

Entwurfsdokumentation Digitales Werkverzeichnis

-

Softwareprojekt Wintersemester 2024/2025
Gruppe WSP3_1



Roni	Aba
Collin	Flügel
Jona	Giese
Sebastian	Kempe
Ricardo	Rieke
Emily	Schmidt

se///

14. Dezember 2025

Tipps und Hilfen

Information: Dieses Kapitel und alle folgenden grauen Boxen dienen als Hilfestellungen und sollen im fertigen Dokument nicht enthalten sein.

Zur Versionsverwaltung während des Softwareprojekts muss *Git* genutzt werden. *Git* führt Textdokumente mit unterschiedlichen Zeilenbearbeitungen automatisch zusammen. Wir empfehlen den Einsatz von \LaTeX für alle Textdokumente. Um das Auto-Merging zu unterstützen, sollte nach jedem Satzende eine neue Zeile im Quelltext begonnen werden. Die *.tex*-Datei dieser PDF verdeutlicht dies. Erkennt *Git*, dass eine gleiche Zeile bearbeitet wurde, wird ein Konflikt auftreten. Dieser kann in der entsprechenden Datei von Hand mittels eines Texteditors behoben werden.

Fußnoten¹ werden für Homepages genutzt. Zitierungen können mittels eines *cite*-Befehls gesetzt, z.B. *citep* [1].

Tipps zur UML-Modellierung können im SE-Wiki² nachgelesen werden. Achtet darauf, dass eure Diagramme stets lesbar (Vektor-Grafiken!) und gut strukturiert sind. Oftmals ist es sinnvoll ein bis zwei Sätze zusätzlich für Diagrammelemente zu formulieren. So können Missverständnisse ausgeschlossen werden, was einen Einfluss auf die Korrektur haben kann. Diagramme für unwichtige Tätigkeiten (z.B. Login / Logout, User erstellen / löschen, Passwort ändern etc.) sind nicht erforderlich. Achtet weiterhin darauf, dass interne Abläufe und Methodenaufrufe in den Frameworks nicht modelliert werden sollen. So soll z.B. das Sequenzdiagramm für das Erstellen einer Rechnung nicht interne Methoden und Abläufe des benutzten Frameworks abbilden, sondern nur diese Operationen zeigen, die Teil eures zukünftigen Codes sind.

¹<https://www.se.informatik.uni-kiel.de/en>

²<https://cau-git.rz.uni-kiel.de/ifi-ag-se/public/teaching/-/wikis/home>

So kann eine TODO-Notiz erzeugt werden

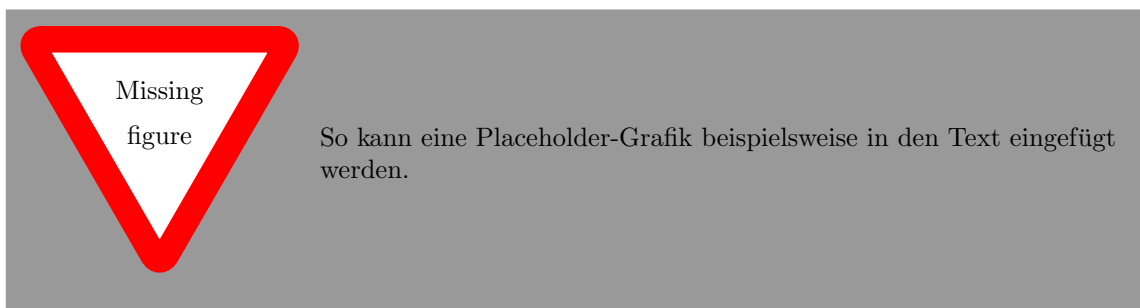


Abbildung 1: Beschreibung

Inhaltsverzeichnis

1	Einleitung	1
1.1	Entwicklungsumgebung	1
1.1.1	Einleitung	1
1.1.2	UML-Diagramme	2
2	Team-Aufteilung	3
3	Komponentendiagramme	4
4	Verteilungsdiagramm	6
5	Klassendiagramme	7
6	Sequenzdiagramme	9
7	Glossar	10

