

# IoT - Device Tracker

System- und Projektdokumentation

**Team** 17

**Mitglieder**

- Pascal Sthamer (psthamer, [pascal.sthamer@smail.th-koeln.de](mailto:pascal.sthamer@smail.th-koeln.de))
- Philipp Seiner (pseiner, [philipp.seiner@smail.th-koeln.de](mailto:philipp.seiner@smail.th-koeln.de))
- Fabian Tsirogiannis (ftsirogi, [fabian\\_pascal.tsirogiannis@smail.th-koeln.de](mailto:fabian_pascal.tsirogiannis@smail.th-koeln.de))

**Version vom** 2022-02-13, 20:00:22 +0100

# Table of Contents

System .....	4
1. Anforderungen .....	5
1.1. Übersicht .....	5
1.2. Akteure und Rollen .....	6
1.3. Anwendungsfälle .....	7
1.4. Daten .....	18
1.5. Qualitätsanforderungen .....	18
1.6. Randbedingungen .....	20
2. Spezifikation .....	22
2.1. Datenschema .....	22
2.2. Verhalten .....	22
2.3. Schnittstellen .....	24
3. Architektur .....	33
3.1. Kontext .....	33
3.2. Komponenten .....	33
3.3. Laufzeitsicht .....	35
3.4. Verteilung .....	39
4. Benutzer .....	43
4.1. Installation .....	43
4.2. Administration .....	45
4.3. Benutzung .....	47
4.4. Entwicklung .....	51
Projekt .....	52
5. Zeitplanung .....	53
6. Verantwortungsbereiche .....	54
6.1. Projektplanungsphase .....	54
6.2. Prototypenintegrations- und Testphase .....	55
6.3. Hauptintegrations- und Testphase .....	55
6.4. Projekt Abgabe .....	56
7. Testprotokoll .....	57
7.1. Automatisierte End-to-End Tests .....	57
7.2. /TF01-1/ Anmeldung eines registrierten Users .....	57
7.3. /TF01-2/ Abmeldung eines Users .....	57
7.4. /TF02/ Registrierung eines Users .....	58
7.5. /TF04/ User anzeigen lassen als System-Admin .....	58
7.6. /TF05/ User bearbeiten als System-Admin .....	58
7.7. /TF06/ User löschen als System-Admin .....	59
7.8. /TF10/ Devices hinzufügen .....	59

7.9. /TF11/ Devices anzeigen .....	60
7.10. /TF12/ Devices bearbeiten .....	60
7.11. /TF13/ Devices entfernen .....	61
7.12. /TF20/ Device Group Hinzufügen .....	61
7.13. /TF21/ Device Group anzeigen .....	62
7.14. /TF22/ Device Group bearbeiten .....	62
7.15. /TF23/ Device Group entfernen .....	63
7.16. /TF30/ Nutzer zu Device Group hinzufügen .....	63
7.17. /TF31/ Nutzer der Device Group anzeigen .....	64
7.18. /TF32/ Nutzer der Device Group bearbeiten .....	64
7.19. /TF33/ Nutzer in Device Group entfernen .....	65
7.20. /TF40/ Metriken senden .....	65
7.21. /TF41/ Aktuelle Metriken einsehen .....	66

# System

Dieser Dokumentteil dokumentiert das entwickelte System aus verschiedenen Blickwinkeln.

# Chapter 1. Anforderungen

In diesem Kapitel werden Anforderungen an das Systems beschrieben. Es entspricht inhaltlich dem klassischen *Lastenheft*. Technische Details und Entscheidungen werden hier nicht beschrieben.

## 1.1. Übersicht

IoT Device Tracker ist ein System, welches die Erfassung und Überwachung von Metriken von IoT Devices erlaubt.

### 1.1.1. Einsatzzweck

Über eine Webseite können IoT Devices verwaltet und Metriken eingesehen werden. Man kann beispielsweise die CPU-Temperatur oder die Raumfeuchtigkeit im Zeitverlauf anzeigen.

IoT Devices können Gruppen zugeordnet werden, jede Gruppe hat eigene Berechtigungen und Metriken. Eine Gruppe von IoT Devices könnte beispielsweise den Standort und die CPU-Temperatur erfassen.

Welche Funktionen auf der Webseite zur Verfügung stehen, hängt von der Gruppe des betrachteten IoT Devices und der Berechtigung des Nutzers ab. Weitere Informationen dazu gibt es [hier](#).

### 1.1.2. Technik

#### (1) MQTT

Für den Nachrichtenaustausch zwischen Daemon und Backend haben uns für MQTT 5 entschieden. Diese neuere Version des MQTT Protokolls, bietet einige Vorteile im Vergleich zu MQTT 3.

#### (2) Daemon

IoT Device Tracker liefert einen lightweight Daemon, welcher Metriken erfasst und per MQTT an unser Backend liefert. Dieser Daemon ist dafür gedacht Metriken von Geräten zu sammeln, welche das MQTT Protokoll nicht von Haus aus unterstützen (wie z.B. Sensoren). Der Daemon wird in [GoLang](#) programmiert und läuft auf mehreren Plattformen. Als MQTT Client kommt [Eclipse Paho](#) zum Einsatz.

#### (3) Backend

Unser Backend ist eine Java 11 Applikation, welche eine RESTful HTTP Schnittstelle bereitstellt. Als Framework kommt [Spring Boot](#) zum Einsatz.

#### (4) MQTT Broker

Als MQTT Broker wird [Eclipse Mosquitto](#) eingesetzt. Die Authentifizierung und Authorisierung erfolgt mit Hilfe von [mosquitto-go-auth](#) und wird via HTTP an das Backend gekoppelt.

#### (5) Datenbanken

Zur persistenten Speicherung der IoT Metriken kommt [InfluxDB 2](#) zum Einsatz. InfluxDB eignet

sich sehr gut zur Speicherung von Zeitstempel-basierten Daten und bietet eine Schnittstelle um MQTT Daten abzufragen und zu speichern: <https://www.influxdata.com/integration/mqtt-monitoring/>.

Für Nutzerdaten und Nicht-Metrik-Daten der IoT Devices wird die relationale Datenbank [PostgreSQL 14](#) verwendet.

## (6) Metrics Collector

Wir nutzen den [Telegraf agent](#), welcher die Metriken via MQTT mitliest und diese regelmäßig in der InfluxDB Datenbank persistiert.

## (7) Frontend

Wir verwenden [Nuxt.js 3](#), welches auf [Vue.js 3](#) und [TypeScript](#) aufbaut, als Frontend Framework. Zur Unterstützung beim Stylen der Webseite, verwenden wir [Tailwind CSS 3](#).

## (8) System-Architekturmuster

- Frontend kommuniziert mit Backend (HTTP oder HTTPS).
- MQTT Broker kommuniziert mit Backend (HTTP oder HTTPS)
- Daemon kommuniziert mit MQTT Broker (MQTT).
- Metrics Collector kommuniziert mit MQTT Broker (MQTT) und InfluxDB (flux).
- Backend kommuniziert mit Datenbanken (flux und SQL).

## (9) Build-Infrastruktur

- Das Frontend wird mit [Yarn](#) und [Nuxt.js](#) gebildet.
- Das Backend wird mit [Gradle](#) gebildet.
- Für die Build- und Release-Automatisierung wird [GitLab CI](#) eingesetzt.

## (10) Ziel-Infrastruktur

Falls nötig läuft der Daemon auf den zu trackenden IoT Devices (nicht nötig bei MQTT Sensoren). Frontend, Backend, MQTT Broker, Metrics Collector und Datenbanken können je nach Wunsch des Betreibers als Docker Container oder direkt auf dem Host-Betriebssystem gestartet werden. Wir wollen nicht vorschreiben, welche Infrastruktur oder Cloud man nutzt, sondern ermöglichen ein flexibles Setup. Die einzige Voraussetzung ist, dass die in (7) genannten Kommunikationswege möglich sind. Im Produktionseinsatz wird dringend empfohlen das Backend und das Frontend nur über HTTPS erreichbar zu haben und entsprechende HTTP Security Header zu setzen. Die HTTPS Konfiguration kann z.B. über einen Proxy (nginx oder Tomcat) erfolgen.

# 1.2. Akteure und Rollen

Ein registrierter Nutzer hat sowohl eine globale Rolle, als auch eine Rolle pro Device Gruppe, in die er hinzugefügt wurde.

### 1.2.1. Globale Rollen

#### Nutzer

Ein Nutzer kann sich auf der Webseite anmelden und neue Device Gruppen erstellen.

#### System-Admin

Ein System-Admin kann Device Gruppen erstellen und hat Admin Zugriff auf jede Gruppe. Er kann neue Nutzer hinzufügen und bestehende Nutzer verwalten. Außerdem kann er die System-Logs einsehen. Hat ein System-Admin keine 2-Faktor-Authentifizierung (TOTP) konfiguriert, gilt er als normaler Nutzer.

### 1.2.2. Device Gruppen Rollen

Zu jeder Device Gruppe können bestehende Nutzer hinzugefügt werden. Die hinzugefügten Nutzer haben eine der folgenden Rollen:

#### Betrachter

Ein Betrachter kann die Metriken aller Geräte der Gruppe einsehen, sowie andere Nutzer in der Gruppe sehen.

#### Admin

Der Ersteller der Gruppe ist automatisch Admin. System-Admins sind implizit Admin jeder Gruppe. Ein Gruppen Admin hat unter anderem folgende Rechte:

- Metriken aller Devices in der Gruppe einsehen
- Neue Geräte in der Gruppe registrieren
- Bestehende Geräte in der Gruppe verwalten
- Nutzer der Gruppe hinzufügen
- Nutzer in der Gruppe verwalten

### 1.2.3. Sonstige Akteure

#### IoT Device

Sendet Metriken via MQTT an das Backend.

## 1.3. Anwendungsfälle



Failed to generate image: PlantUML preprocessing failed: [From <input> (line 3) ]

```
@startuml
```

```
!include UseCaseVariables.puml
```

```
^^^^^
```

```
cannot include UseCaseVariables.puml
```

```
@startuml
```

```
!include UseCaseVariables.puml
```

```
actor :$sys_admin: as sys_admin #Crimson
```

```
actor :$client: as client #Royalblue
```

```
package Admin as account_admin #Crimson {
```

```
    usecase "$LF04" as LF04
```

```
    usecase "$LF05" as LF05
```

```
    usecase "$LF06" as LF06
```

```
}
```

```
package Konto as account #Royalblue {
```

```
    usecase "$LF01_1" as LF01.1
```

```
    usecase "$LF01_2" as LF01.2
```

```
    usecase "$LF02" as LF02
```

```
    usecase "$LF03" as LF03
```

```
    usecase "$LF07" as LF07
```

```
    usecase "$LF08" as LF08
```

```
    usecase "$LF09_1" as LF09.1
```

```
    usecase "$LF09_1" as LF09.2
```

```
    usecase "$LF50" as LF50
```

```
    usecase "$LF51" as LF51
```

```
    usecase "$LF52" as LF52
```

```
    usecase "$LF20" as LF20
```

```
}
```

```
left to right direction
```

```
sys_admin -right- account_admin #Crimson
```

```
client -- account #Royalblue
```

```
sys_admin -|> "Extends" client #Crimson
```

```
@enduml
```

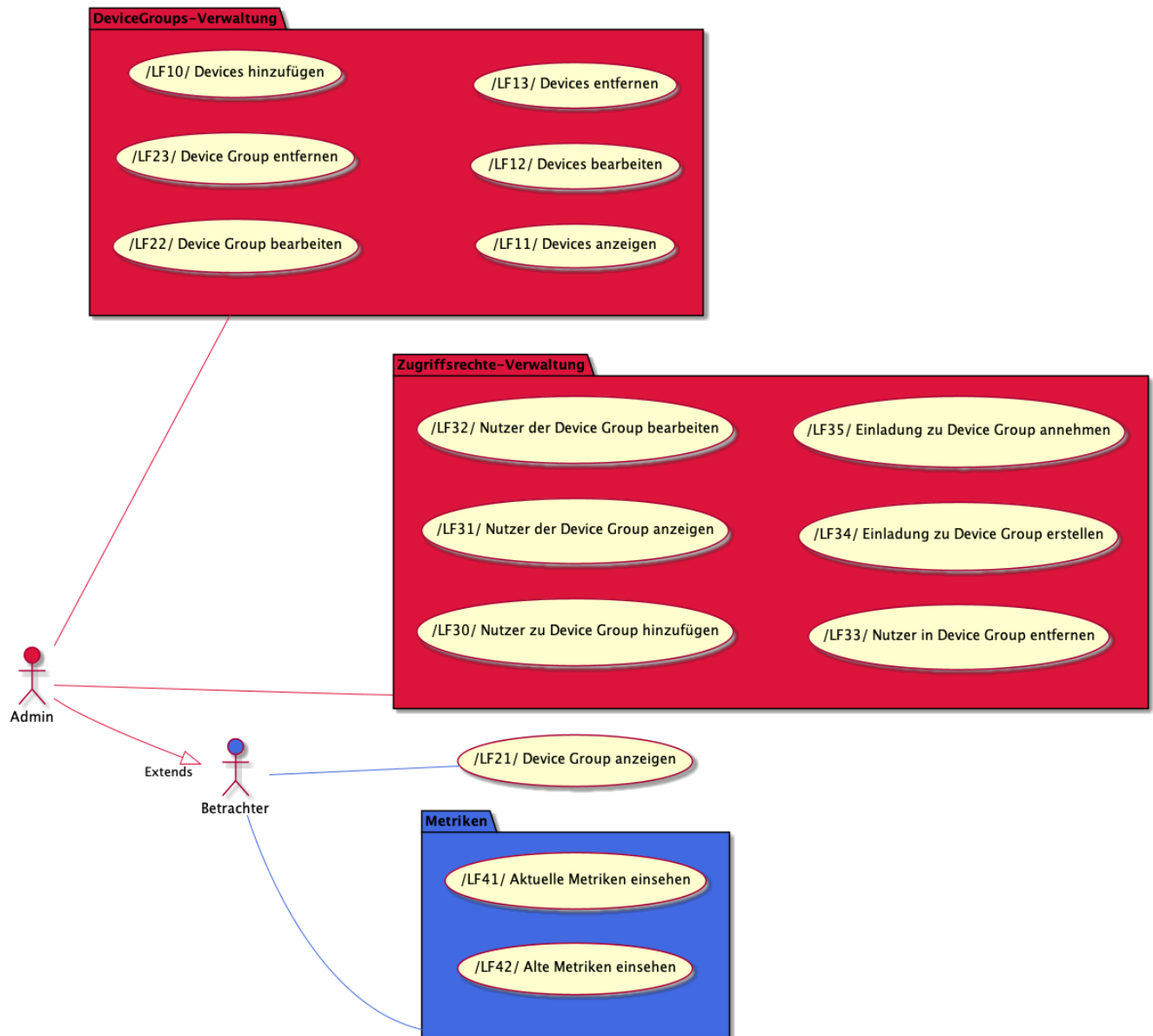


Figure 1. Anwendungsfalldiagramm-Lokal

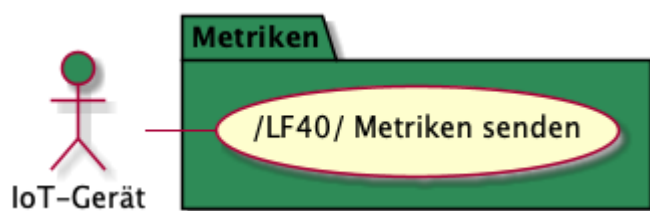


Figure 2. Anwendungsfalldiagramm-IoT-Gerät

### 1.3.1. /LF01.1/ Anmeldung

**Priorität**

Muss

**Akteur**

Nutzer, System-Admin

**Beschreibung** Der Akteur kann sich in der Webanwendung über den Reiter 'Login' mit einem Username, Passwort und TOTP (falls konfiguriert) anmelden.  
Die Anmeldedaten werden mit dem Eintrag in der [PostgreSQL 14](#) Datenbank verglichen ([bcrypt](#)).  
Stimmen die Daten nicht überein wird eine Fehlermeldung angezeigt.  
Stimmen die Login-Informationen überein, werden die zu dem Konto gehörigen Daten über Devices und Device-Groups aus der [PostgreSQL 14](#) Datenbank geholt und dem Akteur in der GUI präsentiert.

