GÖRRES-GYMNASIUM KOBLENZ

BESONDERE LERNLEISTUNG

Konzept und Implementierung einer neuen quelloffenen Produktlösung für den schuleigenen DSBmobile-Vertretungsplan mithilfe von Webscraping

---

Autor:
Philipp Pertermann

Lehrkraft:
Jochen Sauer

25. August 2025

Eidesstattliche Erklärung

Hiermit erkläre ich, Philipp Pertermann, dass ich die vorliegende Arbeit mit dem Titel

„Konzept und Implementierung einer neuen quelloffenen Produktlösung für den schuleigenen DSBmobile-Vertretungsplan mithilfe von Webscraping“

selbstständig verfasst und keine anderen als die angegebenen Quellen benutzt habe. Die Stellen der Arbeit sowie eventuell beigefügte Zeichnungen, Skizzen, graphische Darstellungen oder Daten, die anderen Werken entnommen oder an diese angelehnt sind, habe ich unter Angabe der Quelle als Entlehnung kenntlich gemacht.

Datum: _____

Unterschrift: _____

GÖRRES-GYMNASIUM KOBLENZ

Kurzfassung

Besondere Lernleistung

Konzept und Implementierung einer neuen quelloffenen Produktlösung für den schuleigenen DSBmobile-Vertretungsplan mithilfe von Webscraping

von Philipp Pertermann

Die vorliegende Besondere Lernleistung behandelt die Konzeption und Umsetzung eines Systems zur verbesserten Darstellung des schulischen Vertretungsplans auf Basis der bestehenden DSBmobile-Infrastruktur. Ausgangspunkt war die Beobachtung, dass die bisherige Lösung zwar grundsätzlich funktioniert, im Alltag aber durch eine umständliche Navigation, unübersichtliche Tabellen, fehlende Personalisierung und eingeschränkte Verfügbarkeit oft nicht den Bedürfnissen der Nutzer entspricht.

Das Ziel war daher, eine nutzerfreundliche und quelloffene Erweiterung zu entwickeln, die die bestehenden Daten automatisch übernimmt, aufbereitet und in verständlicher Form darstellt. Dazu wurde eine Architektur entworfen, die die Daten neu strukturiert, übersichtlich darstellt und so den Zugang deutlich vereinfacht. Ergänzend sorgen Mechanismen wie Zwischenspeicherung, automatische Aktualisierung und eine entkoppelte Systemstruktur für Stabilität und Verlässlichkeit.

Im Vordergrund steht dabei die Benutzerfreundlichkeit: Nach der Anmeldung werden die relevanten Informationen sofort sichtbar, Abkürzungen aufgelöst und Unterrichtsausfälle klar hervorgehoben. Durch ein anpassungsfähiges Design bleibt die Anwendung sowohl auf Smartphones als auch auf Desktop-Rechnern gut nutzbar.

Auch Fragen der Sicherheit und des Datenschutzes wurden berücksichtigt: Der Zugang zu den Daten ist geschützt, sensible Inhalte werden nicht veröffentlicht, und in dieser Arbeit wurden alle personenbezogenen Angaben anonymisiert.

Das Ergebnis zeigt, dass ein ursprünglich geschlossenes Informationssystem durch eine moderne, modulare Erweiterung deutlich verbessert werden kann – sowohl in seiner technischen Verlässlichkeit als auch in der täglichen Nutzung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
BDSG	Bundesdatenschutzgesetz
CI/CD	Continuous Integration / Continuous Deployment
CSV	Comma-Separated Values
DOM	Document Object Model
DSGVO	Datenschutz-Grundverordnung
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICS	Internet Calendaring and Scheduling Format (.ics)
JSON	JavaScript Object Notation
JWT	JSON Web Token
OCR	Optical Character Recognition
PDF	Portable Document Format
PWA	Progressive Web App
REST	Representational State Transfer
RSS	Really Simple Syndication
UI	User Interface
XSS	Cross-Site Scripting

1 Einleitung

1.1 Problemstellung

Die digitale Transformation des Schulalltags birgt großes Potenzial zur Optimierung administrativer Prozesse. Ein zentrales Element ist die digitale Bereitstellung des Vertretungsplans, der Schülern und Lehrkräften zeitnah Informationen über Unterrichtsausfälle, -verschiebungen und -vertretungen vermittelt. Seit einigen Jahren werden diese Informationen am hiesigen Gymnasium über die Plattform DSBmobile bereitgestellt, wodurch ein mobiler Zugriff jederzeit auch außerhalb der Schule via Smartphone oder Computer möglich wurde.

Der Autor nutzt als Schüler selbst seit der fünften Klasse täglich diese digitale Lösung, um Veränderungen im Stundenplan zu verfolgen. Die fortlaufende Nutzung hat ihm einen fundierten Einblick in die Funktionalität und die Grenzen des bestehenden Systems ermöglicht. Die tägliche Auseinandersetzung mit DSBmobile zeigt Potenziale zur Optimierung auf, die sowohl die Effizienz steigern als auch das Nutzererlebnis verbessern können.

1.2 Motivation

Eine zentrale Herausforderung des aktuellen Systems liegt in seiner begrenzten Informationsarchitektur: Statt einer personalisierten, nutzerorientierten Darstellung müssen Anwender durch eine Vielzahl von Daten navigieren, um ihre spezifischen Informationsbedürfnisse zu stillen. Dies resultiert in einem vermeidbaren Zeitaufwand – ein Faktor, der angesichts der täglichen Nutzung durch die gesamte Schulgemeinschaft beträchtliche kumulative Auswirkungen hat.

Die Benutzerfreundlichkeit leidet zusätzlich unter einem schulinternen System von Abkürzungen. Insbesondere neue Schüler und Eltern stehen vor der Herausforderung, ein komplexes System aus Kürzeln für Lehrkräfte und Unterrichtsfächer zu entschlüsseln. Diese Umstände führen nicht nur zu Frustration, sondern erhöht auch das Risiko von Missverständnissen und Fehlinterpretationen im Schulalltag. Darüber hinaus wird der Aspekt der Barrierefreiheit, der insbesondere im öffentlichen Dienst einen großen Stellenwert besitzt, bislang nicht hinreichend berücksichtigt.

Aus technischer Perspektive entspricht die aktuelle Implementierung nicht den Qualitätsstandards moderner Softwareentwicklung: Nutzer berichten regelmäßig von Ladeproblemen und temporären Systemausfällen. Besonders kritisch sind diese technischen Unzulänglichkeiten für Schüler und Eltern, die morgens vor Schulbeginn überprüfen möchten, ob die erste Unterrichtsstunde entfällt, um unnötige Wartezeiten vor dem Unterricht zu vermeiden.

1.3 Zielsetzung und zentrale Fragestellung

Aus der Analyse der Schwachstellen lassen sich klare Handlungsfelder ableiten. Die vorliegende *Besondere Lernleistung* (BLL) setzt hier an und entwickelt ein nutzerorientiertes System, das auf der bestehenden DSBmobile-Infrastruktur basiert.

Dieser Ansatz vermeidet zusätzlichen administrativen Implementierungsaufwand und verbessert zugleich die Nutzererfahrung erheblich.

Das vorrangige Ziel dieser Arbeit ist die Entwicklung einer dynamischen Webanwendung, die mittels Webscraping-Technologie die Daten des schulischen Vertretungsplans extrahiert, diese in einer quelloffenen Architektur verarbeitet und dem Nutzer in personalisierter, übersichtlicher Form präsentiert. Durch diesen Ansatz soll die folgende zentrale Fragestellung beantwortet werden:

Wie kann ein bestehendes, geschlossenes Informationssystem durch moderne Softwaretechnologien und nutzerzentriertes Design so erweitert werden, dass es den Anforderungen aller Nutzer gerecht wird?

1.4 Methodisches Vorgehen und Aufbau der Arbeit

Die vorliegende Arbeit verfolgt einen systematischen Ansatz zur Entwicklung der beschriebenen Lösung. Im Anschluss an diese Einleitung werden im zweiten Kapitel die theoretischen Grundlagen erläutert, insbesondere die Funktionsweise von DSBmobile, Techniken des Webscrapings sowie rechtliche und ethische Aspekte.

Das dritte Kapitel widmet sich der Anforderungsanalyse, wobei die Bedürfnisse aller Nutzer erfasst werden. Im vierten Kapitel wird die Konzeption des Systems dargelegt, gefolgt von der technischen Implementierung im fünften Kapitel.

Die Arbeit schließt mit einem Fazit im sechsten Kapitel, in dem die Ergebnisse zusammengefasst und Potenziale für zukünftige Weiterentwicklungen aufgezeigt werden.

2 Theoretische Grundlagen

2.1 DSBmobile: Funktionsweise, Tabellenstruktur, Limitierungen

2.1.1 Systemüberblick

Die Internetanwendung DSBmobile ist ein Produkt der Firma heinekingmedia GmbH. Es stellt eine Kommunikationslösung für Bildungseinrichtungen zur digitalen Bereitstellung von Vertretungsplänen und weiteren organisatorischen Informationen dar. Das System fungiert als digitales schwarzes Brett und stellt digital Dokumente für Schüler und Lehrkräfte bereit.

Die wesentliche Funktionalität von DSBmobile besteht in der strukturierten Darstellung von Vertretungsplänen in Form einer webbasierten Anwendung, die sowohl über Browser als auch über native mobile Apps zugänglich ist. Hierbei werden die ausgewählten Informationen in tabellarischer Form präsentiert, wodurch sich Schüler, Lehrkräfte und Eltern über Änderungen im Schulalltaginformieren können.

2.1.2 Navigationsstruktur

Der Zugang zum System erfolgt über einen Authentifizierungsprozess, bei dem sich Nutzer mit schulspezifischen Zugangsdaten anmelden müssen. Je nach Nutzerrolle (Schüler oder Lehrkraft) werden unterschiedliche Zugangsberechtigungen gewährt, die den Umfang der einsehbaren Informationen bestimmen.

Nach erfolgreicher Authentifizierung präsentiert sich dem Nutzer eine zweigliedrige Hauptnavigation, bestehend aus den Bereichen „Pläne“ und „Aushänge“:

1. **Pläne**: Dieser Bereich stellt grundlegende, längerfristig gültige Dokumente zur Verfügung, darunter
   - Raumpläne
   - Stundenpläne der einzelnen Stufen
   - In der Lehreransicht zusätzlich: spezifische Stundenpläne für einzelne Lehrkräfte und Klassen sowie Aufsichts- und Übersichtslisten
2. **Aushänge**: Hier werden regelmäßig aktualisierte Informationen bereitgestellt, insbesondere
   - Der Vertretungsplan unter der Bezeichnung „DaVinci Touch“
   - Wöchentlich aktualisierte Aufsichtspläne für Klausuren der Oberstufe

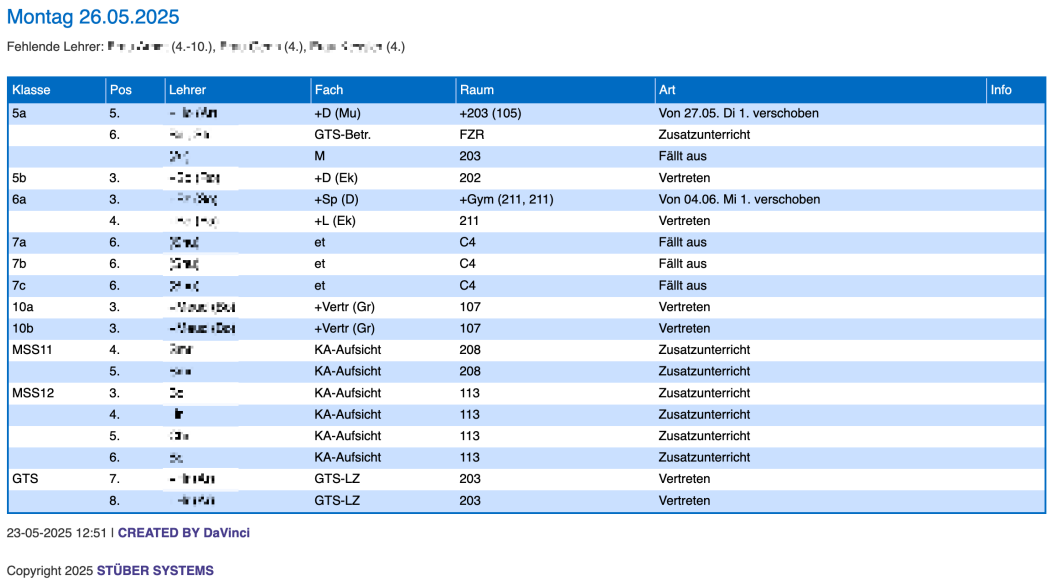
In der Mobilversion sind zusätzlich folgende Bereiche sichtbar:

3. **Startseite**: Wird nicht genutzt und bleibt leer
4. **News**: Könnte zusätzliche Informationen anzeigen, wird derzeit jedoch nicht verwendet

DSBmobile ist hier integriert mit der Zeitplanungssoftware "DaVinci" der Firma Stüber Systems GmbH. DSBmobile fungiert in dieser Konstellation primär als Verteiler für die in DaVinci erstellten Vertretungsinformationen.