Kapitel 1

Einleitung

1.1 Entwicklungsumgebung

1.1.1 Einleitung

Nachfolgend finden Sie die vorläufige Dokumentation für die Entwicklung eines Werkverzeichnis für den Künstler Droege. Ziel dieses Systems ist es, die Anzeige und Filterung der Kunstwerke zu ermöglichen, Informationen über Droege bereitzustellen und eine Benutzerverwaltung zu ermöglichen.

Technologien

In der nachfolgenden Tabelle sind die Technologien, die zur Durchführung des Projekts eingesetzt werden, kurz dargestellt und beschrieben.

Technologie	Beschreibung
Git	Versionskontrollsystem zur kollaborativen Entwicklung
Java	Hauptprogrammiersprache für die Backend-Logik
Spring Boot	Framework zur Erstellung von Java-Webanwendungen
Thymeleaf	Template-Engine für die serverseitige Generierung von HTML
JAXB	Dependency/Library für XML Parsing
Haskell	Für die Automatisierung von Git-Skripte
Bash	Skripterstellung für Automatisierung und Deployment
Gradle	Build-Tool zur Abhängigkeitsverwaltung und Projektkompilierung
Docker	Deployment-Tool
HTML, CSS	Gestaltung und Strukturierung der Benutzeroberfläche
Jakarta Persistence	Für die Objektpersistenz

Technologie	Version	URL
Git	client dependant	git
Java	jdk21	java
Spring Boot	3.4.3	Spring Boot
Thymeleaf	3.1.3	Thymeleaf
JAXB	v2	JAXB
Haskell	The Glorious Glasgow Haskell Compilation System, version 9.4.8	Haskell/ghci
Bash	client-dependent	OS-vorinstalliert
Gradle	8.12.1	gradle
Docker	3.8	Docker
HTML, CSS	HTML5 & CSS3	Browser spezifisch
Jakarta Persistence	3.1.0	Jakarta

1.1.2 UML-Diagramme

Die Entwurfsdokumentation enthält verschiedene UML-Diagramme, die die Systemarchitektur darstellen, darunter:

- **Klassendiagramme:** Zeigen die Struktur der Klassen und deren Beziehungen.
- **Komponentendiagramme:** Veranschaulichen die logische Organisation der Softwaremodule.
- **Verteilungsdiagramme:** Beschreiben die physische Verteilung der Softwarekomponenten auf verschiedenen Systemen.
- **Sequenzdiagramme:** Stellen den Ablauf der Interaktionen zwischen den Systemkomponenten dar.

Diese Diagramme dienen dazu, die Architektur, Abhängigkeiten und Interaktionsabläufe des Systems verständlich zu machen.

Kapitel 2

Team-Aufteilung

Name	Zuständigkeit
Anna	...
Lukas	...
Lea	...
Jan	...
Sarah	...
Tim	...
Hannah	...
Finn	...

Hier soll die Arbeitsaufteilung im Entwicklungsteam angegeben werden. Die Zuständigkeit entspricht dabei auch dem Thema des Einzelvortrags, den jede(r) Teilnehmer(in) hält.

