

Datenbank Verwaltungstool

DIPLOMARBEIT

verfasst im Rahmen der

Reife- und Diplomprüfung

an der

Höheren Abteilung für Medientechnik

Eingereicht von:

David Altenhofer

Sami Abbas Ali

David Precup

Betreuer:

Michael Palitsch-Infanger

Projektpartner:

Sami Abbas Ali, David Precup, Doka GmbH

Leonding, Juli 2022

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2022

David Altenhofer & David Precup

Abstract

The Doka company offers software packages to their customers. Here is the administration still very much in need of improvement. The process of adding new software packages, works as follows: creating new packages, editing and deleting takes place directly via the database, which means that there is no administration program to date. Since everything runs through the database, it becomes clean work prevented. That's why we were commissioned by the Doka company to create a Database Management System based on the JavaScript library React. Before some terms need to be explained:

It should be possible to manage the following tables that have these properties:

- InstallablePackage is the main table, consists of several Installables and can have a description (InstallablePackagesDescriptions) in multiple languages.
- An Installable has an InstallableSyncTemplate and can have an InstallableExecutablePaths.

The program should have the following functions:

- Viewing, installing, adding, editing and deleting so-called Installables / InstallablePackages / InstallableSyncTemplates / InstallableExecutablePaths
- Exporting the current configuration
- Importing a JSON file, where to see the difference of the imported file and the current configuration
- Reset the cache in the backend



Abbildung 1: ERD

Zusammenfassung

Die Firma Doka bietet Softwarepakete an deren Kunden an. Dabei ist das Verwalten noch sehr verbesserungswürdig. Der Prozess des hinzufügens von neuen Softwarepaketen, läuft wie folgt ab: Das Erstellen von neuen Paketen, das Editieren sowie das Löschen erfolgt direkt über die Datenbank, das bedeutet, dass es kein Verwaltungsprogramm bis dato gibt. Dadurch, dass alles über die Datenbank abläuft, wird das saubere Arbeiten verhindert. Deshalb wurden wir von der Firma Doka dazu beauftragt, ein Verwaltungssystem basierend auf der JavaScript library React zu programmieren. Davor müssen einige Begriffe noch erklärt werden:

Folgende Tabellen sollen verwaltet werden können, die diese Eigenschaften haben:

- InstallablePackage ist die wichtigste Tabelle, besteht aus mehreren Installables und kann Beschreibung (InstallablePackagesDescriptions) auf mehreren Sprachen haben.
- Ein Installable hat ein InstallableSyncTemplate und kann ein InstallableExecutablePaths.

Dabei soll das Programm folgende Funktionen haben:

- Das Anzeigen, Installieren, Hinzufügen, Bearbeiten und Löschen von sogenannten Installables / InstallablePackages / InstallableSyncTemplates / InstallableExecutablePaths
- Das Exportieren der aktuellen Konfiguration
- Importieren eines JSON Dateis, wo die Differenz der importierten Datei und der aktuellen Konfiguration zu sehen ist
- Reseten des Cache im Backend