2.1.3 Informationsstruktur des Vertretungsplans

Der Vertretungsplan selbst ist nach Tagen strukturiert, wobei jeder Tag eine separate Seite mit einer tabellarischen Darstellung der Vertretungsinformationen umfasst.



Montag 26.05.2025

Fehlende Lehrer: [redacted] (4.-10.), [redacted] (4.), [redacted] (4.)

Klasse	Pos	Lehrer	Fach	Raum	Art	Info
5a	5.	[redacted]	+D (Mu)	+203 (105)	Von 27.05. Di 1. verschoben	
	6.	[redacted]	GTS-Betr.	FZR	Zusatzunterricht	
		[redacted]	M	203	Fällt aus	
5b	3.	[redacted]	+D (Ek)	202	Vertreten	
6a	3.	[redacted]	+Sp (D)	+Gym (211, 211)	Von 04.06. Mi 1. verschoben	
	4.	[redacted]	+L (Ek)	211	Vertreten	
7a	6.	[redacted]	et	C4	Fällt aus	
7b	6.	[redacted]	et	C4	Fällt aus	
7c	6.	[redacted]	et	C4	Fällt aus	
10a	3.	[redacted]	+Vertr (Gr)	107	Vertreten	
10b	3.	[redacted]	+Vertr (Gr)	107	Vertreten	
MSS11	4.	[redacted]	KA-Aufsicht	208	Zusatzunterricht	
	5.	[redacted]	KA-Aufsicht	208	Zusatzunterricht	
MSS12	3.	[redacted]	KA-Aufsicht	113	Zusatzunterricht	
	4.	[redacted]	KA-Aufsicht	113	Zusatzunterricht	
	5.	[redacted]	KA-Aufsicht	113	Zusatzunterricht	
	6.	[redacted]	KA-Aufsicht	113	Zusatzunterricht	
GTS	7.	[redacted]	GTS-LZ	203	Vertreten	
	8.	[redacted]	GTS-LZ	203	Vertreten	

23-05-2025 12:51 | CREATED BY DaVinci

Copyright 2025 STÜBER SYSTEMS

ABBILDUNG 2.1: Quelle: DSBmobile, 2025

Diese Tabellenstruktur ist das zentrale Element des Vertretungsplans und weist folgende Standardspalten auf:

Spaltenbezeichnung	**Beispielwert**	**Bedeutung**
Klasse	10a	Betroffene Schulklasse
Pos	3.	Unterrichtsstunde (Position der Stunde)
Lehrer	Mu	Kürzel der betroffenen Lehrkraft
Fach	M	Abkürzung des Unterrichtsfachs
Raum	202	Raumbezeichnung
Art	Fällt aus	Art der Vertretungsregelung
Info	*variabel*	Zusätzliche Hinweise

TABELLE 2.1: Datenfelder (Spalten) der Vertretungsdaten

Innerhalb dieser Tabellenstruktur werden schulinterne Abkürzungen und Kürzel verwendet, die nicht selbsterklärend sind:

1. **Fach**: Diese werden mit Abkürzungen dargestellt (z.B. „M" für Mathematik, „Sk" für Sozialkunde). Diese sind im Übrigen auch auf den Raum- und Stundenplänen zu finden.
2. **Kurs**: Insbesondere in der Oberstufe werden statt Fachbezeichnungen spezifische Kursbezeichnungen verwendet (beispielsweise „SK1" für den Leistungskurs eins im Fach Sozialkunde).

3. **Lehrerkürzel**: Lehrkräfte werden durch zweibuchstabige, initialenbasierte Kürzel identifiziert. Diese Kürzel sind über die offizielle Schulwebseite[1] einsehbar und werden dort mit vollständigem Namen sowie Fachkombination aufgeführt. Die Liste muss jedoch separat gepflegt und bei Änderungen des Lehrkörpers regelmäßig aktualisiert werden. Insbesondere zu Schul- und Halbjahreswechseln kann es hierbei zu Verzögerungen kommen. Eine Zuordnung der Kürzel zu den Klarnamen erfordert zudem schulinterne Kenntnisse.

2.1.4 Notationskonventionen

Für die korrekte Interpretation der Vertretungsplaninformationen müssen Nutzer außerdem verschiedene Notationskonventionen kennen:

1. **Klammern**: Ein in Klammern gesetzter Wert (z. B. „(Ar)“ oder „(Mu)“) kennzeichnet die regulär vorgesehene Unterrichtseinheit oder Lehrkraft, nun jedoch ausfallende Information.
2. **Plus-Zeichen**: Ein mit einem Plus-Zeichen versehener Wert (wie etwa „+Mu“ oder „+D“) signalisiert den Ersatz für den regulären Wert.
3. **Vertretungsarten**: Die Spalte „Art“ verwendet verschiedene Bezeichnungen für unterschiedliche Vertretungssituationen:
   - „Fällt aus“/„Selbststudium“ (letzteres in der Oberstufe)
   - „Vertreten“ bei Stundenersatz
   - „Raumänderung“ bei gleichem Fach und Lehrer, aber anderem Raum
   - „Von … auf DD.MM. ddd Pos. verschoben“ bei Stundenplanumstellungen (Datumsangaben nach ISO 8601)
4. **Zusatzinformationen**: Die „Info“-Spalte dient für Informationen, die in keine der anderen Kategorien passen.
5. **Fortsetzungszeilen**: Eine leere Zelle deutet darauf hin, dass sich die Informationen dieser Zeile auf die Klasse der Zeile darüber beziehen. Anstatt die Klasse erneut zu nennen, wird die Zelle leer gelassen.

Zur Veranschaulichung ist folgendes Beispiel angeführt:

Klasse	**Pos**	**Lehrer**	**Fach**	**Raum**	**Art**	**Info**
10a	5.	+Hi (Ku)	+D (Mu)	+203 (105)	Vertreten	
	6.	Ku	KL	105	Zusatzunterricht	Informationsveranstaltung

TABELLE 2.2: Beispielhafter Vertretungseintrag

Diese Notation vermittelt folgende Information: In der Klasse 10a wird die 5. Stunde von Lehrkraft „Hi“ (Hinz) im Fach Deutsch im Raum 203 vertreten. Der regulär vorgesehene Musikunterricht durch Lehrkraft „Ku“ (Kunz) im Raum 105 entfällt. Im Anschluss findet in der Klasse 10a in der 6. Stunde Zusatzunterricht in Form einer Informationsveranstaltung statt. Dieser wird als Klassenleiterstunde („KL“) durch Lehrkraft „Ku“ (Kunz) im Raum 105 abgehalten.

[1] Görres-Gymnasium Koblenz, o. D.

2.1.5 Limitierungen

Trotz der funktionalen Grundstruktur weist das DSBmobile-System mehrere konzeptionelle und nutzungsbezogene Limitierungen auf:

1. **Navigation:** Der Weg zur relevanten Information ist unnötig komplex. Der Nutzer benötigt mindestens vier Klicks (Anwendung öffnen, „Aushänge“ auswählen, „DaVinci Touch“ auswählen, Datum wählen), um lediglich eine Tagesübersicht aller Klassen zu erreichen. Diese wird zusätzlich durch eine unklare Benutzerführung erschwert, bei der die Funktion von Elementen wie „Startseite“, „News“ oder der Bezeichnung „DaVinci Touch“ nicht selbsterklärend ist. Die Informationshierarchie ist ebenfalls suboptimal, da weniger wichtige Abschnitte wie „Fehlende Lehrer“ auffälliger platziert sind als der eigentliche Vertretungsplan und dadurch die Übersicht leidet.
2. **Datenvisualisierung:** Nach der umständlichen Navigation präsentiert sich die Information in einer unergonomischen Form. Die Darstellung als dichte, kleingedruckte Tabelle ohne klare Trennung zwischen Klassen oder visuell hervorgehobene Kennzeichnung verschiedener Vertretungsarten (z. B. Ausfall vs. Raumänderung) erschwert das schnelle Erfassen. Das gewählte Layout entspricht nicht aktuellen Gestaltungsstandards, wodurch der Nutzer zusätzlich belastet wird. Er muss nicht nur die relevanten Informationen identifizieren, sondern auch die schulinternen Notationskonventionen und Abkürzungen im Kopf entschlüsseln.
3. **Benachrichtigungssystem:** Die implementierte Benachrichtigungsfunktion ist in der Praxis kaum nutzbar. Sie bietet lediglich die Wahl zwischen keiner Benachrichtigung und einer generischen Meldung bei *jeder* beliebigen Änderung im System (z. B. „274583: Neue Inhalte verfügbar!“). Da diese Benachrichtigungen keinen Kontext liefern, welche Information sich geändert hat, ist der Nutzer gezwungen, nach jedem Hinweis den gesamten Plan manuell auf für ihn relevante Änderungen zu überprüfen. Dies untergräbt den Zweck einer proaktiven Benachrichtigung.
4. **Verfügbarkeit:** Ein wesentliches Problem ist die geringe Stabilität des Systems. Wiederkehrende Ausfälle machen den Dienst unzuverlässig und verhindern den Zugriff auf Informationen insbesondere in Situationen, in denen sie am dringendsten benötigt werden.

Diese Einschränkungen bilden den Ausgangspunkt für die in dieser Arbeit angestrebte Optimierung der Nutzererfahrung. Ein ergänzendes System greift auf die bestehenden Datenstrukturen zurück, stellt diese in aufbereiteter Form dar und präsentiert sie in einer nutzerorientierten Weise.

2.2 Webscraping: Techniken, Anwendungsbereiche und rechtliche Rahmenbedingungen

Die im vorherigen Abschnitt identifizierten Limitierungen des DSBmobile-Systems erfordern einen technischen Lösungsansatz, der die bestehenden Datenstrukturen nutzt, ohne das Kernsystem zu verändern. Da DSBmobile weder eine offizielle Schnittstelle noch Exportfunktionen bereitstellt, bleibt als einziger Zugangsweg zu den Vertretungsplaninformationen die automatisierte Extraktion der Daten aus der webbasierten Benutzeroberfläche. Webscraping ermöglicht es, die in *Hypertext Markup*

Language (HTML)-Tabellen strukturiert vorliegenden Informationen systematisch zu erfassen und für eine verbesserte Aufbereitung und Präsentation verfügbar zu machen.

