Kommunikation" aus.
4. Mehr Informationen finden Sie unter "Wer darf ein Konto beantragen?".
5. Um den Account zu erstellen und sich anzumelden, klicken Sie auf "Registrieren".
6. Sie werden auf die Startseite weitergeleitet. Dort wird Ihnen nun auch der Filter "Sichtbarkeit" angezeigt.

**OST**  
Ostschweizer  
Fachhochschule

[Zum Login](#)

E-Mail-Adresse

Anrede

Titel

Vorname

Nachname

Organisationseinheit

Rolle

Prüfer Kommunikation

Wer darf ein Konto beantragen?

## Projektverwaltung

1. Wenn Sie angemeldet sind, können Sie mit einem Klick auf "Projektverwaltung" in der Titelleiste die von Ihnen erfassten Projekte anzeigen.
2. Die Suche und Filter auf dieser Seite funktionieren wie auf der Startseite.
3. Um ein Projekt zu bearbeiten oder freizugeben, klicken Sie es in der Liste an.

The screenshot shows the OST project management interface. At the top, there is a navigation bar with 'Forschungsprojekte der OST', 'Projektverwaltung', 'Dashboard', and 'Abmelden'. A search bar is present with the text 'Suche nach Titel, Autor, Abstract' and a search button. Below the search bar, there are two search results:

- Research Hub**: Status: Freigegeben, Erstellt am: 11.12.2024. Description: In der modernen Forschungswelt ist die effiziente Verwaltung und Verbreitung von wissenschaftlichen Publikationen entscheidend für den Erfolg von Forschungsprojekten und die Karriere von Wissenschaftlern. Die Publikationsapp für Forschungsprojekte ist eine innovative Webanwendung, die entwickelt wurde, um Forschern eine integrierte Lösung für das Organisieren, Veröffentlichenden und Teilen Ihrer Forschungsergebnisse zu bieten. Sichtbar für: Alle.
- Test Projekt**: Status: Draft, Erstellt am: 1.12.2024. Description: Dieses Projekt hat noch keine Beschreibung. Bitte füge eine Beschreibung hinzu, um die Projektinformationen weiter zu vervollständigen. Sichtbar für: Ersteller.

On the left side, there is a filter menu with categories: Arbeitsergebnis, Schlagwörter, Orte, Sponsor, Durchführungszeitraum, Datum der Veröffentlichung, Organisationseinheiten, Departemente, and Sichtbarkeit. At the bottom of the filter menu, there are buttons for 'Filter zurücksetzen' and 'Filter anwenden'. A pagination bar at the bottom shows '1' of 10 results.

## Projekterstellung

1. Ein neues Projekt können Sie mit einem Klick auf "Neues Projekt anlegen" oben rechts in der Projektverwaltung oder alternativ in der Titelleiste unter Projektverwaltung -> Neues Projekt anlegen

The screenshot shows the OST project management interface. At the top, there is a navigation bar with 'Forschungsprojekte der OST', 'Projektverwaltung', 'Dashboard', and 'Abmelden'. A dropdown menu is open under 'Projektverwaltung', showing the following options:

- Meine Projekte
- Projekte prüfen
- Neues Projekt anlegen

2 Treffer

## Projektbearbeitung

1. Nach der Erstellung eines neuen Projektes oder dem Klick auf ein Projekt in der Projektverwaltung öffnet sich die Projekterfassungsseite.
2. Hier können Sie alle Informationen zum Projekt in die dafür vorgesehenen Felder eintragen. Mit einem Stern (\*) gekennzeichnete Felder sind Pflichtfelder.
3. Mit einem Klick auf "Speichern" können Sie jederzeit den aktuellen Zwischenstand speichern und später mit der Erfassung weiterfahren.
4. Falls Sie ein falsches Projekt erfasst haben, können Sie es mit einem Klick auf "Löschen" unwiderruflich entfernen.

[← Zurück](#)[✓ Zur fachlichen Prüfung einreichen](#)[Löschen](#)[Speichern](#)

## OST ResearchNavigator

**Status: Draft**

### Titel & Beschreibung

#### Titel des Projekts\*

#### Kurzbeschreibung des Projekts\*

Ein kurze Beschreibung des Projekts, die in der Liste der Suchergebnisse angezeigt wird.

