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

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 wird ein eine Sortier- sowie Suchfunktion geben, um eine optimale Übersicht und schnelle Bedienung zu ermöglichen.
Damit auch alle Daten sicher bleiben, gibt es ein Account-System mit gesicherter Login- und Logour-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 besteht aus einer MariaDB-Datenbank mit den dazugehörigen Funktionen. Außerdem werden alle Funktionalitäten der App in `.json`-Datein und auch das Server-Client-Konzept festgehalten.

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, die Login-Seite, angezeigt. Auf der Login-Seite existiert ein kleines Fenster, wo E-Mail-Adresse und Passwort abgefragt werden. Die Eingabe im Passwort-Feld wird zum Schutz ausgeblendet.
Allen 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 zu den Itemtypen werden die Eigenschaften der, innerhalb des Typs, Items festgehalten. Innerhalb der Anzeige der einzelnen Items innerhalb eines Typs können die bereits vorhandene Items geändert und gelöscht oder neue Items hinzugefügt werden.
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 befindet 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.

`app.js`

Hier wird eine Verbindung zur Datenbank hergestellt und auch wieder geschlossen.

`index.js`

Hier wird ein Port zum Client erstellt, ein HTTP-Server initialisiert.

`/routes index.js`

Hier wird die Homepage gleaden und eine Anfrage an die Datenbank gesendet.

`/routes users.js`

Hier wird die Übersicht der Items geladen.

4.2 client

`angular.json`

Hier wird die Benutzeroberfläche samt Daten und Style geladen

`tsconfig.json`

wichtig?

`tslint.json`

wichtig?

`/e2e`

Hier befindet sich er Text-Client mit Test-Einstellungen

`/src`

Hier befinden sich das Icon, die Startseite als `.html`, der Style als `.css`, die Testdaten des Clients sowie die `config`-Datein.

`/src/app`

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

`/src/assets`

wichtig?

`/src/environments`

wichtig?

5 Datenmodell

Die Datenbank müssen 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 sollen, 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

6.1