Die folgenden Abschnitte erläutern die technischen Grundlagen, rechtlichen Rahmenbedingungen und praktischen Implementierungsaspekte dieser Methodik.

2.2.1 Definition und Grundkonzept

Webscraping bezeichnet den automatisierten Prozess der Extraktion von Daten aus Webseiten. Im Kern handelt es sich um eine Methodik, bei der strukturierte Informationen aus für Menschen konzipierten Webinhalten gewonnen und in maschinenlesbare Formate überführt werden. Diese Technik überbrückt die Kluft zwischen der visuellen Präsentation von Informationen in Webbrowsern und dem Bedarf an strukturierten Daten für automatisierte Systeme.

Die grundlegende Herausforderung ergibt sich aus der Tatsache, dass Webinhalte primär für die visuelle Erfassung durch Menschen konzipiert sind – mit Layouts, Formatierungen und einer Präsentation, die auf menschliche kognitive Fähigkeiten ausgerichtet ist. Webscraping ermöglicht es, diese visuell strukturierten Informationen systematisch zu erfassen und in Datenstrukturen zu transformieren, die für die automatisierte Verarbeitung durch IT-Systeme geeignet sind.

Abgrenzung zu anderen Datenextraktionsmethoden

Webscraping grenzt sich von *Application Programming Interface* (API, Plural: APIs), *Really Simple Syndication* (RSS) bzw. *Atom Syndication Format* (Atom)-Feeds und Datenexportfunktionen ab:

- **APIs**: Offizielle Schnittstellen, die strukturierte Datenformate und Zugriffsmethoden bieten. Sie sind oft limitiert, da nicht alle Dienste APIs bereitstellen oder nur bestimmte Datentypen erlauben.
- **RSS/Atom-Feeds**: Liefern aktualisierte Inhalte und sind zur automatisierten Verarbeitung freigegeben. Sie werden vor allem für Nachrichtenfeeds und Podcasts eingesetzt, unterstützen jedoch meist keine komplexen Strukturen wie Tabellen.
- **Datenexportfunktionen**: (beispielsweise *Comma-Separated Values* (CSV), *JavaScript Object Notation* (JSON) oder *Portable Document Format* (PDF)) beschränken sich auf vom Anbieter vorgegebene Formate.

Webscraping kommt überall dort zum Einsatz, wo diese Alternativen fehlen – zum Beispiel beim DSBmobile-Vertretungsplan ohne API oder Exportfunktion. Übrigens greift auch Banking-Software auf Webscraping zurück, wenn von der Bank keine standardisierte Schnittstelle angeboten wird.

2.2.2 Technische Grundlagen

Strukturierte vs. unstrukturierte Daten im Web

Webinhalte lassen sich entlang eines Spektrums von geringer bis hoher maschineller Verwertbarkeit einordnen:

- **Bildbasierte Inhalte** stellen die unterste Stufe dar: Text liegt lediglich als Grafik vor (z. B. Foto oder Scan eines Dokuments) und ist ohne zusätzliche *Optical Character Recognition* (OCR) nicht maschinenlesbar.
- **Unstrukturierte Daten** sind digital vorhanden, aber ohne klare Ordnung organisiert (z. B. Freitext, narrative Inhalte). Ihre automatische Verarbeitung ist entsprechend schwierig.
- **Teilstrukturierte Daten** besitzen wiederkehrende Muster, jedoch mit Abweichungen und Inkonsistenzen (z. B. Produktlisten mit ähnlichem Aufbau, aber variierendem Format).
- **Strukturierte Daten** folgen definierten Schemata (z. B. HTML-Tabellen mit festen Spalten und semantischem Markup) und eignen sich gut für Scraping.
- **Semantisch angereicherte Daten** stellen den Optimalfall dar. Neben der strukturierten Organisation enthalten sie Metadaten, die Werte zusätzlich beschreiben – etwa ihre Bedeutung, den Datentyp (Integer, Float, Text etc.), Einheiten (z. B. °C, km, €) oder Relationen. Dadurch sind sie unmittelbar maschinenlesbar und für automatisierte Verarbeitungsschritte optimal geeignet.

Der DSBmobile-Vertretungsplan ist ein vergleichsweise günstiger Fall: Er liegt als HTML-Tabelle mit konsistenter Spaltenstruktur vor. Obwohl keine HTML-Klassen oder IDs vergeben sind, bietet die tabellarische Organisation mit definierten Spalten (Klasse, Position, Lehrer, Fach, Raum, Art, Info) eine praktikable Grundlage für das Scraping. Allerdings treten auch Inkonsistenzen auf, beispielsweise wenn in Fortsetzungszeilen die Klassenbezeichnung weggelassen wird. Solche Sonderfälle erfordern zusätzliche Verarbeitungsschritte bei der Datenextraktion.

DOM-Konzept und HTML-Tabellenstrukturen

Das *Document Object Model* (DOM) ist ein wichtiges Konzept für das Webscraping. Es repräsentiert ein HTML-Dokument als hierarchische Baumstruktur, in der jedes Element als Knoten mit definierten Beziehungen zu anderen Elementen dargestellt wird:

- **Mehrstufige Hierarchie**
  - Oberste Ebenen: `<html>` → `<head>` und `<body>`
  - Darunter folgen Sektionen, Blöcke, Container usw.
  - Für Tabellen typisch: `<table>` → `<thead>`/`<tbody>` → `<tr>` → `<th>`/`<td>`
- **Navigation und Selektion**:
  - Elemente können durch ihre Position in der Hierarchie identifiziert werden.
  - Fehlen Klassen oder IDs, dient die reine Struktur (Reihenfolge und Verschachtelung) als Navigationsgrundlage.

Bei HTML-Tabellen ohne explizite Kennzeichnungen, wie im Fall des DSBmobile-Vertretungsplans, ist die implizite Struktur besonders wichtig:

- Die Tabelle selbst kann durch ihre Position im Dokument (z.B. als erste oder einzige Tabelle) identifiziert werden

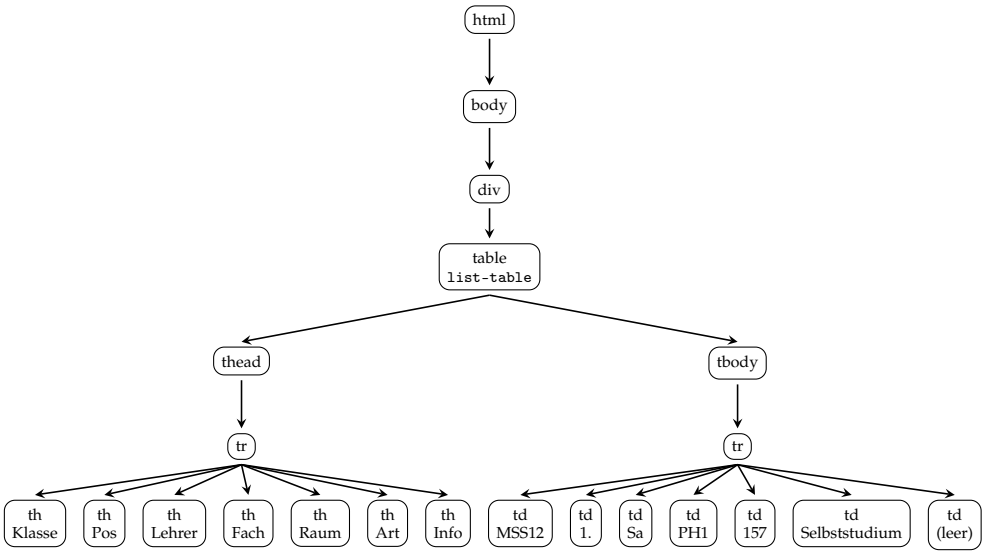


TABELLE 2.3: Baumdarstellung der HTML-Struktur des DSBmobile-Vertretungsplans

- Die Kopfzeile definiert die Semantik der Spalten
- Die Datenzeilen folgen einem konsistenten Muster, wobei besondere Strukturmerkmale wie leere Zellen in der ersten Spalte Informationen über die logische Gruppierung von Zeilen liefern

Selektionsmethoden für tabellenbasierte Daten

Für die Extraktion von Daten aus tabellarischen Strukturen gibt es verschiedene Ansätze, die je nach Anwendungsfall kombiniert werden können. Häufig wird zunächst die Zieltabelle anhand ihrer Position im Dokument ausgewählt, insbesondere wenn keine spezifischen HTML-Kennzeichnungen vorhanden sind. Bei mehreren Tabellen im Dokument kann die Identifikation auch über charakteristische Spaltenüberschriften erfolgen. Zusätzlich ist es oft notwendig, besondere Strukturmerkmale wie leere Zellen in der ersten Spalte zu berücksichtigen, da diese beispielsweise im Vertretungsplan auf eine Fortsetzung der Daten aus der vorherigen Zeile hinweisen.

Für den Vertretungsplan ist daher ein kombinierter Ansatz sinnvoll, der sowohl die tabellarische Grundstruktur als auch die spezifischen Konventionen des jeweiligen Dokuments einbezieht.

2.2.3 Rechtliche und ethische Aspekte

Rechtliche Rahmenbedingungen

Die rechtliche Beurteilung von Webscraping-Aktivitäten berührt mehrere Rechtsbereiche, die unterschiedliche Aspekte des Datenzugriffs und der Datenverwendung regulieren:

Urheberrecht. Das deutsche Urheberrechtsgesetz (UrhG) schützt kreative Werke sowie strukturierte Datensammlungen:

- **Schutz kreativer Elemente**: Das Design, Layout und die visuelle Gestaltung von Webseiten können urheberrechtlichen Schutz genießen (§ 2 UrhG).

- **Datenbankherstellerrecht**: Systematisch angelegte Datensammlungen können durch das Datenbankherstellerrecht (§§ 87a ff. UrhG) geschützt sein, wenn ihre Erstellung eine wesentliche Investition erforderte.
- **Faktendaten vs. kreative Inhalte**: Reine Faktendaten wie Raumnummern, Zeiten oder Fachbezeichnungen genießen als solche einen geringeren urheberrechtlichen Schutzumfang.

Für die Beurteilung des Vertretungsplansystems ist die explizite Regelung im Impressum von DSBmobile von zentraler Bedeutung:

> „Downloads und Kopien dieser Seite sind nur für den privaten, nicht kommerziellen Gebrauch gestattet."[2]

Im Umkehrschluss ergibt sich aus dieser Klausel, dass lediglich die private, nichtkommerzielle Nutzung der Inhalte – wozu auch der Vertretungsplan zählt – zulässig ist. Entscheidend ist hier die Einhaltung der definierten Grenzen: Die **private Nutzung** beschränkt sich auf den persönlichen Gebrauch ohne Verbreitung an Dritte, während der **nicht-kommerzielle Charakter** den Ausschluss jeglicher direkter oder indirekter wirtschaftlicher Verwertung erfordert.

Für die Umsetzung dieser Arbeit folgt daraus insbesondere, dass die schulinternen Zugangsdaten vertraulich zu behandeln sind. Der veröffentlichte Quellcode ist so gestaltet, dass er ohne diese Zugangsdaten keinen Zugriff auf die Inhalte von DSBmobile ermöglicht.