### 1.3.2. /LF01.2/ Abmeldung

**Priorität** Muss

**Akteur** Nutzer, System-Admin

**Beschreibung** Der Akteur kann sich in der Webanwendung über den Reiter 'Logout' abmelden.

### 1.3.3. /LF02/ Registrierung

**Priorität** Muss

**Akteur** Anonym

**Beschreibung** Ein angehender Nutzer kann sich über die Webanwendung unter Angabe eines Nutzernamens, einer E-Mail und eines Passworts (inklusive Bestätigung) registrieren.  
Das Passwort muss mindestens 8 Zeichen lang sein.  
Das registrierte Nutzerkonto wird mit deaktiviertem Verifizierungs-Status in der [PostgreSQL 14](#) Datenbank angelegt.  
Es wird anschließend eine Verifizierungs-E-Mail Adresse mit einem 6-stelligen alphanumerischen Code versendet.

### 1.3.4. /LF03/ Konto löschen

**Priorität** Kann

**Akteur** Nutzer, System-Admin

**Beschreibung** Alle Akteure können ihr eigenes Konto permanent löschen.  
Das Konto und die zugehörigen Daten werden in der [PostgreSQL 14](#) Datenbank gelöscht.

### 1.3.5. /LF04/ User anzeigen

<b>Priorität</b>	Kann
<b>Akteur</b>	System-Admin
<b>Beschreibung</b>	Ein Akteur kann sich alle User anzeigen lassen. Ein Akteur kann über diese Ansicht nach Username filtern. Die Liste aller User wird aus der <a href="#">PostgreSQL 14</a> Datenbank generiert.

### 1.3.6. /LF05/ User bearbeiten

<b>Priorität</b>	Kann
<b>Akteur</b>	System-Admin
<b>Beschreibung</b>	Ein Akteur kann alle User bearbeiten. Die Bearbeitung erfolgt über einen Button 'Bearbeiten' nachdem ein User ausgewählt wurde. Nach der Bearbeitung werden die Änderungen über den Button 'Speichern' gespeichert oder über den Button 'Abbrechen' verworfen. Die gespeicherten Änderungen werden in der <a href="#">PostgreSQL 14</a> Datenbank übernommen.

### 1.3.7. /LF06/ User löschen

<b>Priorität</b>	Kann
<b>Akteur</b>	System-Admin
<b>Beschreibung</b>	Ein Akteur kann jeden User löschen. Das Löschen erfolgt über den Button 'Löschen' und muss nochmals in einem Popup-Fenster bestätigt werden. Das Löschen wird nach doppelter Bestätigung in der <a href="#">PostgreSQL 14</a> Datenbank übernommen.

### 1.3.8. /LF07/ E-Mail Adresse bestätigen

<b>Priorität</b>	Muss
<b>Akteur</b>	Nutzer, System-Admin

**Beschreibung** Ein Akteur kann seine E-Mail Adresse bestätigen, falls sein Nutzerkonto noch nicht verifiziert ist, indem er den 6-stelligen Code aus der Verifizierungs-Mail eingibt.  
Bei erfolgreichem Code wird der Verifizierungs-Status auf Verifiziert gesetzt.

### 1.3.9. /LF08/ E-Mail Adresse ändern

**Priorität** Muss

**Akteur** Nutzer, System-Admin

**Beschreibung** Ein Akteur kann seine E-Mail Adresse ändern. Sein Verifizierungs-Status wird anschließend zurückgesetzt und er bekommt eine Verifizierung-Mail mit einem 6-stelligen alphanumerischen Code an die neue E-Mail Adresse.

### 1.3.10. /LF09.1/ Passwort ändern

**Priorität** Muss

**Akteur** Nutzer, System-Admin

**Beschreibung** Ein Akteur kann sein Passwort ändern. Dafür muss das aktuelle Passwort und das neue Passwort inklusive Bestätigung angegeben werden.  
Loggt den Nutzer überall aus, außer in der aktuellen Session.

### 1.3.11. /LF09.2/ Passwort zurücksetzen

**Priorität** Kann

**Akteur** Nutzer, System-Admin

**Beschreibung** Ein Akteur kann sein Passwort zurücksetzen. Dies erfolgt beim auswählen der Zurücksetzen-Option beim Einloggen. Nach der Eingabe der E-Mail des Benutzers wird eine E-Mail versendet durch die der Akteur dann sein neues Passwort vergeben kann. Davor muss der Akteur aber weiterhin die TOTP Authentifikation nutzen.

### 1.3.12. /LF10/ Devices hinzufügen

**Priorität** Muss

**Akteur** Admin, System-Admin

**Beschreibung** Der Akteur kann IoT Devices zu einer Device Group hinzufügen.  
Devices werden über den Button 'Device hinzufügen' hinzugefügt und es müssen alle Daten außer der ID des Devices angegeben werden.  
Das hinzugefügte Device wird der [PostgreSQL 14](#) hinzugefügt.

### 1.3.13. /LF11/ Devices anzeigen

**Priorität** Muss

**Akteur** Betrachter, Admin, System-Admin

**Beschreibung** Der Akteur kann IoT Devices einer Device Group anzeigen lassen.  
Daten über das Device werden aus der [PostgreSQL 14](#) geladen.

### 1.3.14. /LF12/ Devices bearbeiten

**Priorität** Muss

**Akteur** Admin, System-Admin

**Beschreibung** Der Akteur kann IoT Devices bearbeiten.  
Wenn ein Device ausgewählt wurde, kann über den Button 'Bearbeiten' das Device bearbeitet werden.  
Nach der Bearbeitung werden die Änderungen über den Button 'Speichern' gespeichert oder über den Button 'Abbrechen' verworfen.  
Die geänderten Informationen zu Device werden in die [PostgreSQL 14](#) geschrieben.

### 1.3.15. /LF13/ Devices entfernen

**Priorität** Muss

**Akteur** Admin, System-Admin

**Beschreibung** Der Akteur kann IoT Devices entfernen  
Das Löschen erfolgt über den Button 'Löschen' und muss nochmals in einem Popup-Fenster bestätigt werden.  
Das Löschen wird nach doppelter Bestätigung in die [PostgreSQL 14](#) übernommen.  
Die vom Device generierten Metriken werden in [InfluxDB 2](#) entfernt.

### 1.3.16. /LF20/ Device Group hinzufügen

<b>Priorität</b>	Muss
<b>Akteur</b>	Nutzer, System-Admin
<b>Beschreibung</b>	<p>Der Akteur kann Device Groups hinzufügen.</p> <p>Dabei muss er einen Namen für die Gruppe eingeben und diesen dann bestätigen.</p> <p>Nach der Bestätigung wird eine Device Group mit einer eindeutigen Id und dem eingegebenem Namen erstellt und der Nutzer erhält eine positive Rückmeldung.</p>

### 1.3.17. /LF21/ Device Group anzeigen

<b>Priorität</b>	Muss
<b>Akteur</b>	Betrachter, Admin, System-Admin
<b>Beschreibung</b>	<p>Der Akteur kann Device Groups, auf die er Zugriff hat, anzeigen lassen.</p> <p>Durch einen klick auf eine Device Group, wird eine neue Seite geladen, die Informationen über die ausgewählte Device Group enthält.</p>

### 1.3.18. /LF22/ Device Group bearbeiten

<b>Priorität</b>	Muss
<b>Akteur</b>	Admin, System-Admin
<b>Beschreibung</b>	Der Akteur kann den Namen einer Device Group ändern.

### 1.3.19. /LF23/ Device Group entfernen

<b>Priorität</b>	Muss
<b>Akteur</b>	Admin, System-Admin
<b>Beschreibung</b>	<p>Der Akteur kann Device Groups entfernen.</p> <p>Das erfolgt durch einen Button "Löschen", während der Nutzer sich seine Device Group ansieht. Nach dem Drücken des Buttons erscheint ein Bestätigungsdialog.</p>

### 1.3.20. /LF30/ Nutzer zu Device Group hinzufügen

<b>Priorität</b>	Muss
------------------	------

<b>Akteur</b>	Admin, System-Admin
<b>Beschreibung</b>	Der Akteur kann Nutzer unter Angabe seines Nutzernamens oder der E-Mail Adresse zu einer Device Group hinzufügen. Es muss die Rolle Betrachter oder Admin ausgewählt werden. Betrachter sollte der Default-Wert sein.

### 1.3.21. /LF31/ Nutzer der Device Group anzeigen

<b>Priorität</b>	Muss
<b>Akteur</b>	Betrachter, Admin, System-Admin
<b>Beschreibung</b>	Der Akteur kann Nutzer mitsamt Rolle in einer Device Group anzeigen lassen.

### 1.3.22. /LF32/ Nutzer der Device Group bearbeiten

<b>Priorität</b>	Muss
<b>Akteur</b>	Admin, System-Admin
<b>Beschreibung</b>	Der Akteur kann einem Nutzer in der Device Group eine andere Rolle zuweisen.

### 1.3.23. /LF33/ Nutzer in Device Group entfernen

<b>Priorität</b>	Muss
<b>Akteur</b>	Admin, System-Admin
<b>Beschreibung</b>	Der Akteur kann Nutzer aus einer Device Group entfernen. Erfordert die Bestätigung in einem Dialog.

### 1.3.24. /LF34/ Einladung zu Device Group erstellen

<b>Priorität</b>	Kann
<b>Akteur</b>	Admin, System-Admin



**Beschreibung** Ein Akteur kann Links erstellen, um andere zur seiner Device Group einzuladen.  
Durch eine klick auf ein Kettensymbol wird ein Einladungslink erstellt. Es öffnet sich dann ein kleines Popup-Fenster indem man die Ablaufzeit des Links bestimmen kann, danach wird durch einen klick auf einen Knopf dieser erstellt und kann kopiert werden. Die eingeladenen Nutzer haben die Betrachter Gruppe.

### 1.3.25. /LF35/ Einladung zu Device Group annehmen

**Priorität** Kann

**Akteur** Nutzer

**Beschreibung** Nach Aufruf eines Links aus [LF34](#) wird der angemeldete Nutzer als Betrachter der Gruppe hinzugefügt.

### 1.3.26. /LF40/ Metriken senden

**Priorität** Muss

**Akteur** IoT Device

**Beschreibung** Ein Akteur kann Metriken über MQTT 5 Publish an den Broker senden.

### 1.3.27. /LF41/ Metriken einsehen

**Priorität** Muss

**Akteur** Betrachter, Admin, System-Admin

**Beschreibung** Ein Akteur kann sowohl aktuelle, als auch alte Metriken eines Devices einsehen. Die Metriken werden über eine REST-Schnittstelle vom Backend abgerufen, welches die Metriken wiederum aus der InfluxDB ausliest. Der Zeitraum kann durch ein Dropdown bestimmt werden. Sind keine Metriken für den gewählten Zeitraum vorhanden, soll eine Warnung angezeigt werden.

### 1.3.28. /LF50/ TOTP konfigurieren

**Priorität** Muss

**Akteur** Nutzer, System-Admin

<b>Beschreibung</b>	<p>Die Akteure können über die Webseite einen TOTP Authentikator hinzufügen, falls noch keinen TOTP Authentikator für den Nutzer existiert.</p> <p>Ablauf:</p> <ol style="list-style-type: none"> <li>1. Nutzer klickt auf Button und bekommt einen Dialog angezeigt. Dort muss er zunächst sein aktuelles Passwort eingeben.</li> <li>2. Anfrage an das Backend mit dem Passwort des Nutzers.</li> <li>3. Backend validiert Passwort und prüft, dass zuvor noch kein TOTP Authentikator hinzugefügt wurde. <ol style="list-style-type: none"> <li>3.1 Im Fehlerfall wird ein relevanter Error zurückgesendet.</li> <li>3.2 Im Erfolgsfall, wird ein TOTP Authentikator in der Datenbank angelegt mit random Secret und verified=false. Secret und TOTP QRCode wird zurückgesendet</li> </ol> </li> <li>4. Nutzer bekommt Anweisung den QR Code zu scannen oder den Schlüssel manuell in seine Authentikator App einzutragen.</li> <li>5. Nutzer wird aufgefordert den aktuellen TOTP Token einzugeben.</li> <li>6. Anfrage an das Backend mit aktuellem TOTP Code.</li> <li>7. Backend prüft TOTP Code. <ol style="list-style-type: none"> <li>8.1 Im Fehlerfall wird ein relevanter Error zurückgesendet.</li> <li>8.2 Im Erfolgsfall wird der TOTP Authentikator aktiviert (verified=true), Recovery Codes generiert und alle anderen Sessions zerstört. Anschließend wird eine erfolgreiche Antwort zurückgesendet, welche die generierten Recovery Codes enthält.</li> </ol> </li> </ol>
---------------------	---

### 1.3.29. /LF51/ TOTP Status sehen

<b>Priorität</b>	Muss
<b>Akteur</b>	Nutzer, System-Admin
<b>Beschreibung</b>	Die Akteure können sehen, ob TOTP konfiguriert ist. Falls ja, wird ein Button zum Deaktivieren angezeigt ( <a href="#">LF52</a> ). Falls nein, wird ein Konfigurieren Button angezeigt ( <a href="#">LF50</a> ).

### 1.3.30. /LF52/ TOTP deaktivieren

<b>Priorität</b>	Muss
<b>Akteur</b>	Nutzer, System-Admin
<b>Beschreibung</b>	Die Akteure können TOTP wieder deaktivieren. Der TOTP Authentikator in der Datenbank wird gelöscht.

## 1.4. Daten

### 1.4.1. /LD01/ User

Ein User hat eine ID, einen Nutzernamen, ein Passwort, sowie eine E-Mail Adresse um sich anzumelden. Ein User kann entweder verifiziert sein oder nicht. User sind mehreren IoT Device Gruppen zugeordnet.

User können einen TOTP Authenticator und mehrere Recovery Codes haben.

### 1.4.2. /LD03/ TOTP Authenticator

User können einen oder keinen TOTP Authenticator haben.

Enthält ID, User ID, Secret und Verified.

Ist ein TOTP Authenticator konfiguriert, werden außerdem mehrere Recovery Codes gespeichert.

### 1.4.3. /LD04/ Device

Ein Device hat ID, MAC-Adresse, Namen und Token.

### 1.4.4. /LD05/ Device Group

Besitzt eine ID und einen Namen. Eine Device Group hat mehrere Devices und Device Group User.

Weiterhin besitzt sie noch ein Erstelldatum als Konstante und ein Datum welches festhält wann die Device Group zum letzten Mal bearbeitet wurde.

### 1.4.5. /LD06/ Device Group User

Ordnet User einer Device Group zu. Enthält User ID, Device Group ID und Role.

### 1.4.6. /LD09/ Metric

Enthält mehrere Metriken zu einem eindeutig identifizierten Zeitpunkt (ms Präzision), sowie die dazugehörige Device ID.

