# Einrichtung der FIWARE-Infrastruktur mit Docker

## Inhaltsverzeichnis

[Inhaltsverzeichnis 1](#_Toc518485960)

[Benötigte Software 2](#_Toc518485961)

[Initialisierung der Docker-Container 2](#_Toc518485962)

[Initialisierung von Wirecloud 3](#_Toc518485963)

[Einrichtung der Security-Komponenten 4](#_Toc518485964)

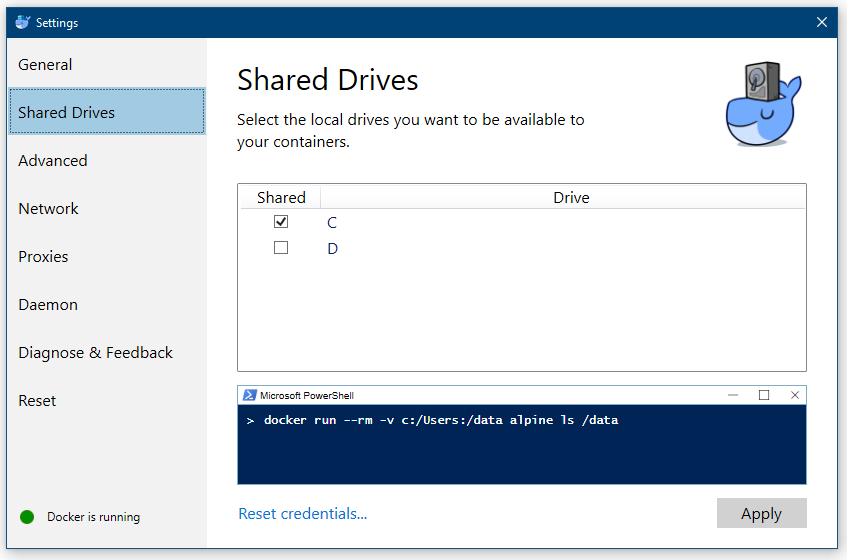
[Einrichtung des Dashboards 10](#_Toc518485965)

[Erstellen von Sensorwerten 12](#_Toc518485966)