Datenschutzrecht. Die Verarbeitung personenbezogener Daten unterliegt den Bestimmungen der Datenschutz-Grundverordnung (DSGVO) sowie des Bundesdatenschutzgesetzes (BDSG):

- **Lehrerkürzel im Vertretungsplan**: Der Vertretungsplan enthält ausschließlich zweibuchstabige Lehrerkürzel (z. B. „Hi", „Ku", „Mu"), die als aus Initialen gebildete Abkürzungen fungieren. Diese stellen für sich genommen keine Klarnamen dar und ermöglichen keine unmittelbare Identifikation der betroffenen Personen. Erst durch die Verknüpfung mit der öffentlich zugänglichen Kollegiumsübersicht[3] auf der Schulwebseite lassen sich die Kürzel den vollständigen Namen und Fachkombinationen der Lehrkräfte zuordnen. Bei diesen Angaben handelt es sich um personenbezogene Daten im Sinne des Art. 4 Nr. 1 DSGVO, jedoch um solche mit geringem Schutzbedarf. Sie sind funktionsbezogene Informationen (Funktionsträgerdaten), die im Zusammenhang mit der dienstlichen Aufgabenwahrnehmung der Lehrkräfte stehen und bereits durch die offizielle Veröffentlichung allgemein zugänglich gemacht werden. Der Schutzumfang ist daher deutlich geringer als bei privaten oder sensiblen personenbezogenen Daten.
- **Rechtmäßigkeit der Verarbeitung**: Für jede Verarbeitung personenbezogener Daten ist eine Rechtsgrundlage erforderlich (Art. 6 DSGVO).
- **Private Verarbeitung**: Nach Art. 2 Abs. 2 lit. c DSGVO gilt die Verordnung nicht für Verarbeitungen, die „durch natürliche Personen zur Ausübung ausschließlich persönlicher oder familiärer Tätigkeiten" erfolgen.

[2] heinekingmedia GmbH, o. D.
[3] Görres-Gymnasium Koblenz, o. D.

Für die vorliegende Arbeit folgt daraus konkret der Verzicht auf die Benennung von Klarnamen. In Abbildungen werden Lehrerkürzel beibehalten, während Klarnamen in Screenshots geschwärzt oder verpixelt dargestellt werden, um dem geringen, aber vorhandenen Schutzbedarf personenbezogener Daten Rechnung zu tragen.

Für ein privates Vertretungsplansystem ist besonders relevant:

1. **Haushaltsausnahme nach Art. 2 Abs. 2 lit. c DSGVO**: Diese Ausnahme kann greifen, wenn das System ausschließlich für den persönlichen oder familiären Gebrauch bestimmt ist und keine kommerzielle oder öffentliche Nutzung erfolgt. In diesem Fall fallen die Datenverarbeitungsvorgänge nicht unter den Anwendungsbereich der DSGVO.
2. Bei einer Nutzung über den rein privaten Bereich hinaus (z. B. durch Bereitstellung für andere Nutzer oder schulische Verwendung) wäre eine eigenständige datenschutzrechtliche Legitimation nach Art. 6 DSGVO erforderlich.

Da die vorliegende Arbeit nicht als persönliche oder familiäre Tätigkeit im Sinne der Haushaltsausnahme einzustufen ist, gilt die DSGVO uneingeschränkt.

Ethische Überlegungen

Eine ressourcenschonende und technisch rücksichtsvolle Implementierung von Webscraping-Mechanismen achtet darauf, die Zielinfrastruktur nicht zu überlasten. Zu diesem Zweck werden angemessene Wartezeiten zwischen den Anfragen eingeführt und die Anzahl der Datenaufrufe auf das notwendige Minimum beschränkt, indem lediglich die tatsächlich benötigten Seiten gezielt extrahiert werden.

Transparenz und Offenheit sind grundlegende Prinzipien einer ethisch fundierten Vorgehensweise.

Die Grundsätze der Datenminimierung und Zweckbindung orientieren sich an datenschutzrechtlichen Vorgaben. Nur die für den definierten Zweck wirklich erforderlichen Informationen dürfen extrahiert werden, und Daten, die nach Erreichen dieses Zwecks nicht mehr benötigt werden, sind unverzüglich zu löschen.

Rechtliche Bewertung und Umsetzungskonsequenzen

Die Betrachtung rechtlicher Aspekte zeigt, dass ein privates Webscraping-System für Vertretungspläne unter den gegebenen Voraussetzungen zulässig ist. Die explizite Erlaubnis für private, nicht-kommerzielle Nutzung im DSBmobile-Impressum sowie die datenschutzrechtliche Haushaltsausnahme bilden eine tragfähige rechtliche Grundlage. Für die praktische Umsetzung bedeutet dies eine konsequente Beschränkung auf den persönlichen Gebrauch ohne Bereitstellung für Dritte sowie den vollständigen Verzicht auf wirtschaftliche Verwertung.

Obwohl das entwickelte System aufgrund der rechtlichen Rahmenbedingungen ausschließlich privat nutzbar ist, wird es im Rahmen dieser Besonderen Lernleistung dennoch nach den Standards einer potenziell veröffentlichbaren Anwendung konzipiert und implementiert. Diese Herangehensweise ermöglicht es, die vollständige Bandbreite moderner Softwareentwicklungsprinzipien zu demonstrieren und eine technisch vollwertige Lösung zu entwickeln, die als Referenz für ähnliche Projekte dienen könnte.

3 Anforderungsanalyse

3.1 Nutzergruppen und deren Anforderungen an das System

Die Basis für ein erfolgreiches System ist das Verständnis der verschiedenen Interessengruppen, die mit ihm interagieren oder von ihm betroffen sind. Für das hier entwickelte Vertretungsplansystem lassen sich mehrere Nutzergruppen unterscheiden, die jeweils eigene Schwerpunkte und Erwartungen mitbringen:

- **Schülerinnen und Schüler** (ca. 850)[1] bilden die größte Nutzergruppe. Für sie steht die schnelle, unkomplizierte und verlässliche Information über aktuelle Planänderungen im Vordergrund. Eine intuitive Bedienung und die jederzeitige Verfügbarkeit auch auf mobilen Endgeräten sind für diese Gruppe besonders entscheidend.
- **Lehrkräfte** (ca. 80)[2] stellen ebenfalls eine zentrale Nutzergruppe dar. Entscheidend ist für sie eine schnelle und zuverlässige Bereitstellung aktueller Änderungen in klarer Form, sowohl mobil als auch auf dem Desktop. So können sie sich ohne großen Aufwand einen Überblick über den eigenen Unterrichtstag verschaffen.
- **Eltern** erwarten Transparenz über den Schulalltag ihrer Kinder. Sie legen Wert auf ein einfach zu bedienendes System, das ihnen ohne technische Hürden Einblick in Änderungen des Unterrichtsablaufs gibt.
- Die **Schulleitung** betrachtet das System aus einer übergeordneten Perspektive. Sie legt besonderen Wert auf Datenschutzkonformität, eine stabile Funktionsweise und die ressourcenschonende Integration in die bestehende IT-Infrastruktur.

Darüber hinaus sind weitere Akteure zu berücksichtigen:

- **Entwickler**, die das System betreuen oder erweitern, erwarten eine saubere Codebasis, die leicht wartbar und flexibel anpassbar ist. Dokumentation und klare Schnittstellen spielen für sie eine zentrale Rolle.
- Der **DSBmobile-Anbieter** schließlich ist indirekt betroffen, da sein System als ursprüngliche Datenquelle dient. Von seiner Seite ist sicherzustellen, dass die Integrität der Systeme nicht gefährdet und die Lizenz- sowie Nutzungsbedingungen eingehalten werden.

Aus dieser Analyse ergibt sich ein breites Spektrum von Anforderungen, die sich inhaltlich, technisch und organisatorisch unterscheiden. Übergreifend zeigt sich dabei der Wunsch nach einer hohen **Datenqualität und -relevanz**. Informationen sollen nicht nur korrekt und zuverlässig, sondern auch in einer für die Zielgruppen verständlichen Form präsentiert werden. Insbesondere das Auflösen interner Abkürzungen trägt erheblich zur Verständlichkeit bei. Ebenso zentral sind eine hohe

[1] Rhein-Zeitung, o. D.
[2] Görres-Gymnasium Koblenz, o. D.

Benutzerfreundlichkeit und Zugänglichkeit, die sich in einem schnellen und intuitiven Zugriff äußern, sowie die optimale Nutzbarkeit auf verschiedenen Endgeräten.

Neben diesen nutzerzentrierten Erwartungen müssen grundlegende **technische und administrative Anforderungen** erfüllt werden. Dazu zählen die Gewährleistung von Systemstabilität und Performanz, die Einhaltung von Datenschutz und Datensicherheit, eine nachhaltige Wartbarkeit und Erweiterbarkeit sowie die Robustheit gegenüber Änderungen an der Datenquelle. Hinzu kommen übergeordnete Ziele wie eine ressourcenschonende Nutzung der bestehenden IT-Infrastruktur und die Einhaltung der Lizenz- und Nutzungsbedingungen des Datenanbieters.

Die Vielfalt dieser Anforderungen – von der benutzerfreundlichen Darstellung im Frontend über die zuverlässige Datenbeschaffung bis hin zu strengen Sicherheits- und Compliance-Vorgaben – verdeutlicht, dass ein monolithischer, in sich geschlossener Ansatz nicht zielführend wäre. Eine solche Architektur wäre starr, schwer wartbar und wenig anpassungsfähig. Stattdessen legt die Analyse nahe, das System in klar abgegrenzte Funktionsbereiche zu unterteilen: die **Beschaffung der Rohdaten**, deren **Aufbereitung und Veredelung** sowie eine separate **Präsentationsschicht**, die ausschließlich auf die benutzerfreundliche Darstellung fokussiert. Durch diesen modularen Aufbau lassen sich die unterschiedlichen Anforderungen der Nutzer gezielt adressieren, und es entsteht eine widerstandsfähige sowie langfristig tragfähige Systemarchitektur.

4 Systemkonzeption

4.1 Gesamtarchitektur

Die Architektur des Gesamtsystems orientiert sich an etablierten Prinzipien moderner Softwareentwicklung. Hervorzuheben ist beispielsweise die klare Trennung der Verantwortlichkeiten (Separation of Concerns), die es ermöglicht, die Aufgaben der Datenextraktion, -verarbeitung und -aufbereitung vollständig vom Frontend zu entkoppeln.[1]

Die Gesamtarchitektur des Systems gliedert sich in fünf aufeinander aufbauende Schichten:

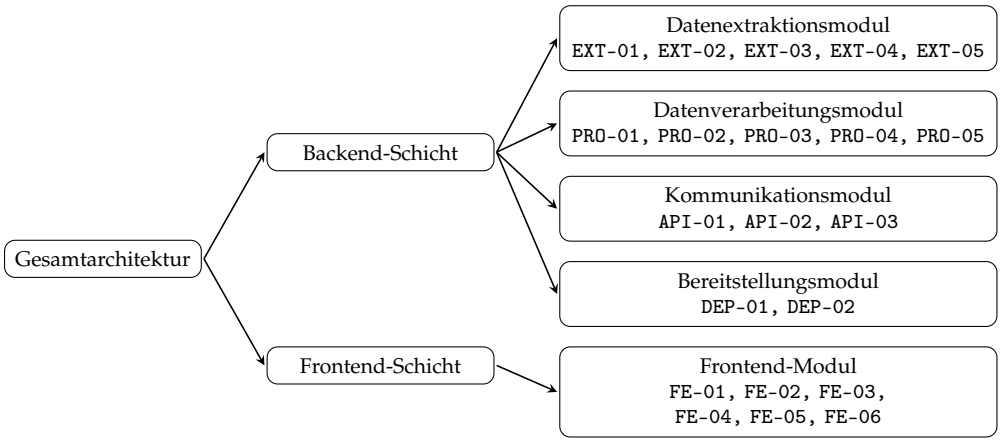


TABELLE 4.1: Baumdarstellung des Schichtenmodells

1. Datenextraktionsmodul

Dieses Modul kommuniziert mit der DSBmobile-API und führt Webscraping durch. DSBmobile stellt eine interne API bereit, die jedoch nur den Zugang zu „Aushänge“ und eine Dokumentenübersicht ermöglicht. Die eigentlichen Vertretungsplaninformationen liegen als HTML-Tabellen vor und müssen per Webscraping extrahiert werden. Die API dient daher primär der Authentifizierung, während der Hauptteil der Datenextraktion über klassisches Webscraping erfolgt. Die spezifischen Anforderungen an dieses Modul sind in der nachfolgenden Tabelle zusammengefasst:

[1] Dijkstra, 1982.