## 1.5. Qualitätsanforderungen

### 1.5.1. /LQ01/ Verarbeitungszeit eines Requests zur Metrikerzeugung

<b>Kategorie</b>	Performance / Antwortzeit
<b>Beschreibung</b>	99% aller Aufrufe an die RESTful-Backend Schnittstelle zur Erzeugung einer Visualisierung der IoT Device Metriken in weniger als 100ms verarbeitet werden.

### 1.5.2. /LQ02/ Durchsatz von Requests zur Metrikerzeugung

<b>Kategorie</b>	Performance / Durchsatz
<b>Beschreibung</b>	Die RESTful-Backend Schnittstelle muss einen Workload von 200 Requests zur Erzeugung von Metrikvisualisierungen pro Sekunde ohne Verschlechterung der Antwortzeiten aus <a href="#">LQ01</a> Request verarbeiten können.

### 1.5.3. /LQ03/ Speicherkapazität für Metrikerzeugungen

<b>Kategorie</b>	Performance / Kapazität
<b>Beschreibung</b>	Die RESTful-Backend Schnittstelle muss 10.000.000.000 Metrikerzeugungen der IoT Devices speichern können.

### 1.5.4. /LQ04/ Skalierbarkeit bei schwankenden Workloads

<b>Kategorie</b>	Performance / Skalierbarkeit
<b>Beschreibung</b>	IoT Device-Tracker muss auf eine langfristige Zunahme und saisonale Schwankungen im Workload durch automatische horizontale Skalierung reagieren können.

### 1.5.5. /LQ05/ Robustheit gegen Ausfall des Frontends

<b>Kategorie</b>	Verlässlichkeit / Robustheit
<b>Beschreibung</b>	IoT Device-Tracker muss über die Web-API verwendbar sein, selbst wenn das Frontend ausfällt und insbesondere keine Device-Metriken liefern kann.

### 1.5.6. /LQ06/ Verfügbarkeit

<b>Kategorie</b>	Verlässlichkeit / Verfügbarkeit
<b>Beschreibung</b>	IoT Device-Tracker muss bei einwandfreiem Deployment zu 99,99% verfügbar sein, d.h. in Summe nicht mehr als 52 Minuten pro Jahr nicht-verfügbar.

### 1.5.7. /LQ07/ Konsistenz von Änderungen der User, Device oder Device-Group Daten

<b>Kategorie</b>	Verlässlichkeit / Konsistenz
------------------	------------------------------

**Beschreibung** Bei Änderungen von User, Device oder Device-Group Daten im Web-Frontend müssen diese innerhalb von 10 Minuten nach Änderung bei neuen Abfragen berücksichtigt werden.

### 1.5.8. /LQ08/ Nachvollziehbarkeit durch Logging

**Kategorie** Wartbarkeit / Nachvollziehbarkeit

**Beschreibung** Interaktionen von Usern mit IoT Device-Tracker müssen im Logging für den System-Admin nachvollziehbar sein.

### 1.5.9. /LQ09/ Deployment fachlicher Änderungen

**Kategorie** Wartbarkeit / Änderbarkeit

**Beschreibung** Änderungen an bestehenden Komponenten müssen konfiguratив durchführbar sein, d.h. dieser Änderungen sollen den laufenden Betrieb nicht unterbrechen.

### 1.5.10. /LQ10/ Two-Factor-Login für System-Administratoren

**Kategorie** Sicherheit

**Beschreibung** System-Administratoren müssen sich über einen Two-Factor-Login-Verfahren authentifizieren.

## 1.6. Randbedingungen

### 1.6.1. /LC01/ Beliebiges Deployment

Soll sowohl in Public, Private, als auch Hybrid Cloud lauffähig sein.

### 1.6.2. /LC02/ Sprache

Die Webseite soll auf Englisch lokalisiert sein, jedoch mit der Möglichkeit weitere Sprachen hinzuzufügen.

### 1.6.3. /LC03/ Plattformunabhängiges Design

Es sollen Metriken von IoT Devices mit einer Vielzahl von Betriebssystemen und Architekturen erfasst werden können.

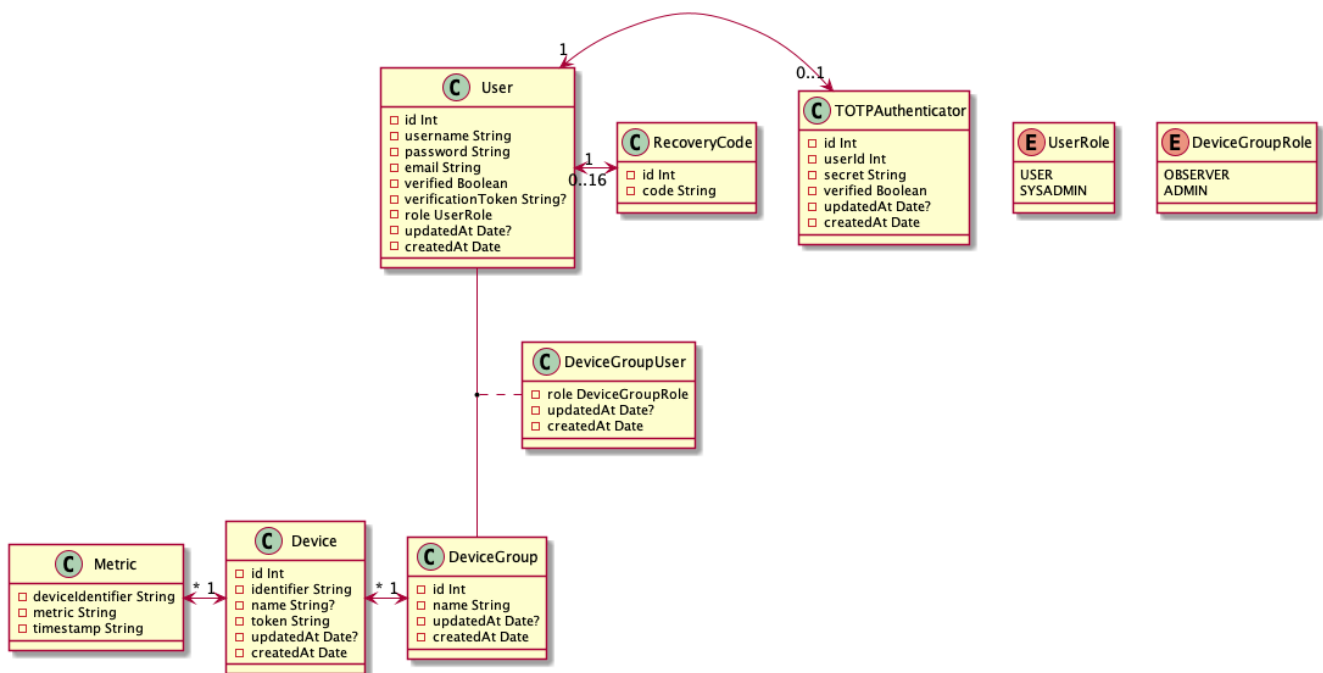
#### **1.6.4. /LC04/ Responsive Design**

Die Webseite soll sowohl auf mobilen, als auch auf Desktop-Endgeräten korrekt dargestellt werden. Es sollen sowohl der Landscape, als auch der Portrait Modus unterstützt werden.

# Chapter 2. Spezifikation

In diesem Kapitel wird spezifiziert, wie sich das System *nach außen* verhalten soll, d.h. welche Schnittstellen-Funktionen via UI oder API bereitgestellt werden, welche Entitäten das System verarbeitet und wie diese zueinander stehen und welches Verhalten das System zeigt. Der innere Aufbau des Systems ist nicht Gegenstand dieses Kapitels.

## 2.1. Datenschema



Die zu trackenden *Devices* sind in *DeviceGroups* organisiert.

Einer *DeviceGroup* können beliebig viele *Devices* zugeordnet werden, aber eine *Device* hat immer genau eine *DeviceGroup*.

Zusätzlich können einer *DeviceGroup* eine Menge von *Usern* zugewiesen werden. Die Beziehung  $User \leftrightarrow DeviceGroup$  hat weitere Attribute, wie die Rechte (Rolle) des Nutzers, welche in *DeviceGroupUser* festgehalten.

Die *Metrics* der *Devices* sind nicht wohldefiniert, um beliebige Metriken zu unterstützen.

*User* können einen *TOTPAuthenticator*, inklusive *RecoveryCodes* konfiguriert haben und sind beliebig vielen *DeviceGroups* zugeordnet.

## 2.2. Verhalten

### 2.2.1. Account

#### Lifecycle

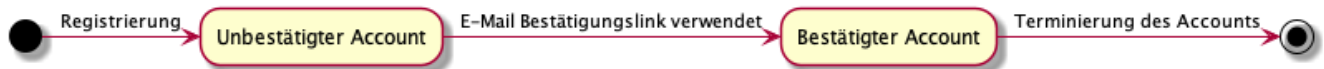


Figure 3. Lebenszyklus eines Accounts

Nach der Registrierung muss der Account durch einen Code, welcher per E-Mail versendet wird, bestätigt werden.

## Login Session



Figure 4. Session eines angemeldeten Users

Die Anmeldung eines Benutzers kann nur bei erfolgreicher Registrierung des Benutzers erfolgen. Die Anmeldung eines Benutzers folgt dem gängigen Schema einer Online-Anmeldung.

## 2.2.2. Device Group

### Lifecycle



Figure 5. Lifecycle einer Device Group

Durch das Erstellen einer Device Group wird diese erst existent.

## User Rollen

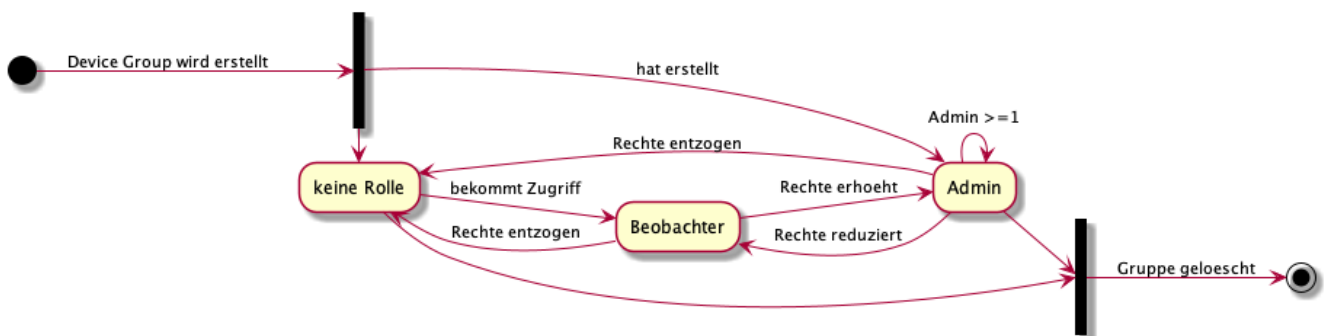


Figure 6. Benutzer Rollen innerhalb einer Device Group

Wird eine Device Group erstellt, erhält der Ersteller dieser Device Group automatisch die Rolle "Admin".

Eine Device Group muss immer mindestens einen "Admin" besitzen.

Benutzer die eine Device Group nicht erstellt haben, haben zunächst "keine Rolle".

Wird einem Benutzer innerhalb der Device Group Zugriff erteilt, wechselt dieser in die "Beobachter" Rolle.



Werden "Admin" oder "Beobachter" die Rechte entzogen, besitzen diese wieder "keine Rolle".  
"Beobachter" können zu "Admins" werden, wenn deren Rechte erhoeht werden.  
"Admins" werden zu "Beobachter", wenn deren Rechte reduziert werden.

### 2.2.3. TOTP Authenticator

#### Authenticator anlegen



Figure 7. Anlegen von TOTP Authenticator

Die TOTP Authentifizierung kann zusätzlich zur Kennwort Authentifizierung hinzugefügt werden.  
Für System-Admins ist diese auch verpflichtend.

## 2.3. Schnittstellen

### 2.3.1. User Interface

Ist man nicht angemeldet, sieht man lediglich die Login-Seite ([\[LF01.1\]](#)).

Dashboard

← Login [Register](#)

Email address  
Jane Doe

Password  
\*\*\*\*\*

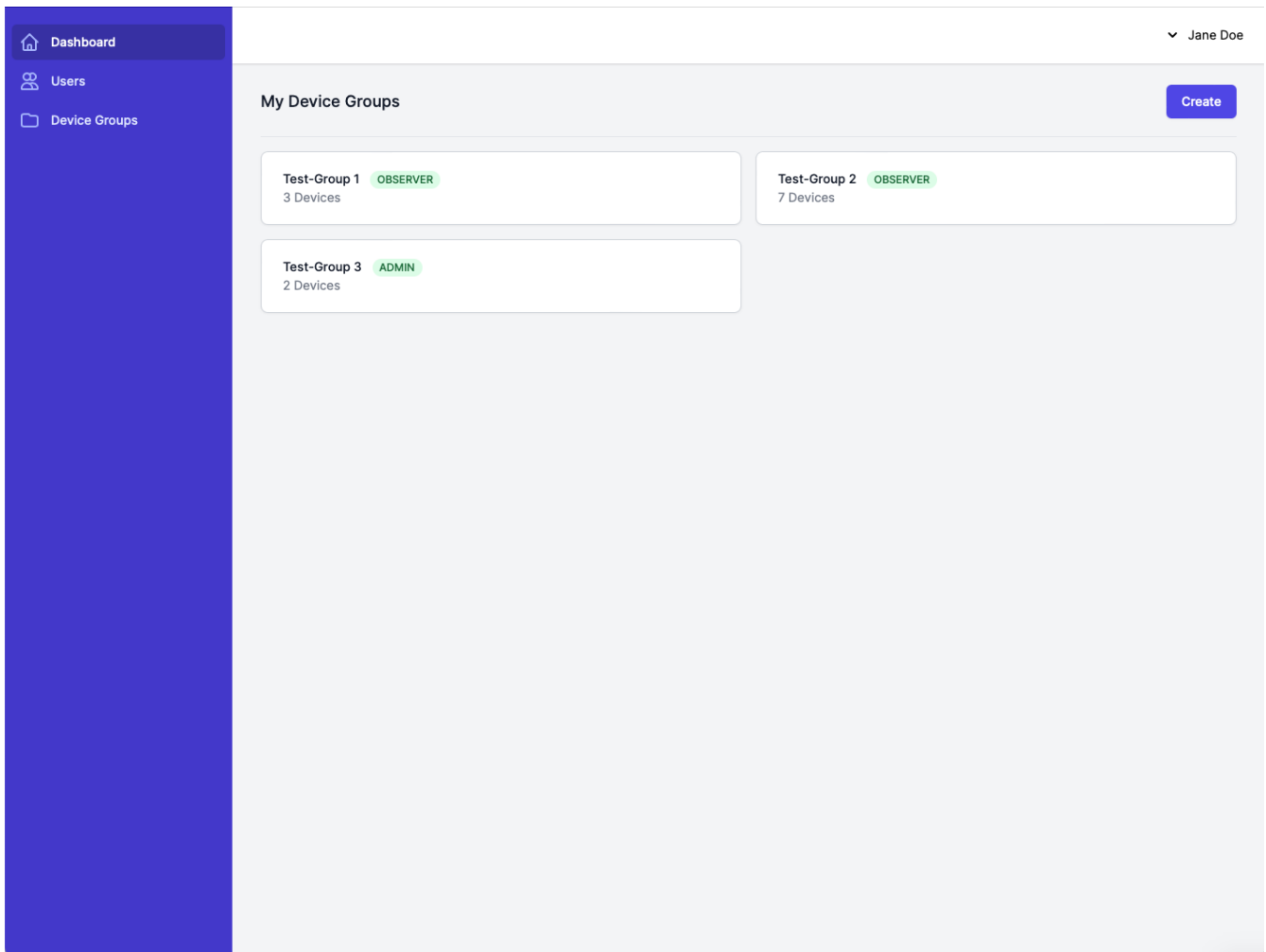
☐ Remember me [Forgot your password?](#)

Sign in

Durch einen Klick auf [Forgot your password?](#) kann man den Vorgang zur Passwort-Zurücksetzung einleiten ([\[LF09.1\]](#)). Mit einem Klick auf [Register](#) gelangt man zur Registrierungs-Seite ([\[LF02\]](#), [\[LF07\]](#)).

