

---

# Projekt Ausleihbar

Software- und Plattformarchitektur

## Autoren

Andreas Burri  
Dominic Rohner  
Giuseppe Zaino  
Luca Zaugg  
Marcel Borter  
Séverin Kiener

## Dozent

Fabian Hirter

Abgabetermin: 09.03.2025

08.03.2025

Schweizerische  
Fachschule

**TEKO**

## Inhaltsverzeichnis

<b>1. Einleitung</b>	<b>3</b>
<b>2. Ausgangslage</b>	<b>3</b>
<b>3. Hauptteil</b>	<b>3</b>
<b>4. Variantenentscheid</b>	<b>4</b>
4.1. Backend . . . . .	4
4.2. Frontend . . . . .	5
<b>5. Ergebnisse</b>	<b>5</b>
5.1. Aktueller Stand . . . . .	6
<b>6. Diskussion</b>	<b>7</b>
<b>7. Empfehlungen</b>	<b>7</b>
<b>8. Ausblick</b>	<b>8</b>
<b>9. Quellen</b>	<b>9</b>
<b>Anhang</b>	<b>10</b>
<b>A. ADRs</b>	<b>10</b>
A.1. Codebase . . . . .	10
A.2. Markdown Linter . . . . .	11
A.3. Web Applikation . . . . .	12
A.4. Framework . . . . .	13
<b>B. Domain-Driven Design</b>	<b>14</b>
B.1. Ubiquitous Language . . . . .	14
B.2. Bounded Context . . . . .	14

## Abbildungsverzeichnis

1. Module Webapplikation . . . . .	4
2. Registrationsformular . . . . .	6

## **Glossar**

ADR	Architectural Decision Record
CI/CD	Continuous Delivery / Continuous Deployment
TDD	Test Driven Development
DDD	Domain Driven Development
DDoS	Distributed-Denial-of-Service-Angriffe
PHP	PHP Hypertext Preprocessor

## 1. Einleitung

In unserem Softwareprojekt entwickeln wir als Gruppe eine Webapplikation, mit der man Gegenstände ausleihen kann. Die Idee dahinter ist, dass Dinge, die man nur selten braucht, nicht von jedem gekauft werden müssen, sondern einfach geteilt werden können.

Um die Webseite zu erstellen, arbeiten wir in mehreren Schritten. Zuerst planen wir die wichtigsten Funktionen und die Softwarearchitektur. Da wir nach Test Driven Design vorgehen, schreiben wir anschliessend die Tests und beginnen die Webseite zu programmieren. Dabei legen wir besonderen Wert auf eine einfache Bedienung, eine sichere Verwaltung der Nutzerdaten und eine zuverlässige Plattform.

Dieses Projekt hilft uns, praktische Erfahrung in der Softwareentwicklung zu sammeln und zu lernen, wie man als Team zusammenarbeitet. Am Ende haben wir eine funktionierende Webseite, die das Ausleihen von Gegenständen erleichtert.

## 2. Ausgangslage

Im Rahmen unseres Kurses lernen wir die nötigen Fähigkeiten, um ein Softwareprojekt erfolgreich umzusetzen. Dazu gehört das Planen, die verschiedenen Softwarearchitekturen, die verschiedenen Plattformen und die Zusammenarbeit im Team. Unser Projekt bietet uns die Möglichkeit, dieses Wissen praktisch anzuwenden.

Um dieses Projekt umzusetzen, nutzen wir die im Unterricht erlernten Methoden und Werkzeuge, wie zum Beispiel Versionskontrolle, Markdown und CI/CD Pipelines.

In der Gruppe haben wir unterschiedliche Erfahrungen und Fähigkeiten. Deshalb ist auch das Wissen über die behandelten Themen sehr unterschiedlich verteilt.

