

Contents

1	Übersicht	1
2	Einführung	1
3	Anforderungen	2
3.1	Funktionale Anforderungen an die Web-Applikation	2
3.2	Nicht-funktionale Anforderungen an die Web-Applikation	3
3.3	Zusätzliche Anforderungen	3
4	Tech Stack	4
4.1	Framework	4
4.2	Programmiersprache	5
4.3	Datenbank	5
5	Arbeitsprozess	6
5.1	Entwicklungsmodell	6
5.2	Entwicklungsumgebung	6
5.3	Versionskontrolle	6
6	Nutzerhilfe	7
6.1	Sidebar	7
6.2	Navbar	7
6.3	Anmeldung	7
6.4	Ausleihen & Zurückgeben	8
6.5	Bestandsinformation	10
6.6	Nutzerverwaltung	13
6.7	Gruppen	14
6.8	Sensortypen	14
6.9	Dateiverwaltung	17
6.10	QR-Code	18
6.11	Einstellungen	18
7	Technische Umsetzung	18
7.1	Struktur der Web-Applikation	18
7.2	Datenbank	21
7.3	Cookies	23
7.4	Deployment	24
8	Diskussion	24
9	Ausblick	24

1 Übersicht

2 Einführung

Die Verwaltung von Messmitteln am Fachgebiet Fahrzeugantriebe der Technischen Universität Berlin gestaltet sich schwierig. Zurzeit werden Messmittel dezentral verwaltet. Die handelnden Personen besitzen wenig Information über den Ausleihzustand einzelner Messmittel über verschiedene Projekte hinweg. Daher hat sich das Fachgebiet entschlossen, im Rahmen des Moduls “Projekt Fahrzeugantriebe” eine Web-Applikation zur Verwaltung von Messmitteln zu erstellen. Diese Web-Applikation soll die Ausleihe und Rückgabe einzelner Messmittel über einen QR-Code realisieren. Darauf hinaus sollen Messmittel gruppiert, zusätzliche Informationen, wie zum Beispiel Datenblätter für Messmittel, bereitgestellt und der Bestand von Messmitteln

erfasst werden. Um die unterschiedlichen Verantwortlichkeiten der Akteure zu berücksichtigen, ist zudem eine rechtebasierte Nutzerverwaltung vorzusehen.

Der Inhalt dieses Berichtes umfasst die Dokumentation des Arbeitsprozesses, die Beschreibung des finalen Produktes hinsichtlich technischer Umsetzung und Funktionalität und zeigt zusätzlich Anknüpfungspunkte für Folgeprojekte auf.

Messmittel = Sensor

3 Anforderungen

Aus der vom Fachgebiet bereitgestellten Aufgabenstellung wurden unmittelbar die folgenden Anforderungen abgeleitet: Aufbauend auf einer Literaturrecherche soll ein Konzept zur Erstellung einer Web-Applikation entworfen werden. Die Umsetzung besteht aus der Programmierung der entworfenen Web-Applikation sowie dem Anlegen einer zugehörigen Datenbank auf dem Server des Fachgebiets. Die verwendeten Technologien sind dabei begründet frei zu wählen. Die konkreten Anforderungen an die Web-Applikation sind dabei aus dem in der Aufgabenstellung beschriebenen Funktionstest abgeleitet worden:

- Erstellen von Sensoren
- Erstellen von Nutzern
- Matching unterschiedlicher Sensortypen mit jeweils einem QR-Code
- Ausleihe und Rückgabe von Sensoren
- Konsistente Datenverwaltung in Form einer Datenbank

Diese Anforderungen stellen den minimalen Satz an Anforderungen dar. Während der Bearbeitung des Projektes zeigte sich, dass zusätzliche Anforderungen notwendig sind, um einen zweckmäßigen Einsatz am Fachgebiet sicherzustellen. Die Web-Applikation soll beispielsweise sowohl von Studenten, wissenschaftlichen Mitarbeitern und speziell geschultem Personal (üblicherweise wissenschaftliche Mitarbeiter) mit unterschiedlichen Verantwortlichkeiten genutzt werden. Daher müssen die Minimalanforderungen erweitert werden. Nutzer besitzen unterschiedliche Rechte und müssen sich gegenüber der Web-Applikation authentifizieren. Der finale Satz an Anforderungen kann den folgenden Abschnitten zu funktionalen und nicht-funktionalen Anforderungen entnommen werden.

3.1 Funktionale Anforderungen an die Web-Applikation

Funktionale Anforderungen spiegeln den Funktionsumfang der Web-Applikation wider. Die Vielzahl funktionaler Anforderungen bedingt ein Aufgliedern in verschiedene Funktionsbereiche, die im Folgenden detailliert beschrieben werden.

3.1.1 Sensorverwaltung

Die Sensoren am Fachgebiet stehen untereinander in Beziehung: Ein Versuchsaufbau kann sich aus verschiedenen Sensoren zusammensetzen. Dabei ist weniger der Sensor als viel mehr seine Art von Interesse. So ist es zum Beispiel unerheblich von welchem Hersteller oder aus welcher Serie ein konkreter Sensor ist, solange die gleiche Funktionalität erbracht wird. Der konkrete Sensor wird als *Untertyp* bezeichnet. Die Funktionalität, die alle *Untertypen* verbindet, spiegelt sich im *Typen* wider. Ein Versuchsaufbau ist schließlich eine *Gruppe*, die verschiedene *Typen* beinhaltet. Zusammenfassend lässt sich festhalten:

Jeder Sensor hat einen *Untertypen*. Mehrere nur geringfügig unterschiedliche *Untertypen* werden in einem *Typen* zusammengefasst. Mehrere *Typen* können Teil einer *Gruppe* sein. Ein *Typ* kann Teil mehrerer Gruppen sein.

Ausgehend von diesen Definitionen sind die folgenden Anforderungen zu erfüllen:

- Erstellen und Entfernen von *Gruppen*, *Typen* und *Untertypen*
- Umbenennen von *Gruppen*, *Typen* und *Untertypen*
- Bestandserfassung auf Ebene der *Untertypen*

- Matching von QR-Codes auf Ebene der *Typen*
- Verknüpfung von *Gruppen* mit *Typen* und von *Typen* mit *Untertypen*

3.1.2 Dateiverwaltung

- Hochladen, Umbenennen und Löschen von Datenblättern im PDF-Format für *Gruppen*, *Typen* und *Untertypen*
- Herunterladen von einzelnen Datenblättern als PDF-Datei oder mehreren Datenblättern komprimiert in einer zip-Datei

3.1.3 Nutzerverwaltung

Wie eingangs beschrieben, ist eine rechtebasierte Verwaltung notwendig. Dazu werden drei Rollen angelegt. Der *Benutzer* kann Sensoren ausleihen und zurückgeben. Er kann seinen Benutzernamen und sein Passwort ändern. Der *Moderator* kann zusätzlich *Benutzer* anlegen. Der *Administrator* kann zusätzlich auf die Sensorverwaltung zugreifen und Nutzer mit einer beliebigen Rolle anlegen. Eine genaue Aufschlüsselung der Rechte der drei Rollen - und somit der Anforderungen an die Nutzerverwaltung - kann Abbildung ... entnommen werden.

TODO: Rechte-Matrix

3.1.4 Ausleihverwaltung

- Ausleihe und Rückgabe von *Untertypen* beliebiger Menge innerhalb eines verfügbaren Rahmens durch Nutzer oder für beliebigen Nutzer durch *Administrator*
- Abschreiben von *Untertypen* durch *Administratoren*
- Anzeige von Datenblättern für *Gruppen*, *Typen* und *Untertypen*

3.1.5 Bestandsinformation

Um ohne Betreten des Lagerortes ermitteln zu können, wie viele Elemente eines *Typen* oder *Untertypen* verfügbar sind oder festzustellen, welcher Nutzer einen benötigten *Typen* ausgeliehen hat, ist eine Übersicht über den Bestand und die Ausleihhistorie zu implementieren.

3.2 Nicht-funktionale Anforderungen an die Web-Applikation

Für einen nachhaltigen Einsatz der Web-Applikation sind die folgenden nicht-funktionalen Anforderungen zu erfüllen:

- Intuitive Nutzerführung
- Konsistenz durch wiedererkennbares Layout und Design
- Performance

Aufgrund ihres nicht-funktionalen Charakters ist die Erfüllung nicht an konkrete Bedingungen geknüpft. Alle unternommenen Bestrebungen zur Erfüllung der funktionalen Anforderungen sind stets hinsichtlich der hier aufgeführten nicht-funktionalen Anforderungen zu bewerten.

3.3 Zusätzliche Anforderungen

Zusätzlich soll eine Datenbank für die konsistente Verwaltung der anzulegenden Daten genutzt werden. Die Web-Applikation und die Datenbank sollen auf einer virtuellen Maschine (*VM*), die auf einem Server des Fachgebiets abgelegt wird, betrieben werden.

3.3.1 Datenbank

Die Datenbank soll die konsistente Datenverwaltung bewerkstelligen. Sie enthält Tabellen, welche entsprechend der funktionalen Anforderungen der Web-Applikation zu gestalten und miteinander zu verknüpfen sind.

Hierzu müssen sowohl ein geeignetes Datenbankmodell als auch ein konkretes Datenbankmanagementsystem ausgewählt werden.