The image shows a web application interface. On the left is a solid blue sidebar with a white house icon and the text 'Dashboard'. The main content area has a light gray background. Centered in this area is a white rounded rectangle containing a registration form. The form is titled 'Register' with a back arrow icon on the left and a 'Login' link on the right. It contains four input fields: 'Username' with the text 'Jane Doe', 'Email address' with the text 'jane.doe@email.net', 'Password' with masked characters '\*\*\*\*\*', and 'Confirm password' also with masked characters '\*\*\*\*\*'. At the bottom of the form is a blue button with the white text 'Register'.

Ist man angemeldet, sieht man initial das Dashboard. Dort sieht man alle Device Groups, in die man hinzugefügt wurde (LF21). Mit dem Button **Create** kann man eine neue Device Group anlegen (LF20). Dazu öffnet sich ein Dialog.



Klickt man auf eine Device Group, gelangt man auf eine neue Seite, welche weitere Informationen über die Device Group darstellt ([LF21](#), [LF11](#), [LF31](#)) - insbesondere die zugehörigen Devices und User.

Über den Button **Edit** oben rechts, kann der Name der Device Group bearbeitet werden ([LF22](#)) und die Device Group kann gelöscht werden ([LF23](#)).

Über den Button **Create Device** kann ein neues Device angelegt werden ([LF10](#)).

Um einen User zu bearbeiten ([LF32](#)) oder aus der Device Group zu entfernen ([LF33](#)), gibt es den **Edit**-Button neben einem User in der Tabelle.

Der Button **Add User** erlaubt es, einen User zur Device Group hinzuzufügen ([LF30](#)). Eingabemasken zum Erstellen und Bearbeiten öffnen sich jeweils in einem Dialog auf der aktuellen Seite.

Dashboard

Team 17 Gruppe (★▽★) ADMIN

Edit Delete







Devices

Raspberry Pi  
dc:a6:32:26:ea:8e

Droni's Home (・・・) /  
eui-a81758fffe058ba7

Create Device ...

Users

NAME	EMAIL	ROLE	MFA ENABLED	
Philipp	philipp.seincher@smail.th-koeln.de	OBSERVER	Enabled	 
Fabian	fabian_pascal.tsirogiannis@smail.th-koeln.de	ADMIN	Disabled	 
Pascal (You)	pascal.sthamer@smail.th-koeln.de	ADMIN	Disabled	
team17	team17@iotdevicetracker.com	ADMIN	Enabled	 

Add User

Klickt man auf ein Device, gelangt man auf eine neue Seite. Dort kann man das Device über den **Edit**-Button bearbeiten (LF12) und löschen (LF13), sowie einen Zeitraum auswählen und die zugehörigen Metriken einsehen (LF41, [LF42]).

Pro Metrik ist eine eigene Grafik und ein eigener Grafik-Typ (z.B. eine Landkarte) denkbar. Die hier angezeigte Grafik ist nur ein sehr simples Beispiel.



Als System-Admin hat man über die Seitenleiste Zugriff auf 2 weitere Seiten, welche die Verwaltung aller User ([LF04](#), [LF05](#), [LF06](#)) und Device Groups erlauben.

## Users

USERNAME	EMAIL	MFA ENABLED	ROLE	
pmansour0	cstanesby0@unc.edu	Enabled	USER	<a href="#">Edit</a>
sleckie1	otottem1@opensource.org	Disabled	USER	<a href="#">Edit</a>
estables2	gfrudd2@netvibes.com	Enabled	USER	<a href="#">Edit</a>
jadelsberg3	rmawditt3@fema.gov	Disabled	SYSADMIN	<a href="#">Edit</a>
cabrams4	tblampey4@unblog.fr	Enabled	USER	<a href="#">Edit</a>
aeggle5	ptrench5@hatena.ne.jp	Enabled	USER	<a href="#">Edit</a>
rnuzzetti6	ohamsher6@blogs.com	Enabled	USER	<a href="#">Edit</a>
ltrembey7	fprop7@myspace.com	Enabled	USER	<a href="#">Edit</a>
jemnoney8	bcornwall8@ca.gov	Enabled	USER	<a href="#">Edit</a>
adearnley9	scalcutt9@ucoz.ru	Disabled	USER	<a href="#">Edit</a>
gbrezlawa	agioania@ca.gov	Enabled	USER	<a href="#">Edit</a>
mlonghirstb	bsproulsb@diggg.com	Disabled	USER	<a href="#">Edit</a>
kmiettinenc	amcquaidec@ifeng.com	Disabled	USER	<a href="#">Edit</a>
eamblerd	brosebladed@tumblr.com	Enabled	USER	<a href="#">Edit</a>
bfernante	jstrahane@blogger.com	Disabled	USER	<a href="#">Edit</a>

Showing 1 to 15 of 50 users

[Previous](#)[Next](#)

Dashboard

Users

Device Groups

Jane Doe

Device Groups

NAME	#DEVICES	
Fintone	06	Show
Fix San	3663	Show
Prodder	48448	Show
Asoka	5693	Show
Trippledex	88557	Show
Zontrax	2877	Show
Biodex	893	Show
Fix San	0	Show
Zamit	7833	Show
Tres-Zap	77	Show
Y-find	1	Show
Bamity	8	Show
Veribet	711	Show
Temp	2139	Show
Konklab	2000	Show

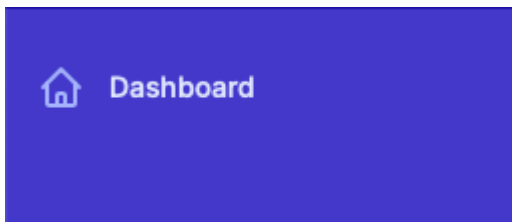
Showing 1 to 15 of 50 groups

Previous

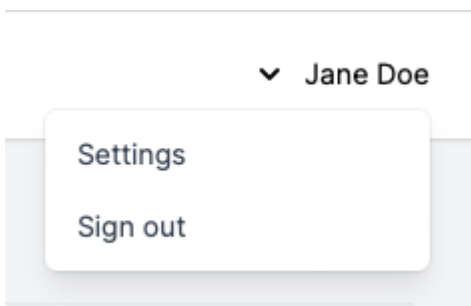
Next

localhost:3000/admin/device-groups\* in neuem Tab öffnen

Ist man kein System-Admin, fehlen diese Punkte in der Seitenleiste.



Um auf die Account-Einstellungen zuzugreifen und sich auszuloggen ([LF01.2]), klickt man oben rechts auf seinen Nutzernamen. Es erscheint ein Dropdown. Klickt man auf **Settings**, gelangt man zu einer neuen Seite, auf der man E-Mail Adresse und Passwort ändern ([LF08, LF09]), seinen Account löschen ([LF03]), sowie TOTP konfigurieren ([LF50, LF51, LF52]) kann.



Mobil passt sich das Layout der Seite entsprechend an ([LC04]). Die Seitenleiste wird ausgeblendet und ist über ein Burger-Menü oben links erreichbar.



## My Device Groups

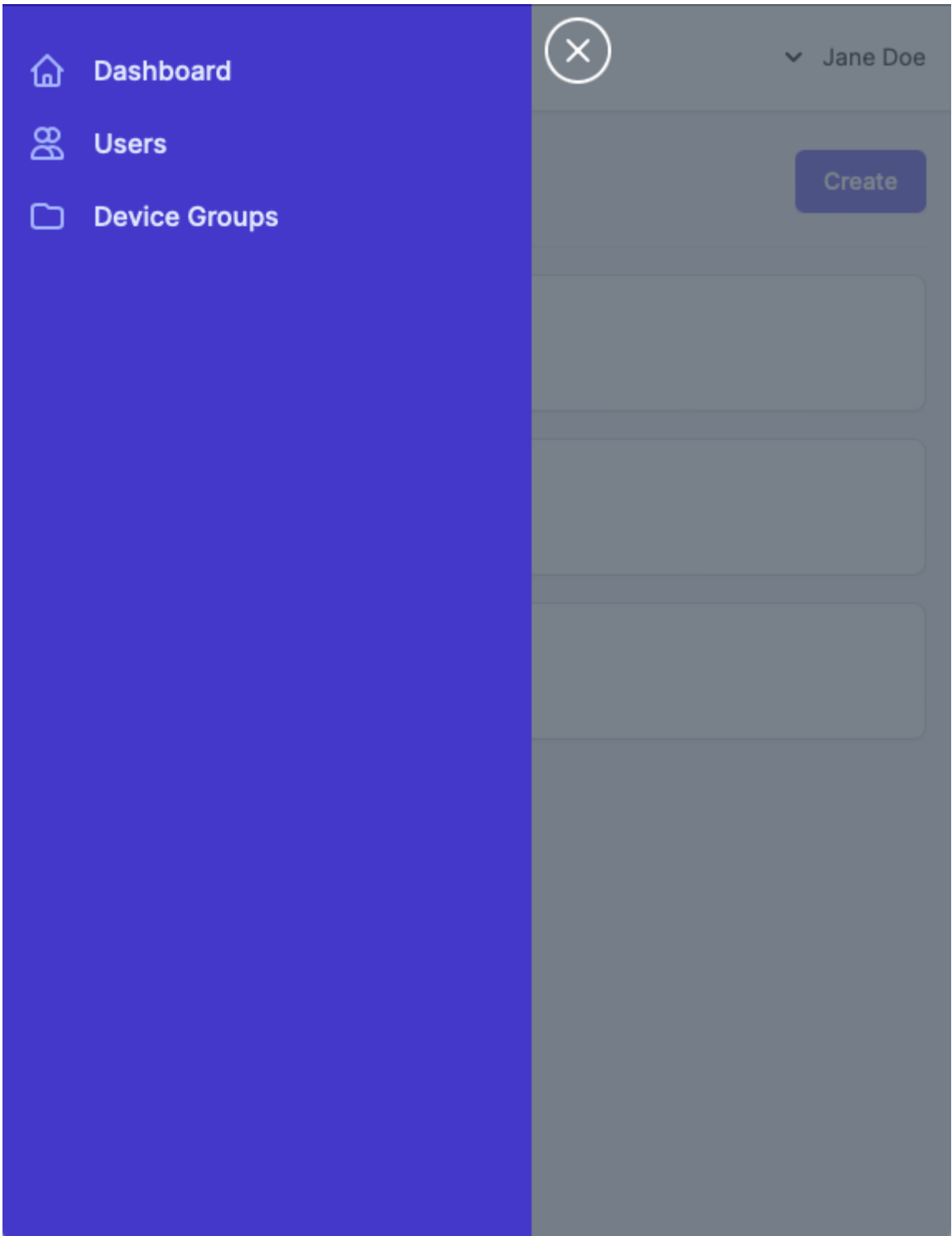
Create

**Test-Group 1** **OBSERVER**  
3 Devices

**Test-Group 2** **OBSERVER**  
7 Devices

**Test-Group 3** **ADMIN**  
2 Devices

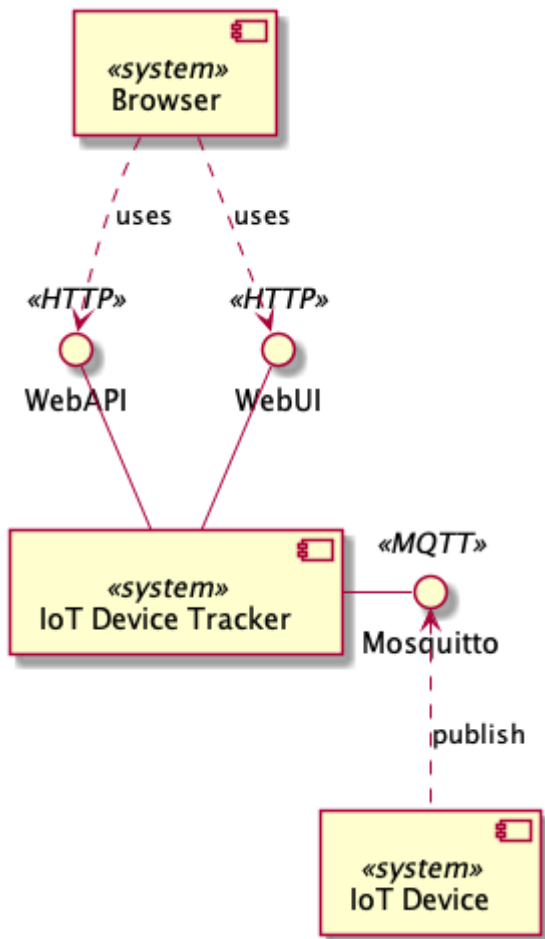




# Chapter 3. Architektur

Dieses Kapitel beschreibt den Zusammenhang des Systems mit Nachbarsystemen, den inneren Aufbau des Systems und seiner (späteren) Code-Basis, innere Abläufe und die (spätere) Verteilung des Systems und seiner Deployment-Artefakte.