Ziel dieser Studienarbeit ist die Implementierung eines Minimum Viable Product (MVP) der gewünschten Lösung. Den ersten Schritt bildete die Formulierung der Anforderungen an die Anwendung in Form von Use Cases. Daraus wurden die wichtigsten nicht-funktionalen Anforderungen abgeleitet, um eine sinnvolle und robuste Architektur mit dem Fokus auf Erweiterbarkeit zu entwerfen.

Um die Übersichtlichkeit zu wahren, ist eine maximale Zeichenanzahl vorgegeben.

377 / 1000

## File Upload

1. Um Dateien hochzuladen, scrollen Sie in der Projekterfassung nach unten. Dort findest du den Abschnitt "Datei wählen und hochladen".
2. Folgen Sie den dort beschriebenen Schritten 1-3, um eine Datei hinzuzufügen.
3. Achten Sie auf den Dateityp! Für die Dateitypen "Bild", "Titelbild" und "Video" können nur entsprechende Dateiformate hochgeladen werden.
4. Die Dateien werden nach dem Hochladen in der Liste darunter angezeigt.
5. Mit einem Klick auf den Dateinamen können Sie die Datei im Browser öffnen oder herunterladen.
6. Um die Datei wieder zu löschen, klicken Sie auf das Mülleimer-Symbol rechts davon

### Datei wählen und hochladen (optional)

1. Wähle eine Datei, die du hochladen möchtest.
  2. Wähle den passenden Dateitypen aus, dieser bestimmt die Art und Weise wie der hochgeladene Inhalt angezeigt wird.
  3. Klicke auf "Hochladen" um die Datei dem Projekt hinzuzufügen.
- Die maximale Dateigröße beträgt 50 MB.

+ Datei auswählen

↑ Datei hochladen

× Abbrechen

Ziehen Sie eine Datei per Drag & Drop hierher, um sie hochzuladen

#### Hochgeladene Dateien

Dateiname

Dateityp

filter\_schlagwoerter.png

Bild



### Projekt zur Prüfung einreichen

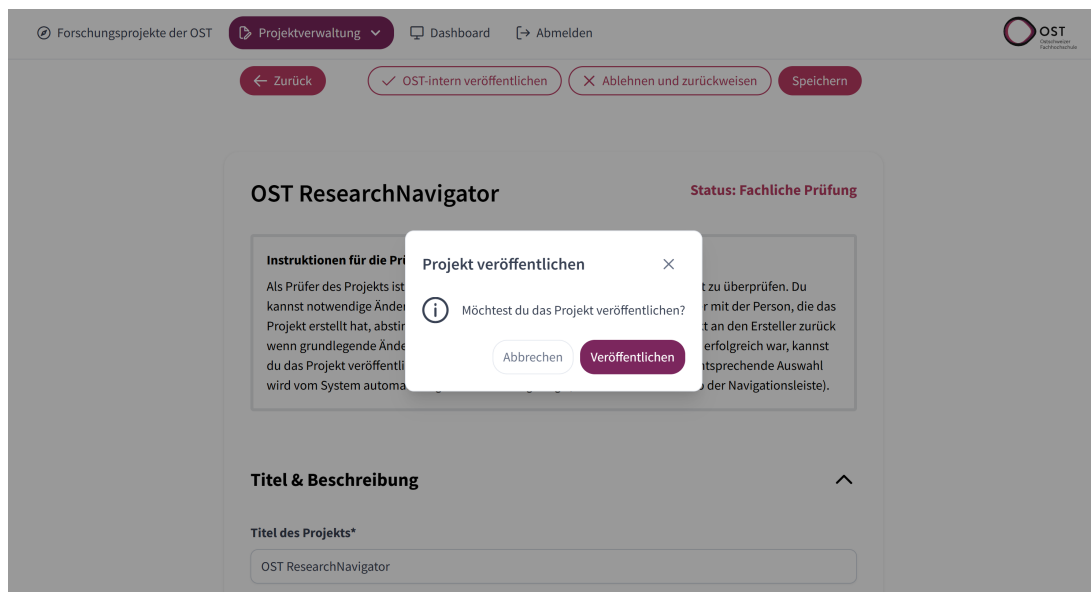
1. Sobald alle Pflichtfelder ausgefüllt sind, können Sie das Projekt für die fachliche Prüfung freigeben.
2. Klicken Sie dafür auf "Zur fachlichen Prüfung einreichen"
3. Im geöffneten Fenster wählen Sie die gewünschte fachliche Prüfperson und klicken dann auf "Zur Prüfung einreichen". VORSICHT: Sie können das Projekt danach NICHT mehr bearbeiten!
4. Sie werden auf Ihre Projektverwaltungsseite weitergeleitet. Das Projekt wird Ihnen noch angezeigt, ist aber ausgegraut und Sie können es nicht mehr anklicken.