Kapitel 3

Komponentendiagramme

- Die Ansicht des Werkverzeichnisses ist offen und kann für Gäste der Seite auch ohne Log-In stattfinden. Die Komponente *Werkverzeichnisansicht* ermöglicht es, Motivgruppen und Motive zu wählen sowie zwischen den zugehörigen Werken zu wechseln. Dazu kommen die Filter-, Sortier- und Suchfunktionen des Werkverzeichnisses. Neben der standardmäßigen Galerieansicht kann auch zu einer Listenansicht gewechselt werden.
- Es können auch PDF-Steckbriefe zu den Werken exportiert werden. Diese Funktionalitäten stecken in der Komponente *PDF-Export*.
- Für die Anmeldung als Nutzer sowie als Admin gibt es eine *Authentifizierungskomponente*, die Zugriff auf weitere Komponenten gewährt. Sie übernimmt auch die Registrierung von neuen Nutzern und enthält von Spring bereitgestellte Funktionalitäten für die Nutzerverwaltung und Sicherheitskonfiguration.
- Ist der Nutzer angemeldet, hat er Zugriff auf die *Benutzerbereich-Verwaltungskomponente*. Diese ermöglicht es ihm, seine Suchanfragen sowie seine Notizen zu den Werken zu speichern und zu verwalten.
- Die *Kunstwerk-Verwaltungskomponente* fasst Funktionalitäten zusammen, die sowohl dem Nutzer als auch dem Admin zur Verfügung stehen. Der Nutzer kann Anfragen zum Hinzufügen, Editieren sowie Löschen eigener Werke stellen. Der Admin kann den Anfragen entweder zustimmen oder diese ablehnen. Außerdem verwaltet der Admin hier, von welchen Werken die Standorte angezeigt werden dürfen.
- Auf die *Benutzer-Verwaltungskomponente* hat nur der Admin Zugriff. Diese Komponente erlaubt es ihm, Nutzer zu löschen.
- In der Datenbank werden alle Produktdaten gespeichert. Zugriffe laufen über *Repository-Klassen*, die das von Spring zur Verfügung gestellte Repository-Interface erweitern.
- Die *Lido-Parser-Komponente* fasst die Funktionalitäten zusammen, die die Werkdaten der digicult-Datenbank abfragen und so aufbereiten, dass sie in unsere Datenbank überführt werden können.

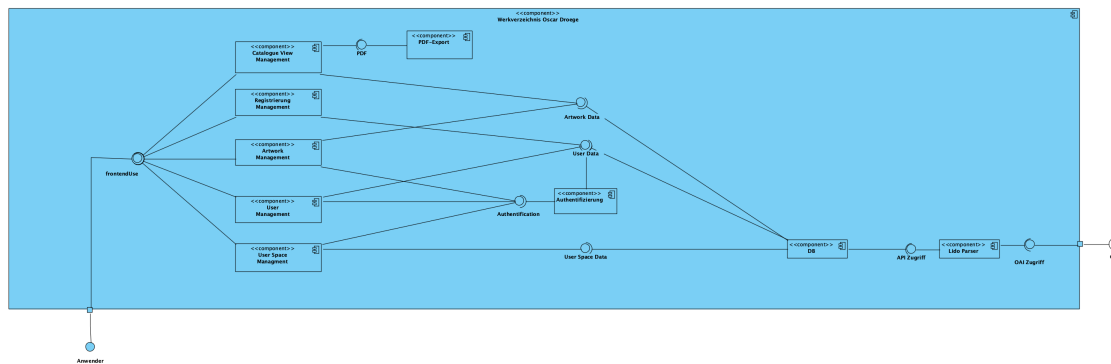


Abbildung 3.1: Komponentendiagramm - A

Kapitel 4

Verteilungsdiagramm

- Der Docker-Container auf dem Webserver, bzw. der virtuellen Maschine des Rechenzentrums stellt eine isolierte Umgebung mit allen Abhängigkeiten für die Anwendung dar.
- Die JAR inklusive der Datenbank laufen auf dem Tomcat-Applicationserver.
- Die Kommunikation zwischen dem Anwender und der Anwendung findet über HTTPS statt.
- Die Abfrage von Daten aus der digicult-Datenbank findet ebenfalls über HTTPS statt.