3.3.2 Deployment

Als Deployment wird die Integration der Web-Applikation und der Datenbank in die bestehende Infrastruktur bezeichnet. Dazu sind folgende Schritte notwendig:

- Auswahl einer Virtualisierungssoftware
- Einrichten einer *VM*
- Installation von Servern, Wartungssoftware und Programmiersprache
- Transfer von Datenbank und Web-Applikation auf *VM*

4 Tech Stack

Als Tech Stack wird die Summe der verwendeten Technologien bezeichnet. Dazu gehören zum Beispiel die Programmiersprache, das Framework für die Web-Applikation, die Datenbanksoftware, aber auch die Entwicklungsumgebung und weitere Software, die im Entwicklungsprozess verwendet wird.

4.1 Framework

Zuallererst muss das Framework zur Erstellung der Web-Applikation gewählt werden. Dieses legt normalerweise die zu verwendende Programmiersprache fest und setzt möglicherweise Restriktionen in Bezug auf weitere Software. Eine Web-Applikation zeichnet sich dadurch aus, dass sie im Webbrower ausführbar ist. Der Webbrower ist in der Lage, Dateien im HTML-Format (HTML: Hyper Text Markup Language) darzustellen. Das HTML-Format spezifiziert dabei ausschließlich die Struktur der Webseite. Um die visuelle Erscheinung der Webseite zu beeinflussen, können Regeln in CSS-Dateien (CSS: Cascading Style Sheet) hinterlegt werden. Für interaktives Verhalten existiert die Sprache JavaScript, die es ermöglicht, das HTML-Dokument dynamisch anzupassen.

Ein Framework zur Erstellung von Web-Applikationen bietet ein Grundgerüst für Layout sowie Funktionalität und stellt einen Server bereit. Für das Layout werden beispielsweise makroskopische Komponenten, wie Dashboards und Landing Pages, oder mikroskopische Komponenten, wie Inputs, Tabellen und Plots, bereitgestellt. Die Funktionalität wird abstrahiert und der Zustand der Web-Applikation modelliert. Der Server bearbeitet Anfragen von Clients, also Nutzern der Web-Applikation. Frameworks können in beliebigen Programmiersprachen implementiert werden, solange eine Schnittstelle zwischen der vom Framework verwendeten Sprache und einer dem Brower verständlichen Sprache existiert. Frameworks können hinsichtlich verschiedener Kriterien unterschieden werden. Backend-Frameworks integrieren neben einem Server meist auch noch Datenbanken, wohingegen Frontend-Frameworks ihren Fokus mehr auf dem visuellen Part legen. Multipage-Frameworks enthalten mehrere Seiten, wohingegen Singlepage-Frameworks nur eine einzige Seite darstellen.

Für die Bearbeitung dieses Projektes wurde das Framework Shiny gewählt, das in der Programmiersprache R implementiert ist. Hierbei handelt es sich um ein Singlepage-Framework, das als Backend den sogenannten Shiny Server enthält. Maßgeblich für die Entscheidung war, dass die beiden Autoren über Erfahrung im Umgang mit R und im Speziellen mit Shiny verfügen. Darüber hinaus zeichnet sich Shiny durch folgende Eigenschaften und Vorzüge aus:

- Moderne Templates
- Fokus auf konkrete Funktionalität
- Reaktives Zustandsmodell
- Modularisierbarkeit
- Für Anwendungsfälle ausreichende Performance
- Einfache Integration von Datenbanken

4.2 Programmiersprache

R ist eine Multiparadigmen-Programmiersprache. Je nach Anwendungsfall kann somit zum Beispiel objektorientiert oder funktional programmiert werden. R verfügt einen Pool an Standardbibliotheken und kann einfach durch selbstgeschriebene und frei verfügbare Packages erweitert werden. Das Comprehensive R Archive Network (CRAN) stellt eine Vielzahl von quelloffenen Bibliotheken zur Verfügung beispielsweise `{shiny}`, das die Funktionalitäten des Frameworks beinhaltet. Die folgenden Tabellen geben Aufschluss über die im Projekt verwendeten Packages und ihren Zweck. Packages können von anderen Packages abhängen. Es wird daher darauf verzichtet auf untergeordnete Bibliotheken einzugehen.

4.2.1 Packages für `{shiny}`

Package	Beschreibung
<code>{bs4Dash}</code>	AdminLTE-Template
<code>{DT}</code>	DataTables für <code>{shiny}</code>
<code>{htmltools}</code>	HTML-Repräsentation in R
<code>{rclipboard}</code>	Zwischenablage
<code>{shinydisconnect}</code>	Verbindungsverlustbildschirm
<code>{shinyjs}</code>	Integration von Custom-JavaScript
<code>{waiter}</code>	Ladebildschirm

4.2.2 Packages für die Programmierung

Package	Beschreibung
<code>{Cairo}</code>	PDF-/PNG-/SVG-Erstellung
<code>{DBI}</code>	Datenbankinterface
<code>{dplyr}</code>	Datentransformationen
<code>{glue}</code>	String-Erzeugung
<code>{lubridate}</code>	Datumsformat
<code>{RSQLite}</code>	SQLite-Datenbank
<code>{stringr}</code>	String-Manipulation
<code>{tibble}</code>	Tabellenformat
<code>{purrr}</code>	Funktionale Programmierung
<code>{qrbase}</code>	Erstellung von QR-Codes
<code>{renv}</code>	Packagemanagement
<code>{yaml}</code>	YAML-Dateiformat

4.3 Datenbank

Zur konsistenten Datenverwaltung wird eine Datenbank benötigt. Datenbanken sind in der Lage, Anfragen von verschiedenen Clients zu bearbeiten und dabei zu gewährleisten, dass bestimmte Regeln hinsichtlich der Datenstruktur und Ausprägung der Daten eingehalten werden. Es existieren verschiedene Datenbankmodelle, unter anderem das Netzwerk-, das objektorientierte, das hierachische oder das relationale Datenbankmodell.¹ Diese unterscheiden sich hinsichtlich der Verknüpfung der beteiligten Daten. Aufgrund der hohen Flexibilität und der weiten Verbreitung wurde das relationale Datenbankmodell ausgewählt. Dieses speichert die Daten in miteinander verknüpften Tabellen. Die Tabellenzeilen enthalten Beobachtungen, die Tabellenspalten stellen die beobachtbaren Merkmale dar. Zur eindeutigen Identifikation erhält jede Zeile eine Identifikationsnummer. Die Spalte der Identifikationsnummern wird als Primärschlüssel (*Primary Key*) bezeichnet. Um verschiedene Tabellen miteinander zu verknüpfen, werden Identifikationsnummern referenziert. Eine Spalte, die auf einen Primärschlüssel einer anderen Tabelle verweist, wird als Fremdschlüssel (*Foreign Key*) bezeichnet.

¹Laudon, K.C.; Laudon, J.P.; Schroeder, D.: *Wirtschaftsinformatik - Eine Einführung*, Pearson, 2015, S. 295-300

Es gibt eine Vielzahl verschiedener relationaler Datenbankmanagementsysteme, die sich hinsichtlich ihrer Anwendungsbereiche und Skalierbarkeit unterscheiden. Für Projekte kleinen und mittleren Umfanges (unter 100.000 Aufrufe / Tag²) eignet sich SQLite. Hierbei werden alle Tabellen in einer einzigen Datei mit dem Suffix *.sqlite* gespeichert. Der Zugriff auf die Datenbank erfolgt grundsätzlich über die *Structured Query Language* (SQL). Für die Programmiersprache R gibt es die Packages {DBI} und {RSQLite}, die eine direkte Schnittstelle zur Datenbank bereitstellen.

5 Arbeitsprozess

Neben der technischen Umsetzung ist insbesondere der Arbeitsprozess von herausragender Bedeutung. Durch diesen wird festgelegt, in welcher Weise die Anforderungen letztendlich umgesetzt werden. Für Projekte im Allgemeinen beinhaltet der Arbeitsprozess das Zeit-, Personal- und Aufgabenmanagement. Ein gut strukturierter Arbeitsprozess hilft schließlich dabei, Ergebnisse effizient und nachvollziehbar zu erzielen. In der Softwareentwicklung wird der Arbeitsprozess zusätzlich durch die Wahl von Entwicklungsmodell, Entwicklungsumgebung und Versionskontrolle bestimmt.

5.1 Entwicklungsmodell

In der Softwareentwicklung existiert eine Vielzahl verschiedener Modelle zur Bewältigung eines Projektes. Die klassischen Modelle (Wasserfallmodell, V-Modell) fokussieren sich darauf, Phasen sequentiell abzuarbeiten. Im Kontrast dazu stehen die agilen Modelle, in denen alle Phasen wiederholt durchlaufen werden. Agile Modelle sind in der Regel deutlich flexibler, da die Anforderungen kontinuierlich angepasst werden können. Für kleine Entwicklungsteams bietet sich die Verwendung eines agilen Entwicklungsmodells auch deswegen an, weil der Koordinationsaufwand zwischen den Teilnehmern gering ist. Das Entwicklungsmodell legt nicht nur fest, wie das Projekt auf der Makroebene strukturiert ist, sondern auch wie auf der Mikroebene konkret programmiert wird.