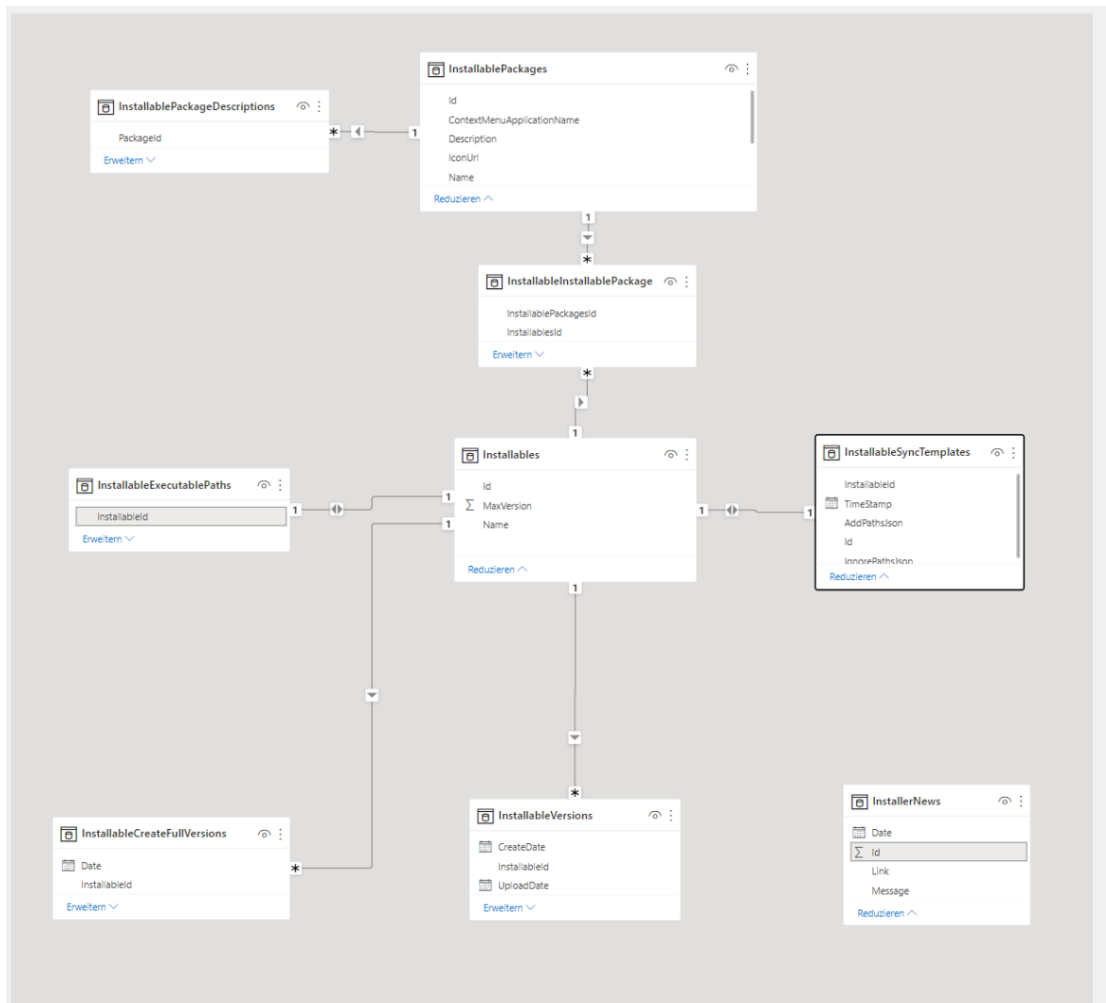


Abbildung 2: ERD

Autoren der Diplomarbeit

David Altenhofer

Aufgabenbereich Frontend und Authentifizierungslogik

Name: David Altenhofer

Geburtsdatum: 8. Juli 2003

E-Mail: david.altenhofer@gmail.com

Bildungsweg

2007 bis 2011 Volksschule Doppl

2011 bis 2015 NMS Hart

seit 2015 HTL Leonding, Informatik

Berufliche Erfahrung:

Sommer 2016 Colour & Point, Softwareentwickler

Sommer 2018 Runtastic, Softwareentwickler

Sprachliche Kenntnisse:

Deutsch Muttersprache

Englisch Fließend

Danksagung

An dieser Stelle möchten wir uns bei denen bedanken, die uns bei der Durchführung dieses Projektes beraten und unterstützt haben. Besonderer Dank, gilt Alex und Mathias, die als Mitarbeiter unserer Partnerfirma Doka, uns im Laufe des Projekts fachlich geholfen haben, bei der Planung als auch bei der Durchführung und Lösung von Hindernissen. Ein weiteres Dankeschön an die Firma Doka, die uns unser Diplomarbeitsprojekt im Rahmen eines Praktikums im Sommer zur Verfügung gestellt haben.

Zu guter letzt möchten wir uns noch bei unserem Betreuer Herrn Prof Palitsch-Infanger, für die Beratung und Begleitung dieser Diplomarbeit, bedanken.