ID	Anforderung
EXT-01	Das System muss sich mit den hinterlegten Zugangsdaten bei der DSBmobile-API authentifizieren können.
EXT-02	Das System muss die URLs zu den tagesaktuellen Vertretungsplänen extrahieren.
EXT-03	Das System muss alle Zeilen und Spalten aus der HTML-Tabelle des Vertretungsplans auslesen.
EXT-04	Das System muss die Logik der Fortsetzungszeilen (leere erste Spalte) korrekt interpretieren und die Zeilen der richtigen Klasse zuordnen.
EXT-05	Das System muss die Daten der Lehrkräfte (Kürzel, Name, Fächer) von der Schulwebseite extrahieren können.

TABELLE 4.2: Anforderungen an das Datenextraktionsmodul

2. Datenverarbeitungsmodul

Dieses Modul transformiert die Rohdaten in eine kursbasierte Struktur. Dabei werden Abkürzungen für Fächer und Lehrernamen durch vollständige Bezeichnungen ersetzt. Abschließend erfolgt eine Schema-basierte Validierung der Datenintegrität. Die detaillierten Anforderungen für die Datenverarbeitung sind in folgender Tabelle aufgeführt:

ID	Anforderung
PRO-01	Das System muss die tagesbasierten Rohdaten in eine klassen- und kursbasierte Struktur umwandeln.
PRO-02	Das System muss jeden Vertretungseintrag in ein strukturiertes JSON-Format überführen.
PRO-03	Das System muss die in den Daten enthaltenen Lehrerkürzel durch die vollen Nachnamen ersetzen.
PRO-04	Das System muss gängige Fachabkürzungen durch die vollen Fachbezeichnungen ersetzen (außer bei MSS-Kursen).
PRO-05	Das System muss die verarbeiteten Daten gegen ein vordefiniertes JSON-Schema validieren, um die Datenintegrität sicherzustellen.

TABELLE 4.3: Anforderungen an das Datenverarbeitungsmodul

3. Kommunikationsmodul

Dieses Modul verbindet Backend und Frontend über eine Programmierschnittstelle. Es implementiert Authentifizierung, Ratenlimitierung und Caching für optimale Performance und Sicherheit. Die funktionalen Anforderungen an die API-Schnittstelle werden in der folgenden Tabelle spezifiziert:

ID	Anforderung
API-01	Die API muss einen Endpunkt zur Benutzerauthentifizierung bereitstellen, der bei Erfolg ein JWT zurückgibt.
API-02	Die API muss einen geschützten Endpunkt bereitstellen, der die vollständigen, verarbeiteten Vertretungsdaten im JSON-Format ausliefert.
API-03	Die API muss einen öffentlichen Endpunkt zur Systemüberwachung (Healthcheck) anbieten.

TABELLE 4.4: Anforderungen an das Kommunikationsmodul

4. Bereitstellungsmodul

Dieses Modul orchestriert den automatisierten Systembetrieb. Eine zeitgesteuerte Komponente (Scheduler) koordiniert die regelmäßige Datenaktualisierung, während die containerisierte Infrastruktur einen stabilen Rund-um-die-Uhr-Betrieb gewährleistet. Die Anforderungen für den stabilen Betrieb des Systems sind in nachstehender Tabelle definiert:

ID	Anforderung
DEP-01	Der Datenextraktions- und Verarbeitungsprozess muss automatisiert und in regelmäßigen Intervallen ausgeführt werden, während der API-Prozess dauerhaft parallel läuft.
DEP-02	Das System muss als containerisierte Anwendung (Docker) bereitgestellt werden können.

TABELLE 4.5: Anforderungen an das Bereitstellungsmodul

5. Frontend-Modul

Dieses Modul realisiert die nutzerzentrierte Darstellung und Interaktion. Der Fokus liegt auf einer intuitiven Benutzeroberfläche mit optimierter Nutzererfahrung für alle Endgeräte. Die Anforderungen an die Benutzerschnittstelle sind in der nachfolgenden Tabelle zusammengestellt:

ID	Anforderung
FE-01	Das Frontend muss ein benutzerfreundliches Anmeldeformular mit Eingabevalidierung bereitstellen.
FE-02	Das Frontend muss eine intuitive Klassenauswahl mit persistenter Speicherung der Nutzereinstellung ermöglichen.
FE-03	Vertretungsdaten müssen in einer übersichtlichen, responsiven Tabellenform dargestellt werden.
FE-04	Unterrichtsausfälle und Selbststudium müssen visuell hervorgehoben werden.
FE-05	Das Frontend muss Funktionen zum Durchsuchen und Filtern der Vertretungsdaten bieten.
FE-06	Die Anwendung muss auf Desktop-, Tablet- und Mobilgeräten übersichtlich und benutzerfreundlich dargestellt werden.

TABELLE 4.6: Anforderungen an das Frontend-Modul

4.2 Backend-Schicht

Das Backend ist in Python implementiert. Diese Wahl begründet sich durch die ausgezeichnete Eignung für Datenverarbeitungsaufgaben und die umfangreiche Sammlung an Bibliotheken für Webscraping-Anwendungen. Zusätzlich ist die Lesbarkeit von Python im Vergleich zu anderen Low-Level-Programmiersprachen sehr hoch, was die Arbeit damit effizienter macht.

4.2.1 Datenextraktionsschicht

Die Datenextraktion ist die größte Herausforderung des Systems und ist deshalb besonders sorgfältig konzipiert. Um zuverlässig auf die Vertretungsplandaten zuzugreifen, kombiniert das System zwei sich ergänzende Zugriffswege: den Zugriff über die offizielle API sowie das klassische HTML-Scraping.

Die erste Methode erfolgt über die API-basierte Datenextraktion mithilfe des `PyDSB`-Moduls.[2] Dieses Modul wurde ursprünglich von Moritz Jannasch entwickelt und zwischen August 2019 und Januar 2023 kontinuierlich verbessert. Die letzte offizielle Version 2.3.0 (veröffentlicht am 12. Januar 2023) wird als Basis für die hier vorgestellte Implementierung genutzt.

Das `PyDSB`-Modul ermöglicht eine direkte Kommunikation mit der DSBmobile-API und stellt nach erfolgreicher Authentifizierung einen autorisierten Zugang zu allen verfügbaren Ressourcen wie Vertretungsplänen, Aushängen und News her. Für die Zwecke dieser Arbeit wurde das ursprüngliche Modul umfassend überarbeitet und an die spezifischen Anforderungen des Projekts angepasst. Zu diesen wesentlichen Verbesserungen zählen:

- Ein erweitertes Logging-System
- Ein robusteres Fehlerbehandlungssystem, das verschiedene Arten von Netzwerk- und Authentifizierungsfehlern abfängt und angemessen behandelt
- Optimierte Methoden zum gezielten Abrufen und Verarbeiten der Vertretungsplandaten

Es ist jedoch anzumerken, dass das `PyDSB`-Modul mit ursprünglich etwa 66 Codezeilen lediglich einen sehr kleinen Teil dieser Gesamtarbeit darstellt. Allein der Backend-Teil umfasst über 1200 eigens geschriebene Codezeilen. Der Hauptteil der Entwicklung liegt in der weiteren Verarbeitung, Strukturierung, Bereitstellung und Sicherung der Daten sowie in der Frontend-Implementierung zur Darstellung der Vertretungsinformationen.

Das Modul liefert eine Übersicht der verfügbaren Pläne und Dokumente, jedoch noch nicht die eigentlichen strukturierten Vertretungsdaten. Für die eigentliche Extraktion der Vertretungsplandaten kommt ein spezialisierter HTML-Parser zum Einsatz. Die Bibliothek `BeautifulSoup` wurde gewählt, da sie eine robuste und fehlertolerante Verarbeitung von HTML-Dokumenten ermöglicht. Der Parser analysiert die Tabellenstruktur und extrahiert systematisch die relevanten Informationen.

Die in Kapitel 2.1 beschriebenen tabellarischen Strukturen mit ihren spezifischen Notationskonventionen sind besonders wichtig zu beachten. Die Extraktion berücksichtigt dabei auch Besonderheiten wie Fortsetzungszeilen, die durch leere Zellen in

[2] Jannasch, 2023.

der ersten Spalte gekennzeichnet sind und für das korrekte Verständnis der Daten essentiell sind.

4.2.2 Datenverarbeitungsmodul

Nach der erfolgreichen Extraktion durchlaufen die Rohdaten einen mehrstufigen Verarbeitungsprozess, der sie für die Endnutzung aufbereitet. Die zentrale Innovation besteht hierbei in der grundlegenden Umstrukturierung der Daten: Während der ursprüngliche Vertretungsplan primär nach Tagen organisiert ist, werden die Daten im neuen System nach Klassen und Kursen strukturiert. Die Neuorganisation ermöglicht eine wesentlich effizientere Filterung nach den für den jeweiligen Nutzer relevanten Informationen.

Die Umwandlung erfolgt hier aus der tabellarischen Darstellung in ein hierarchisches JSON-Format. Diese Strukturierung nach dem Schema „Klasse → Vertretungen“ anstelle von „Tag → Vertretungen aller Klassen“ entspricht auch wesentlich besser der Denkweise der Nutzer, die typischerweise nur an den Vertretungen ihrer eigenen Klasse interessiert sind.

Ein weiteres Element der Datenverarbeitung ist die Anreicherung der Informationen. Hierbei werden die Lehrerkürzel sowie die Fachnamen durch vollständige Begriffe ersetzt. Die Lehrerkürzel werden über den zum Basisskript separaten Teacher-Scraper beschafft, der die Lehrerdaten von der offiziellen Schulwebseite[3] extrahiert und dem Basisskript zur Verfügung stellt. Die Fachbezeichnungen stammen dagegen aus einer eigenen Lookup-Tabelle. Dieses kann nun gebündelt zum einen die Klassennamen („M“ → „Mathe“), aber auch Lehrerkürzel („(Mu) +Mü“ → „(Mustermann) +Müller“) ersetzen. Wie bereits in der Anforderungsanalyse beschrieben, hilft dies enorm neuen Schülern sowie Elternteilen beim Verständnis des Vertretungsplans.

Die Datenspeicherung erfolgt in einem dreistufigen System, das die verschiedenen Verarbeitungsstadien abbildet:

1. Die ursprünglich extrahierten HTML-Daten werden unverändert für Audit-Zwecke und zur Fehleranalyse bis zum nächsten Programmlauf im Log aufbewahrt.
2. Die aus den Rohdaten transformierten, jedoch noch nicht vollständig angereicherten Daten werden zwischengespeichert.
3. Die vollständig verarbeiteten und angereicherten Daten bilden die primäre Quelle für die API.

Dies gewährleistet die Nachvollziehbarkeit des Programms und erleichtert die Fehlersuche bei Bugs oder unerwartetem Verhalten.

Die Konsistenz der Datenstrukturen wird durch den Einsatz von JSON-Schema[4] sichergestellt. Ein JSON-Schema ist eine standardisierte Beschreibungssprache, mit der die Struktur und der Inhalt von JSON-Daten definiert werden. Es legt fest, welche Felder erlaubt sind, welchen Datentyp sie haben sollen (z. B. String, Zahl, Boolean) und ob sie erforderlich sind. Das JSON-Schema des DSBmobile-Backends[5]

[3] Görres-Gymnasium Koblenz, o. D.