The screenshot shows the 'Projektverwaltung' (Project Management) page for 'OST Research Navigator'. The project status is 'Draft'. A modal dialog box titled 'Projekt zur Prüfung einreichen' (Project for review) is open, prompting the user to select a reviewer. The selected reviewer is 'Livio Mauchle'. The dialog includes 'Abbrechen' (Cancel) and 'Zur Prüfung einreichen' (Submit for review) buttons. In the background, a red warning box states: 'Das Projekt wurde von... Zurückgewiesen am 15... Begründung: Das Projekt weist erhebliche Mängel auf und benötigt präzisere Ausarbeitungen. Bitte arbeiten Sie es... Grundlagen stärker.' The main form fields visible include 'Titel des Projekts\*' (OST ResearchNavigator) and 'Kurzbeschreibung des Projekts\*'.

5. Sobald sowohl die fachliche als auch die kommunikative Prüfperson Ihr Projekt validiert hat, wird es freigeschaltet und ist per Suche auffindbar
6. Wird das Projekt von einer Prüfperson zurückgewiesen, kommt es für die Überarbeitung zurück in Ihre Projektverwaltung

## Projekt validieren

1. Navigieren Sie über Projektverwaltung -> Projekte prüfen auf die Prüferübersicht.
2. Darauf sehen Sie alle Projekte, welche Ihnen für die fachliche Prüfung zugewiesen sind sowie alle Projekte, die aktuell für eine kommunikative Prüfung eingereicht sind.
3. Mit einem Klick auf eines der Projekte können Sie es in der Erfassungsansicht öffnen.
4. Sie können alle Informationen zum Projekt überarbeiten.
5. Wenn Sie mit dem Projekt zufrieden sind, können Sie es veröffentlichen bzw. weiterreichen. Im Falle einer fachlichen Prüfung klicken Sie dafür auf "OST-intern veröffentlichen" oder "Zur Kommunikationsprüfung einreichen", bei einer kommunikativen Prüfung auf "Veröffentlichen". Bestätigen Sie danach mit einem erneuten Klick auf "Veröffentlichen". VORSICHT: Die Veröffentlichung kann NICHT rückgängig gemacht werden.



6. Sind Sie mit einem Projekt unzufrieden, können Sie es auch zurückweisen. Klicken Sie dafür auf "Ablehnen und zurückweisen"
7. Geben Sie eine Begründung ein, z.B. was fehlt oder was besser beschrieben werden soll.
8. Mit einem Klick auf "Zurückweisen" wird das Projekt an den Erfasser zurückgereicht, welcher es nochmals überarbeiten kann

Forschungsprojekte der OST Projektverwaltung Dashboard Abmelden OST  
OST-extern veröffentlichen Ablehnen und zurückweisen Speichern

**Projekt zurückweisen** X

Möchtest du das Projekt an den Ersteller zurückweisen?

**Begründung**

Das Projekt weist erhebliche Defizite in der fachlichen Herleitung und Methodik auf. Bitte arbeiten Sie es präziser aus und berücksichtigen Sie relevante Standards und wissenschaftliche Grundlagen stärker.

Zeichenlimit 205/2000

Abbrechen Zurückweisen

**OST Research**

**Instruktionen für die**  
Als Prüfer des Projekts kannst notwendige Änderungen am Projekt erstellt hat, aber wenn grundlegende Änderungen du das Projekt veröffentlicht wird vom System automatisch

**Titel & Beschreibung**

**Titel des Projekts\***  
OST ResearchNavigator

**us: Fachliche Prüfung**  
überprüfen. Du mit der Person, die das in den Ersteller zurück erfolgreich war, kannst bereichende Auswahl (Navigationselemente).