Inhaltsverzeichnis

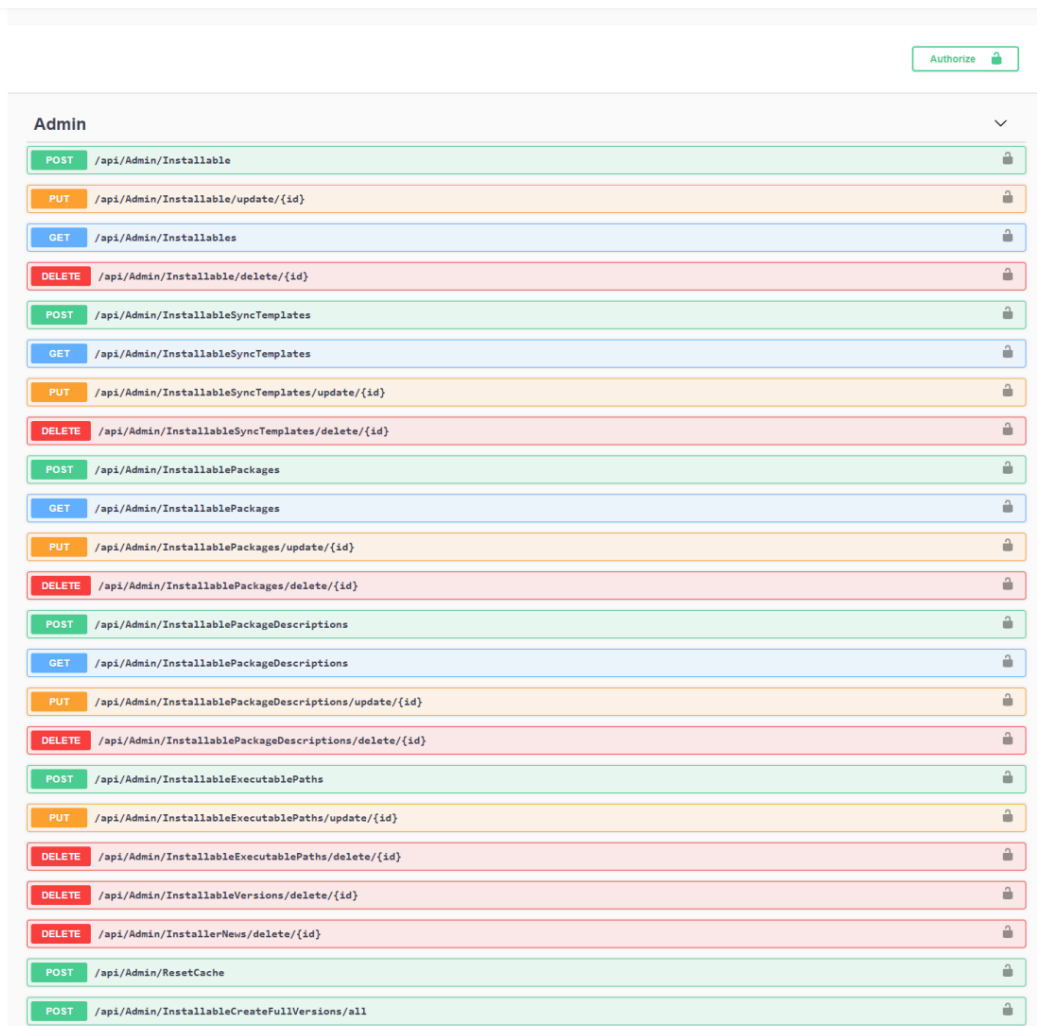
1	Einleitung	1
1.1	Ausgangssituation	1
1.2	Aufgabenstellung	4
1.3	Zielsetzung	4
2	Umfeldanalyse	5
3	Technologien	6
3.1	React	6
3.2	Tailwind CSS	8
3.3	Dev Ops	8
3.4	Adobe XD	8
4	Umsetzung	9
4.1	Einteilung & Design	9
5	Zusammenfassung	11
	Literaturverzeichnis	IX
	Abbildungsverzeichnis	X
	Tabellenverzeichnis	XI
	Quellcodeverzeichnis	XII
	Anhang	XIII

