



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 Felder, darunter Pflichtfelder, 'Unique'-Felder 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 in Angular entwickelt und dient als Benutzeroberfläche, auf der sich alle Funktionen der Web-App übersichtlich finden und genutzt werden. Es existiert außerdem eine REST-API zwischen Back-End und Front-End.

3 Grundsätzliche Struktur- und Entwurfsprinzipien

3.1 Programmiersprache

Das gesamte Projekt wird in **TypeScript** geschrieben. Das Front-End benutzt für den Style von **HTML CSS**, welches durch den **sass**-Precompiler generiert wird.

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 die REST-API miteinander verbunden, getestet und ggf. angepasst. Der Style des Userinterfaces wird dann zum Schluss erstellt.

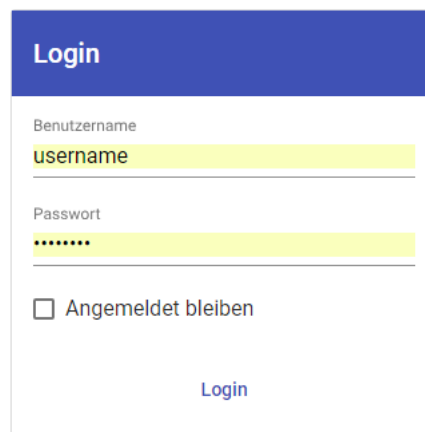
3.3 Back-End

Alle Eingaben, die der Nutzer im Front-End tätigt, werden über die REST-API ins Back-End übergeben, vom Server auf Korrektheit überprüft und anschließend 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.

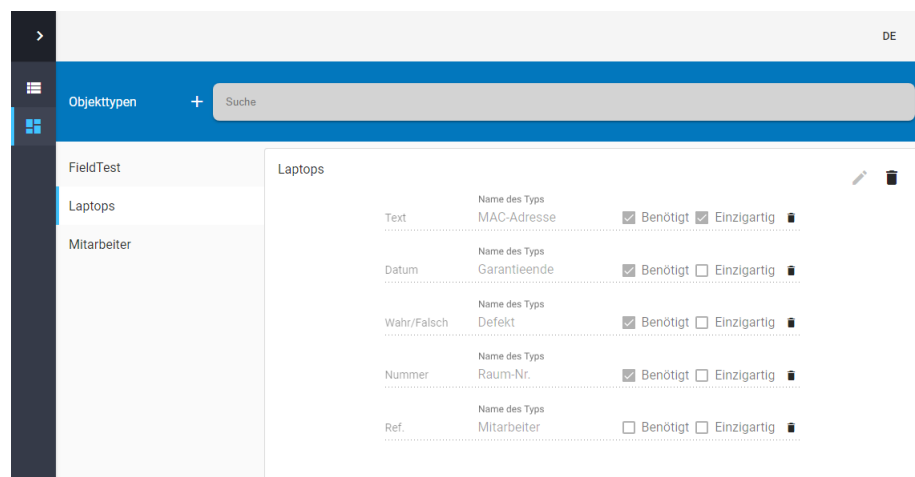
3.4 Front-End

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 (noch nicht implementiert). Gästen wird dabei nur die Startseite und die Login-Seite angezeigt. Auf der Login-Seite existiert ein 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 containing the word 'Login'. Below the header, there are two input fields: 'Benutzername' with the text 'username' and 'Passwort' with masked characters '.....'. Below these fields is a checkbox labeled 'Angemeldet bleiben'. At the bottom of the form is a blue button labeled 'Login'.

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

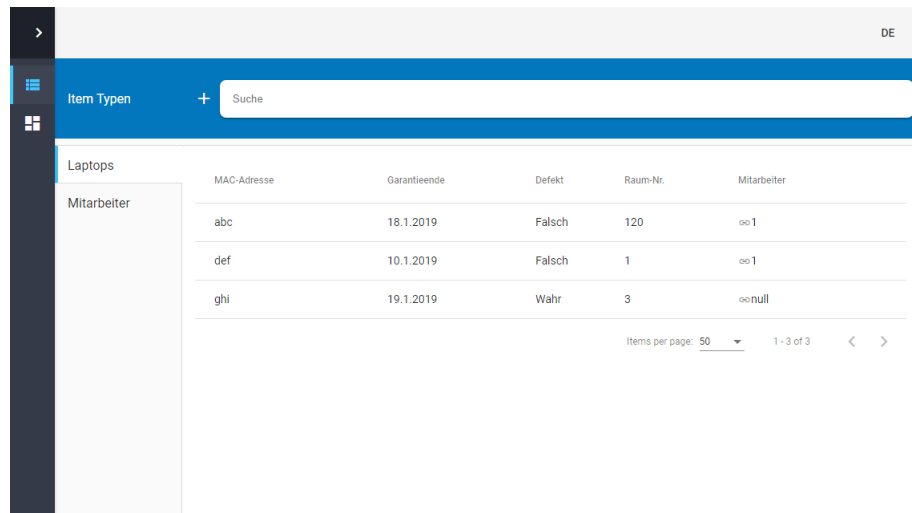


The image shows a web interface for managing object types. On the left is a sidebar with a menu containing 'Objekttypen', 'FieldTest', 'Laptops', and 'Mitarbeiter'. The main area is titled 'Laptops' and contains a table of settings. The table has columns for the field name, the type name, and checkboxes for 'Benötigt' and 'Einzigartig'. The settings are as follows:

Text	Name des Typs	Benötigt	Einzigartig
MAC-Adresse	MAC-Adresse	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Datum	Garantieende	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Wahr/Falsch	Defekt	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Nummer	Raum-Nr.	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Ref.	Mitarbeiter	<input type="checkbox"/>	<input type="checkbox"/>

Übersicht Objekttypen mit Einstellungen

Durch einen Button links im Menü gelangt man dann zu der Übersicht der einzelnen Items innerhalb eines Objekttyps. Über einen Tab links kann man zwischen den Objekttypen switchen.

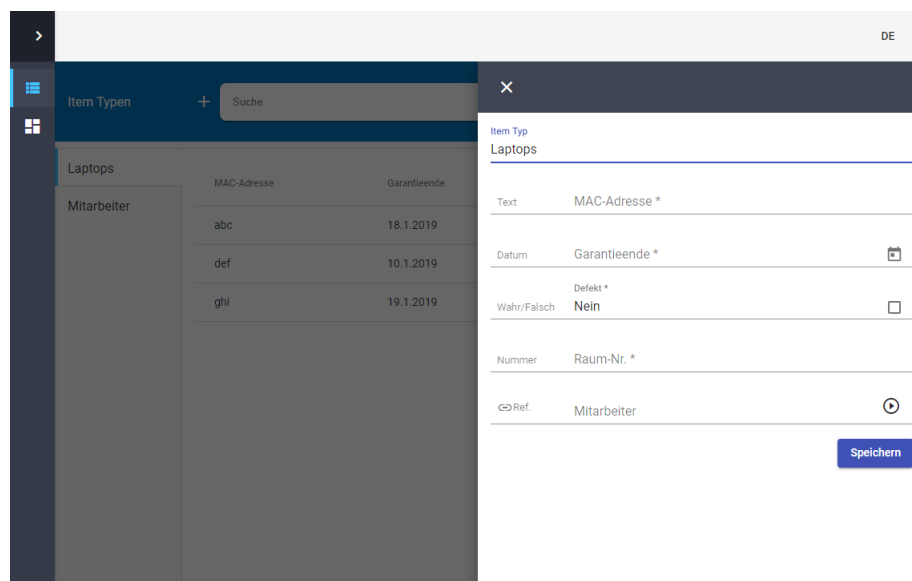


MAC-Adresse	Garantieende	Defekt	Raum-Nr.	Mitarbeiter
abc	18.1.2019	Falsch	120	∞1
def	10.1.2019	Falsch	1	∞1
ghi	19.1.2019	Wahr	3	∞null

Items per page: 50 1 - 3 of 3 < >

Übersicht Items eines Typs

Dort können neue Items hinzugefügt, bearbeitet und gelöscht werden.



Item Typ: Laptops

Text: MAC-Adresse *

Datum: Garantieende *

Defekt *: Wahr/Falsch ☒ Nein

Nummer: Raum-Nr. *

Ref.: Mitarbeiter

Speichern

Außerdem gelangt man über den Button links oben zum Sprachwechsler und zum Logout. 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-Server 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 Back-Ends 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, außerdem gibt es Modellklassen für die Datenbank.

`/src/types.ts`

War schon beim letzten Release dabei aber nicht aufgelistet. Unwichtig???

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/company`

Hier befinden sich alle nötigen Komponenten und Funktion, um die Unternehmensverwaltung zu realisieren.

`/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 unter anderem das Anzeigen von Items, im Bezug auf dessen Typen, und die Suchleiste.

`/src/app/models`

Hier werden alle benötigten Interfaces erzeugt, welche dann in Komponente importiert werden können, sollten sie gebraucht werden.

`/src/app/roles`

Unwichtig, da noch nicht implementiert?

`/src/app/shared`

Unwichtig, da noch nicht implementiert?

`/src/app/types`

Hier werden alle Komponenten der Seite, welche mit Itemtypen zu tun haben, und die dazugehörigen Funktionen definiert und initialisiert. Dazugehörig ist, zum Beispiel 'Globale Pflichtfelder' und das Anzeigen von Itemtypen.

`/src/app/user`

Hier werden alle Komponenten, zur Darstellung von Benutzern, und die entsprechenden Funktionen definiert und initialisiert.

`/src/app/shell`

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

`/src/environments`

Hier befinden sich die Einstellungen für die verschiedenen Umgebungen, in denen die Applikation laufen soll.

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.