Wir haben uns dafür entschieden, das agile Modell des Pair Programming - einer Unterform des Extreme Programming - einzusetzen. Hierbei arbeiten stets zwei Programmierer (also im vorliegenden Fall alle) gemeinsam an der Erstellung von Programmcode. Vorteile hiervon liegen im stetigen Informationsaustausch, gemeinsamer Entscheidungsfindung und geringerer Fehlerhäufigkeit. Als nachteilig wird allgemeinhin der doppelte Personalaufwand angesehen.

5.2 Entwicklungsumgebung

Als Entwicklungsumgebung wird die Software bezeichnet, die zur Erstellung und Verwaltung des Programmcodes genutzt wird. Für die Programmiersprache R empfiehlt es sich, die Entwicklungsumgebung RStudio zu verwenden. Diese ermöglicht es, Projekte anzulegen, die Web-App für das Testen unmittelbar auszuführen und den Code mit Git und GitHub für die Versionsverwaltung zu integrieren. Eine interaktive Konsole und eine integrierte Hilfe erleichtern den Arbeitsprozess.

5.3 Versionskontrolle

Versionskontrolle ist aus vielerlei Gründen für den Softwareentwicklungsprozess unerlässlich. Sie ermöglicht es,

- Versionen zu verwalten,
- Bugs durch Differenzbildung zwischen verschiedenen Versionen zu finden und zu beheben sowie
- den Projektfortschritt zeitlich und inhaltlich nachzuvollziehen.

Für die Versionskontrolle wurde Git in Verwendung mit GitHub eingesetzt. In Git werden inkrementelle Änderungen durch sogenannte Commits erfasst. Jeder Commit ist dabei mit einem Kommentar versehen. Durch sogenannte Branches können verschiedene Personen gleichzeitig zum Projekt beitragen oder verschiedene Features gleichzeitige entwickeln. Branches können wieder zusammengeführt werden (Merging). Ein Ordner,

²SQLite Consortium: *Appropriate Uses for SQLite*, URL: <https://www.sqlite.org/whentouse.html>, Letzter Aufruf: 14.03.2021

der mit Git initialisiert wurde, wird als Repository bezeichnet. Repositories können im Internet verfügbar gemacht werden und dann auf beliebigen Computern heruntergeladen werden.

Die Online-Plattform GitHub erleichtert die Kollaboration über Git. Sie stellt einen Ablageort für das Repository bereit und unterstützt den Arbeitsprozess durch ein Ticket-System (Issues). In diesem können Fehler und Verbesserungsvorschläge gemeldet werden. Die Issues können darüber hinaus als Notizblock für geplante Features verwendet werden.

6 Nutzerhilfe

Die folgenden Abschnitte fassen wesentliche Merkmale der Web-Applikation zusammen und erläutern ihre Bedienung. Die technischen Details können im anschließenden Kapitel nachgeschlagen werden. Die Web-Applikation ist ein Dashboard. Dieses besitzt eine Sidebar, eine Navbar und einen Body, vergleiche Abbildung 1.

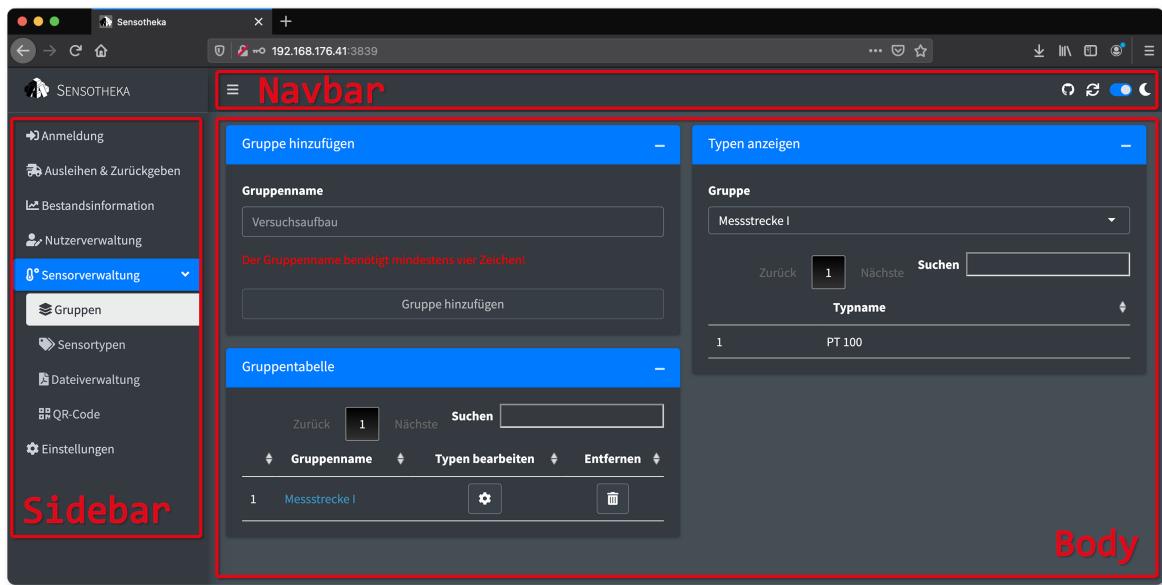


Abbildung 1: Übersicht über das Dashboard

6.1 Sidebar

In der Sidebar kann über einen Reiter die Funktionalität ausgewählt werden, die im Body dargestellt wird. Die Anzahl an Auswahlmöglichkeiten hängt vom Anmeldestatus und der Benutzerrolle ab.

6.2 Navbar

Die Navbar enthält:

- einen Link zum GitHub-Repository, das den Quellcode der Web-Applikation enthält,
- einen Reload-Button, mit dem eine Aktualisierung der Daten erzwungen werden kann,
- einen Toggle, mit dem zwischen Tag- und Nachtmodus gewechselt werden kann.

6.3 Anmeldung

Dieser Reiter verändert sich in Abhängigkeit des Anmeldestatus.

6.3.1 Status: Nicht angemeldet

Registrierte Benutzer können sich durch Angabe ihres Benutzernamens und ihres Passwortes anmelden, vergleiche Abbildung 2. Nicht-registrierte Benutzer müssen sich von einem Moderator oder Administrator (Wissenschaftliche Mitarbeiter) registrieren lassen.

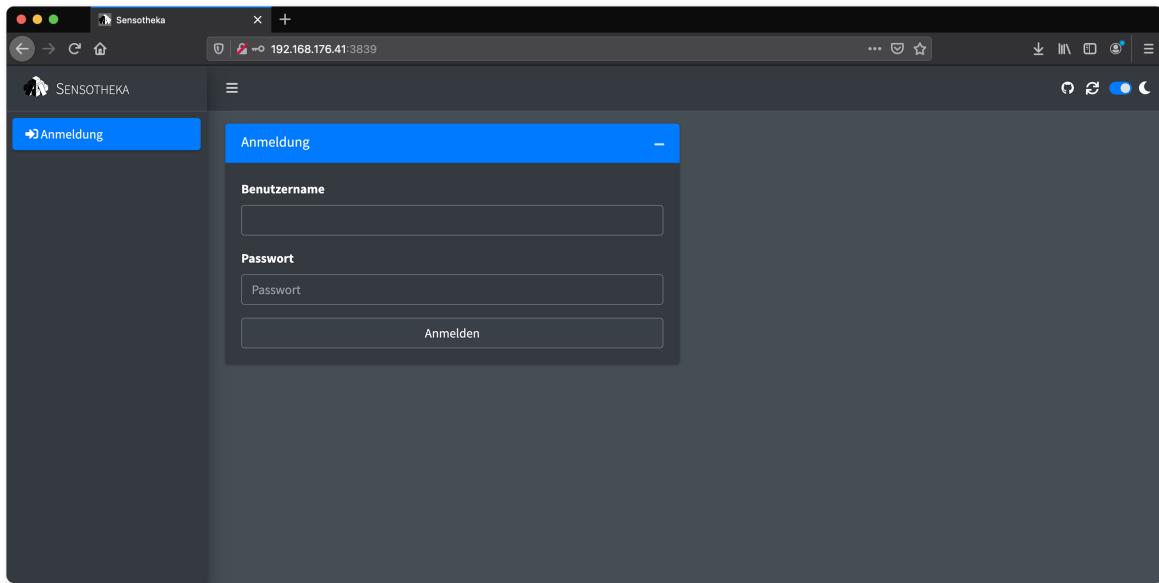


Abbildung 2: Anmeldung - nicht angemeldet

6.3.2 Status: Angemeldet

Nach erfolgreicher Anmeldung wird der Benutzer mit Informationen über sein Nutzungsverhalten versorgt, vergleiche Abbildung 3. Dargestellt werden:

- der Benutzername und der Benutzerstatus,
- die Dauer seit der momentanen Anmeldung,
- die Dauer seit der letzten Anmeldung,
- die Anzahl der Anmeldungen.

Benutzer können sich zudem abmelden.

6.4 Ausleihen & Zurückgeben

Die Ausleihe und Rückgabe setzt sich aus drei Abschnitten zusammen, vergleiche Abbildung 4. Im ersten Abschnitt *Ausleihen & Zurückgeben* kann die konkrete Operation vorgenommen werden. Zusätzliche Informationen stellen die Abschnitte *Gruppen des ausgewählten Typs* und *Dateien* bereit.