1 Einleitung

1.1 Ausgangssituation

Seit Jahren müssen die Mitarbeiter der Firma die Datenbank Einträge sehr umständlich ändern, da kein User Interface dafür vorhanden ist. Unser Ziel war es, genau das zu Ändern. Ich habe gemeinsam mit meinem Team ein Web-Frontend erstellt, welches das Hinzufügen, Bearbeiten und Löschen der Einträge in der Datenbank um ein vielfaches vereinfacht. Nun kann auf das unpraktische Verwenden von Swagger verzichtet werden und auf ein paar wenige Klicks reduziert werden.

Gegeben war bereits ein fertiges Backend mit einer API



Admin		
POST	/api/Admin/Installable	🔒
PUT	/api/Admin/Installable/update/{id}	🔒
GET	/api/Admin/Installables	🔒
DELETE	/api/Admin/Installable/delete/{id}	🔒
POST	/api/Admin/InstallableSyncTemplates	🔒
GET	/api/Admin/InstallableSyncTemplates	🔒
PUT	/api/Admin/InstallableSyncTemplates/update/{id}	🔒
DELETE	/api/Admin/InstallableSyncTemplates/delete/{id}	🔒
POST	/api/Admin/InstallablePackages	🔒
GET	/api/Admin/InstallablePackages	🔒
PUT	/api/Admin/InstallablePackages/update/{id}	🔒
DELETE	/api/Admin/InstallablePackages/delete/{id}	🔒
POST	/api/Admin/InstallablePackageDescriptions	🔒
GET	/api/Admin/InstallablePackageDescriptions	🔒
PUT	/api/Admin/InstallablePackageDescriptions/update/{id}	🔒
DELETE	/api/Admin/InstallablePackageDescriptions/delete/{id}	🔒
POST	/api/Admin/InstallableExecutablePaths	🔒
PUT	/api/Admin/InstallableExecutablePaths/update/{id}	🔒
DELETE	/api/Admin/InstallableExecutablePaths/delete/{id}	🔒
DELETE	/api/Admin/InstallableVersions/delete/{id}	🔒
DELETE	/api/Admin/InstallerNews/delete/{id}	🔒
POST	/api/Admin/ResetCache	🔒
POST	/api/Admin/InstallableCreateFullVersions/all	🔒

Abbildung 3: Swagger UI - Endpoints

POST	/api/Admin/InstallableCreateFullVersions/{installableId}	🔒
GET	/api/Admin/InstallableCreateFullVersions	🔒
InstallableQuery ▾		
GET	/api/InstallableQuery/InstallableVersions/GetVersionInfos/{culture}	🔒
GET	/api/InstallableQuery/InstallableVersions/{id}	🔒
GET	/api/InstallableQuery/InstallableExecutablePaths	🔒
GET	/api/InstallableQuery/InstallerNews	🔒
InstallableUpload ▾		
POST	/api/InstallableUpload/RequestUpload	🔒
POST	/api/InstallableUpload/NotifyUpload	🔒
GET	/api/InstallableUpload/InstallableSyncTemplates	🔒
GET	/api/InstallableUpload/GetInstallableVersionUploadSettingQuery/{installableId}	🔒
POST	/api/InstallableUpload/InstallerNews	🔒
Schemas ▾		
InstallableDto >		
InstallablePackageDescriptionDto >		
InstallablePackageDto >		
InstallableSyncTemplateDto >		
InstallableVersionDto >		
InstallableVersionInfo >		
InstallableVersionPackageDto >		
InstallableVersionPackageInfo >		
InstallerNewsDto >		
SyncPathDto >		

Abbildung 4: Swagger UI - Endpoints

1.2 Aufgabenstellung

Um den Umgang mit der Datenbank für alle Mitarbeiter zu vereinfachen, soll ein leicht verständliches, intuitives Interface entwickelt werden. Die Web-Anwendung darf nur für autorisierte Benutzer in der Abteilung der Firma verfügbar sein. Deswegen ist eine Authentifizierungslogik zu implementieren. Erst nach erfolgreichen Anmelden mit einem gültigen Doka-Account, soll die Website erreichbar sein und Änderungen in der Datenbank durchgeführt werden können.