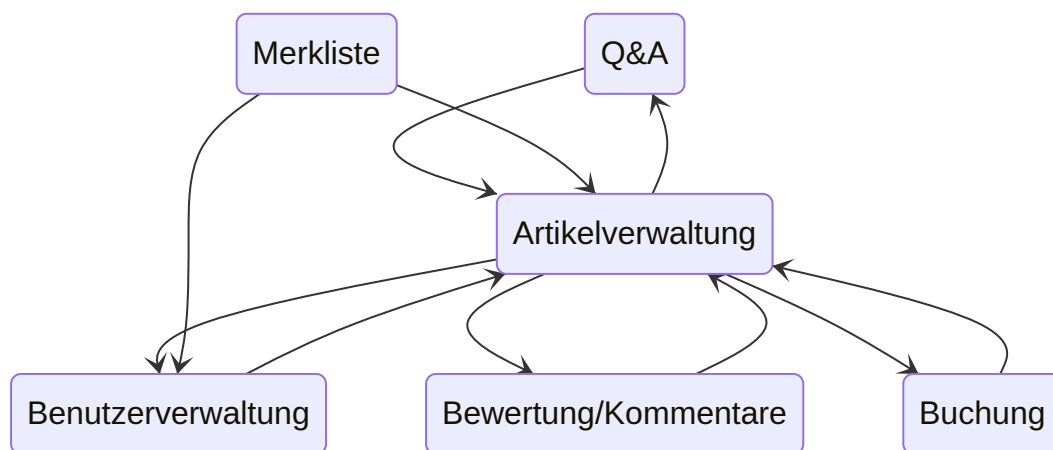
## 3. Hauptteil

In unserem Projekt entwickeln wir eine Webapplikation, über die Nutzerinnen und Nutzer verschiedene Gegenstände ausleihen können. Das primäre Ziel ist es, eine benutzerfreundliche Webseite bereitzustellen, die sowohl für die Anwender als auch für uns als Administratoren einfach zu bedienen und zu verwalten ist. Die Anwendung soll folgende Funktionen erfüllen:

- Benutzerverwaltung: Registrierung, Anmeldung und Verwaltung der Nutzerkonten.
- Artikelverwaltung: Auflistung der verfügbaren Artikel (z. B. Bücher, Werkzeuge, elektronische Geräte) mit detaillierten Informationen.

- Buchungssystem: Reservierungs- und Ausleihmanagement der Artikel.
- Bewertung/Kommentare: Bewertungen und Kommentare zu ausgeliehenen Artikeln
- Benachrichtigungsfunktion: Automatische Benachrichtigungen (E-Mail, Push-Benachrichtigungen) bei anstehenden Rückgaben oder Verzögerungen.
- UI: Ein übersichtliches und responsives Interface, das leicht zu navigieren ist.

Die Webapplikation wurde in verschiedene Module unterteilt worden. Abbildung 1 zeigt die entstandenen Module.



**Abbildung 1:** Module Webapplikation

## 4. Variantenentscheid

Bei der Wahl des Frontends als auch des Backends haben wir uns mit unterschiedlichen Varianten auseinandergesetzt, um eine gute und performante Lösung zu realisieren, welche alle unsere Anforderungen erfüllt.

### 4.1. Backend

Da in der Gruppe nicht grosse Erfahrungen mit Backend Frameworks vorhanden war, haben wir uns einige Framework angeschaut, welche wir bereits kannten. Zusätzlich haben wir auch im Internet nach weiteren Frameworks gesucht. Es stand auch zur Diskussion gar kein Framework zu verwenden.

Nach der Recherche und dem Austauschen der Erfahrungen in der Gruppe haben wir folgende drei Varianten genauer angeschaut:

- Laravel (PHP) [1]: Bietet ein bereits integriertes Benutzerverwaltungssystem, umfangreiche Dokumentation und eine grosse Community. Der Aufwand sich in Laravel einzuarbeiten ist zwar grösser, bietet aber bereits viele Funktionen an, welche benutzt werden können. Die Zeit welche bei der Einarbeitung in das Framework benötigt wird, kann aber bei der Verwendung der Laravel Funktionen wieder eingespart werden.
- Symfony (PHP) [2]: Bietet eine grosse Community. Ähnlich bei Laravel sind bereits viele Funktionen vorhanden. Jedoch wird für die Einarbeitung auch mehr Zeit benötigt.
- Express.js (Node.js) [3]: Sehr flexibel und schlank, erfordert jedoch mehr Konfigurationsaufwand für typische Aufgaben (z.B. Benutzerverwaltung).

Da Laravel bereits viele Module bietet, wie z.B Benutzerverwaltung, welche einfach in unsere Applikation eingebunden werden kann, haben wir uns für Laravel entschieden (siehe ADR im Anhang).