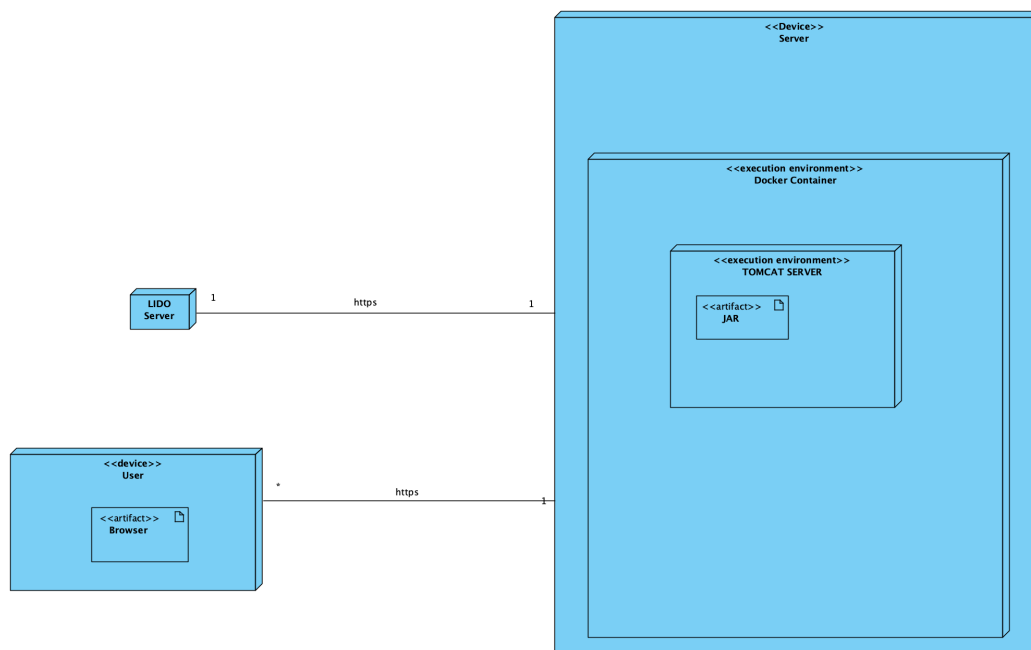


Abbildung 4.1: Komponentendiagramm - A

Kapitel 5

Klassendiagramme

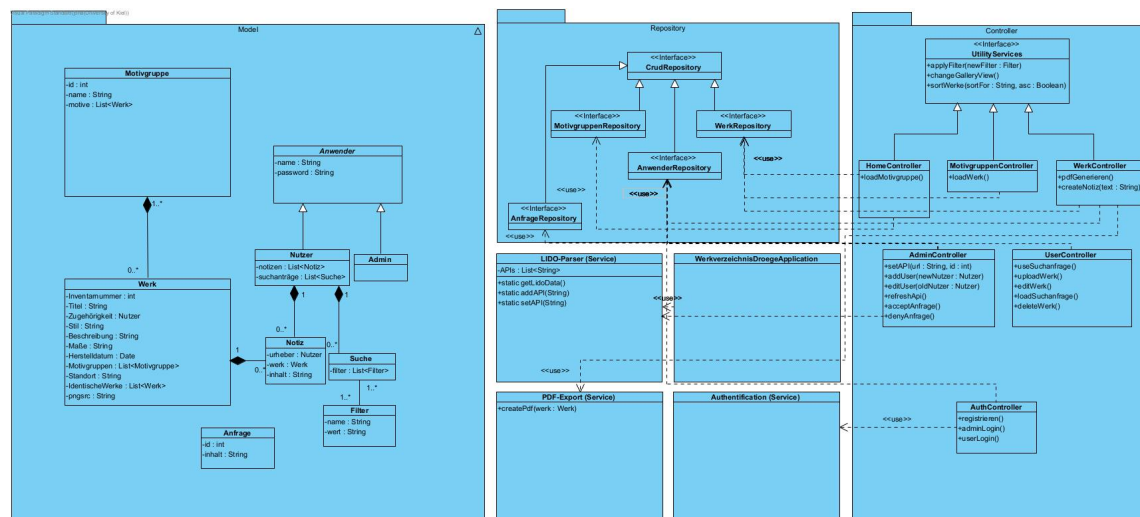


Abbildung 5.1: Klassendiagramm

Name	Aufgabe
(Package) Model	Enthält Klassen mit Grundstruktur verwendeter Daten.
(Package) Repository	Stellt verschiedene Datenbanken zum Speichern und Abrufen einzelner Daten zur Verfügung.
LIDOParser	Ein Service, welcher gespeicherte APIs durchgeht, Lido-Dateien abruft, in Werk-Objekte überführt und persistent speichert.
PDF-Export	Ein Service, welcher zu einem übergebenen Werk eine PDF-Datei erstellt
HomeController	Übernimmt die Aufgabe, die Interaktionen des Anwenders mit der Hauptseite zu verarbeiten. Z.B. das Wählen einer Motivgruppe.
MotivgruppenController	Übernimmt die Aufgabe, die Interaktionen des Anwenders mit der Motivgruppenseite zu verarbeiten. Z.B. das Wählen einer Referenzbildes.
MotivController	Übernimmt die Aufgabe, die Interaktionen des Anwenders mit der Werkseite zu verarbeiten. Z.B. das Erstellen einer PDF-Datei zu einem Werk.