[4] `https://json-schema.org/overview/what-is-jsonschema`

[5] `https://github.com/PrtmPhlp/DSB-Backend/blob/master/schema/schema.json`

überprüft die Struktur des erstellten Datensatzes und signalisiert eine Fehlermeldung, sobald die Anforderungen nicht erfüllt sind, um sowohl den Nutzer als auch das Frontend vor korrupten oder fehlerhaften Daten zu schützen.[6]

4.2.3 Kommunikationsmodul

Eine *Representational State Transfer* (REST)-API ist ein Architekturstil für Webschnittstellen, der auf den Prinzipien von Ressourcenorientierung, einheitlicher Schnittstellenstruktur und zustandsloser Kommunikation basiert. Ressourcen (z. B. Vertretungsplandaten) werden über eindeutige URLs adressiert und mittels standardisierter HTTP-Methoden wie `GET`, `POST`, `PUT` oder `DELETE` abgerufen oder verändert. Der Einsatz von REST erleichtert die Integration verschiedener Systeme, da die Schnittstelle leicht verständlich, plattformunabhängig und gut skalierbar ist.

Die API bildet die entscheidende Schnittstelle zwischen Backend und Frontend und wurde nach den Prinzipien eines ressourcenorientierten REST-Designs konzipiert. Sie zeichnet sich durch Einfachheit, Konsistenz und eine klare semantische Struktur aus.

Der Hauptendpunkt `/api/` liefert die vollständigen Vertretungsplandaten und bildet damit die zentrale Datenquelle für das Frontend. Ergänzend bietet der Endpunkt `/login` die notwendige Authentifizierungsfunktionalität, während `/healthcheck` zur Systemüberwachung dient. Die bewusste Beschränkung auf wenige, klar definierte Endpunkte erhöht die Verständlichkeit der API und reduziert potenzielle Fehlerquellen.

Die Authentifizierung erfolgt über `JSON Web Tokens` (JWT), die nach erfolgreicher Anmeldung generiert und bei jedem Zugriff auf geschützte Ressourcen überprüft werden. Dieses Verfahren ermöglicht eine Implementierung der API ohne serverseitige Sitzungsverwaltung („stateless“) und unterstützt dadurch die Skalierbarkeit des Systems.

Zur Sicherstellung einer stabilen Systemleistung begrenzt die API die Anzahl der Anfragen pro Client auf 10 pro Sekunde. Dies schützt vor Überlastung und verhindert eine unverhältnismäßige Beanspruchung durch einzelne Nutzer. Ergänzend werden häufig angefragte Daten im Arbeitsspeicher des Servers zwischengespeichert (sogenanntes „In-Memory-Caching“), sodass sie bei wiederholten Zugriffen nicht jedes Mal neu aus einem persistenten Speicher geladen werden müssen.

Die API legt besonderen Wert auf eine konsistente und aussagekräftige Fehlerbehandlung. Jeder Fehlerfall wird durch einen angemessenen HTTP-Statuscode sowie eine strukturierte Fehlermeldung kommuniziert, was die Fehlerdiagnose und -behebung erheblich erleichtert.

4.2.4 Bereitstellungsmodul

Das Bereitstellungsmodul orchestriert den automatisierten Betrieb und gewährleistet kontinuierliche Verfügbarkeit. Es umfasst zwei zentrale Komponenten: Zeitsteuerung der Datenprozesse (Scheduler) und Containerisierung.

[6] Ein Beispiel für eine Schema-validierte Datei findet sich unter: https://github.com/PrtmPhlp/DSB-Backend/blob/master/schema/schema-sample.json

Scheduler Ein separater Prozess koordiniert parallele Abläufe und sorgt für die regelmäßige Datenaktualisierung, während die API durchgehend verfügbar bleibt. Das System folgt dabei einem dualen Ansatz: Die API läuft dauerhaft, während Datenextraktion und -verarbeitung in konfigurierbaren Intervallen erfolgen. Diese Trennung optimiert die Ressourcennutzung und erhöht die Stabilität. Kommt es zu Fehlern bei der Datenextraktion, bleibt die API funktionsfähig und greift auf die zuletzt verarbeiteten Daten zurück. Eine automatische Wiederholung sorgt dafür, dass temporäre Störungen aufgefangen werden.

Containerisierung und Infrastruktur Docker bildet das Fundament für eine portable und isolierte Bereitstellung. Durch die Kapselung aller Abhängigkeiten in standardisierte Container entsteht eine konsistente Ausführungsumgebung unabhängig von der Infrastruktur. Das bekannte „It works on my machine“[7], bei dem Software nur in der Entwicklungsumgebung funktioniert, wird dadurch vermieden.

Das Multi-Container-Setup trennt Backend und Frontend für unabhängige Skalierung und Wartung. Docker Compose orchestriert die Infrastruktur und definiert Dienst-Beziehungen. Ergänzend wird das *Prinzip der geringstmöglichen Privilegien*[8] umgesetzt: Jeder Container erhält nur die Rechte, die er tatsächlich benötigt. Dadurch bleiben mögliche Sicherheitslücken lokal begrenzt.

Ein weiterer zentraler Bestandteil ist die automatisierte Bereitstellung neuer Software-Versionen. Über ein *Continuous Integration / Continuous Deployment* (CI/CD)-System werden Änderungen am Quellcode automatisch getestet, gebaut und als Docker-Container in die GitHub Container Registry hochgeladen. Von dort erfolgt die Installation direkt auf dem Server, sodass neue Versionen Sicherheitsupdates oder Verbesserungen ohne manuelles Eingreifen sofort und zuverlässig live gehen.

Diese Kombination aus Zeitsteuerung, Containerisierung und CI/CD gewährleistet einen stabilen, sicheren und rund um die Uhr verfügbaren Betrieb.

4.3 Frontend-Modul

Das Frontend wurde mit besonderem Augenmerk auf Benutzerfreundlichkeit und Leistung entwickelt. Für die Umsetzung kam `TypeScript`[9] in Kombination mit dem modernen Webframework `Next.js`[10] zum Einsatz. Diese Technologien ermöglichen eine zuverlässige und reaktionsschnelle Benutzeroberfläche.

`TypeScript` erweitert JavaScript um statische Typisierung und hilft dabei, Fehler frühzeitig zu erkennen und die Codequalität insgesamt zu verbessern. `Next.js` baut auf React auf und bringt viele hilfreiche Funktionen wie serverseitiges Rendering, intelligentes Routing und Performance-Optimierungen direkt mit. Im Vergleich zu herkömmlichen Methoden mit einfachem HTML, CSS und JavaScript bietet `Next.js` eine strukturierte Grundlage, die die Entwicklung moderner, dynamischer Webanwendungen deutlich erleichtert und professioneller gestaltet.

[7] https://www.reddit.com/r/ProgrammerHumor/comments/cw58z7/it_works_on_my_machine/

[8] https://www.cloudflare.com/de-de/learning/access-management/principle-of-least-privilege/

[9] https://www.typescriptlang.org/

[10] https://nextjs.org/

4.3.1 Benutzeroberfläche und Benutzererfahrung

Die Benutzeroberfläche folgt dem Prinzip „Content First“ und stellt die relevanten Vertretungsinformationen mit minimaler Interaktion in den Mittelpunkt. Der Nutzer muss sich lediglich einmalig anmelden und seine Klasse auswählen. Diese Daten werden im lokalen Speicher des Browsers („Local Storage“) abgelegt, sodass beim nächsten Laden der Webseite sofort die gewählte Klasse sowie alle Vertretungen für den aktuellen Tag angezeigt werden.

Wichtige Inhalte – wie zum Beispiel Unterrichtsausfälle – werden hervorgehoben, um sofort ins Auge zu fallen. Fällt eine Stunde aus oder findet im Selbststudium statt, wird dies durch einen auffälligen orangenen Markierungseffekt sichtbar gemacht. Weniger wichtige Informationen, etwa welche Lehrkraft bei einem Ausfall betroffen ist, erscheinen dagegen in einem dezenten Grauton. Dadurch bleibt der visuelle Fokus auf den entscheidenden Inhalten.

Die Interaktion wurde bewusst minimalistisch gehalten: Das System reagiert unmittelbar auf Eingaben und reduziert die erforderlichen Klicks auf ein Minimum. Ergänzend sorgt ein responsives Design dafür, dass sich die Darstellung automatisch an unterschiedliche Endgeräte und Bildschirmgrößen anpasst.

4.4 Sicherheitskonzept und Datenschutz

Die erste Sicherheitsebene bildet die Authentifizierung. Sie stellt sicher, dass nur berechtigte Nutzer Zugriff auf die Vertretungsinformationen erhalten. Hierfür kommt eine passwortbasierte Anmeldung zum Einsatz. Nach erfolgreichem Login wird ein sogenannter JWT-Token erzeugt, der den Nutzer gegenüber dem System eindeutig identifiziert.

Dieser Token wird in einem *HTTP-only Cookie* gespeichert – also in einem Cookie, auf das JavaScript im Browser nicht zugreifen kann. Dadurch ist er besonders gut vor sogenannten *XSS-Angriffen*[11] (Cross-Site Scripting) geschützt. Bei solchen Angriffen versuchen Angreifer, schädlichen Code in Webseiten einzuschleusen, um sensible Daten wie Zugangstoken auszulesen. Da der Token in diesem Fall nicht über JavaScript erreichbar ist, bleibt er vor solchen Angriffen sicher.

Zusätzlich wird bei jeder Anfrage an die API überprüft, ob der Token noch gültig und korrekt ist. Die Gültigkeit ist zeitlich begrenzt, um das Risiko eines Missbrauchs gestohlener Tokens weiter zu minimieren.

Auch andere gängige Angriffsszenarien wurden berücksichtigt. So ist das System dadurch abgesichert, dass alle dynamischen Inhalte beim Einfügen in die Webseite automatisch so aufbereitet werden, dass keine schädlichen Skripte ausgeführt werden können – selbst dann, wenn die Daten von Nutzern stammen oder aus externen Quellen geladen werden.

Zum Schutz vor *Brute-Force-Angriffen*, bei denen automatisiert zahlreiche Passworte ausprobiert werden, wurde eine Ratenbegrenzung implementiert. Wird eine Anmeldung mehrfach hintereinander falsch eingegeben, wird der Zugriff für eine gewisse Zeit blockiert. So wird automatisiertes Erraten von Passwörtern effektiv unterbunden.

[11] https://owasp.org/www-community/attacks/xss/

5 Implementierung

In den vorangegangenen Kapiteln wurden Problemstellung (Kapitel 1), theoretische Grundlagen (Kapitel 2), die Anforderungen (Kapitel 3) sowie die Systemkonzeption (Kapitel 4) hergeleitet. Dieses Kapitel beschreibt die praktische Umsetzung. Eine vollständige und zeilenweise Dokumentation aller Komponenten würde den vorgesehenen Rahmen der Arbeit sprengen und hätte einen geringen Mehrwert gegenüber dem bereitgestellten, versionierten und kommentierten Quellcode im Repository.[1] Stattdessen wird ein repräsentativer, vertiefter Blick auf zwei exemplarische Teilbereiche gegeben:

1. **Beispiel Backend:** Verarbeitungspipeline vom Roh-HTML zur strukturierten kursbasierten JSON-Ausgabe, am Beispiel des Parsing- und Normalisierungsschritts.
2. **Beispiel Frontend:** Nutzerfluss (Login → Kurswahl → Darstellung eines Vertretungstags) mit Fokus auf Responsive Design, Performance und Usability.

Die Auswahl dieser Beispiele folgt dem Ziel, sowohl die datengetriebene als auch die nutzerzentrierte Seite der Implementierung abzubilden.