## 4.2. Frontend

Ähnlich wie beim Backend, war der Wissensstand über Frontend Frameworks auch sehr unterschiedlich.

Wir haben nach Konsolidierung der Frameworks, folgende Varianten etwas genauer angeschaut:

- Vue.js [4]: Kann einfach in unsere Webapplikation integriert werden. Bietet eine grosse Dokumentation. Einige Gruppenmitglieder haben damit bereits Erfahrungen gemacht.
- React [5]: Grosse Community und hohe Flexibilität. Allerdings ist das Einbinden zusätzlicher Bibliotheken für grundlegende Features (z. B. Routing, State Management) sehr häufig nötig, was unter Umständen zu höherer Komplexität führt.
- Angular [6]: Vollumfängliches Framework mit vielen Funktionalitäten. Für kleinere oder mittlere Projekte kann Angular jedoch oft überdimensioniert wirken.

Jedes Frontend Framework hat ihre eigenen Vor und Nachteile, da wir aber im Team bereits einige Erfahrungen mit Vue.js gemacht haben, haben wir uns für dieses Framework entschieden (siehe Anhang A.4).

## 5. Ergebnisse

Durch die Wahl von Laravel im Backend und Vue.js im Frontend haben wir eine solide Grundstruktur geschaffen, die den Anforderungen unserer Webanwendung erfüllt. Im Einzelnen sind folgende Resultate zu erwarten:

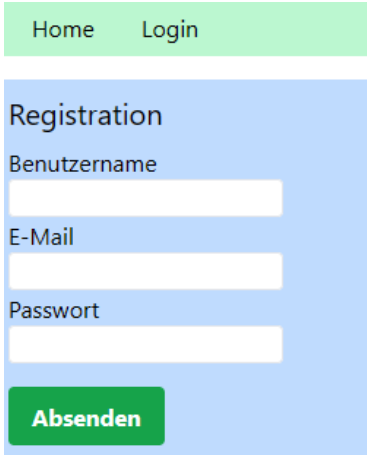
- Schnelle Entwicklung: Dank der out-of-the-box-Funktionen von Laravel und den leicht erweiterbaren Vue-Komponenten können wir zügig mit unserem Projekt starten.

- Benutzerverwaltung: Die bereits in Laravel eingebundene Benutzerverwaltung sorgt für ein solides Sicherheitsfundament. Rollen- und Rechteverwaltung lassen sich unkompliziert erweitern.
- Erweiterbares System: Sowohl Laravel als auch Vue.js ermöglichen eine modulare Architektur. Neue Funktionen (z. B. zusätzliche Kategorien, Buchungen, Benachrichtigungsmethoden) lassen sich schnell integrieren, ohne das gesamte System umstrukturieren zu müssen.
- Gutes Benutzererlebnis: Durch das responsive Design und das reaktive Frontend ist eine hohe Nutzerfreundlichkeit sichergestellt. Die Darstellung von verfügbaren Artikeln erleichtert dem Benutzer die Bedienung der Applikation.
- Zukunftssicherheit: Beide Technologien haben eine grosse Community und werden aktiv weiterentwickelt. Damit können wir sicherstellen, dass sowohl Updates als auch zukünftige Erweiterungen problemlos durchgeführt werden können.

Mit Laravel als Backend-Framework und Vue.js als Frontend-Framework sind wir in der Lage, eine leistungsstarke, benutzerfreundliche und erweiterbare Webapplikation zum Ausleihen von Gegenständen zu realisieren.

## 5.1. Aktueller Stand

Während der Projektarbeit wurde die Benutzerverwaltung mit Registrations- und Loginfunktionalität implementiert. Zudem konnte ein einfaches Dashboard mit Funktionalitäten zum Erstellen, Bearbeiten und Löschen von Gegenständen implementiert werden. Abbildung 2 zeigt das erstellte Formular zur Benutzerregistration im Frontend.



Home Login

Registration

Benutzername

E-Mail

Passwort

Absenden

**Abbildung 2:** Registrationsformular

## 6. Diskussion

Im Verlauf der Entwicklung unseres Projekts haben wir verschiedene Herausforderungen gemeistert und wertvolle Erfahrungen gesammelt.