6.4.1 Ausleihen & Zurückgeben

Zunächst muss ein Typ ausgewählt werden. Wenn die Web-Applikation über einen QR-Code aufgerufen wurde (siehe QR-Code), ist der zum QR-Code gehörende Typ bereits ausgewählt. Im nächsten Schritt muss ein zugehöriger Untertyp ausgewählt werden. Die konkrete Operation kann durch Klicken auf einen der verfügbaren Buttons angestoßen werden. Im sich darauf öffnenden Dialog muss die Menge angegeben und die Ausführung der Operation bestätigt werden. Abbildung 5 gibt einen Überblick über den Abschnitt.

Administratoren können zusätzlich den ausführenden Nutzer auswählen, um Operationen für diesen

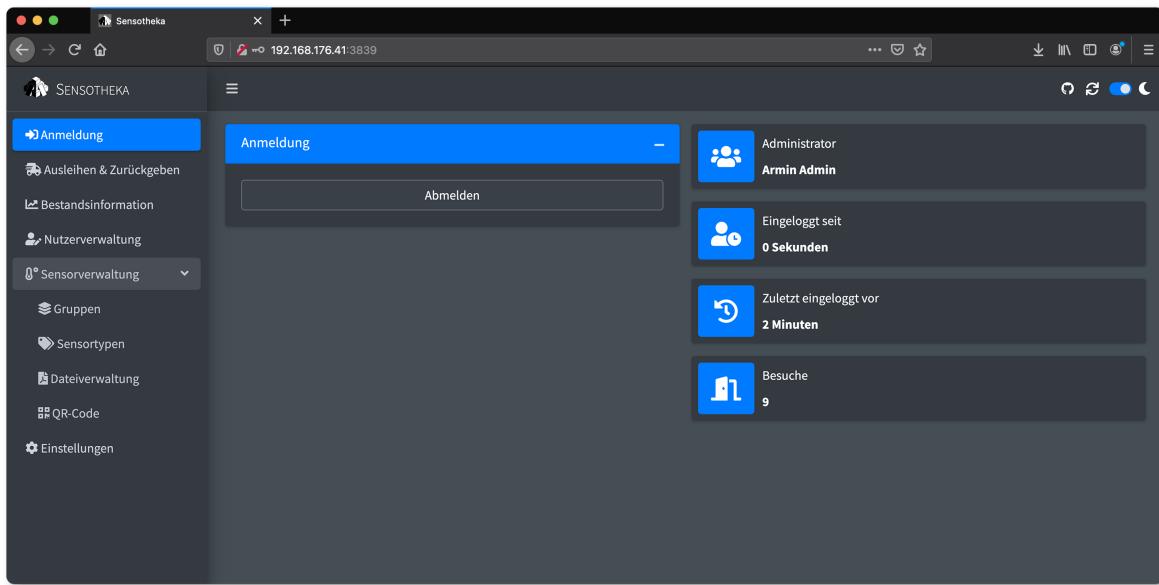


Abbildung 3: Anmeldung - angemeldet

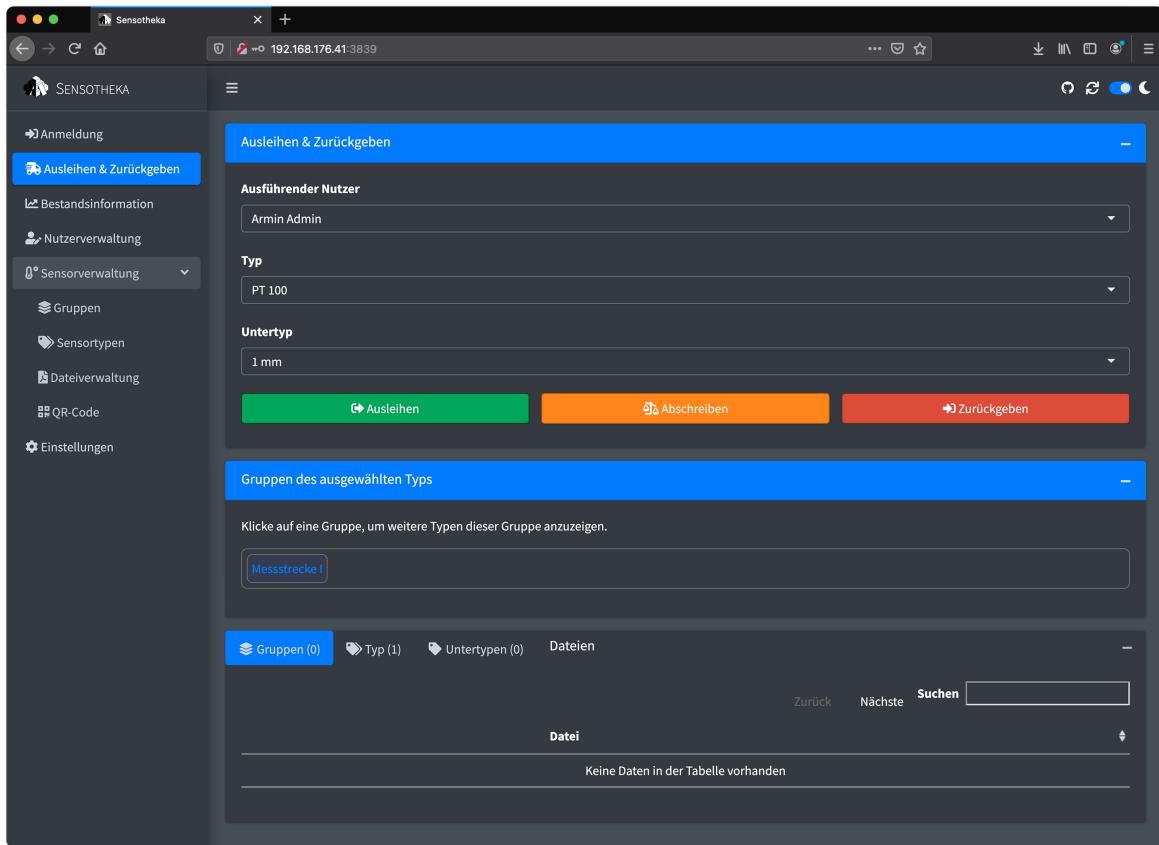


Abbildung 4: Übersicht: Ausleihen & Zurückgeben

durchzuführen. Das ist zum Beispiel notwendig, wenn ein Student das Fachgebiet verlassen hat und es versäumt hat, alle ausgeliehenen Sensoren zurückzugeben.

Es können bis zur maximal verfügbaren Menge Elemente ausgeliehen werden.

Es können bis zur maximal ausgeliehenen Menge Elemente zurückgegeben werden.

Es können bis zur maximal im Lager verfügbaren Menge Elemente abgeschrieben werden. Dies ist zum Beispiel notwendig, wenn ein Element ausfällt oder verloren geht.

Abbildung 5: Abschnitt: Ausleihen & Zurückgeben

6.4.2 Gruppen des ausgewählten Typs

Alle Gruppen des ausgewählten Typs werden aufgelistet. Durch Klicken auf eine Gruppe öffnet sich ein Dialog, in dem alle Typen dieser Gruppe angezeigt werden, vergleiche Abbildung 6. Durch Klicken auf einen Typen wird dieser Typ für eine weitere Operation ausgewählt. Somit können alle Elemente einer Gruppe (zum Beispiel ein Versuchsaufbau) komfortabel hintereinander ausgeliehen / zurückgegeben werden.

6.4.3 Dateien

Hier werden alle dem ausgewählten Typ zugeordneten Dateien angezeigt. Dabei ist die Anzeige - wie Abbildung 7 entnommen werden kann - in drei Reiter gegliedert:

- *Gruppen*: Alle Dateien, die Informationen zu Gruppen des ausgewählten Typs enthalten
- *Typ*: Alle Dateien, die Informationen zum ausgewählten Typ enthalten
- *Untertypen*: Alle Dateien, die Informationen zu Untertypen des ausgewählten Typs enthalten. Es ist zu beachten, dass die Wahl des Untertypen keinen Einfluss auf die angezeigten Dateien hat, da Dateien für alle Untertypen dargestellt werden.

Die Ziffer neben dem Reittitel gibt Auskunft darüber, wie viele Dateien in der jeweiligen Kategorie vorhanden sind. Indem auf einen Dateinamen geklickt wird, öffnet sich das zugehörige PDF in einem separaten Browserfenster oder wird über den PDF-Viewer angezeigt. Alle Dateien einer Kategorie können in einem Archiv (.zip) heruntergeladen werden.

6.5 Bestandsinformation

Die Bestandsinformation gliedert sich in zwei Abschnitte, vergleiche Abbildung 8. In der Bestandsübersicht werden Informationen zum Lagerbestand von Sensoren dargestellt, wohingegen die Ausleihübersicht Informationen über ausgeliehene Sensoren enthält.

The screenshot shows two stacked interface sections. The top section has a blue header bar with the text "Gruppen des ausgewählten Typs". Below this is a dark grey area containing the instruction "Klicke auf eine Gruppe, um weitere Typen dieser Gruppe anzuzeigen." and a single button labeled "Messstrecke I". A large blue downward arrow points from the bottom of this section to the top of the second section. The second section has a dark grey header bar with the text "Enthaltene Typen" and a small "x" icon. Below this is another dark grey area containing the instruction "Klicke auf einen Typen, um die Ausleihe bzw. die Rückgabe vorzubereiten." and two buttons labeled "PT 100" and "Multimeter".