5.1 Technologischer Stack und Entwicklungsumgebung

Der Stack orientiert sich an Pragmatismus und Wartbarkeit:

- **Backend:** Python 3.12, Bibliotheken: `requests` (HTTP), `BeautifulSoup` (HTML-Parsing), `jsonschema` (Validierung), `Flask` (API), `schedule` (Abarbeitung), `PyDSB` (Eigenes Modul: Authentifizierungs-Wrapper der inoffiziellen mobilen DSB-API)
- **Frontend:** TypeScript[2] mit `Next.js`[3], Komponenten-basiertes Rendering, zustandsarme Architektur, Skelett-Ladezustand
- **Deployment:** Docker-Container[4] (Backend und Frontend getrennt), orchestriert via `docker compose`. Healthchecks und getrennte Lebenszyklen für API und Scraping-Prozess.
- **Sicherheit:** JWT-basierte Zugriffskontrolle; Token wird im HTTP-only Cookie gehalten (Schutz vor XSS[5]), Rate-Limitierung (10 req/s) schützt Ressourcen.
- **Automatisierung:** Scheduler-Prozess führt periodisch (alle 2 Minuten, konfigurierbar) den Extraktions- und Verarbeitungslauf aus, während die API durchgehend erreichbar bleibt.
- **Qualität:** Logische Trennung in Phasen (Extraktion, Verarbeitung u. Anreicherung, Validierung, Bereitstellung), strukturierte JSON-Schemata sichern Konsistenz.

[1] https://github.com/PrtmPhlp/DSB-Backend
[2] https://www.typescriptlang.org/
[3] https://nextjs.org/
[4] https://docs.docker.com/get-started/docker-overview/
[5] https://www.cloudflare.com/de-de/learning/security/threats/cross-site-scripting/

5.2 Schichtenmodell

In Kapitel 4 wurde das Schichtenmodell eingeführt. Die Implementierung folgt diesem Modell direkt:

Datenextraktion: Authentifizierung via `PyDSB`, anschließendes Einholen dynamischer „DaVinci Touch"-Basis-URL, Ermittlung der Tages-Links und der einzelnen HTML-Tagesseiten.

Datenverarbeitung: Parsen jeder Tabelle zu einem Mapping `Tag` → `Kurs` → `Zeilen`. Anschließende Invertierung in kursbasierte JSON-Repräsentationen; Anreicherung (Lehrer-/Fachersetzung); Validierung via Schema.

API: Bereitstellung über REST-Endpunkte (`/login`, `/api/`, `/healthcheck`).

Bereitstellung: Scheduler kapselt zyklische Verarbeitung; Logging liefert telemetrische Nachvollziehbarkeit; Container liefern reproduzierbare Ausführung.

Frontend: Minimalistische, latenzarme, mobilefreundliche Darstellung mit Authentifizierung, Kurspersistenz und visueller Hervorhebung von Ausfällen.

5.3 Beispiel Backend: Parser und Datenpipeline

5.3.1 Kontext und Ziel

Die Rohquelle ist eine tabellarische HTML-Struktur mit Besonderheiten wie Fortsetzungszeilen und Kürzeln. Ziel ist, daraus ein kursorientiertes JSON zu erzeugen, das sofort nach Klassen gefiltert werden kann. Gerade diese Umorientierung (Tag → Kurs) bildet den Kernnutzen für die spätere Nutzererfahrung.

5.3.2 Designentscheidungen

Zentrale Designentscheidungen bei der Umsetzung waren die klare Trennung von Datenabruf (Fetch) und Auswertung (Parse), wodurch die Komponenten unabhängig voneinander getestet werden können. So lässt sich beispielsweise eine gespeicherte HTML-Datei direkt in den Parser einspeisen, ohne dass zuvor ein Netzabruf erforderlich ist. Für Fortsetzungszeilen wurde bewusst nur ein minimaler Kontext (`last_course`) verwendet, anstatt eine komplexe Zustandsmaschine zu implementieren. Fehlerfälle werden mithilfe des Logging-Mechanismus kategorisiert und gespeichert, beispielsweise als Warnung bei fehlenden Tabellen oder als Debug-Meldung bei verwaisten Fortsetzungszeilen. Schließlich wird die Verarbeitung der einzelnen Tage parallelisiert, da die Seiten voneinander unabhängig sind und so eine effizientere Ausführung ermöglicht wird.

5.3.3 Codeauszug: Fortsetzungslogik

Der folgende gekürzte Auszug zeigt die wesentliche Schleife, die Fortsetzungszeilen korrekt zuordnet.

```
def parse_single_day(self, day_key, day_url) -> Dict[str, List[List[str]]]:
    day_data: Dict[str, List[List[str]]] = {}
    soup = self._fetch_day_html(day_url)
    table = soup.find("table")
    if not table:
        self.logger.warning("No <table> for day '%s'", day_key)
```

```
        return day_data

    last_course: Optional[str] = None

    for row in table.find_all("tr"):
        cols = row.find_all("td")
        if not cols:
            self.logger.debug("Skipped non-data row")
            continue

        first_cell = cols[0].get_text(strip=True)

        # Leere erste Spalte => Fortsetzungszeile
        if first_cell in ("", "\xa0"):
            if last_course is None:
                # Inkonsistenter Zustand -> überspringen
                self.logger.debug("Orphan continuation row skipped: %s",
                                  [c.get_text(strip=True) for c in cols])
                continue
            course_name = last_course
        else:
            course_name = first_cell
            last_course = course_name

        row_values = [c.get_text(strip=True) for c in cols]
        day_data.setdefault(course_name, []).append(row_values)

    return day_data
```

Unmittelbar nach dem Abruf wird defensiv geprüft, ob eine Tabelle überhaupt vorhanden ist; fehlt sie, endet die Funktion kontrolliert mit einer leeren Struktur. Der Parser speichert nur einen minimalen Zustand: `last_course` merkt sich die zuletzt erkannte Klasse und ermöglicht damit, nachfolgende Zeilen ohne eigenen Klasseneintrag korrekt anzuhängen. Erscheint eine Fortsetzungszeile, bevor überhaupt eine Klasse erkannt wurde, wird sie verworfen. Dies ist eine bewusste Entscheidung zugunsten von Datenintegrität statt riskantem „Raten“. Die Schleife normalisiert jede Zeile in eine Liste von Strings; auf dieser einfachen Repräsentation bauen alle weiteren Schritte (Invertierung, Ersetzung, Validierung) auf. Dieser Ansatz hält die Funktion klein, testbar und deterministisch: Für identisches HTML entsteht immer dasselbe Dictionary.

5.4 Beispiel Frontend: Dynamische Webanwendung

5.4.1 Was bedeutet „dynamisch“?

Als dynamisch werden Webanwendungen bezeichnet, die ihre Inhalte nicht ausschließlich in Form statischer HTML-Dateien ausliefern, sondern während der Laufzeit anpassen. Neue Daten können im Hintergrund nachgeladen und direkt in die bestehende Oberfläche integriert werden, ohne dass die gesamte Seite neu geladen werden muss. Grundlage dafür ist das Zusammenspiel einer serverseitigen Datenquelle (z. B. über eine API), clientseitiger Logik zur Verwaltung des Anwendungszustands und eines Frameworks, das Änderungen effizient in die Darstellung im Browser überträgt. In der hier entwickelten Anwendung übernehmen **TypeScript** die Definition klarer Datenstrukturen und die frühzeitige Fehlererkennung, **Next.js**

als React-basiertes Framework das serverseitige Rendern und Routing, sowie die Kombination aus der UI-Komponentenbibliothek **shadcn/ui** und **Tailwind CSS** die konsistente Gestaltung der Oberfläche. Dazu gehört insbesondere die Anpassung an unterschiedliche Bildschirmgrößen (Responsivität) sowie die Unterstützung eines Dark Modes.

5.4.2 Benutzerfluss

1. **Seitenaufruf:** Der Nutzer öffnet die Anwendung. Im Hintergrund prüft das System anhand gespeicherter Cookies, ob bereits eine gültige Anmeldung vorliegt.
2. **Anmeldung:** Ist der Nutzer nicht eingeloggt, wird ihm ein kompaktes Anmeldeformular angezeigt. Nach Eingabe von Benutzerdaten authentifiziert sich das System beim Backend.
3. **Session-Aufbau:** Bei erfolgreicher Anmeldung speichert das System ein JWT in einem HTTP-only Cookie. Zusätzlich wird die Kurspräferenz des Nutzers im lokalen Speicher des Browsers abgelegt, sodass sie bei späteren Sitzungen automatisch verfügbar ist.
4. **Laden der Vertretungsdaten:** Das System ruft die relevanten Vertretungsplandaten von der API ab und bereitet sie für die Anzeige auf. Währenddessen sieht der Nutzer Lade-Skeletons, die den Seitenaufbau andeuten.
5. **Navigation im Plan:** Der Nutzer wählt seinen Kurs aus. Im Hintergrund filtert das System die Daten entsprechend und zeigt nur die relevanten Einträge. Über die Paginierung kann der Nutzer zudem den gewünschten Tag auswählen.
6. **Verfeinerung der Ansicht:** Der Nutzer kann die Anzeige weiter eingrenzen, z. B. indem er nur Unterrichtsausfälle filtert oder die Suchfunktion für bestimmte Fächer/Lehrkräfte verwendet. Das System führt die Filterung clientseitig auf den bereits geladenen Daten aus, ohne erneuten API-Aufruf.

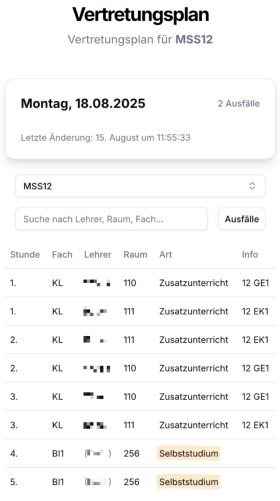


ABBILDUNG 5.1: Mobile Ansicht

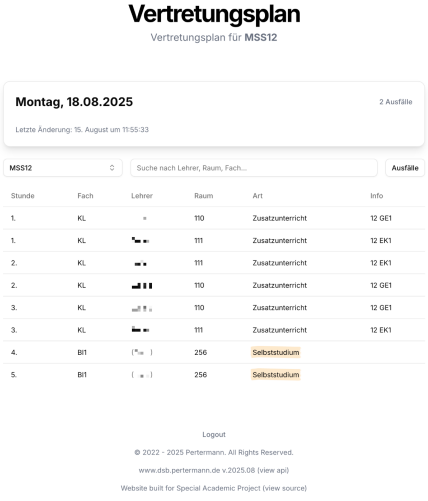


ABBILDUNG 5.2: Desktop-Ansicht

5.4.3 Responsivität und Zugänglichkeit

Die Oberfläche ist so gestaltet, dass sie sich flexibel an unterschiedliche Bildschirmgrößen anpasst. Auf mobilen Endgeräten werden Schriftgrößen automatisch reduziert, damit Inhalte auch bei geringer Displaybreite gut lesbar bleiben. Elemente, die auf größeren Bildschirmen in einer Zeile angeordnet sind – Kurswahl, Suchfeld und Filterung nach Ausfällen – werden auf Smartphones in zwei Zeilen verteilt: In der ersten Zeile befindet sich die Kurswahl, in der zweiten Zeile folgen Suchfeld und Filterung.

Auch die Tabelle reagiert auf die verfügbare Breite. Zunächst werden Spalten soweit wie möglich zusammengeschoben, damit keine horizontale Navigation notwendig ist. Erst wenn der Platz nicht mehr ausreicht, wird horizontales Scrollen ermöglicht. Auf diese Weise bleibt die Darstellung übersichtlich, ohne dass Informationen abgeschnitten oder unlesbar werden.

