



UNIVERSITÄT
LEIPZIG

Wintersemester 2018/19
Softwaretechnikpraktikum

App zur Inventarisierung von Unternehmenswerten

Entwurfsbeschreibung

Gruppe: ak18b

Betreuer: Benjamin Lucas Friedland, Michael Fritz

Gruppenmitglieder: Alexander Zwisler, Leon Kamuf, Leon Rudolph, Maurice Eisenblätter,
Maximilian Gläfcke, Robin Seidel, Sina Opitz, Steve Woywod

Inhaltsverzeichnis

1	Visionen und Ziele	1
2	Rahmenbedingungen und Produktübersicht	1
2.1	Back-End	1
2.2	Front-End	1
3	Grundsätzliche Struktur- und Entwurfsprinzipien	1
3.1	Programmiersprache	1
3.2	Vorgehensweise	1
3.3	Back-End	2
3.4	Front-End	2
4	Struktur- und Entwurfprinzipien einzelner Pakete	3
4.1	server	4
4.2	client	4
5	Datenmodell	5
6	Glossar	5

1 Visionen und Ziele

Unser Ziel ist es, eine frei konfigurierbare Web-App für Google Chrome zur vollständigen Inventarisierung zu entwickeln und optimieren. Um eine optimierte Übersicht zu gewährleisten, kann das gesamte Inventar in Itemtypen gegliedert werden, die unterschiedliche Eigenschaften besitzen. Innerhalb der Itemtypen existiert auch eine vollständige Anzeige der einzelnen Items. Für alle Itemtypen können Pflichtfelder, 'Unique'-Felder und sonstige erstellt werden, um eine hohe Konfigurierbarkeit zu gewährleisten. Außerdem kann eingestellt werden, welche Felder in der Item-Auflistung und welche nur bei der Detailansicht angezeigt werden. Weiterhin gibt es eine Sortier- sowie Suchfunktion, um eine optimale Übersicht und schnelle Bedienung zu ermöglichen.

Damit auch alle Daten sicher bleiben, gibt es ein Account-System mit gesicherter Login- und Logout-Funktion. Der Admin kann außerdem Benutzergruppen erstellen und beliebig konfigurieren, um Rechte zu vergeben. So können nur bestimmte Nutzer beispielsweise Items bearbeiten oder löschen.

Auch die Mehrsprachigkeit (Deutsch und Englisch) ist ein Ziel.

2 Rahmenbedingungen und Produktübersicht

Das Projekt basiert auf einem Server-Client-Grundkonzept mit einem Back-End und Front-End. Zur Speicherung der Daten wird eine MariaDB und als Schnittstelle zwischen Client und Server eine REST-API verwendet.

2.1 Back-End

Das Back-End bildet die Schnittstelle zwischen der Datenbank (MariaDB) und dem Client. Es sorgt im Wesentlichen dafür, dass die vom Client übertragenen Daten vollständig und korrekt sind sowie für das Erstellen, Manipulieren und Löschen der dynamischen Datensätze (Items). Der Server selbst versteht nur HTTP-Anfragen vom Content-Type 'applications/json' und antwortet auch in diesem.

2.2 Front-End

Das Front-End wird komplett aus Angular entwickelt und dient als Benutzeroberfläche, auf der sich alle Funktionen der Web-App übersichtlich finden und genutzt werden. Es existiert außerdem ein Kommunikationskanal zwischen Back-End und Front-End.

3 Grundsätzliche Struktur- und Entwurfsprinzipien

3.1 Programmiersprache

Das gesamte Projekt wird in TypeScript geschrieben. Das Front-End hat zudem Style-Elemente, die mithilfe von CSS festgehalten werden.

3.2 Vorgehensweise

Die Entwicklung findet zuallererst im Back-End statt, wo die Funktionalitäten implementiert und optimiert werden, bevor dazu ein passendes Front-End entwickelt wird. Anschließend wird beides über den Kommunikationskanal miteinander verbunden, getestet und ggf. angepasst. Der Style des Userinterfaces wird dann zum Schluss erstellt.

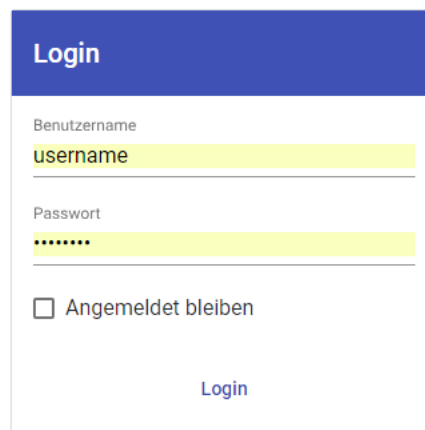
3.3 Back-End

Alle Eingaben, die der Nutzer im Front-End tätigt, werden über den Kommunikationskanal ins Back-End übergeben, vom Server auf Korrektheit überprüft und anschließend mithilfe der REST-API in der Datenbank gespeichert. Des Weiteren existiert eine Serverstruktur, welche Daten (bei Sortier- und Suchanfragen) aus der Datenbank ausliest und an die anfragenden Clients weiterleitet.