## 9.2.5 API Dokumentation mittels Swagger UI

18.12.24, 22:34

Swagger UI



Select a definition

Source v1

### Source 1.0 OAS3

<http://localhost:5050/swagger/v1/swagger.json>

Authorize

### Account ^

POST /api/account/register ^

Parameters

Try it out

No parameters

Request body

application/json

Example Value Schema

```
{
  "email": "string",
  "instituteId": 0,
  "role": 0,
  "firstName": "string",
  "lastName": "string",
  "phoneNumber": "string",
  "isCommValidator": true,
  "gender": "string",
  "title": "string"
}
```

Responses



Code	Description	Links
200	Success	No links

POST /api/account/login



## Institute



GET /api/institutes/all



## Project



GET /api/projects/{projectId}



POST /api/projects/{projectId}



DELETE /api/projects/{projectId}



GET /api/projects/{projectId}/edit



POST /api/projects



GET /api/projects/statistics



POST /api/projects/{projectId}/submit



POST /api/projects/{projectId}/reject



GET /api/projects/selectionValues



POST /api/projects/{projectId}/files



DELETE /api/projects/{projectId}/files/{fileId}



## Search



- GET /api/search/filterDetails
- GET /api/search/projects
- GET /api/search/projects/editable
- GET /api/search/projects/validation

### Schemas

```
ContributorDto {
  id integer($int32)
  description string
  nullable: true
  contributorType string
  nullable: true
  isEditor boolean
  projectId integer($int32)
  userId integer($int32)
  nullable: true
}
```

```
InstituteDto {
  id integer($int32)
  name string
  nullable: true
}
```

LinkDto

ProjectEditDto

ProjectFileDto

ProjectMetadataDto

RegionDto

**SearchTagDto**

**SignUpDto**

**UserDto**

**VerificationDto**

## 9.2.6 Auszug aus Dokumentation des Codes - Übersicht Interfaces

18.12.24, 22:46

OST ResearchNavigator Backend: Source.Services.Abstractions Namespace Reference

### Source.Services.Abstractions Namespace Reference

---

#### Classes

---

interface **IAccessControlService**

Defines methods for checking user access to resources. [More...](#)

interface **IAccountService**

Defines methods for managing user accounts, including registration, authentication, and user information retrieval. [More...](#)

interface **IFileService**

Defines the operations for managing project files, including adding, deleting, and generating presigned URLs. [More...](#)

interface **IInstituteService**

Defines operations for managing institutes. [More...](#)

interface **IProjectService**

Defines operations and businesslogic for creating, retrieving, updating, and managing projects. [More...](#)

interface **ISearchService**

Provides search functionalities for projects, including filtering and retrieving filter details. [More...](#)

interface **IValidationService**

Provides functionality for validating and managing project submissions and rejections. [More...](#)

class **SearchService**

---

Generated by  1.12.0

## 9.2.7 Auszug aus Dokumentation des Codes - IProjectService

18.12.24, 22:48

OST ResearchNavigator Backend: Source.Services.Abstractions.IProjectService Interface Reference

### Source.Services.Abstractions.IProjectService Interface Reference

Defines operations and businesslogic for creating, retrieving, updating, and managing projects. [More...](#)

#### Public Member Functions

Task< <b>ProjectDetailsDto</b> >	<b>GetByIdAsync</b> (int projectId, CancellationToken cancellationToken=default)	Retrieves the details of a project by its unique identifier.
Task< <b>ProjectEditDto</b> >	<b>GetByIdForEditAsync</b> (int projectId, CancellationToken cancellationToken=default)	Retrieves a project by its unique identifier in an editable form.
Task< <b>ProjectEditDto</b> >	<b>CreateAsync</b> (CancellationToken cancellationToken=default)	Creates a new project with default settings and returns it in an editable form.
Task< <b>ProjectEditDto</b> >	<b>UpdateAsync</b> ( <b>ProjectEditDto</b> projectEditDto, CancellationToken cancellationToken=default)	Updates the specified project with the provided data.
Task< <b>SelectionValuesDto</b> >	<b>GetSelectionValuesAsync</b> (CancellationToken cancellationToken=default)	Retrieves a set of selection values (e.g., institutes, regions, tags) used for populating project fields.
Task	<b>DeleteAsync</b> (int projectId, CancellationToken cancellationToken=default)	Permanently deletes the specified project.
Task< <b>StatisticsDto</b> >	<b>GetStatisticsAsync</b> ( <b>ProjectSearchRequestDto</b> searchRequestDto, CancellationToken cancellationToken=default)	Retrieves statistical data for projects based on the specified search criteria.
Task	<b>UpdateProjectValuesAsync</b> ( <b>Project</b> project, <b>ProjectEditDto</b> dto)	Updates the values of a given project entity based on the provided ProjectEditDto.

#### Detailed Description

Defines operations and businesslogic for creating, retrieving, updating, and managing projects.