AdminController	Übernimmt die Aufgabe, die Interaktionen des Admins mit der Adminseite zu verarbeiten. Z.B. Interaktionen mit Anwenderdatenbank oder aktualisieren der Werkdatenbank.
UserController	Übernimmt die Aufgabe, die Interaktionen des Nutzers mit der Nutzerseite zu verarbeiten. Z.B. Senden von Anfragen.
AuthController	Übernimmt Anfragen zum Registrieren oder Einloggen des Anwenders.

Tabelle 5.1: Klassenbeschreibung

Kapitel 6

Sequenzdiagramme

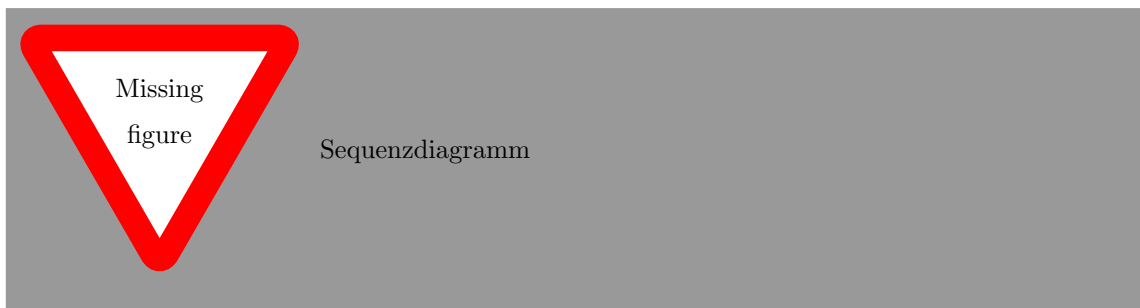


Abbildung 6.1: Sequenzdiagramm - A

Das dynamische Verhalten des Systems wird mittels Sequenzdiagrammen modelliert. Hier müssen wahrscheinlich geräteübergreifende Aufrufe modelliert werden. Findet dafür eine geeignete Notation und nutzt diese durchgehend! Achtet weiterhin darauf, dass die anderen Methoden im Klassendiagramm zu finden sind. Manche Sequenzen erfordern sicherlich eine kurze schriftliche Beschreibung.

Kapitel 7

Glossar

Literaturverzeichnis

- [1] Mary Shaw. Writing good software engineering research papers: minitutorial. In *Proceedings of the 25th International Conference on Software Engineering (ICSE 2003)*, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.