1.3 Zielsetzung

Die Web-Anwendung soll in einem ansprechenden Design entwickelt werden und ein leicht verständlicher unkomplizierter Umgang muss auch für Nicht-Fachleute gegeben sein.

2 Umfeldanalyse

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Citing [1] properly.

Was ist eine Globally Unique Identifier (GUID)? Eine GUID kollidiert nicht gerne.

Kabellose Technologien sind in abgelegenen Gebieten wichtig [2].

3 Technologien

3.1 React

React ist eine **JavaScript-Programmbibliothek** zur Erstellung von webbasierten Benutzeroberflächen. Jede React-Webanwendung besteht aus wiederverwendbaren Komponenten, die Teile der Benutzeroberfläche bilden – wir können eine separate Komponente für unsere Navigationsleiste haben, eine für die Fußzeile, eine andere für den Hauptinhalt und so weiter.

Diese wiederverwendbaren Komponenten machen die Entwicklung einfacher, weil man wiederkehrenden Code nicht wiederholen muss. Man muss nur die Logik erstellen und die Komponente in jeden Teil des Codes importieren, in dem sie benötigt wird. React ist außerdem eine Single-Page-Anwendung. Anstatt also jedes Mal, wenn eine neue Seite gerendert werden soll, eine Anfrage an den Server zu senden, wird der Inhalt der Seite direkt von den React-Komponenten geladen. Das führt zu einem schnelleren Rendering ohne Neuladen der Seite.

In den meisten Fällen wird für die Erstellung von React-Apps die Syntax JSX (JavaScript XML) verwendet, eine Syntaxerweiterung von JavaScript. Damit kann man die Logik von JavaScript und die Logik der Benutzeroberfläche auf einzigartige Weise kombinieren. Mit JSX muss kein DOM integriert werden, weil man einfach Methoden wie **document.getElementById**, **querySelector** und andere DOM-Manipulationsmethoden verwendet. Die Verwendung von JSX ist zwar nicht obligatorisch, aber sie macht die Entwicklung von React-Anwendungen einfacher.

Im oberen Code-Beispiel wird eine funktionale React-Komponente verwendet, um ein Stück Text im Browser darzustellen. Der Name der Komponente ist **App**. Es wird eine Variable vor der Funktion **render()** erstellt. Diese Variable wird mit geschweiften Klammern an das Markup übergeben. Das ist kein HTML, sondern die Syntax für das Schreiben von Code mit JSX.

```
function App() {  
  const greetings = "Hello World";  
  return (  
    <div className="App">  
      <h1> {greetings} </h1>  
    </div>  
  );  
}
```

Abbildung 5: JSX-Example in React [2]

3.2 Tailwind CSS

Tailwind CSS ist ein Utility-First-CSS-Framework. Es stellt vorgefertigte CSS-Klassen bereit, die verwendet werden können, um HTML-Elemente schnell und einfach zu gestalten. Im Vergleich mit anderen CSS-Frameworks, die vorgefertigte UI-Komponenten bereitstellen, fokussiert sich Tailwind mehr auf Low-Level-Bausteine wie Abstände (Paddings und Margins), Schriftarten und Farbe, um benutzerdefinierte Styles zu erstellen.

Entwickler können vorgefertigte CSS-Klassen hinzufügen, anstatt für jedes Element benutzerdefiniertes CSS zu schreiben. Das beschleunigt nicht nur den Entwicklungsprozess, sondern kann sich auch positiv auf die Performance auswirken. Die Dokumentation von Tailwind CSS ist umfangreich und die große Community erleichtert den Einstieg und bietet Unterstützung.