Auch die Datensicherung und die damit verbundene Rechteverwaltung wird im Back-End definiert und somit die Anzeige der Elemente im Front-End gesteuert.

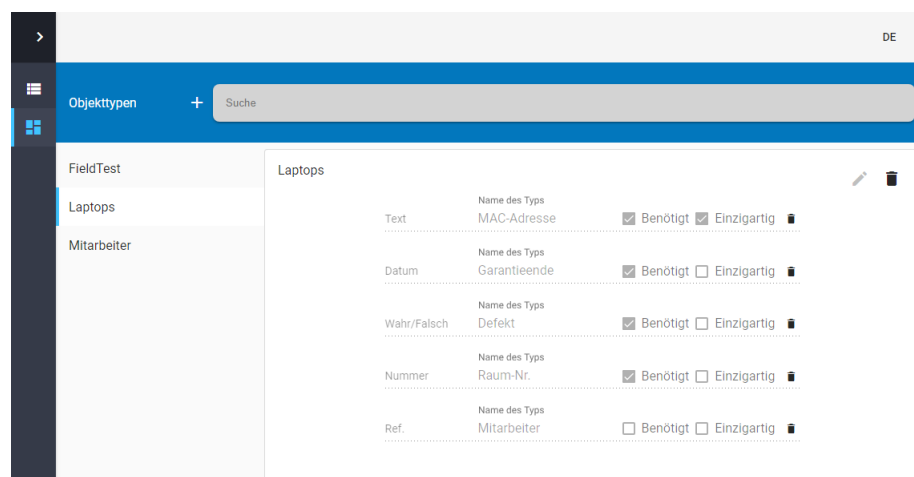
3.4 Front-End

Die im Back-End definierte Rechteverwaltung steuert die Anzeige im Front-End. Während Benutzer, die alle Rechte besitzen, Buttons zum Ändern und Löschen angezeigt bekommen, fehlen diese bei anderen Benutzern. Gästen wird dabei nur die Startseite und die Login-Seite, angezeigt. Auf der Login-Seite existiert ein kleines Fenster, wo Username und Passwort abgefragt werden. Die Eingabe im Passwort-Feld wird zum Schutz ausgeblendet.



The image shows a login form with a blue header bar containing the word "Login". Below the header, there are two input fields: "Benutzername" (Username) with the text "username" entered, and "Passwort" (Password) with masked characters "*****". Below the password field is a checkbox labeled "Angemeldet bleiben" (Stay logged in). At the bottom of the form is a blue button labeled "Login".

Allen eingeloggtten Benutzern wird eine Übersicht der Itemtypen und einzelnen Items angezeigt. Dort können auch neue Itemtypen erstellt und bereits vorhandene Itemtypen bearbeitet werden. Bei den Einstellungen der Itemtypen werden die Eigenschaften der Items festgehalten.



The image shows a screenshot of a web application interface. On the left is a dark sidebar with a menu containing "Objekttypen" (Object Types) and a search bar. The main content area has a blue header with "Objekttypen" and a search bar. Below the header, there is a list of object types: "FieldTest", "Laptops", and "Mitarbeiter". The "Laptops" type is selected, and its settings are displayed in a table. The table has columns for "Name des Typs" (Type Name), "Benötigt" (Required), and "Einzigartig" (Unique). The rows are:

Name des Typs	Benötigt	Einzigartig
MAC-Adresse	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Garantieende	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Defekt	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Raum-Nr.	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Mitarbeiter	<input type="checkbox"/>	<input type="checkbox"/>

Übersicht Itemtypen mit Einstellungen

>

Übersicht Items eines Typs

Innerhalb der Anzeige der einzelnen Items, eines Typs, können die bereits vorhandenen Items geändert und gelöscht oder neue Items hinzugefügt werden.

Item Typ	Suche			
Laptops				
Mitarbeiter				
MAC-Adresse	Garantieende	Defekt	Raum-Nr.	Mitarbeiter
abc	18.1.2019	Falsch	120	∞ 1
def	10.1.2019	Falsch	1	∞ 1
ghi	19.1.2019	Wahr	3	∞ null
Items per page: 50 1 - 3 of 3 < >				

Item Typ

Laptops

Text

MAC-Adresse *

Datum

Garantieende *

Defekt *

Wahr/Falsch

Nein

Nummer

Raum-Nr. *

∞ Ref.

Mitarbeiter

Speichern

Am Rand kann über ein Button auf das Menü zugegriffen werden, wo man zu den Einstellungen gelangt, die Sprache auswählen und sich ausloggen kann. Dem Admin wird außerdem ein Reiter für die Rechteverwaltung angezeigt, wo er einzelne Benutzergruppen erstellen und konfigurieren kann.