5.4.4 Visuelle und semantische Hervorhebung

Ausfälle oder Phasen des Selbststudiums werden durch farblich markierte Hintergründe („Highlight-Flächen“) sowie durch eine eindeutige Beschriftung gekennzeichnet. Ursprünglich vorgesehene Lehrkräfte werden in Klammern angegeben, die farblich abgeschwächt dargestellt sind, damit das Auge schneller die aktive Vertretung erfasst. Um die Abhängigkeit von reiner Farbkennung zu verringern und die Barrierefreiheit zu verbessern, werden Farben stets in Verbindung mit Text eingesetzt.

5.4.5 Skeleton Loading

Beim erstmaligen Aufruf der Anwendung werden während des Ladeprozesses formähnliche graue Platzhalter („Skeletons“) angezeigt. Diese erscheinen sowohl, solange einzelne Interface-Elemente noch nachgeladen werden, als auch während der Abruf der Vertretungsplandaten von der API erfolgt. So erhält der Nutzer frühzeitig Eindruck von Layout-Struktur und relativer Gewichtung der Informationen (z. B. Position einer Tabelle oder Überschrift), was die subjektiv empfundene Wartezeit verkürzt. Sobald die Daten eintreffen, ersetzen die tatsächlichen Inhalte nahtlos die Platzhalter, ohne Layoutsprünge.

ABBILDUNG 5.3: Darstellung von Skeleton-Elementen

5.4.6 Sicherheit und Datenhaltung

Der JWT liegt ausschließlich im HTTP-only Cookie (Schutz vor XSS-Auslesen). Das Frontend parst keine Token selbst, sondern fragt geschützte Endpunkte an und interpretiert nur Erfolg/Fehler. Läuft der Token ab, wird kontrolliert in den Login-Zustand zurückgekehrt. Suchbegriffe oder Filter werden nicht serverseitig protokolliert; sie bleiben im temporären UI-Kontext.

6 Fazit und Ausblick

6.1 Zusammenfassung und Beantwortung der Fragestellung

Die vorliegende Arbeit beschäftigte sich mit der Konzeption und Implementierung eines optimierten Systems zur Darstellung von Vertretungsplandaten auf Basis der bestehenden DSBmobile-Infrastruktur. Ausgangspunkt war die zentrale Fragestellung:

Wie kann ein bestehendes, geschlossenes Informationssystem durch moderne Softwaretechnologien und nutzerzentriertes Design so erweitert werden, dass es den Anforderungen aller Nutzer gerecht wird?

6.1.1 Anforderungen und Herausforderungen

Die Entwicklung orientierte sich an drei zentralen Anforderungskomplexen, die aus der Analyse in Kapitel 3 hervorgingen:

- **Informationsarchitektur und Benutzerfreundlichkeit:** Die zentrale Herausforderung bestand darin, den Informationsfluss so zu reorganisieren, dass Nutzer ohne überflüssigen Aufwand genau die für sie relevanten Daten auffinden. Der Paradigmenwechsel von tages- zu kursorientierter Darstellung war hier ein Schlüsselkonzept.
- **Technische Robustheit und Zuverlässigkeit:** Im Schulalltag ist höchste Verfügbarkeit unverzichtbar. Das System musste daher nicht nur unter Normalbedingungen, sondern auch bei Störungen der Primärquelle zuverlässig funktionieren und resilient reagieren.
- **Sicherheit und Datenschutz:** Der Umgang mit sensiblen, schulinternen Daten erforderte Sicherheitsmechanismen bei Authentifizierung, Autorisierung und Verarbeitung personenbezogener Informationen – unter Einhaltung geltender Datenschutzbestimmungen.

6.1.2 Umsetzung im entwickelten System

Das in den Kapiteln 4 und 5 beschriebene System adressiert diese Anforderungen durch gezielt eingesetzte Konzepte und Technologien:

- **Informationsarchitektur und Benutzerfreundlichkeit:** Die Umstellung vom tages- auf ein kursbasiertes Format ermöglicht es Nutzern, sofort ihre relevanten Vertretungen zu sehen. Aufgelöste Kürzel und Fachabkürzungen beseitigen Verständnishürden, farbliche Hervorhebungen lenken den Blick auf Ausfälle. Der Suchaufwand sinkt deutlich, der Informationszugang wird beschleunigt.
- **Technische Robustheit und Zuverlässigkeit:** Eine klar getrennte Schichtenarchitektur mit Datenbeschaffung, Verarbeitung, API und Frontend schafft ein modulares und wartbares System. Docker sichert konsistente Umgebungen,

Caching und Ratenlimitierung halten das System auch bei Problemen stabil. CI/CD ermöglicht schnelle, automatisierte Updates.

- **Sicherheit und Datenschutz:** Die Authentifizierung erfolgt über JWT in HTTP-only Cookies als Schutz vor XSS. Ratenbegrenzung und Logging sichern zusätzlich ab. Personenbezogene Daten sind nur nach Login zugänglich; für diese Arbeit wurden alle sensiblen Inhalte verpixelt oder anonymisiert.
- **Nutzerorientierung:** Schüler profitieren von mobiler Zugänglichkeit und einfacher Bedienung, Eltern von Transparenz, Lehrkräfte von einem strukturierten Überblick, und die Schulleitung von einem stabilen, sicheren und ressourcenschonenden System.

6.1.3 Beantwortung der Fragestellung

Auf Basis dieser Ergebnisse lässt sich die zentrale Frage klar beantworten: **Ein geschlossenes Informationssystem kann durch den gezielten Einsatz von Webscraping, moderner Softwarearchitektur und nutzerzentriertem Design erfolgreich erweitert werden, ohne Änderungen am Ursprungssystem vorzunehmen.**

Das System zeigt, dass durch:

- effiziente Datenextraktion und -transformation,
- nutzerorientierte Informationsarchitektur,
- technologische Entkopplung und
- konsequente Berücksichtigung von Sicherheit und Datenschutz

die identifizierten Defizite des bestehenden DSBmobile-Systems überwunden werden können. Damit belegt die Arbeit, dass selbst ein ursprünglich restriktiv angelegtes Informationssystem durch ein modulares Erweiterungskonzept sowohl funktional als auch in der Nutzererfahrung erheblich verbessert werden kann.

6.2 Potenzielle Weiterentwicklungen und persönliches Fazit

6.2.1 Mögliche Weiterentwicklungen

Das entwickelte System stellt einen funktionsfähigen Prototyp dar, der bewusst offen für Erweiterungen konzipiert ist. Naheliegende Weiterentwicklungen sind:

1. **Personalisierte Push-Benachrichtigungen:** Konfiguration als Progressive Web App (PWA) mit Service Worker und Web Push API, um Nutzer über kursrelevante Änderungen in Echtzeit zu informieren.
2. **Integration mit dem persönlichen Stundenplan:** Verknüpfung von Vertretungs- und Regeldaten, ggf. durch OCR-Verfahren, um Bilddateien in maschinenlesbare Strukturen zu transformieren.
3. **Kalenderintegration:** Export von Vertretungsinformationen als iCalendar-Dateien (.ics) oder per WebCal-Link in Systeme wie Google Kalender, Apple Kalender oder Outlook.
4. **Analytische Funktionen:** Aufbereitung von Daten für die Schulverwaltung, z. B. zur Identifikation von Mustern oder zur Personalplanung.

5. **Native mobile Apps:** Umsetzung in Swift (iOS) und Kotlin/Java (Android) für optimierte mobile Nutzung, allerdings mit eigenem Entwicklungs- und Wartungsaufwand.

6.2.2 Persönliches Fazit

Mit dieser vielschichtigen, aber in sich geschlossenen Architektur belegt die Arbeit, dass selbst ein ursprünglich restriktiv konzipiertes Informationssystem durch ein modulares Erweiterungskonzept sowohl funktional als auch in der Nutzererfahrung grundlegend verbessert werden kann, ohne das Ausgangssystem zu gefährden oder zusätzlichen Administrationsaufwand zu erzeugen. Die Bearbeitung hat mir zudem ein vertieftes Verständnis für die Relevanz nutzerzentrierter Gestaltung vermittelt und gezeigt, wie technische Konzepte konkrete Alltagsprobleme lösen können.

6.2.3 Reflexion der Arbeit

Rückblickend war die Entwicklung des Systems von mehreren Herausforderungen geprägt. Die größte technische Schwierigkeit lag in der zuverlässigen Datenextraktion aus den HTML-Tabellen des bestehenden Systems. Schon kleine Abweichungen in der Struktur konnten zu fehlerhaften Ergebnissen führen. Die konsequente Trennung von Datenabruf und -auswertung erwies sich hier als besonders wertvoll, da sie eine gezielte Fehlersuche und eine hohe Testbarkeit ermöglichte. Auch das Logging strukturierter Fehlermeldungen war im Entwicklungsprozess unverzichtbar, da dadurch Unstimmigkeiten frühzeitig sichtbar wurden.

Eine zweite Herausforderung lag in der Balance zwischen Einfachheit und Funktionsumfang. Viele potenzielle Erweiterungen – etwa eine komplexere Nutzerverwaltung – wurden bewusst nicht umgesetzt, um die Kernfunktionalität stabil und übersichtlich zu halten. Dieser Fokus auf das Wesentliche erwies sich im Nachhinein als richtige Entscheidung, da er die Robustheit und Benutzerfreundlichkeit des Systems deutlich erhöhte.

Wesentlich war zudem die sichere Gestaltung der Authentifizierung. Anstatt sensible Zugangsdaten dauerhaft im Frontend zu speichern, erfolgt die Anmeldung über Token, die ausschließlich in einem HTTP-only Cookie hinterlegt werden. Diese Maßnahme schützt effektiv vor dem Auslesen durch Skripte im Browser und vermittelt praxisnah, welche Anforderungen an sichere Sitzungsverwaltung in schulischen Softwareprojekten gestellt werden.

Insgesamt hat die Bearbeitung gezeigt, wie wichtig eine klare Architektur, strukturiertes Vorgehen und bewusste Abgrenzungen im Funktionsumfang für den Erfolg eines Projekts sind. Für eine zukünftige Weiterentwicklung würde ich diesen Ansatz beibehalten, aber noch stärker auf Automatisierung von Tests und eine frühzeitige Einbindung von Feedback der späteren Nutzer achten.

Literaturverzeichnis

Dijkstra, Edsger W. (1982). „On the role of scientific thought“. In: *Selected Writings on Computing: A Personal Perspective*. New York, NY, US: Springer-Verlag, S. 60–66. ISBN: 0-387-90652-5. URL: https://www.cs.utexas.edu/users/EWD/transcriptions/EWD04xx/EWD447.html (besucht am 15. 08. 2025).

DSBmobile (23. Mai 2025). *Screenshot eines Vertretungsplans*. Nicht öffentlich zugänglich; Zugriff nur nach Login mit Passwort. URL: https://dsbmobile.de (besucht am 25. 05. 2025).

Görres-Gymnasium Koblenz (o. D.). *Kollegiumsübersicht*. URL: https://www.goerres-koblenz.de/kollegium (besucht am 30. 06. 2025).

heinekingmedia GmbH (o. D.). *Impressum*. URL: https://heinekingmedia.de/impressum/ (besucht am 25. 05. 2025).

Jannasch, Moritz (2023). *PyDSB: Python API for DSBmobile (Version 2.3.0)*. Version 2.3.0, veröffentlicht am 12. Januar 2023. URL: https://github.com/mojansch/pydsb (besucht am 15. 08. 2025).

Rhein-Zeitung (o. D.). *Verändert Koblenzer Görres-Gymnasium sein Profil?* URL: https://www.rhein-zeitung.de/lokales/koblenz-region/veraendert-koblenzer-goerres-gymnasium-sein-profil_arid-4037743.html (besucht am 15. 07. 2025).