## 3.1. Kontext



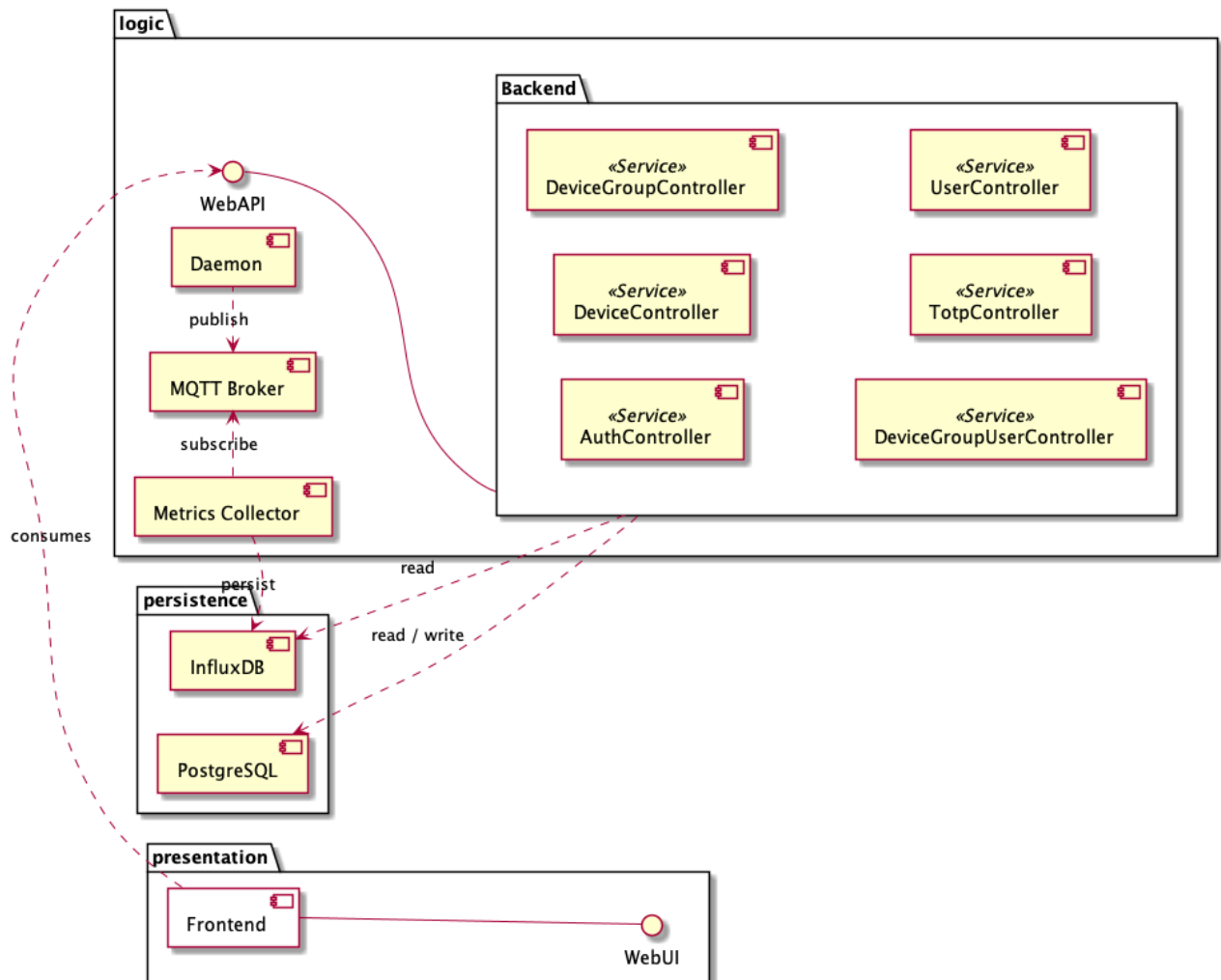
Der IoT Device Tracker ist im Wesentlichen ein in sich geschlossenes System, welches ohne externe Systeme funktioniert.

Es bietet nach außen eine HTTP Schnittstelle, welche die Weboberfläche ausliefert. Diese wird über einen Browser von Endnutzern aufgerufen.

Außerdem wird eine weitere HTTP Schnittstelle, nämlich eine RESTful API, angeboten. Diese wird ebenfalls vom Browser aufgerufen, um verschiedene Aktionen auszuführen.

Zusätzlich bietet der IoT Device Tracker eine MQTT Schnittstelle, mit welcher registrierte IoT Devices Metriken senden können.

## 3.2. Komponenten



Der IoT Device Tracker ist ein System, welches aus mehreren Komponenten besteht, die über das Netzwerk miteinander kommunizieren. Erst das Zusammenspiel dieser Komponenten ermöglicht die Funktionalität des IoT Device Tracker.

### 3.2.1. Komponente MQTT Broker

Der MQTT Broker stellt die MQTT Schnittstelle bereit, über die IoT Devices Metriken an das System senden können.

### 3.2.2. Komponente Metrics Collector

Der Metrics Collector nutzt Telegraf, um die Metriken via MQTT mitzulesen und regelmäßig in der InfluxDB zu persistieren.

### 3.2.3. Komponente Frontend

Das Frontend liefert die Web-Oberfläche via HTTP aus.

### 3.2.4. Komponente Backend

Das Backend beinhaltet die Geschäftslogik und stellt die RESTful Web API bereit.

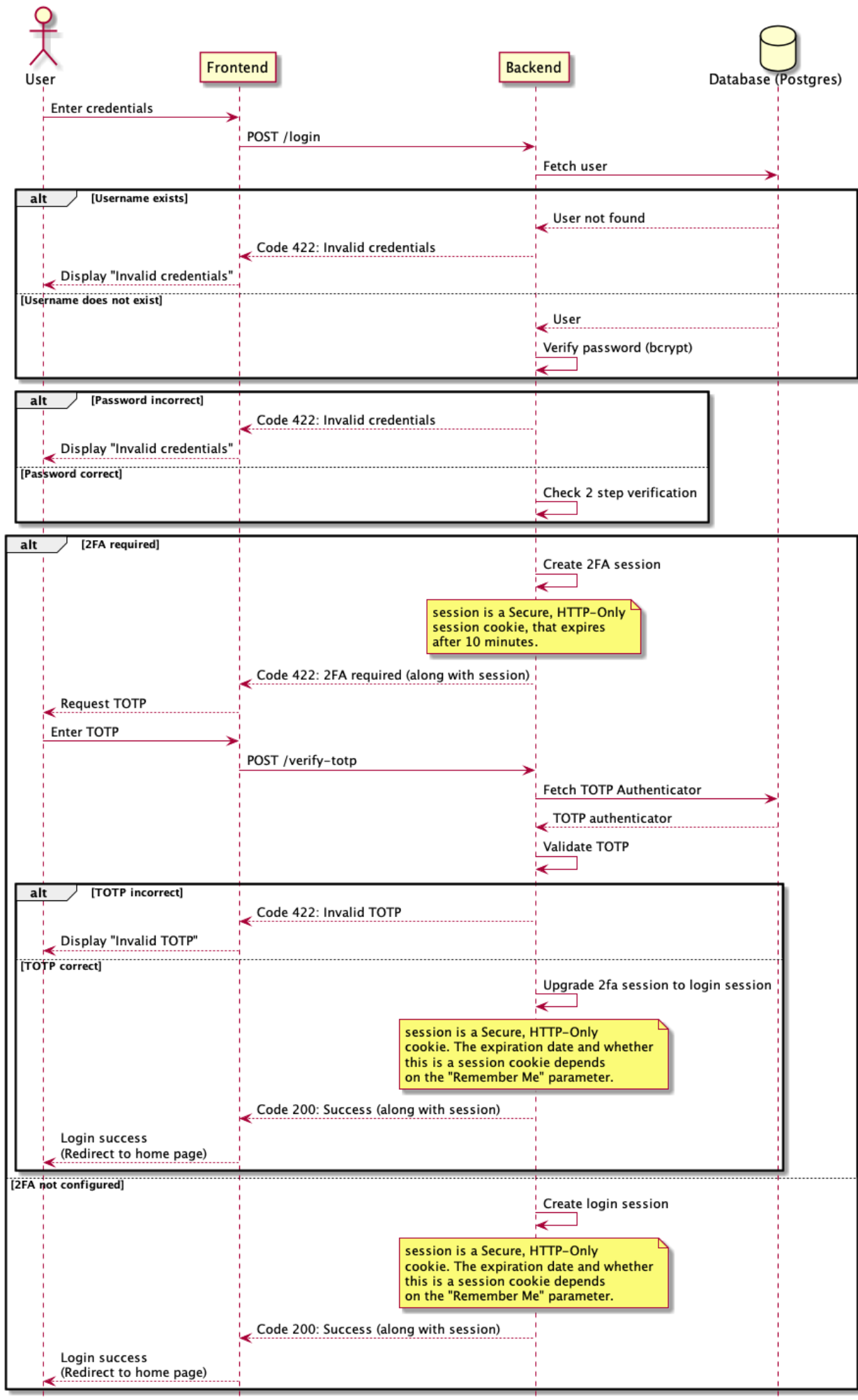
## 3.3. Laufzeitsicht

### 3.3.1. Anmeldung

Die folgende Abbildung stellt den Anmeldevorgang dar. Dieser variiert je nachdem, ob der Nutzer die 2-Faktor-Authentifizierung aktiviert hat oder nicht.

Eingabefehler, wie z.B. ein unbekannter Nutzernamen, ein falsches Passwort oder ein ungültiger TOTP-Token, werden dem Frontend durch das Backend mit Hilfe eines HTTP Response-Codes 422 signalisiert. Je nach Response-Body, reagiert das Frontend, indem es den Nutzer um eine erneute bzw. weitere Eingabe (im Falle von aktiviertem TOTP) bittet.

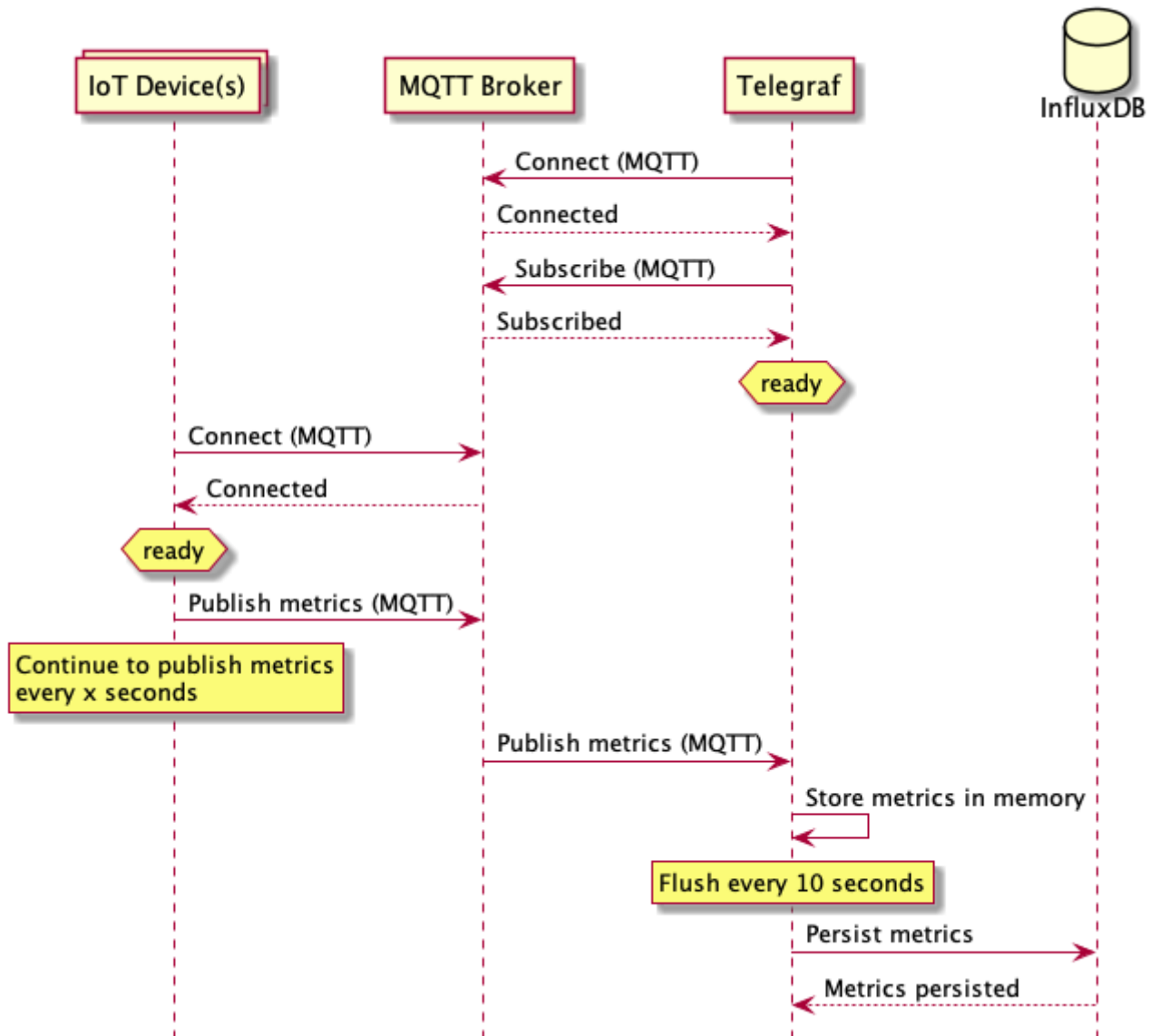




### 3.3.2. Metriken erfassen

Ein Kernbestandteil des IoT Device Trackers, ist die Erfassung von Metriken. Eine beliebige Anzahl IoT Devices, welche das MQTT Protokoll entweder nativ unterstützen oder unseren Daemon installiert haben, melden sich beim MQTT Broker (Mosquitto) an und veröffentlichen regelmäßig ihre Metriken.

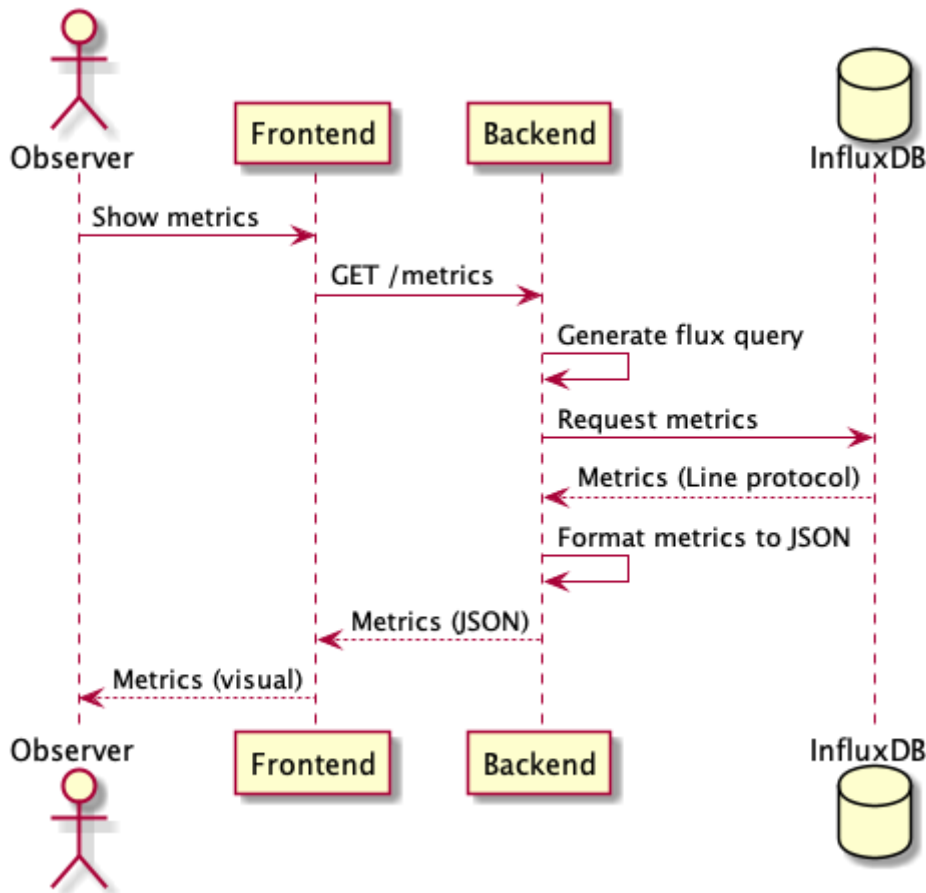
Eine Telegraf Instanz, welche auch mit dem MQTT Broker verbunden ist, liest alle Metriken mit und persistiert diese regelmäßig in InfluxDB.



### 3.3.3. Gespeicherte Metriken auslesen

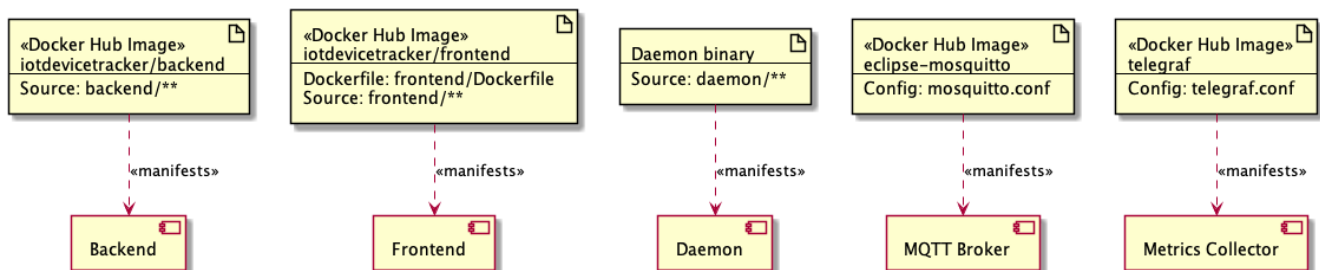
Live-Metriken können über MQTT ausgelesen werden. Der MQTT Broker speichert jedoch keine vergangenen Metriken. Diese werden ausschließlich in der InfluxDB persistiert.