Abbildung 6: Abschnitt: Gruppen des ausgewählten Typs

The screenshot shows a dark grey interface with several tabs at the top: "Gruppen (0)", "Typ (1)" (which is highlighted in blue), "Untertypen (0)", and "Dateien". Below the tabs are search and navigation controls: "Zurück", a page number "1", "Nächste", "Suchen" (Search) with a text input field, and a refresh icon. A horizontal line separates this from the main content area. The content area is titled "Datei" and lists one item: "1 Datenblatt - PT 100.pdf". At the bottom of the content area is a button labeled "Verzeichnis herunterladen" (Download Index).

Abbildung 7: Abschnitt: Dateien

The screenshot shows the Sensotheka software interface. On the left, a sidebar lists various menu items: Anmeldung, Ausleihen & Zurückgeben, Bestandsinformation (which is selected and highlighted in blue), Nutzerverwaltung, Sensorverwaltung (with a dropdown arrow), Gruppen, Sensortypen, Dateiverwaltung, QR-Code, and Einstellungen.

The main area has a blue header bar labeled "Bestandsübersicht". Below it, there is a search bar with the placeholder "Typ" containing "PT 100" and a checkbox "Nur kritische Bestände anzeigen". A navigation bar at the top right includes "Zurück", a page number "1", "Nächste", and a search input field.

The first table displays stock levels:

	Untertyp	Verfügbar	Maximal verfügbar
1	1 mm	55	58
2	1,5 mm	7	27
3	2 mm	25	35

Below this is a tab bar with "Gesamt", "Nach Untertyp", "Nach Nutzer", "Transaktionen" (which is selected and highlighted in blue), and "Ausleihübersicht".

The second table displays transaction history:

	Nutzer	Typ	Untertyp	Datum	Menge
1	Bernd Benutzer	PT 100	1,5 mm	2021-03-18 16:48:51	5
2	Armin Admin	PT 100	1,5 mm	2021-03-18 16:48:42	-10
3	Modesta Moderator	PT 100	2 mm	2021-03-18 16:48:32	2
4	Modesta Moderator	PT 100	2 mm	2021-03-18 16:48:27	-12
5	Bernd Benutzer	PT 100	1,5 mm	2021-03-18 16:48:14	-15
6	Armin Admin	PT 100	2 mm	2021-03-18 16:47:51	10
7	Armin Admin	PT 100	2 mm	2021-03-18 16:47:42	35
8	Armin Admin	PT 100	1,5 mm	2021-03-18 16:47:30	3
9	Armin Admin	PT 100	1,5 mm	2021-03-18 16:47:19	27
10	Modesta Moderator	PT 100	1 mm;2 (gelöscht)	2021-03-18 16:41:46	3

Abbildung 8: Übersicht: Bestandsinformation

6.5.1 Bestandsübersicht

Zunächst muss ein Typ ausgewählt werden. Für diesen Typen werden tabellarisch alle Untertypen dieses Typs aufgelistet. Die Spalte *Verfügbar* enthält die gelagerte Menge, wohingegen die Spalte *Maximal verfügbar* die maximal gelagerte Menge (kein Sensor dieses Untertypen ausgeliehen) enthält. Über die Checkbox *Nur kritische Bestände anzeigen* kann ausgewählt werden, ob nur Untertypen angezeigt werden sollen, für die die verfügbare Menge kleiner als ihr kritischer Bestand ist. Der kritische Bestand eines Untertypen kann von einem im Reiter Sensortypen festgelegt werden.

6.5.2 Ausleihübersicht

Die Ausleihübersicht besteht aus vier Reitern, die unterschiedliche Fragestellungen in Bezug auf ausgeliehene Sensoren beantworten, vergleiche Abbildung 9.

Reiter	Beschreibung
Gesamt	Übersicht über alle Untertypen mit einer gegenwärtig ausgeliehenen Menge größer Null.
Nach Untertyp	Übersicht über alle Nutzer, die den gewählten Untertypen zurzeit ausgeliehen haben. Hiermit kann herausgefunden werden, wer Ansprechpartner ist, falls alle Elemente eines Untertyps ausgeliehen sind.
Nach Nutzer	Übersicht über alle Untertypen, die der gewählte Nutzer zurzeit ausgeliehen hat.
Transaktionen	Übersicht über alle Transaktionen, die jemals im Rahmen der Sensorverwaltung stattgefunden haben. Ein Nutzer sieht ausschließlich seine eigenen Transaktionen. Ein sieht alle Transaktionen. Dazu gehören neben Ausleih- und Rückgabeoperationen auch Abschreibungen und sonstige Mengenänderungen.

6.6 Nutzerverwaltung

In der *Nutzerverwaltung* können neue Nutzer hinzugefügt und in der *Nutzertabelle* verwaltet werden, vergleiche Abbildung 10.

6.6.1 Nutzer hinzufügen

Neue Nutzer können unter Angabe eines Benutzernamens und eines zur Sicherheit zweimal anzugebenden Passwortes hinzugefügt werden. Ein kann ausschließlich hinzufügen, wohingegen ein Benutzer mit jeder Rolle hinzufügen kann.

6.6.2 Nutzertabelle

In der Nutzertabelle können Nutzer verwaltet werden. Neben dem Benutzernamen und dem gegenwärtigen Status gibt es drei Spalten, die Buttons enthalten, mit denen Eigenschaften eines Nutzer angepasst werden können.

Spalte	Beschreibung
Status ändern	Dieser Button öffnet einen Dialog, in dem ein neuer Status für den Nutzer ausgewählt werden kann.

Spalte	Beschreibung
Entfernen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass der ausgewählte Nutzer gelöscht werden soll. Es ist zu beachten, dass diese Operation nicht rückgängig gemacht werden kann. Transaktionen, die der ausgewählte Nutzer vorgenommen hat, bleiben jedoch weiterhin erhalten. Es können nur Nutzer gelöscht werden, die zurzeit keine Sensoren ausgeliehen haben. Falls der Nutzer nicht in der Lage ist, die Sensoren eigenständig zurückzugeben, kann der die ausgeliehenen Sensoren im Reiter Ausleihen & Zurückgeben für diesen Nutzer zurückgeben. Ein kann nur entfernen, die er selbst hinzugefügt hat.
Passwort zurücksetzen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass das Passwort des Nutzers zurückgesetzt werden soll. Das Passwort wird auf das Standardpasswort <i>1234</i> zurückgesetzt und sollte vom Nutzer sofort danach im Reiter Einstellungen geändert werden.

6.7 Gruppen

Der Reiter *Gruppen* erlaubt es, neue Gruppen hinzuzufügen und bestehende Gruppen zu verwalten, vergleiche Abbildung 11.

6.7.1 Gruppe hinzufügen

Eine neue Gruppe kann durch Angabe eines Gruppennamens hinzugefügt werden.

6.7.2 Gruppentabelle

In der *Gruppentabelle* können Gruppen bearbeitet werden.

Spalte	Beschreibung
Gruppename	Ein Klick auf einen Gruppennamen öffnet einen Dialog, in dem der Gruppename angepasst werden kann.
Typen bearbeiten	Dieser Button öffnet einen Dialog, in dem die zur ausgewählten Gruppe zugehörigen Typen bearbeitet werden können.
Entfernen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass die ausgewählte Gruppe gelöscht werden soll. Es ist zu beachten, dass diese Operation nicht rückgängig gemacht werden kann.

6.7.3 Typen anzeigen

Zunächst muss eine Gruppe ausgewählt werden. In der Tabelle werden alle zur ausgewählten Gruppe zugehörigen Typen dargestellt.

6.8 Sensortypen

Der Reiter *Sensortypen* erlaubt es, neue Typen und Untertypen hinzuzufügen und bestehende Typen und Untertypen zu verwalten, vergleiche Abbildung 12.

6.8.1 Typ hinzufügen

Ein neuer Typ kann durch Angabe eines Typnamens hinzugefügt werden.

The screenshot displays four tables arranged in a grid:

- Ausleihübersicht (Top Left):** Shows loans by type and subtype. Filtered for PT 100 and 1 mm. One loan record is shown.
- Ausleihübersicht (Top Right):** Shows loans by type, subtype, quantity, and last borrowed date. Filtered for PT 100 and 1 mm. Three records are shown.
- Nach Nutzer (Bottom Left):** Shows loans by user. Filtered for Bernd Benutzer. Two records are shown.
- Transaktionen (Bottom Right):** Shows transaction details. Filtered for PT 100 and 1 mm. Six records are shown, with columns for Rückgegebene Menge, Ausgeliehene Menge, and Bestandsänderungen.

Abbildung 9: Übersicht: Ausleihübersicht

The screenshot shows two windows side-by-side:

- Nutzer hinzufügen (Left):** A form for adding a new user. Fields include Benutzername (Max Mustermann), Passwort, Passwort bestätigen, and Status (Administrator). Error messages indicate the password must be at least 4 characters long.
- Nutzertabelle (Right):** A table listing existing users with actions for status change and deletion.