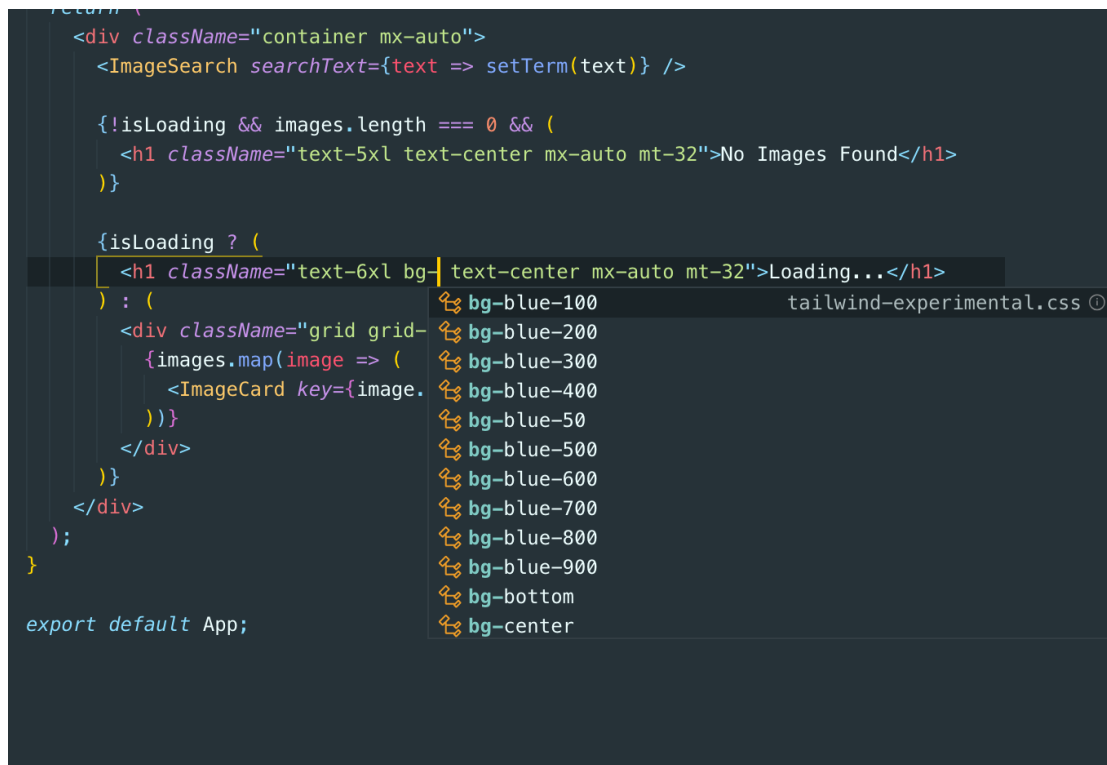


Abbildung 6: Tailwind - vorgefertigte CSS-Klassen [2]

3.3 Dev Ops

3.4 Adobe XD

4 Umsetzung

4.1 Einteilung & Design

Sobald die Festlegung und die Aufteilung der Arbeitspakete in der Firma geregelt waren, wurde mit dem Use-Case-Diagramm begonnen. Es haben sich alle Teamkollegen der Diplomarbeit daran beteiligt und nach einigen Änderungen, Diskussionen und Empfehlungen war das Use-Case-Diagramm nach zwei Tagen finalisiert.