Um diese Metriken einzusehen, wird eine Anfrage an das Backend gemacht, welches die Daten aus der InfluxDB holt, ggf. aggregiert und anschließend in Form von JSON an das Frontend zurückliefert. Das Frontend kann die Metriken dann je nach Typ visualisieren (z.B. in Form eines Graphen oder einer Weltkarte).



## 3.4. Verteilung

### 3.4.1. Build-Artefakte



Wir liefern die Komponenten unseres Systems in Form von Docker Images aus, welche bereits alle benötigten Abhängigkeiten beinhalten.

Für die Komponenten MQTT Broker und Metrics Collector existieren bereits Images, welche wir aus der Docker Registry beziehen.

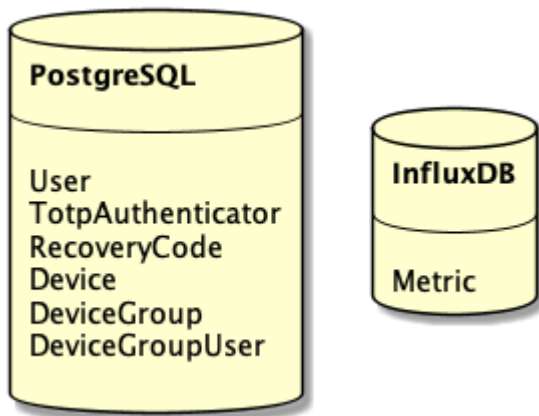
Images für Frontend und Backend werden aus unserem Quellcode erzeugt.

Der Daemon wird als einzige Komponente nicht als Docker Image ausgeliefert. Das liegt daran, dass die IoT Devices, auf denen der Daemon laufen soll, oft nicht die benötigte Rechenleistung für Docker haben und es hier auch vor allem auf Effizienz ankommt. Es gilt also die benötigte Rechenleistung und somit auch den Stromverbrauch zu minimieren. Deshalb setzen wir an dieser Stelle auf Binaries, welche für viele Plattformen erzeugt werden.



Migrationen für die Erstellung und fortlaufende Modifizierung des Datenbank-Schemas, werden über [Flyway](#) verwaltet und im Source-Code gespeichert.

### 3.4.2. Daten-Verteilung

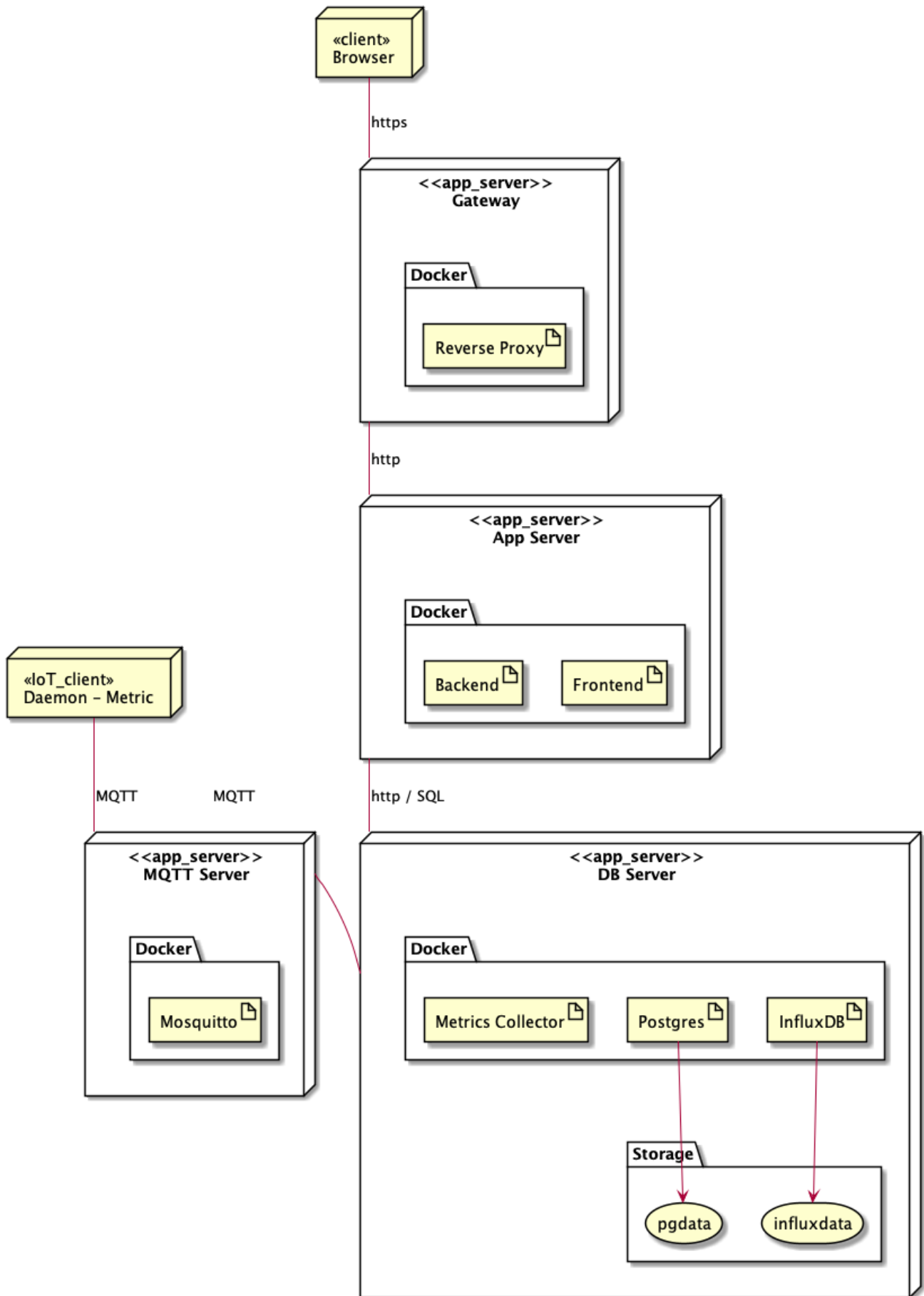


Der IoT Device Tracker arbeitet mit zwei Datenbanken.

Zum Einen setzen wir auf eine relationale PostgreSQL Datenbank zur Speicherung der wohl-definierten Entitäten, welche untereinander relationale Abhängigkeiten haben.

Für die sehr dynamischen Metriken setzen wir auf eine NoSQL Datenbank. Wir erwarten hier auch die größte Datenmenge, in Form der Metriken, weshalb sich eine NoSQL Datenbank auch aufgrund der Skalierbarkeit, besser eignet.

### 3.4.3. Deployment



Der IoT Device Tracker lässt sich sehr flexibel bereitstellen. Die Abbildung oben zeigt ein relativ simples Beispiel-Deployment, welches auf 3 Server setzt.

## MQTT Server

Dieser Server stellt den MQTT Broker, in Form eines Docker Containers für Eclipse Mosquitto, bereit. Der konfigurierte Port muss öffentlich aus dem Internet heraus erreichbar sein.

## App Server

Dieser Server stellt das Frontend und das Backend in Form von Docker Containern bereit. Beide Komponenten sollten nur im lokalen, gesicherten Netz über die konfigurierten Ports erreichbar sein. Es wird unverschlüsselt über HTTP kommuniziert.

## DB Server

Der DB Server muss sich im gleichen Netz, wie der App Server befinden. Auf ihm laufen die beiden Datenbanken, sowie der Metrics Collector, in Form von Telegraf. Die Datenbanken nutzen persistenten Speicher auf dem Host-System und sollten nicht aus dem Internet heraus erreichbar sein.

## Gateway

Diese Komponente ist öffentlich im Internet via HTTPS erreichbar und terminiert den verschlüsselten Traffic. Der Proxy leitet Anfragen an den App Server weiter. Im Internet wird nur verschlüsselt kommuniziert, während im internen Netz unverschlüsselt kommuniziert wird.

## Alternative Deployments

Wie bereits angesprochen, lässt sich der IoT Device Tracker sehr flexibel bereitstellen. Das oben genannte Deployment ist ein Beispiel, welches für die meisten Anwender gut funktionieren sollte. Man könnte jedoch einige Änderungen vornehmen, unter Anderem:

- Alles auf einem Server laufen lassen (Insbesondere für Dev- und Testumgebungen geeignet)
- Auch intern verschlüsselt kommunizieren, sodass das Frontend und Backend eine HTTPS-, anstatt einer HTTP-Schnittstelle anbieten
- Den Storage auf einen (oder mehrere) externe Server auslagern und übers Netzwerk dem DB Server verfügbar machen. Das ist insbesondere bei einem Deployment in der Cloud denkbar. So kann man auch die Datensicherheit durch Replikation erhöhen.
- Postgres und InfluxDB auf verschiedenen Servern laufen lassen, falls die Last für einen Server zu hoch wird.
- Die Komponenten Frontend, Backend oder InfluxDB horizontal über mehrere Server skalieren und über einen Load Balancer die Last verteilen. Das ist insbesondere für maximale Verfügbarkeit und bei sehr viel Traffic sinnvoll.

# Chapter 4. Benutzer

Dieses Kapitel beschreibt die Verwendung des Systems durch die [identifizierten Akteure](#).

## 4.1. Installation

Der IoT Device Tracker kann auf vielfältige Weise installiert (deployed) werden. In der folgenden Anleitung wird erklärt, wie man den IoT Device Tracker auf einer einzigen Maschine deployen kann. Dabei werden alle Komponenten des Gesamt-Systems mittels docker-compose auf einer einzigen Maschine installiert. Wir gehen in diesem Beispiel davon aus, dass ausschließlich IPv4 verwendet wird.

### 4.1.1. Mindest-Voraussetzungen

Erfahrungsgemäß reicht ein VPS mit 2 vCPUs, 4GB RAM und 20GB Speicherplatz für den Anfang aus. Sobald mehr Traffic erwartet wird, sollte das System jedoch geupgradet werden und die einzelnen Komponenten ggf. auf mehrere Maschinen verteilt werden.

### 4.1.2. System-Voraussetzungen

Es wird ein Linux Host-Betriebssystem vorausgesetzt, welches eine aktuelle Version von Docker und Docker Compose installiert hat.

Installationsanleitung Docker: <https://docs.docker.com/engine/install/>.

Installationsanleitung Docker Compose: <https://docs.docker.com/compose/install/>

Die Ports 80 und 443 müssen freigeschaltet sein.

### 4.1.3. Domain-Voraussetzungen

Es wird eine Domain (hier: example.com) vorausgesetzt, über die der IoT Device Tracker später erreichbar sein wird. Für die Domain müssen einige A Records aufgesetzt werden, welche auf die öffentliche IPv4 Adresse der Maschine (hier: 1.2.3.4) zeigen:

1. A www.example.com 1.2.3.4
2. A api.example.com 1.2.3.4
3. A influxdb.example.com 1.2.3.4
4. A mosquito.example.com 1.2.3.4

### 4.1.4. Sonstige Voraussetzungen

Es wird ein SMTP Server zum Senden von E-Mails vorausgesetzt.

### 4.1.5. Sample Deployment kopieren

Im Ordner sample\_deployment befinden sich einige Konfigurationsdateien, welche ein guter Startpunkt sind. Kopiere alle Dateien in diesem Ordner in einen beliebigen Ordner auf der

Maschine.

#### 4.1.6. Standard-Zugangsdaten ändern

Bearbeite nun die Datei `docker-compose.yaml` wie folgt:

1. Zeile 20-21: Beliebigen Nutzernamen und Passwort für die Postgres Datenbank wählen
2. Zeile 34-36: Beliebige Zugangsdaten, Organisations- und Bucket-Namen wählen. Der Name der Organisation ist meistens der Name des Unternehmens, welches den IoT Device Tracker deployed.
3. Zeile 82-84: Datenbank Zugangsdaten aus Schritt 1 eintragen
4. Zeile 85-88: Zugangsdaten für den SMTP Server eintragen

#### 4.1.7. Reverse Proxy konfigurieren

Bearbeite nun die Datei `nginx/nginx.conf` und ersetze überall `example.com` durch deine Domain.

#### 4.1.8. SSL Zertifikate aufsetzen

nginx benötigt gültige SSL Zertifikate. Diese kann man beispielsweise über Certbot (Let's Encrypt) kostenlos erhalten: <https://certbot.eff.org/instructions>.

Sollen andere Zertifikate verwendet werden, muss der bestehende Pfad (`/etc/letsencrypt/...`) in `nginx/nginx.conf` und `docker-compose.yaml` angepasst werden.

#### 4.1.9. Anwendung das erste mal starten

Alle Komponenten müssen nun über den Befehl `docker-compose up -d` gestartet werden.

Anschließend sollte unter <https://influxdb.example.com> die InfluxDB erreichbar sein. Logge dich ein und navigiere zu Data → Tokens.

Dort müssen wir nun 2 Tokens für den Metrics Collector (Telegraf) und das Backend erstellen.

##### Metrics Collector konfigurieren

1. Klicke auf Generate Token → Read/Write Token und wähle als Beschreibung Metrics Collector
2. Wähle sowohl bei Read als auch bei Write "Scoped" aus. Bei Write muss nun der Bucket (Standard: `iotdt`) ausgewählt werden.
3. Klicke anschließend auf Save. Ein neuer Eintrag sollte in der Liste erschienen sein.
4. Klicke auf "Metrics Collector Token" und kopiere den erstellten Token.
5. Füge den kopierten Token in die Datei `telegraf/telegraf.conf` in Zeile 23 ein. In Zeile 24 und 25 müssen außerdem die aktualisierten InfluxDB Zugangsdaten eingetragen werden.

##### Backend konfigurieren

Generiere einen weiteren Token in der InfluxDB, indem du wie zuvor verfährt. Vergib diesmal nur Read und keine Write Permissions auf den Bucket (Standard: `iotdt`).

Kopiere den neu erstellten Token wie zuvor und füge ihn in der `docker-compose.yaml` in Zeile 90

(INFLUX\_TOKEN: ) ein.

#### 4.1.10. MQTT Broker Zugangsdaten konfigurieren

Nun ändern wir die Zugangsdaten für den MQTT Broker.

1. Starte eine interaktive Shell in den Mosquitto Container: `docker-compose exec mosquitto /bin/bash`
2. Lösche die Standardzugangsdaten: `rm /etc/mosquitto/auth/password_file`
3. Denke dir 2 Passwörter aus (bzw. generiere 2). Das erste ist für den admin Zugang und das zweite für den Metrics Collector.
4. Führe folgenden Befehl aus, um das admin Passwort zu setzen: `echo "admin:$(/mosquitto/pw -p 'My Admin Password')>> /etc/mosquitto/auth/password_file`
5. Führe folgenden Befehl aus, um das telegraf Passwort zu setzen: `echo "telegraf:$(/mosquitto/pw -p 'My Telegraf Password')>> /etc/mosquitto/auth/password_file`
6. Trage das telegraf Passwort in der `telegraf/telegraf.conf` Datei in Zeile 4 ein.