#### Member Function Documentation

◆ **CreateAsync()**

Task< **ProjectEditDto** >

Source.Services.Abstractions.IProjectService.CreateAsync ( CancellationToken **cancellationToken** = default )

Creates a new project with default settings and returns it in an editable form.

**Parameters**

**cancellationToken** A token to observe while waiting for the task to complete.

**Returns**

A ProjectEditDto representing the newly created project.

**Exceptions**

**UnauthorizedAccessException** Thrown if the current user is not authorized to create a project.

◆ DeleteAsync()

Task Source.Services.Abstractions.IProjectService.DeleteAsync ( int **projectId**,  
CancellationToken **cancellationToken** = default )

Permanently deletes the specified project.

**Parameters**

**projectId** The unique identifier of the project to delete.

**cancellationToken** A token to observe while waiting for the task to complete.

**Returns**

A task that represents the asynchronous delete operation.

**Exceptions**

**NotFoundException** Thrown if the project does not exist.

**UnauthorizedAccessException** Thrown if the current user is not authorized to delete the project.

◆ GetByIdAsync()

Task< **ProjectDetailsDto** >

Source.Services.Abstractions.IProjectService.GetByIdAsync ( int **projectId**,  
CancellationToken **cancellationToken** = default )

Retrieves the details of a project by its unique identifier.

#### Parameters

**projectId** The unique identifier of the project.  
**cancellationToken** A token to observe while waiting for the task to complete.

#### Returns

A ProjectDetailsDto containing detailed information about the project.

#### Exceptions

**NotFoundException** Thrown if no project with the specified *projectId* is found.  
**UnauthorizedAccessException** Thrown if the current user is not authorized to view the project.

### ◆ GetByIdForEditAsync()

Task< **ProjectEditDto** >

Source.Services.Abstractions.IProjectService.GetByIdForEditAsync ( int **projectId**,  
CancellationToken **cancellationToken** = default )

Retrieves a project by its unique identifier in an editable form.

#### Parameters

**projectId** The unique identifier of the project.  
**cancellationToken** A token to observe while waiting for the task to complete.

#### Returns

A ProjectEditDto that can be used to modify project properties.

#### Exceptions

**NotFoundException** Thrown if the project does not exist.  
**UnauthorizedAccessException** Thrown if the current user is not authorized to edit the project.

### ◆ GetSelectionValuesAsync()

Task< **SelectionValuesDto** >

Source.Services.Abstractions.IProjectService.GetSelectionValuesAsync ( CancellationToken **cancellationToken** = default )

Retrieves a set of selection values (e.g., institutes, regions, tags) used for populating project fields.

**Parameters**

**cancellationToken** A token to observe while waiting for the task to complete.

**Returns**

A SelectionValuesDto containing lists of available selection values.

◆ **GetStatisticsAsync()**

Task< **StatisticsDto** >

Source.Services.Abstractions.IProjectService.GetStatisticsAsync ( **ProjectSearchRequestDto** searchRequestDto,  
CancellationToken **cancellationToken** = default )

Retrieves statistical data for projects based on the specified search criteria.

**Parameters**

**searchRequestDto** An object specifying the criteria for filtering projects.

**cancellationToken** A token to observe while waiting for the task to complete.

**Returns**

A StatisticsDto containing counts and other statistical data for matching projects.

◆ **UpdateAsync()**



Task< **ProjectEditDto** >  
Source.Services.Abstractions.IProjectService.UpdateAsync ( **ProjectEditDto** projectEditDto,  
CancellationToken cancellationToken = default )

Updates the specified project with the provided data.

**Parameters**

- projectEditDto** An object containing the updated project information.
- cancellationToken** A token to observe while waiting for the task to complete.

**Returns**

A ProjectEditDto reflecting the updated state of the project.

**Exceptions**

- NotFoundException** Thrown if the project does not exist.
- UnauthorizedAccessException** Thrown if the current user is not authorized to update the project.

#### ◆ UpdateProjectValuesAsync()

Task Source.Services.Abstractions.IProjectService.UpdateProjectValuesAsync ( **Project** project,  
**ProjectEditDto** dto )

Updates the values of a given project entity based on the provided ProjectEditDto.

**Parameters**

- project** The project entity to update.
- dto** The data transfer object containing updated project information.

**Returns**

A task that represents the asynchronous update operation.

The documentation for this interface was generated from the following file:

- Source/Services/Abstractions/**IProjectService.cs**