Benutzername	Status	Status ändern	Entfernen	Passwort zurücksetzen
Armin Admin	Administrator	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bernd Benutzer	Benutzer	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Modesta Moderator	Moderator	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Abbildung 10: Übersicht: Nutzerverwaltung

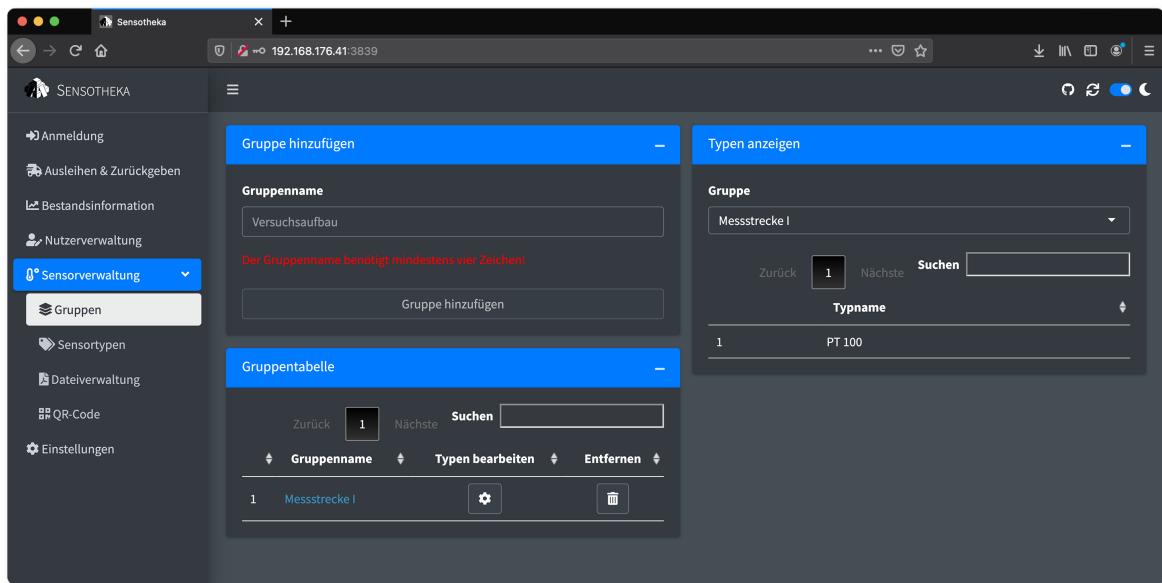


Abbildung 11: Übersicht: Nutzerverwaltung

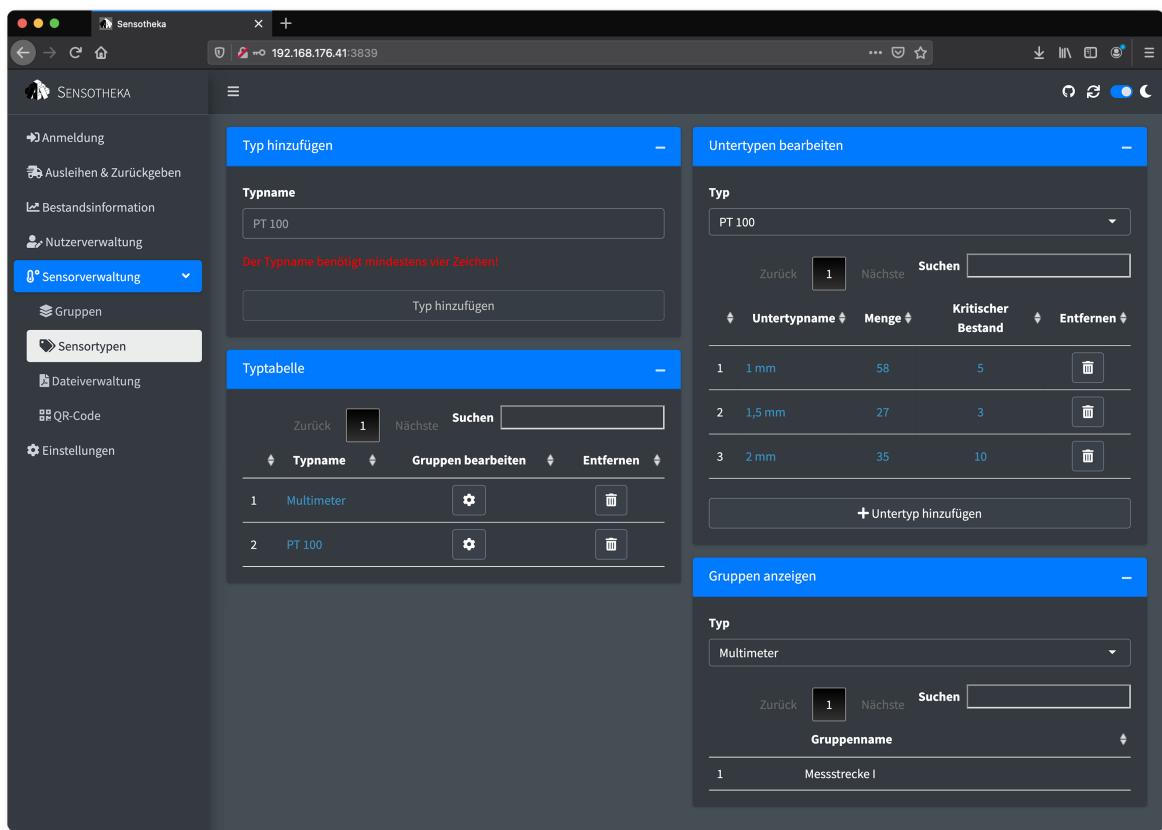


Abbildung 12: Übersicht: Sensortypen

6.8.2 Typtabelle

In der *Typtabelle* können Typen bearbeitet werden.

Spalte	Beschreibung
Typname	Ein Klick auf einen Typnamen öffnet einen Dialog, in dem der Typname angepasst werden kann.
Gruppen bearbeiten	Dieser Button öffnet einen Dialog, in dem die zum ausgewählten Typ zugehörigen Gruppen bearbeitet werden können.
Entfernen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass der ausgewählte Typ gelöscht werden soll. Es ist zu beachten, dass diese Operation nicht rückgängig gemacht werden kann. Ein Typ kann nur gelöscht werden, wenn kein Element seiner Untertypen ausgeliehen ist. Transaktionen, die den ausgewählten Typen betreffen, bleiben weiterhin erhalten. Das Entfernen eines Typen schließt das Entfernen aller Untertypen dieses Typen ein.

6.8.3 Gruppen anzeigen

Zunächst muss ein Typ ausgewählt werden. In der Tabelle werden alle zum ausgewählten Typen zugehörigen Gruppen dargestellt.

6.8.4 Untertypen bearbeiten

Zunächst muss ein Typ ausgewählt werden. In der Tabelle können Untertypen des ausgewählten Typen bearbeitet werden.

Spalte	Beschreibung
Untertypname	Ein Klick auf einen Untertypnamen öffnet einen Dialog, in dem der Untertypname angepasst werden kann.
Menge	Ein Klick auf die maximal verfügbare Menge öffnet einen Dialog, in dem die maximal verfügbare Menge des Untertypen angepasst werden kann. Es ist zu beachten, dass die maximal verfügbare Menge nicht auf einen Wert gesetzt werden kann, der kleiner der Anzahl gegenwärtig ausgeliehener Elemente ist.
Kritischer Bestand	Ein Klick auf den kritischen Bestand öffnet einen Dialog, in dem der kritische Bestand des Untertypen angepasst werden kann. Der kritische Bestand kann als Filterkriterium in der <i>Ausleihübersicht</i> des Reiters Bestandsinformation verwendet werden.
Entfernen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass der ausgewählte Untertyp gelöscht werden soll. Es ist zu beachten, dass diese Operation nicht rückgängig gemacht werden kann. Ein Untertyp kann nur gelöscht werden, wenn keine Elemente von diesem ausgeliehen sind. Transaktionen, die den ausgewählten Untertypen betreffen, bleiben weiterhin erhalten.

6.9 Dateiverwaltung

In der *Dateiverwaltung* können PDF-Dateien als Informationsmaterial in den Reitern *Gruppen*, *Typen* und *Untertypen* hochgeladen werden. Die Reiter unterscheiden sich ausschließlich in der Auswahl des zu beschreibenden Objektes. Durch Klick auf *Datei hochladen* kann eine PDF-Datei auf dem lokalen Dateisystem ausgewählt werden. Hochgeladene Dateien werden in einer Tabelle angezeigt.

Spalte	Beschreibung
Datei	Ein Klick auf einen Dateinamen öffnet einen Dialog, in dem der Dateiname angepasst werden kann.
Herunterladen	Dieser Button lädt die Datei herunter.
Löschen	Dieser Button öffnet einen Dialog, in dem bestätigt werden muss, dass die ausgewählte Datei gelöscht werden soll. Es ist zu beachten, dass diese Operation nicht rückgängig gemacht werden kann.

6.10 QR-Code

Zur beschleunigten Ausleihe und Rückgabe kann ein QR-Code für Typen erstellt werden, vergleiche Abbildung 13. Dazu muss zunächst der Typ und die Serverdomäne ausgewählt werden. Der erstellte QR-Code und der Link, auf den der QR-Code verweist, werden eingeblendet. Der QR-Code kann im Anschluss als PDF-, PNG- oder SVG-Datei heruntergeladen werden. Dazu kann zusätzlich die Breite und Höhe der zu erstellenden Datei in Millimeter angegeben werden.