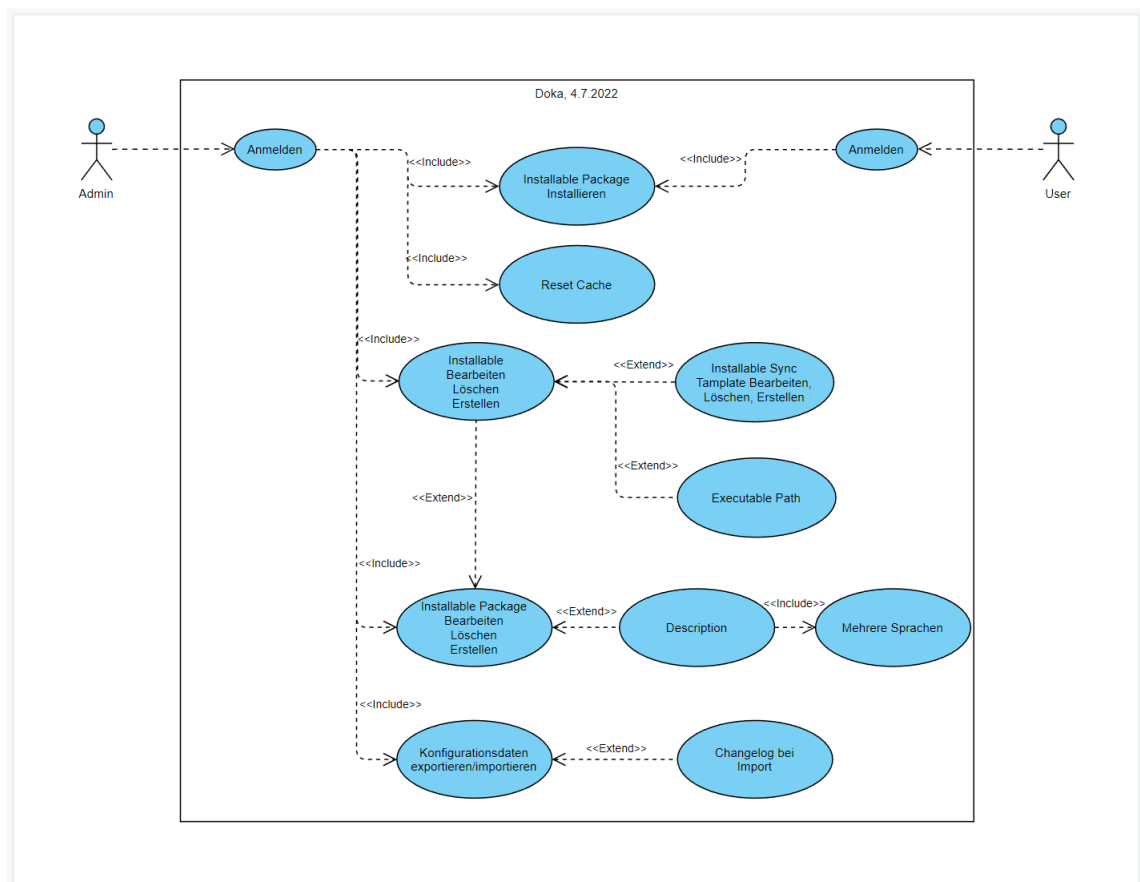


Abbildung 7: Use-Case-Diagramm [2]

Ein Use-Case-Diagramm oder Anwendungsfalldiagramm, visualisiert die von außen sichtbare Interaktion von Akteuren mit dem zu entwickelnden System. Es besteht aus dem System, zugehörigen Anwendungsfällen und Akteuren und setzt diese miteinander in Beziehung.

System: Was wird beschrieben?

Akteur: Wer benutzt das System?

Use Case: Was machen die Akteure?

5 Zusammenfassung

Aufzählungen:

- Itemize Level 1
 - Itemize Level 2
 - Itemize Level 3 (vermeiden)
- 1. Enumerate Level 1
 - a. Enumerate Level 2
 - i. Enumerate Level 3 (vermeiden)

Desc Level 1

Desc Level 2 (vermeiden)

Desc Level 3 (vermeiden)

Literaturverzeichnis

- [1] P. Rechenberg, G. Pomberger *et al.*, *Informatik Handbuch*, 4. Aufl. München – Wien: Hanser Verlag, 2006.
- [2] Association for Progressive Communications, „Wireless technology is irreplaceable for providing access in remote and scarcely populated regions,” 2006, letzter Zugriff am 23.05.2021. Online verfügbar: <http://www.apc.org/en/news/strategic/world/wireless-technology-irreplaceable-providing-access>

Abbildungsverzeichnis

1	ERD	II
2	ERD	IV
3	Swagger UI - Endpoints	2
4	Swagger UI - Endpoints	3
5	JSX-Example in React [2]	7
6	Tailwind - vorgefertigte CSS-Klassen [2]	8
7	Use-Case-Diagramm [2]	9

Tabellenverzeichnis

Quellcodeverzeichnis

Anhang