Je nach Benutzergruppe werden Buttons und die damit verbundenen Funktionen ausgeblendet. So sehen nicht alle den Button, um Items zu löschen oder Itemtypen zu erstellen.

4 Struktur- und Entwurfprinzipien einzelner Pakete

Das Projekt besitzt zwei Pakete, den `client` und den `server`. Innerhalb des Projekt-Pakets, befinden sich außerdem alle Packages, die benötigt werden und die REST-API als Schnittstelle zwischen dem Client und Server. In jedem Paket befinden sich die Auflistung der benutzten Frameworks und Tools, ein Text-Paket mit Testdaten und die `src`-Pakete, wo die Funktionalitäten festgehalten werden.

4.1 server

Im `server`-Paket wird der HTTP-Sever initialisiert, die einzelnen Seiten (ggf. mit Sichtbarkeitseinschränkungen) geladen und die Verbindung zur Datenbank hergestellt.

`/types`

Hier werden für TypeScript fehlende Typen wie zum Beispiel MariaDB erfasst und niedergeschrieben.

`/src/app.ts`

Ist der Core des Backends und initialisiert die Datenbank und den Express-Server

`/src/index.ts`

Hier wird die Config eingelesen und der HTTP(S)-Server sowie die App(`/src/app.ts`) gestartet.

`/src/api`

Hier werden alle Endpunkte der API definiert sowie der Funktion implementiert.

`/src/database`

Hier werden alle Datenbank-Abfragen definiert und implementiert.
Später sollen auch Modellklassen für Datenbank hier rein kommen.

4.2 client

`/src/assets`

Hier werden die Sprachen für den Sprachregler definiert.

`/src/app`

Hier befinden sich die einzelnen Komponenten, aus welchen sich die Web-App zusammensetzt.

`/src/app/confirm-dialog`

Hier wird sichergestellt, dass alle Änderungen, die vom User vorgenommen werden können, überprüft werden.

`/src/app/items`

Hier werden alle Komponenten der Seite, die mit den Items zu tun haben und die dazugehörigen Funktionen definiert und initialisiert. Dies beinhaltet das Hinzufügen, Ändern und Löschen von Items, die Elemente eines Itemtyps, das Anzeigen von Items und Itemtypen, und die Suchleiste. Auch werden hier die Datentypen der einzelnen Elemente und das Interface gesetzt.

`/src/app/shell`

Hier wird das grobe Userinterface definiert. Dazu gehören die Authentifizierung, Sprachselektor, Navigation und das Login-Fenster.

`/src/util`

was passiert hier?

/src/environments

wichtig?

/e2e

Hier befindet sich der Test-Client.

5 Datenmodell

Die Datenbank muss sehr dynamisch sein, da die einzelnen Daten, die gespeichert werden sollen, frei konfigurierbar sind.

Bei den Itemtypen werden die einzelnen Eigenschaften, die jedes Item des Typs besitzen soll, definiert. Es muss dabei ein **key** gesetzt werden, um den Datensatz eindeutig zu bestimmen. Diesen Key setzt der Benutzer. Außerdem legt der Benutzer selbst fest, welche Felder Pflichtfelder sind. Die Items innerhalb des Typs werden anhand der Einstellungen des Itemtyps festgestellt. Bei jedem Hinzufügen und Ändern eines Items wird darauf geachtet, dass ein eindeutiger Key gesetzt und alle Pflichtfelder erfüllt wurden. Erst dann speichert der Server die eingegeben Daten in die Datenbank.

6 Glossar

Web-App

Eine Webanwendung (auch Webapplikation, kurz Web-App) ist eine Anwendung nach dem Server-Client-Prinzip. Der große Unterschied zu den klassischen Desktop-Apps besteht darin, dass die Anwendung nicht lokal auf dem Computer, sondern auf einem Webserver online ausgeführt wird.

Server-Client-Prinzip

Das Client-Server-Prinzip beschreibt eine Möglichkeit, innerhalb eines Netzwerks zu kommunizieren. Der sogenannte Client kann dabei Dienste und Daten vom Server abfragen, die dann vom Server bearbeitet und an den Client zurück übermittelt werden.

Back-End / Front-End

Viele Vorgänge und Anwendungen lassen sich in Front- und Back-End unterteilen. Front-End ist hierbei das Userinterface, welches der Benutzer sieht.

Im Back-End hingegen befinden sich die Prozesse, die im Hintergrund ausgeführt werden und die Benutzung der Anwendung ermöglicht.

In unserem Fall ist der Server das Back-End und der Client das Front-End.

MariaDB

MariaDB ist eine freies Datenbankmanagementsystem, welches durch eine Abspaltung aus MySQL entstanden ist.

REST-API

Die **R**epresentational **S**tate **T**ransfer des **A**pplication **P**rogramming **I**nterfaces ist eine Programmierschnittstelle zwischen Server und Client.

TypeScript

TypeScript ist eine Scriptsprache, die von jedem Browser und Betriebssystem unterstützt wird. Sie ist eine Erweiterung von Javascript.