#### 4.1.11. Alle Services neu starten

Führe die Befehle `docker-compose down` und `docker-compose up -d` nacheinander aus.  
Das System ist einsatzbereit!

## 4.2. Administration

### 4.2.1. Logging

Logs können mithilfe von `docker-compose` eingesehen werden, dabei können auf dem laufenden System folgende Befehle verwendet werden:

- `docker-compose logs frontend`
- `docker-compose logs backend`
- `docker-compose logs nginx`
- `docker-compose logs telegraf`
- `docker-compose logs postgres`
- `docker-compose logs sonarqube`
- `docker-compose logs mosquitto`

Für Mosquitto gibt es noch eine Log Datei die detailreicher ist, einsehbar unter `mosquitto/log/mosquitto.log`.

### 4.2.2. Konfiguration

#### Umgebungsvariablen

Es können folgende Umgebungsvariablen zur Konfiguration genutzt werden.

Frontend:

- `HOST`
- `PORT`

Backend:

#### 1. SQL Datenbank

- `SPRING_DATASOURCE_URL` (z.B. `jdbc:postgresql://localhost:5432/iotdtdev`)
- `SPRING_DATASOURCE_USERNAME`
- `SPRING_DATASOURCE_PASSWORD`

#### 2. NOSQL/INFLUX Datenbank

- `INFLUX_URL`
- `INFLUX_USERNAME`
- `INFLUX_PASSWORD`
- `INFLUX_TOKEN`
- `INFLUX_ORG`
- `INFLUX_BUCKET`

#### 3. Mail Server

- `SPRING_MAIL_HOST`
- `SPRING_MAIL_PORT`
- `SPRING_MAIL_USERNAME`
- `SPRING_MAIL_PASSWORD`

#### 4. Logging Level

- `LOGGING_LEVEL_ROOT`: (`info`; `warn`; `error`; `debug`; `trace`)

Daemon:

- `MQTT_URL`
- `MQTT_PASSWORD`

### Komponenten Konfigurationen

Die einzelnen Komponenten können basierend auf dem `sample_deployment` angepasst werden.

#### 4.2.3. UI-Dashboard

Als System-Admin hat man Zugriff auf alle Entitäten und kann diese auch bearbeiten.

Dies erfolgt durch die hinzugefügten Optionen in der Menüauswahl:

[Extended Menu] | `../abbildungen/admin/ui/Extended_Menu.png`

Die Verwaltung der Benutzer erfolgt dann in einer Liste die folgend aussieht:

[User List] | [../abbildungen/admin/ui/User\\_List.png](#)

Die einzelnen Benutzer können von hier entweder gelöscht werden oder bearbeitet werden, das Popup bei der Bearbeitung bietet folgende Optionen:

[User Edit] | [../abbildungen/admin/ui/User\\_Edit.png](#)

---

Weiterhin können auch alle existenten Device-Groups eingesehen werden unter "Device Groups":

[Group List] | [../abbildungen/admin/ui/Group\\_List.png](#)

Durch drücken auf **Show** wird man dann ins Dashboard für die jeweilige Gruppe weitergeleitet, unabhängig ob man teil davon nun ist. Weiterhin besitzt ein System-Admin in jeder Gruppe implizit die Admin-Rolle auch wenn er gar nicht teil davon ist.

## 4.2.4. Release

Ein Release kann mithilfe eines Tags im Gitlab erstellt werden.

Dabei wird das Frontend und Backend Image erstellt und mit dem **latest** docker-tag getagt sowie mit dem Namen des Gitlab-Tags.

Gepushed wird auf Docker-Hub.

Wenn ein Tag mit dem Namen **0.0.1** erstellt wurde sind dann folgende images mit tags abrufbar:

- **iotdevicetracker/frontend:latest**
- **iotdevicetracker/frontend:0.0.1**
- **iotdevicetracker/backend:latest**
- **iotdevicetracker/backend:0.0.1**

Vollständige Pipeline in Gitlab:

[Pipeline Success] | [../abbildungen/admin/release/Pipeline\\_Success.png](#)

## 4.3. Benutzung

### 4.3.1. Account

#### Registrierung

Man kann einen Account anlegen.

[Form Registration] | [../abbildungen/user/ui/Form\\_Registration.png](#)

Dieser muss jedoch nach der Registrierung verifiziert werden.



[Verification] | [../abbildungen/user/ui/Verification.png](#)

Gibt man diesen ein wird man zum Dashboard weitergeleitet.

## Login

Einloggen kann man sich nach einer Registrierung mit der Eingabe des Benutzernamen oder der E-Mail.

[Form Login] | [../abbildungen/user/ui/Form\\_Login.png](#)

Bei einem erfolgreichen Login wird man zum Dashboard weitergeleitet.

Ein Logout erfolgt über das drücken des eigenen Benutzernamen und dann auf **Logout**.

[Self] | [../abbildungen/user/ui/Self.png](#)

## Einstellungen

Die Einstellungen erreicht man durch das drücken des eigenen Benutzernamen und dann auf **Settings**.

[Self] | [../abbildungen/user/ui/Self.png](#)

Unter den Einstellungen können nun Account-Details verändert oder der Account gelöscht werden.

[Self Settings] | [../abbildungen/user/ui/Self\\_Settings.png](#)

## TOTP

Zum Einrichten von TOTP muss man in den Account Einstellungen zunächst gelangen und von dort aus **Start 2FA Setup** drücken.

Als ersten Schritt müssen wir unser Passwort eingeben.

[Totp 1] | [../abbildungen/user/ui/Totp\\_1.png](#)

Danach können wir mithilfe einer Totp-App den QR Code Scannen oder den manuellen Schlüssel nutzen.

Die jeweilige Totp-Authenticator müsste nach dem Setup einen 6-Stelligen Code ausgeben mit dem das Setup verifiziert werden kann.

[Totp 2] | [../abbildungen/user/ui/Totp\\_2.png](#)

Die Recovery Codes sollte man an einem sicheren Platz aufbewahren so dass diese nicht verloren gehen können.

Diese Codes können dann verwendet werden wenn man Zugang zu seinem Totp-Authenticator verloren hat.

[Totp 3] | [../abbildungen/user/ui/Totp\\_3.png](#)

Nun ist Totp erfolgreich eingerichtet.

Um Totp wieder zu deaktivieren kann man die Schaltfläche **Disable 2FA** betätigen.

[Totp 4] | ../../abbildungen/user/ui/Totp\_4.png

Dafür müssen wir nochmal das Passwort zum bestätigen eingeben.

[Totp 5] | ../../abbildungen/user/ui/Totp\_5.png

Beim Login sieht die Totp-Abfrage dann so aus:

[Form Login Totp] | ../../abbildungen/user/ui/Form\_Login\_Totp.png

### 4.3.2. Gruppen

Gruppen entsprechen einer Sammlung an Geräten und werden genutzt um den Zugriff auf diese zu regeln.

#### Anlegen

Durch drücken der Schaltfläche **Create Device Group** ... erscheint ein Popup zur Erstellung einer Gruppe.

[Group Create] | ../../abbildungen/user/ui/Group\_Create.png

Hier muss nun ein Name für die Gruppe vergeben werden.

[Group Create Form] | ../../abbildungen/user/ui/Group\_Create\_Form.png

Schlussendlich wird im Dashboard die neue Gruppe angezeigt.

[Group List] | ../../abbildungen/user/ui/Group\_List.png

Durch drücken auf die Gruppe gelangt man nun auf das Gruppen-Dashboard.

[Group Edit] | ../../abbildungen/user/ui/Group\_Edit.png

#### Bearbeiten

Vom Gruppen-Dashboard können wir durch drücken auf die **Edit** Schaltfläche nun die Gruppe bearbeiten.

[Group Edit] | ../../abbildungen/user/ui/Group\_Edit.png

Beim bearbeiten kann man nur den Gruppennamen verändern.

[Group Edit Popup] | ../../abbildungen/user/ui/Group\_Edit\_Popup.png

#### Löschen

Um die Gruppe zu löschen muss man auf die **Delete** Schaltfläche drücken und mit einem weiteren druck auf **Delete** bestätigen.

[Group Edit] | ../../abbildungen/user/ui/Group\_Edit.png

## User Hinzufügen

Durch drücken auf die Schaltfläche **Add User** kann ein Benutzer hinzugefügt werden.

[Group User Change] | *../abbildungen/user/ui/Group\_User\_Change.png*

Dabei gibt man den Namen des Nutzers ein und vergibt ihm eine Rolle.

[Group User Add] | *../abbildungen/user/ui/Group\_User\_Add.png*

## User Bearbeiten/Entfernen

Der Benutzer kann nun auch bearbeitet/entfernt werden wenn man auf die Schaltfläche mit dem Stift/Mülleimer drückt.

[Group User Change] | *../abbildungen/user/ui/Group\_User\_Change.png*

Es kann dabei die Rolle des Nutzers verändert werden oder diesem die Zugriffsrechte auf die Gruppe entzogen werden.

## 4.3.3. Geräte

### Anlegen

Ein Gerät kann angelegt werden wenn man im Gruppen-Dashboard auf die Schaltfläche **Create Device** ... drückt.

[Device Create] | *../abbildungen/user/ui/Device\_Create.png*

Dabei erscheint dann ein Popup bei dem man einen Identifier eingeben muss (z.B. MAC-Adresse) und noch einen Namen vergeben kann.

[Device Create Form Full] | *../abbildungen/user/ui/Device\_Create\_Form\_Full.png*

Danach wird der Token zurückgegeben den man sichern sollte für den Daemon. Dieser kann nämlich später nicht mehr abgerufen werden.

[Device Create Form Result] | *../abbildungen/user/ui/Device\_Create\_Form\_Result.png*

Schließlich kann man sich das Device-Dashboard anzeigen lassen.

[Device] | *../abbildungen/user/ui/Device.png*

### Bearbeiten

Vom Device-Dashboard kann man auf die **Edit** Schaltfläche drücken und dort den Namen vom Device verändern.

[Device Edit] | *../abbildungen/user/ui/Device\_Edit.png*

## Löschen

Vom Device-Dashboard kann man auf die **Delete** Schaltfläche drücken und dort erneut auf **Delete** den Löschvorgang bestätigen.

## Metriken

Es können paar Abfrageparameter eingestellt werden.

### Parameter-Timeframe

Man kann eine der Optionen aus dem Dropdown auswählen um zu bestimmen wie weit die Daten die man abfragen möchte zurückliegen.

[Device Timeframe] | *../..abbildungen/user/ui/Device\_Timeframe.png*

### Parameter-Refresh

Man kann eine der Optionen aus dem Dropdown auswählen um zu bestimmen wie oft die Daten aktualisiert werden sollen.

[Device Refresh] | *../..abbildungen/user/ui/Device\_Refresh.png*

### Parameter-Fields

Man kann eine oder mehrere der Optionen aus dem Dropdown auswählen, diese Daten werden dann im Graphen dargestellt.

[Device Fields] | *../..abbildungen/user/ui/Device\_Fields.png*

[Device] | *../..abbildungen/user/ui/Device.png*

## 4.4. Entwicklung

Informationen zur Entwicklung können sie der README.md im Hauptverzeichnis entnehmen.

# Projekt

Dieser Dokumentteil dokumentiert Details zur Projektdurchführung.

# Chapter 5. Zeitplanung

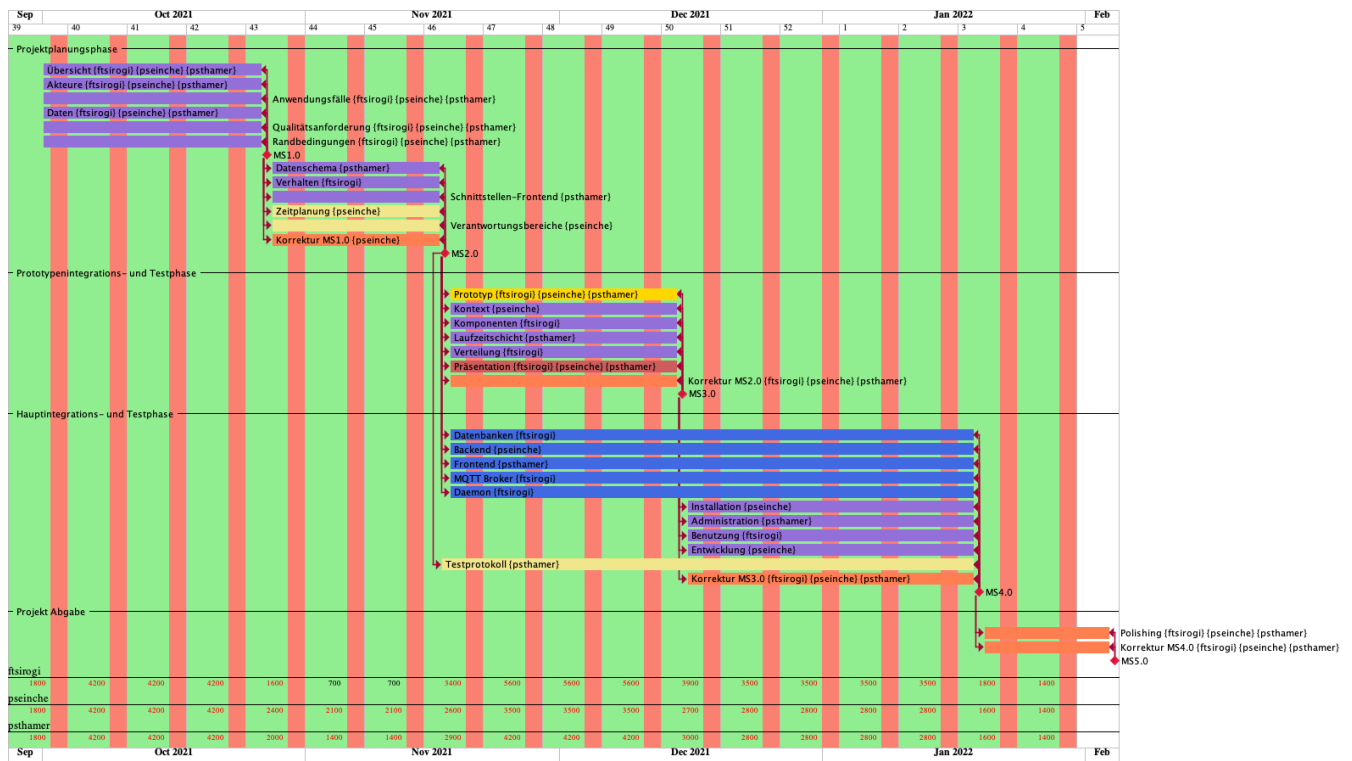


Figure 8. Zeitplanung

# Chapter 6. Verantwortungsbereiche

## 6.1. Projektplanungsphase

### 1. Übersicht:

- ftsirogi
- pseinche
- psthamer

### 2. Akteure:

- ftsirogi
- pseinche
- psthamer

### 3. Anwendungsfälle:

- ftsirogi
- pseinche
- psthamer

### 4. Daten:

- ftsirogi
- pseinche
- psthamer

### 5. Qualitätsanforderung:

- ftsirogi
- pseinche
- psthamer

### 6. Randbedingungen:

- ftsirogi
- pseinche
- psthamer

### 7. Datenschema:

- psthamer

### 8. Verhalten:

- ftsirogi

### 9. SchnittstellenFrontend:

- psthamer

### 10. Zeitplanung

- pseinche

11. Verantwortungsbereiche:

- pseinche

12. Korrektur MS1.0:

- pseinche

## 6.2. Prototypenintegrations- und Testphase

1. Prototyp:

- ftsirogi
- pseinche
- psthamer

2. Kontext:

- pseinche

3. Komponenten:

- ftsirogi

4. Laufzeitschicht:

- psthamer

5. Verteilung:

- ftsirogi

6. Präsentation:

- ftsirogi
- pseinche
- psthamer

7. Korrektur MS2.0:

- ftsirogi
- pseinche
- psthamer

## 6.3. Hauptintegrations- und Testphase

1. Datenbanken:

- ftsirogi

2. Backend:

- pseinche

3. Frontend:



- psthamer
- 4. MQTT Broker:
  - ftsirogi
- 5. Daemon:
  - ftsirogi
- 6. Installation:
  - pseinche
- 7. Administration:
  - psthamer
- 8. Benutzung:
  - ftsirogi
- 9. Entwicklung:
  - pseinche
- 10. Testprotokoll:
  - psthamer
- 11. Korrektur MS3.0:
  - ftsirogi
  - pseinche
  - psthamer

## 6.4. Projekt Abgabe

1. Polishing:
  - ftsirogi
  - pseinche
  - psthamer
2. Korrektur MS4.0:
  - ftsirogi
  - pseinche
  - psthamer

# Chapter 7. Testprotokoll

## 7.1. Automatisierte End-to-End Tests

Für einige Funktionen existieren bereits automatisierte E2E-Tests (siehe frontend/README.md). Wir planen die automatisierten End-to-End Tests in Zukunft noch zu erweitern. Da diese sehr zeitaufwendig sind, haben wir den IoT Device Tracker zunächst ausgiebig manuell getestet. Das Protokoll für die manuellen Tests finden Sie auf den folgenden Seiten.