Eine der grössten Herausforderungen war die Wahl der richtigen Technologien für unser Vorhaben. Da das Team unterschiedliche Erfahrungsstände in der Softwareentwicklung hatte, war es wichtig, eine Lösung zu finden, mit der man schnell Fortschritte erzielen kann, aber trotzdem alle Funktionen unserer Anwendung erfüllen kann. Zudem sollte eine ausführliche Dokumentation für die Einarbeitung vorhanden sein.

Die Entscheidung für Laravel als Backend-Framework und Vue.js als Frontend-Framework hat sich als sinnvoll erwiesen, da wir mit diesen Technologien schnell Fortschritte erzielen konnten und bereits existierende Module verwenden konnten. Allerdings stellte die fehlende Erfahrung mit Laravel und Vue.js für viele Teammitglieder eine Herausforderung dar. Die Einarbeitung in diese Technologien erforderte Zeit und eine intensive Auseinandersetzung mit den jeweiligen Frameworks. Dennoch hat sich diese Entscheidung langfristig bewährt, da Laravel bereits viele nützliche Module, wie die Benutzerverwaltung, mitbringt, die wir direkt nutzen konnten. Dies sparte Entwicklungszeit und ermöglichte eine effizientere Umsetzung der geplanten Funktionen. Rückblickend würden wir diese Technologieentscheidung erneut treffen, da sie uns eine stabile und erweiterbare Basis für unsere Anwendung geboten hat.

Ein weiterer wichtiger Punkt war die Zusammenarbeit im Team. Die Anwendung von Versionskontrolle und Continuous Integration/Continuous Deployment (CI/CD) hat dazu beigetragen, eine effiziente Arbeitsweise zu etablieren. Dennoch gab es Herausforderungen, insbesondere in der Koordination von Aufgaben und der Sicherstellung einer einheitlichen Codebasis. Das regelmässige Absprechen der Frontend und Backend Gruppe haben jedoch geholfen, diese Probleme zu minimieren.

Die Tests, die im Rahmen des Test Driven Designs (TDD) erstellt wurden, haben dazu beigetragen, die Qualität der Software sicherzustellen und Fehler frühzeitig zu identifizieren. Jedoch konnten nicht für alle Funktionen die Tests geschrieben werden.

## 7. Empfehlungen

Aufgrund unserer Erfahrungen im Projekt empfehlen wir für ähnliche Projekte folgende Vorgehensweisen:

- Frühe Technologieentscheidung: Eine frühe Festlegung auf geeignete Technologien erleichtert die Einarbeitung und vermeidet spätere Umstellungen, die Zeit und Ressourcen kosten.



- Testgetriebene Entwicklung (TDD): Die Einhaltung von TDD hilft, eine stabile und fehlerfreie Anwendung zu entwickeln und macht es einfacher, neue Features hinzuzufügen.
- Effektive Kommunikation im Team: Regelmässige Meetings und klare Aufgabenverteilung reduzieren Missverständnisse und steigern die Produktivität.
- Modularer Aufbau der Software: Eine modulare Architektur erleichtert die Wartung und Erweiterung der Anwendung.
- Benutzerzentriertes Design: Die Benutzerfreundlichkeit sollte stets im Mittelpunkt stehen, da eine intuitive Bedienung die Akzeptanz der Anwendung erhöht.

## 8. Ausblick

Unser Projekt hat eine solide Grundlage geschaffen, auf der zukünftige Erweiterungen aufbauen können. In der weiteren Entwicklung könnten folgende Verbesserungen und neue Funktionen umgesetzt werden:

- Deployment und Testing via CI/CD Pipelines: Das gesamte Deployment und Testing sollte via CI/CD Pipelines gemacht werden. Damit wird sichergestellt, dass neue Features effizient integriert, getestet und ausgerollt werden. Automatisierte Tests und Deployments würden dabei helfen, Fehler frühzeitig zu erkennen und die Stabilität der Anwendung zu gewährleisten.
- Erweiterte Such- und Filterfunktionen: Eine bessere Kategorisierung und Filterung der Artikel würde die Navigation und das Auffinden relevanter Gegenstände erleichtern.
- Automatisiertes Benachrichtigungssystem: Die Implementierung der automatisierten Benachrichtigungen für Rückgaben und Erinnerungen steht noch aus und müssten in einer späteren Entwicklungsphase umgesetzt werden.
- Erweiterung auf weitere Anwendungsbereiche: Neben physischen Gegenständen könnten auch Dienstleistungen oder digitale Inhalte verliehen werden.
- Sicherheitsüberprüfung: Vor einer finalen Veröffentlichung der Webapplikation müssen sämtliche sicherheitsrelevanten Aspekte wie das Abspeichern der Benutzerdaten und Passwörter nochmals überprüft werden, um Datenschutz, Zugriffskontrollen und Schutz vor Angriffen sicherzustellen.
- Schutz vor DDoS-Angriffen: Um die Verfügbarkeit der Plattform zu gewährleisten, sollten geeignete Schutzmassnahmen gegen DDoS implementiert werden.
- Mobile App: Eine App für Mobile-Phones könnte die Nutzungsmöglichkeiten erweitern und die Benutzerfreundlichkeit verbessern.

Mit diesen Erweiterungen kann die Plattform weiterentwickelt und noch attraktiver für die Nutzer gemacht werden.

Durch unser Projekt konnten wir nicht nur technische Fähigkeiten erlernen, sondern auch die im Unterricht erlernten Konzepte in der Praxis anwenden. Diese Erfahrung wird uns in zukünftigen Softwareentwicklungsprojekten von Nutzen sein.

## 9. Quellen

- [1] *Laravel - The PHP Framework for Web Artisans*. URL <https://laravel.com/docs/12.x>. - abgerufen am 2025-02-28
- [2] *Symfony - High Performance PHP Framework for Web Development*. URL <https://symfony.com/doc/7.2/index.html>. - abgerufen am 2025-02-28
- [3] *Express.js: Node.js Web Application Framework*. URL <https://expressjs.com/>. - abgerufen am 2025-02-28
- [4] *Vue.js - Introduction*. URL <https://vuejs.org/guide/introduction.html>. - abgerufen am 2025-02-28
- [5] *React: Quickstart*. URL <https://react.dev/learn>. - abgerufen am 2025-02-28
- [6] *Angular - What is Angular?* URL <https://angular.dev/overview>. - abgerufen am 2025-02-28

## Anhang

### A. ADRs

#### A.1. Codebase

Datum: 2024-11-07

##### A.1.1. Status

Status: Accepted on 2024-11-07

##### A.1.2. Kontext

Wir benötigen eine gemeinsame Codebasis (Repository), über die wir effizient Code austauschen, gemeinsam arbeiten und den Code versionieren können. Ein weiterer wichtiger Punkt, ist die Möglichkeit, bei jedem Commit automatisierte Pipelines auszuführen und frühzeitig mögliche Fehler zu erkennen. Eine solche Lösung sollte möglichst einfach für das Team zu bedienen sein.

##### A.1.3. Entscheid

Nach eingehender Recherche und Diskussionen haben wir uns für die Nutzung von Github entschieden. Github bietet eine ausgereifte Versionsverwaltung mittels Git, unterstützt kollaboratives Arbeiten und stellt Hilfsmittel zur Dokumentation zur Verfügung. Zudem sind einige Teammitglieder bereits mit Github vertraut, was die Einarbeitungszeit verkürzt. Ein Vorteil ist die CI/CD Funktion über "Github Actions", die wir nutzen werden um die automatisierten Pipelines bei jeder Codeänderung auszuführen. Dadurch können wir eine gewisse Qualitätssicherung unseres Projekts etablieren.

##### A.1.4. Konsequenzen

Wir verwenden GitHub als zentrale Codebasis mit Git zur Versionskontrolle und setzen auf Pull Requests für transparente Code Reviews. Github Actions ermöglicht uns automatisierte CI/CD-Pipelines, um Builds, Tests und Deployments effizient abzuwickeln.

## A.2. Markdown Linter

Datum: 2024-11-14

### A.2.1. Status

Status: Accepted on 2024-11-14

### A.2.2. Kontext

Wir möchten unsere Markdown Dateien möglichst konsistent und fehlerfrei halten. Dazu gehört eine einheitliche Formatierung wie auch das Sicherstellen, dass keine Syntax Fehler und Fehler in der Struktur der Markdown-Datei entstehen. Dafür haben wir einen geeigneten Linter gesucht, der uns bei der Qualitätskontrolle unserer Markdown-Dokumente unterstützt.