Der ausgedruckte QR-Code kann im Lager platziert werden. Dort muss er zum Beispiel mit einem auf einem mobilen Endgerät installierten, handelsüblichen QR-Code-Reader eingescannt werden, wodurch die Web-Applikation geöffnet wird. und werden - falls bereits angemeldet, siehe Cookies - sofort, ansonsten nachdem sie sich angemeldet haben, auf den Reiter Ausleihen & Zurückgeben weitergeleitet, wo der eingescannte Typ bereits vorausgewählt ist. Sie müssen dann nur noch den Untertyp auswählen.

6.11 Einstellungen

In den Einstellungen können der Benutzername und das Passwort geändert werden, vergleiche Abbildung 14.

6.11.1 Benutzernamen ändern

Um den eigenen Benutzernamen zu ändern, muss ein neuer Benutzername eingegeben werden. Nach Eingabe des Passworts und Bestätigen wird der Benutzername geändert.

6.11.2 Passwort ändern

Um das eigene Passwort zu ändern, muss ein neues Passwort zweimal eingegeben und bestätigt werden.

7 Technische Umsetzung

Die nachfolgenden Abschnitte erweitern die Nutzerhilfe um die Beschreibung der technischen Umsetzung ausgewählter Elemente der Web-Applikation. Die ausgewählten Elemente sollen dabei insbesondere eine Vielzahl des verwendeten Methodenspektrums darlegen, sodass eine nachhaltige Nutzung über die Laufzeit des Projektes hinaus erleichtert wird. Darüber hinaus sollen interessante Ansätze vorgestellt werden.

7.1 Struktur der Web-Applikation

Strukturell handelt es sich bei der Web-Applikation um eine Sammlung verschiedener Dateien innerhalb eines Ordners. Der Einstiegspunkt in die Web-Applikation ist die Datei `app.R`. Alle anderen Dateien werden zur Laufzeit der Web-Applikation eingebunden. Die Web-Applikation wird gestartet, indem der Shiny Server die Datei `app.R` ausführt. In `app.R` werden Initialisierungen durchgeführt und die Funktionen `ui` und `server` definiert. Die Datei endet mit dem Aufruf `shinyApp(ui, server)`, wobei die `ui`-Funktion das visuelle Layout der Web-Applikation beschreibt und die `server`-Funktion Eingaben verarbeitet. Mithilfe von sogenannten Modulen kann der Code strukturiert werden. Jedes Modul setzt sich dabei wiederum aus einer `ui`- und einer `server`-Funktion zusammen. Module bilden Teilaspekte, die unter Umständen wiederverwendet werden können, ab.

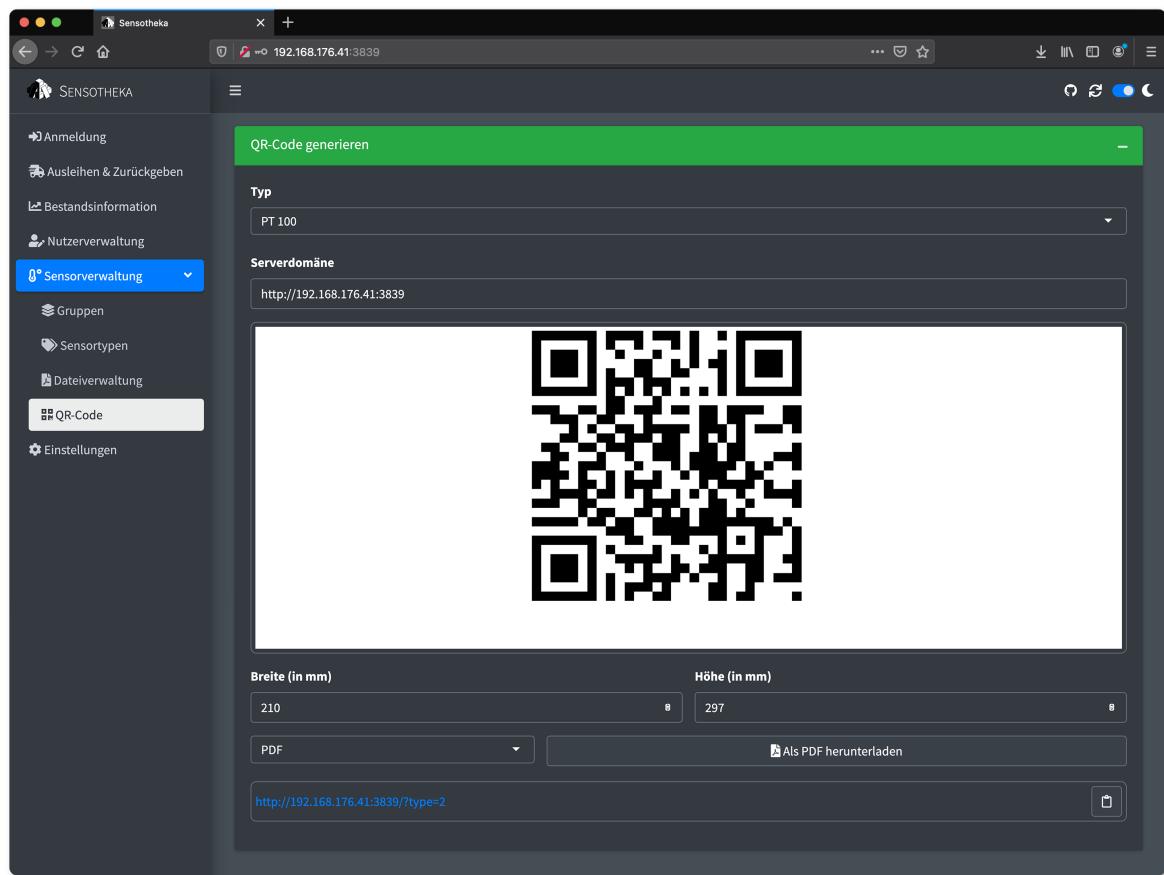


Abbildung 13: Übersicht: QR-Code

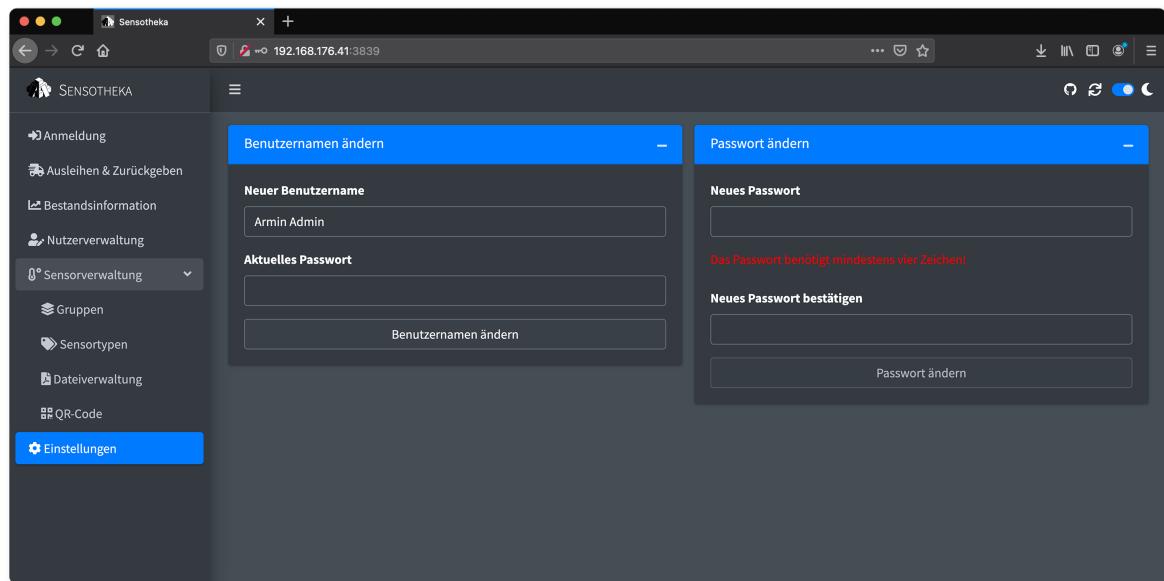


Abbildung 14: Übersicht: Einstellungen

7.1.1 Struktur von app.R

Wie bereits beschrieben, ist die Datei `app.R` der Einstiegspunkt der Web-Applikation. Da sie Einfluss auf das Verhalten vieler weiterer Module hat, ist ihr Aufbau näher zu betrachten. Zur Übersicht werden die Inhalte verkürzt dargestellt.