## 7.2. /TF01-1/ Anmeldung eines registrierten Users

<b>Anwendungsfall</b>	<a href="#">LF01-1</a>
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"><li>1. <a href="https://www.iotdevicetracker.software/auth/login">https://www.iotdevicetracker.software/auth/login</a> aufrufen</li><li>2. Username eingeben</li><li>3. Passwort korrekt eingeben</li><li>4. Anmelden klicken</li></ol>
<b>erwartetes Ergebnis</b>	Der User ist erfolgreich angemeldet und befindet sich auf dem Dashboard
<b>Ergebnis</b>	Der User befindet sich auf dem Dashboard und kann eine beliebige Aktion ausführen

## 7.3. /TF01-2/ Abmeldung eines Users

<b>Anwendungsfall</b>	<a href="#">LF01-2</a>
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"><li>1. Der User muss eingeloggt sein</li><li>2. Auf den Benutzernamen oben rechts klicken</li><li>3. "Logout" auswählen</li></ol>
<b>erwartetes Ergebnis</b>	Der Benutzer ist von der Webanwendung abgemeldet
<b>Ergebnis</b>	In PostgreSQL ist die Spring Session beendet und der Cookie-Eintrag des Users ist aus der Datenbank entfernt.

## 7.4. /TF02/ Registrierung eines Users

**Anwendungsfall**      **LF02**

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"><li>1. <a href="https://www.iotdevicetracker.software/auth/register">https://www.iotdevicetracker.software/auth/register</a> aufrufen</li><li>2. Username eingeben</li><li>3. Email Adresse in validem Format eingeben</li><li>4. Passwort &gt;= 8 Zeichen eingeben</li><li>5. Eingabe mit identischem Passwort wiederholen</li><li>6. Button Register klicken</li><li>7. Verifikations-Token aus Email eingeben</li></ol>
<b>erwartetes Ergebnis</b>	Ein neuer Nutzer wird erfolgreich angelegt und befindet sich beim Loginfenster
<b>Ergebnis</b>	Der User ist in der Datenbank hinterlegt und kann sich mit den Zugangsdaten über <a href="https://www.iotdevicetracker.software/auth/login">https://www.iotdevicetracker.software/auth/login</a> anmelden

## 7.5. /TF04/ User anzeigen lassen als System-Admin

**Anwendungsfall**      **LF04**

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"><li>1. Der User muss System-Admin Rechte haben</li><li>2. Der User muss eingeloggt sein</li><li>3. "Users" im Menu auswählen</li></ol>
<b>erwartetes Ergebnis</b>	Alle registrierten User der Webanwendung werden aufgelistet
<b>Ergebnis</b>	Die angezeigten User stimmen mit den Einträgen der Datenbank USER überein.

## 7.6. /TF05/ User bearbeiten als System-Admin

**Anwendungsfall**      **LF05**

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss System-Admin Rechte haben</li> <li>2. Der User muss eingeloggt sein</li> <li>3. "Users" im Menu auswählen</li> <li>4. Das Stift-Icon zum Bearbeiten des Users auswählen</li> <li>5. Einen der User-Werte ändern</li> </ol>
<b>erwartetes Ergebnis</b>	Die Daten des Users haben sich im UI geändert.
<b>Ergebnis</b>	Die Daten des Users haben sich im UI und in der PostgreSQL Datenbank USER geändert.

## 7.7. /TF06/ User löschen als System-Admin

<b>Anwendungsfall</b>	LF06
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss System-Admin Rechte haben</li> <li>2. Der User muss eingeloggt sein</li> <li>3. "Users" im Menu auswählen</li> <li>4. Das Eimer-Icon zum Löschen des Users auswählen</li> <li>5. Einen der User-Werte ändern</li> </ol>
<b>erwartetes Ergebnis</b>	Die Daten des Users haben sich im UI geändert.
<b>Ergebnis</b>	Die Daten des Users haben sich im UI und in der PostgreSQL Datenbank USER geändert.

## 7.8. /TF10/ Devices hinzufügen

<b>Anwendungsfall</b>	LF10
-----------------------	------

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss sich innerhalb einer Device Group befinden</li> <li>2. Auf "Create Device" klicken</li> <li>3. Die MAC-Adresse des Geräts als Identifier eingeben</li> <li>4. Einen beliebigen Namen für das Gerät auswählen</li> <li>5. Auf "Create" klicken</li> <li>6. Den Token des Geräts aufbewahren</li> <li>7. Daemon auf dem Device installieren und starten</li> </ol>
<b>erwartetes Ergebnis</b>	Das neue Device wird unter Devices gelistet und kann nach der Installation und dem Start des Daemon Metriken an die Anwendung schicken.
<b>Ergebnis</b>	Das neue Device wird in der PostgreSQL Datenbank DEVICE gelistet und besitzt die korrekten Einträge. Das IoT Gerät kann nach Installation und Start der Daemon Metriken senden und diese sind in der InfluxDB als auch Frontend korrekt sichtbar.

## 7.9. /TF11/ Devices anzeigen

<b>Anwendungsfall</b>	<a href="#">LF11</a>
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss sich innerhalb einer Device Group befinden</li> <li>2. Der User muss ein korrekt angelegtes Device auswählen</li> <li>3. Metric des Device auswählen</li> <li>4. Passenden Timeframe wählen</li> </ol>
<b>erwartetes Ergebnis</b>	Das Device wird mit korrektem Namen, korrekter MAC und den Metriken angezeigt.
<b>Ergebnis</b>	Die in der Webanwendung gezeigten Parameter stimmen mit denen der PostgreSQL Datenbank Device überein. Die Metriken in der Webanwendung stimmen mit den in InfluxDB persistierten Daten überein.

## 7.10. /TF12/ Devices bearbeiten

<b>Anwendungsfall</b>	<a href="#">LF12</a>
-----------------------	----------------------

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss sich innerhalb einer Device Group befinden</li> <li>2. Das zu bearbeitende Device auswählen</li> <li>3. Auf "Edit" klicken</li> <li>4. Namen des Devices bearbeiten</li> <li>5. Mit Klick auf "Edit" bestätigen</li> </ol>
<b>erwartetes Ergebnis</b>	Das Device wird mit geändertem Namen, korrekter MAC und den Metriken angezeigt.
<b>Ergebnis</b>	Die in der Webanwendung geänderten Parameter sind in der PostgreSQL Datenbank DEVICE ebenfalls geändert worden.

## 7.11. /TF13/ Devices entfernen

<b>Anwendungsfall</b>	<a href="#">LF13</a>
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Der User muss sich innerhalb einer Device Group befinden</li> <li>2. Das zu löschende Device auswählen</li> <li>3. Auf "Delete" klicken</li> <li>4. Das Pop-up durch klicken auf "Delete" bestätigen</li> </ol>
<b>erwartetes Ergebnis</b>	Das Device ist aus der Device-Group entfernt worden.
<b>Ergebnis</b>	Das Device ist in der Webanwendung entfernt worden und befindet sich nicht mehr in der PostgreSQL Datenbank.

## 7.12. /TF20/ Device Group Hinzufügen

<b>Anwendungsfall</b>	<a href="#">LF20</a>
<b>Rolle</b>	User
<b>Voraussetzung</b>	Account besitzen und angemeldet sein.

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Dashboard aufrufen</li> <li>2. Auf 'Create Device Group' klicken</li> <li>3. Device Group in Eingabefeld 'Name' benennen</li> <li>4. Auf den Button 'Create' drücken</li> </ol>
<b>erwartetes Ergebnis</b>	Eine Device Group wird erfolgreich angelegt und wird im Dashboard angezeigt
<b>Ergebnis</b>	Die Device Group wurde in der Datenbank angelegt und kann geöffnet werden über ein Click auf die Schaltfläche, die diese repräsentiert.

## 7.13. /TF21/ Device Group anzeigen

<b>Anwendungsfall</b>	<a href="#">LF21</a>
<b>Rolle</b>	User
<b>Voraussetzung</b>	Device Group existiert bereits
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Dashboard aufrufen</li> <li>2. Schaltfläche identifizieren die, die zuvor erstellte Device Group repräsentiert</li> <li>3. Auf die Schaltfläche drücken</li> </ol>
<b>erwartetes Ergebnis</b>	Die Device Group wird übersichtlich dargestellt mit den zugehörigen Usern sowie Devices.
<b>Ergebnis</b>	Der User befindet sich in der Übersicht zur Device Group und kann weitere Änderungen von dort vornehmen.

## 7.14. /TF22/ Device Group bearbeiten

<b>Anwendungsfall</b>	<a href="#">LF22</a>
<b>Rolle</b>	User
<b>Voraussetzung</b>	Device Group existiert bereits

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. Button 'Edit' klicken</li> <li>3. Im Eingabefeld 'Name' Namensänderung durchführen</li> <li>4. Durch Klick auf 'Edit' bestätigen</li> </ol>
<b>erwartetes Ergebnis</b>	Die Namensänderung wurde erfolgreich vollzogen.
<b>Ergebnis</b>	Der Name wurde in der Datenbank geändert.

## 7.15. /TF23/ Device Group entfernen

<b>Anwendungsfall</b>	<a href="#">LF23</a>
<b>Rolle</b>	User
<b>Voraussetzung</b>	Device Group existiert bereits
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. Button 'Delete' klicken</li> <li>3. Im Dialog auf den Button 'Delete' drücken</li> </ol>
<b>erwartetes Ergebnis</b>	Device Group wurde gelöscht.
<b>Ergebnis</b>	Device Group wurde aus der Datenbank entfernt, zugehörige Objekte werden auch gelöscht.

## 7.16. /TF30/ Nutzer zu Device Group hinzufügen

<b>Anwendungsfall</b>	<a href="#">LF30</a>
<b>Rolle</b>	User
<b>Voraussetzung</b>	Device Group existiert bereits



<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. Unter Users auf den Button 'Add Users' klicken</li> <li>3. Im Eingabefeld 'Username' Benutzernamen festlegen</li> <li>4. Im Dropdown 'Role' die Rolle 'OBSERVER' oder 'ADMIN' auswählen</li> <li>5. Auf Button 'Add User' klicken</li> </ol>
<b>erwartetes Ergebnis</b>	User wird hinzugefügt.
<b>Ergebnis</b>	In der Datenbank wird ein Eintrag angelegt der die Zugriffsrechte für den erstellten Nutzer beschreibt.

## 7.17. /TF31/ Nutzer der Device Group anzeigen

<b>Anwendungsfall</b>	<a href="#">LF31</a>
<b>Rolle</b>	Admin
<b>Voraussetzung</b>	Device Group existiert bereits
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. 'Users' zeigt alle User an, mindestens den Ersteller der Gruppe selbst</li> </ol>
<b>erwartetes Ergebnis</b>	User der Device-Group sind sichtbar.
<b>Ergebnis</b>	User werden als Liste angezeigt

## 7.18. /TF32/ Nutzer der Device Group bearbeiten

<b>Anwendungsfall</b>	<a href="#">LF32</a>
<b>Rolle</b>	Admin
<b>Voraussetzung</b>	Device Group existiert bereits und ein anderer User wurde hinzugefügt zur Device Group

<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. Unter Users einen User aus der Liste auswählen und 'Edit' Icon klicken</li> <li>3. Role im Dropdown verändern</li> <li>4. Auf Button 'Edit' klicken</li> </ol>
<b>erwartetes Ergebnis</b>	Role wurde angepasst.
<b>Ergebnis</b>	Angepasster User hat eine neue Rolle in der Device Group

## 7.19. /TF33/ Nutzer in Device Group entfernen

<b>Anwendungsfall</b>	<a href="#">LF33</a>
<b>Rolle</b>	Admin
<b>Voraussetzung</b>	Device Group existiert bereits und ein anderer User wurde hinzugefügt zur Device Group
<b>Beschreibung</b>	<p>Folgende Schritte sind auszuführen +</p> <ol style="list-style-type: none"> <li>1. Device Group auswählen</li> <li>2. Unter Users einen User aus der Liste auswählen und 'Delete' Icon klicken</li> <li>3. Bestätigen durch Drücken auf 'Revoke access' Button.</li> </ol>
<b>erwartetes Ergebnis</b>	User befindet sich nicht mehr in der Gruppe und hat keinen Zugriff.
<b>Ergebnis</b>	Angepasstem User wurden die Zugriffsrechte genommen.

## 7.20. /TF40/ Metriken senden

<b>Anwendungsfall</b>	<a href="#">LF40</a>
<b>Rolle</b>	-
<b>Voraussetzung</b>	Der Besitzt eines IoT Gerätes

<b>Beschreibung</b>	Folgende Schritte sind auszuführen +	
	<ol style="list-style-type: none"> <li>1. Auf dem Gerät  '<code>MQTT_URL="https://mosquitto.iotdevicetracker.software"</code>  '<code>MQTT_PASSWORD="device-token" ./daemon</code>' ausführen</li> <li>2. Daemon auf dem IoT Gerät ausführen</li> </ol>	
<b>erwartetes Ergebnis</b>	Daemon versendet Metriken CPU-Last als Metrik	
<b>Ergebnis</b>	Daemon versendet CPU-Last als Metrik	

## 7.21. /TF41/ Aktuelle Metriken einsehen

<b>Anwendungsfall</b>	<a href="#">LF41</a>	
<b>Beschreibung</b>	Folgende Schritte sind auszuführen +	
	<ol style="list-style-type: none"> <li>1. Daemon starten</li> </ol>	
<b>erwartetes Ergebnis</b>	Aktuelle Metrik Cpu-Last wird über die Konsole angezeigt.	
<b>Ergebnis</b>	Daemon versendet über Device Metrik Cpu-Last an Mosquitto.	
<b>NOTE</b>	Es ist ein bekannter Bug, dass diese Datei im GitLab-Preview nicht richtig angezeigt wird. Sie dient in erster Linie dem PDF-Export der Gesamtdokumentation.	