### A.2.3. Entscheid

Nach Prüfung verschiedener Optionen haben wir uns für das *markdownlint-cli* aus dem NPM Repo entschieden. Dadurch lassen sich die Markdown Dateien automatisiert überprüfen und einheitlich formatieren.

### A.2.4. Konsequenzen

Wir setzen den *markdownlint-cli* Linter für unser Projekt ein. Dadurch arbeiten alle Teammitglieder einheitlich und bei Bedarf können wir den Linter anpassen oder austauschen. Mit dem Linter werden die Dokumente auf Fehler überprüft.

### **A.3. Web Applikation**

Datum: 2024-11-28

#### **A.3.1. Status**

Status: Accepted on 2024-11-28

#### **A.3.2. Kontext**

Wir haben uns Gedanken über verschiedene Lösungsansätzen gemacht und dabei den Fokus darauf gelegt, wie sich unsere Anwendung auf verschiedene Endgeräten bereitstellen lässt. Eine Web-Applikation bietet folgende entscheidende Vorteile:

- Eine Codebase für alle Geräte
- Kann breit gebraucht werden
- Wendet sich an Nutzer mit beliebigen Geräten
- Keine Installation nötig

#### **A.3.3. Entscheid**

Auf Basis dieser Überlegungen haben wir beschlossen, eine Webapplikation zu entwickeln. Diese sollte von allen gängigen Browsern unterstützt werden und responsive sein.

#### **A.3.4. Konsequenzen**

Wir erstellen eine plattformunabhängige Webanwendung die ohne Installation auskommt und sich leicht aktualisieren lässt.

## **A.4. Framework**

Datum: 2024-12-12

### **A.4.1. Status**

Status: Accepted on 2024-12-12

### **A.4.2. Kontext**

Im Zuge unserer ersten Implementierungen mussten wir ein Framework auswählen, das unsere Anforderungen an Responsivness, Entwicklerfreundlichkeit und Performance erfüllt. Bei unseren Recherchen haben wir nicht nur die technische Leistungsfähigkeit verschiedener Frameworks, sondern auch deren Community-Support und vorhandene Dokumentation überprüft. Da wir eine solide Basis für die Entwicklung unserer Webanwendung schaffen wollten, waren insbesondere integrierte Lösungen für wiederkehrende Aufgaben wie Authentifizierung und Benutzerverwaltung von Bedeutung, zudem muss das Framework folgende Anforderungen erfüllen:

- Support für alle gängigen Browser
- Responsivness
- Entwicklerfreundlich
- Gute Performance

### **A.4.3. Entscheid**

Nach der Auswertung unserer Internet Recherchen und einer ersten kurzen Evaluierung haben wir uns entschieden für das Backend das Framework Laravel (<https://laravel.com/>) einzusetzen. Laravel ist ein verbreitetes PHP-Framework mit einer grossen Community und Fokus auf Entwicklerfreundlichkeit bietet. Laravel bringt bereits eine nutzerfreundliche Authentifizierungs- und Benutzerverwaltung mit sich. Für das Frontend setzen wir Vue ein (<https://vuejs.org/>). Vue ermöglicht mit seiner flexiblen und komponentenbasierten Architektur, schnelle Entwicklungszeiten und gute Performance. Zudem bietet es genügend Dokumentation um sich einzuarbeiten.

### **A.4.4. Konsequenzen**

Wir setzen PHP mit dem Framework Laravel für die serverseitige Logik ein. Vue wird im Frontend eingesetzt mit dem sich Funktionen schnell umsetzen lassen.

## B. Domain-Driven Design

### B.1. Ubiquitous Language

Da wir im Team in Deutsch kommunizieren, jedoch in Englisch programmieren, haben wir die Begriffe in beiden Sprachen definiert.

Folgende Begriffe wurden definiert:

Deutsch	Englisch
Benutzerverwaltung	User management
Artikelverwaltung	Thing management
Benutzer	User
Benachrichtigung	Notification
Bewertung	Review
Buchung	Reservation
Merkliste	Wishlist
Kommentare	Comments
Fragen & Antworten	Q&A

Folgende Begriffe können unterschiedliche Bedeutungen haben:

- Bewertung (Benutzer, Artikel und App)

### B.2. Bounded Context

Wir haben nur einen Kontext in unserer App.