- Laden benötigter Packages mit `library()`. Die Mehrzahl aller Packages wird über den `::-`-Operator referenziert. Nur die Packages, die zwangsläufig mit `library()` geladen werden müssen, also Packages, die funktionsrelevanten Code mit dem Aufruf von `library()` verbinden, werden hier geladen.

```
library(shiny)
library(shinyjs)
library(dplyr)
library(qrcode)
```

- Die Funktion `source_directory()` wird manuell gesourcet und danach dazu verwendet, alle `.R`-Dateien in den Ordner `modules` und `db/func` zu sourcen.

```
source("init/source_directory.R")

source_directory(
  path = "modules",
  encoding = "UTF-8",
  modifiedOnly = FALSE,
  chdir = TRUE,
  recursive = TRUE,
  envir = if (source_to_globalenv) globalenv() else environment()
)
```

- Die `ui`-Funktion bindet alle CSS- und JavaScript-Dateien ein und aktiviert spezielle Packages. Die Funktion `container_ui()`, die das Layout des Dashboardes festlegt, wird aufgerufen. In `container_ui()` wird für jeden Reiter eine eigene `ui`-Funktion aufgerufen.

```
# UI -----
ui <- htmltools::tagList(
  htmltools::includeScript("www/js/dark-mode.js"),
  htmltools::includeCSS("www/css/styles.css")
  container_ui(
    id = "container"
  ),
  rclipboard::rclipboardSetup(),
  shinyjs::useShinyjs(),
  waiter::use_waiter()
)
```

- In der `server`-Funktion wird die `.values`-Environment angelegt. Diese enthält die folgenden Elemente:

Name	Beschreibung
<code>query\$type</code>	Aus Query-String ausgelesene Typ-ID
<code>settings</code>	Liste, die Namenslängen festlegt und Dictionaries enthält
<code>update</code>	Liste, die <code>reactiveVal()</code> s enthält, die bei Datenaktualisierung getriggert werden
<code>user</code>	Liste, die Informationen zum angemeldeten Nutzer enthält
<code>yaml</code>	Inhalt von <code>app.yml</code>

Diese Environment wird jeder `server`-Funktion übergeben, sodass die Werte in jeder `server`-Funktion

verfügbar sind.

- In der `server`-Funktion von `app.R` wird die `server`-Funktion `container_server()` aufgerufen.

7.1.2 Ordnerstruktur

- `.gitignore` Index aller Dateien, die nicht der Versionskontrolle unterliegen
- `app.R` Einstiegspunkt in die Web-Applikation
- `app.yml` Konfigurationsdatei
- `db`
 - `db.sqlite` Datenbank
 - `func` Funktionssammlung zur Interaktion mit der Datenbank
- `files` Dateiensammlung für Dateiverwaltung. Enthält drei Unterordner für Gruppen, Typen und Untertypen. Jeder dieser Unterordner enthält für jedes Element der jeweiligen Kategorie einen Ordner mit der Identifikationsnummer des jeweiligen Elementes als Namen
- `init/source_directory.R` Hilfsfunktion, die beim Start der Web-Applikation alle `.R`-Dateien einliest
- `modules` Shiny-Module und sonstige Funktionalität
 - `container.R` Definition des Dashboards
 - `dt_options.R` Optionen für `DT::datatable`
 - `object` Generalisierte Module für Gruppen, Typen und Untertypen (z.B. Element hinzufügen, umbenennen etc.)
 - `sidebar_menu.R` Definition der Sidebar
 - `tab_file_management` Dateiverwaltung
 - `tab_group` Gruppen
 - `tab_login` Anmeldung
 - `tab_operate` Ausleihen & Zurückgeben
 - `tab_qrcode` QR-Code
 - `tab_reporting` Bestandsinformation
 - `tab_settings` Einstellungen
 - `tab_type` Typen
 - `tab_user_management` Nutzerverwaltung
 - `utils.R` Hilfsfunktionen
- `renv`, `renv.lock` Package-Management
- `www`
 - `css` CSS-Dateien
 - `favicon.ico` Sensotheka-Icon
 - `js` JavaScript-Dateien

7.2 Datenbank

Die Datenbank ist eine SQLite-Datenbank. Sie liegt in `db/db.sqlite`. Der Zugriff erfolgt über das Package `{DBI}`.

7.2.1 Struktur

Alle Tabellen enthalten die Spalte `rowid`, die den Primärschlüssel der jeweiligen Tabelle enthält. Diese kann somit zur eindeutigen Identifikation einer Zeile verwendet werden. Die Beschränkung bezieht sich immer nur auf die nicht gelöschten Elemente (`removed = 0`).

7.2.1.1 circulation

Spalte	Einschränkungen	Beschreibung
<code>user_id</code>		Identifikationsnummer des ausführenden Nutzers

Spalte	Einschränkungen	Beschreibung
subtype_id		Identifikationsnummer des bearbeiteten Untertyps
quantity		Bearbeitete Menge
time		Zeitpunkt der Transaktion
op_type	1 oder 2	Transaktionsart: 1 - Ausleihen und Zurückgeben, 2 - Bestandsänderungen

7.2.1.2 groups

Spalte	Einschränkungen	Beschreibung
group_name		Gruppenname
removed	0 oder 1	0 - existent, 1 - gelöscht

7.2.1.3 group_type

Spalte	Einschränkungen	Beschreibung
group_id		Identifikationsnummer der Gruppe
type_id		Identifikationsnummer des Typs

7.2.1.4 subtype

Spalte	Einschränkungen	Beschreibung
type_id		Identifikationsnummer des Typs
subtype_name	innerhalb eines Typs	Untertypenname
quantity		Bestandsmenge
removed	0 oder 1	0 - existent, 1 - gelöscht

7.2.1.5 type

Spalte	Einschränkungen	Beschreibung
type_name		Typname
removed	0 oder 1	0 - existent, 1 - gelöscht

7.2.1.6 user

Spalte	Einschränkungen	Beschreibung
hash		Verhaschter Nutzernname (notwendig für Cookies)
name		Nutzernname
status	admin, mod oder user	Nutzerrolle
password		Verhashtes Passwort
added_from		Nutzeridentifikationsnummer des Hinzufügenden
time_added		Zeitpunkt der Kontoeröffnung
time_current_logged		Zeitpunkt der letzten Anmeldung
time_previous_logged		Zeitpunkt der vorletzten Anmeldung
times_logged		Anzahl der Anmeldungen

Spalte	Einschränkungen	Beschreibung
removed	0 oder 1	0 - existent, 1 - gelöscht

7.2.2 Zugriff

```
library(DBI)
library(RSQLite)

# Verbinde die Datenbank mit R
db <- dbConnect(SQLite(), "db/db.sqlite")

# Tabellennamen
dbListTables(db)

## [1] "circulation" "group_type"   "groups"      "subtype"     "type"       "user"
```

Beispielhaft wird die Tabelle “circulation” betrachtet. Diese Tabelle speichert alle Transaktionen, die von Nutzern durchgeführt worden sind, also zum Beispiel Ausleihen, Rückgaben und Mengenänderungen.

```
dbReadTable(db, "circulation")
```

Der Ordner db/func enthält eine Sammlung von über 80 Hilfsfunktionen, die zum Erstellen, Abfragen und Modifizieren der Datenbank verwendet werden. Die Interaktion mit der Datenbank erfolgt ausschließlich über diese Hilfsfunktionen. Dadurch werden Redundanzen ausgeschlossen und es ist darüber hinaus einfacher, die korrekte Funktionalität zu gewährleisten.

Beispielhaft wird die Funktion db_get_borrowed_quantity(db, subtype_id) betrachtet. Unter Angabe der Datenbank db und einer oder mehrerer Untertypenidentifikationsnummern subtype_id wird die ausgeliehene Menge dieser Untertypen zurückgegeben. In {DBI} werden Abfragen an die Datenbank mithilfe von dbGetQuery() gestellt. Das Resultat der Abfrage wird anschließend weiterverarbeitet. Der Nutzen der Hilfsfunktion wird zusätzlich dadurch verdeutlicht, dass sowohl die abzufragende Tabelle sowie die beteiligten Zeilen und Spalten nicht gesondert angegeben werden müssen.

```
# Funktionsdefinition
db_get_borrowed_quantity <- function(db, subtype_id) {
  borrowed <- DBI::dbGetQuery(
    db,
    "SELECT SUM(quantity) AS borrowed FROM circulation
     WHERE subtype_id = ? AND op_type = 1",
    params = list(subtype_id)
  )$borrowed

  ifelse(is.na(borrowed), 0, borrowed)
}

# Abfrage der ausgeliehenen Menge für mehrere Untertypen
db_get_borrowed_quantity(db, 1:5)

## [1] 0 13 0 0 2
```

7.3 Cookies

Cookies sind Textdaten, die der Client mit jeder Anfrage an den Server mitsendet. Die Web-Applikation setzt zwei Cookies. Die Cookies sind jeweils für einen Zeitraum von sieben Tagen gültig.

7.3.1 dark-mode

Dieses Cookie kann die Werte "true" oder "false" enthalten. Das Cookie wird gesetzt, wenn der Nutzer die Toggle für den Nachtmodus verwendet. Beim Start der Web-Applikation wird das Cookie verwendet, um den initialen Status für den Nachtmodus zu bestimmen. Falls das Cookie nicht vorhanden ist, wird die Einstellung des Browsers (`window.matchMedia('prefers-color-scheme: dark')`) ausgelesen und im Cookie gesetzt.

7.3.2 user

Dieses Cookie enthält einen verhaschten Nutzernamen und dient zur Identifikation eines angemeldeten Nutzers. Das Cookie wird gesetzt, wenn sich ein Nutzer anmeldet. Beim Start der Web-Applikation wird das Cookie verwendet, um den Nutzer automatisch anzumelden. Falls das Cookie nicht vorhanden ist, muss der Nutzer sich manuell anmelden.

7.3.3 Implementierung

Zur Verwaltung der Cookies wird die Bibliothek js-cookie verwendet. Die Datei `www/js/cookies.js` enthält Hilfsfunktionen zum Lesen und Schreiben von Cookies und verbindet diese mit Inputwerten in Shiny. Die Hilfsfunktionen werden in R mithilfe von `shinyjs::extendShinyjs()` eingebunden.

7.4 Deployment

8 Diskussion

9 Ausblick