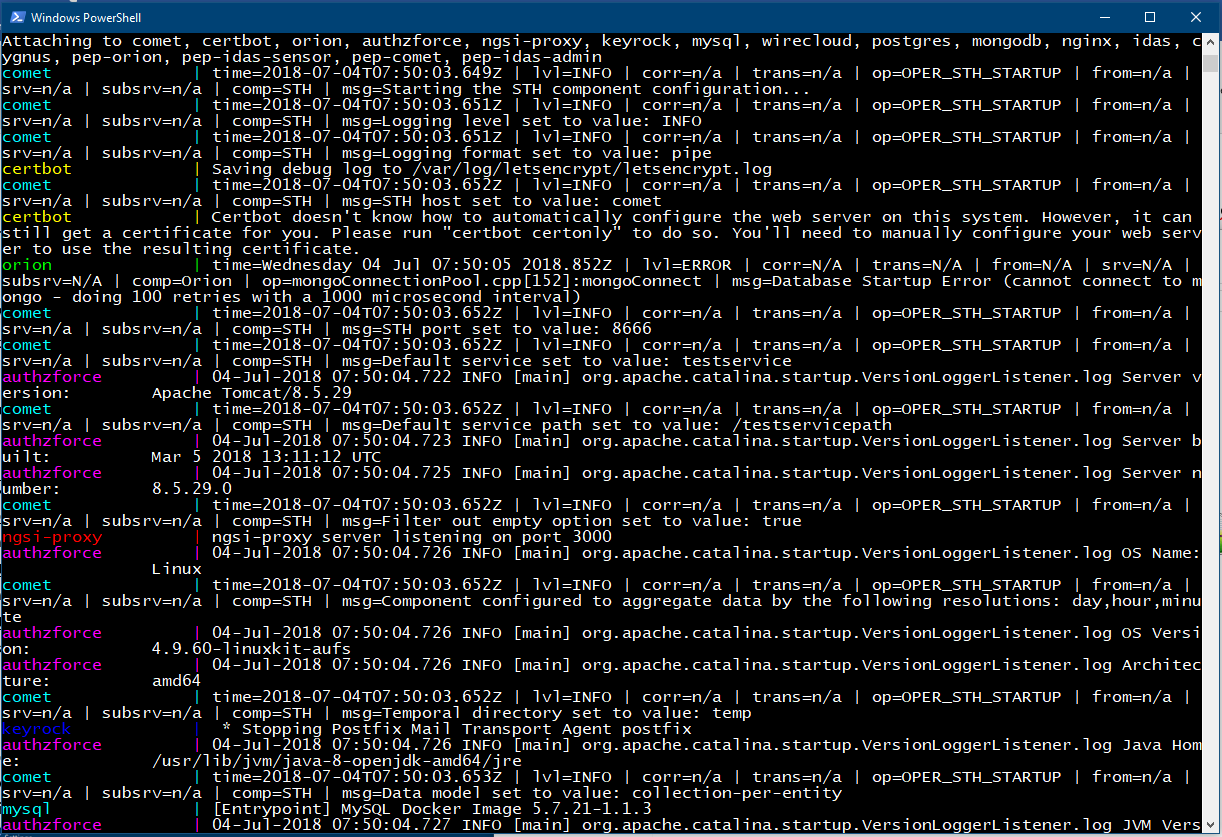
## Benötigte Software

* Docker Community Edition (inklusive Docker Compose)  
  Link: <https://docs.docker.com/docker-for-windows/install/>
* Python 2.7  
  Link: <https://www.python.org/download/releases/2.7/>
* Python-Bibliothek „requests“  
  Installation: Befehl ***pip install requests*** in Konsole ausführen

## Initialisierung der Docker-Container

Nach dem Starten des Docker-Clients muss als Erstes die Laufwerksfreigabe im Einstellungsmenü festgelegt werden:

Dieser Schritt ist notwendig, da es sonst zu Problemen mit dem Einhängen von Dateien und Volumes in Containern kommen kann.

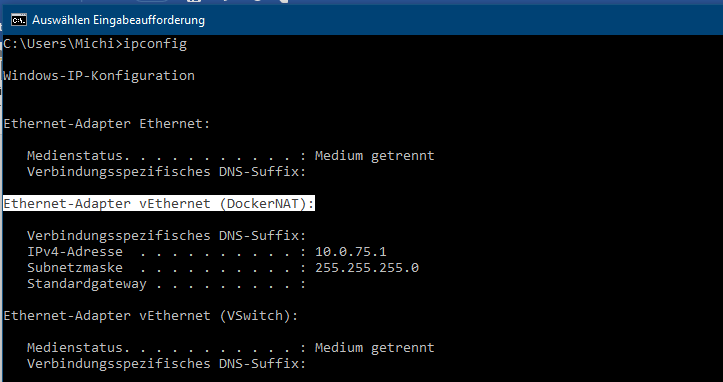
Als nächstes muss eine PowerShell-Konsole im Ordner „[…]DockerInfrastructure\Docker“ geöffnet werden. Die Initialisierung der Container wird mit dem Befehl ***docker-compose up*** gestartet.  
Docker lädt nun die benötigten Images von Docker Hub und bereitet die Container für den ersten Start vor. Wenn die Initialisierung abgeschlossen ist, werden die Log-Meldungen der einzelnen Container angezeigt:

## Initialisierung von Wirecloud

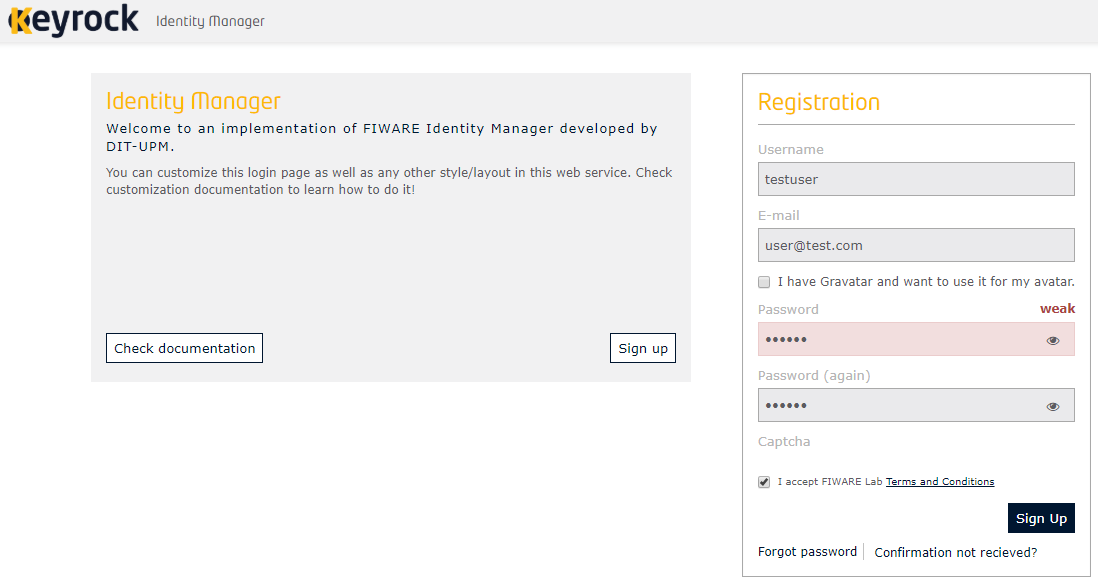
Bevor Wirecloud verwendet werden kann, muss erst dessen Datenbank initialisiert werden. Dafür müssen die folgenden zwei Befehle in einem weiteren PowerShell-Fenster ausgeführt werden:

***docker exec -it wirecloud /docker-entrypoint.sh initdb***

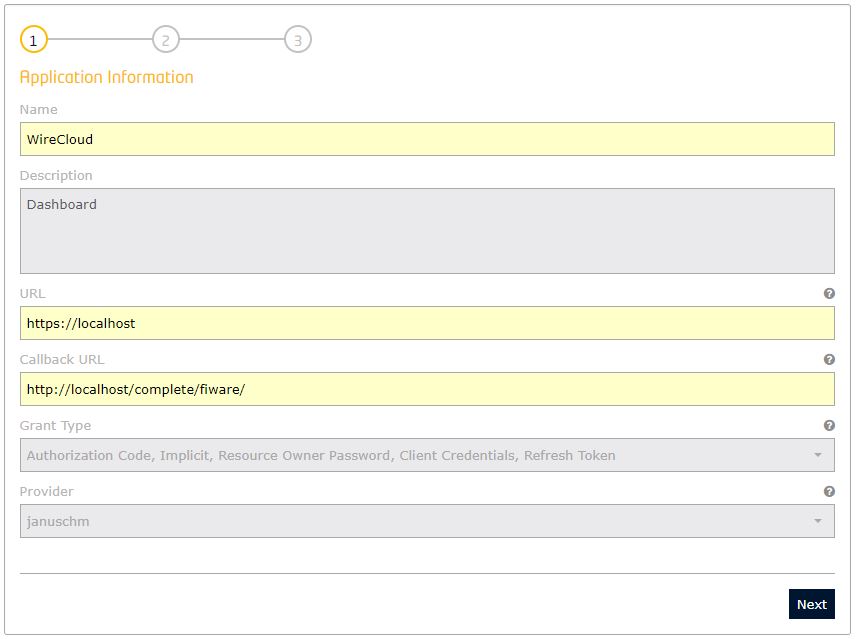
***docker exec -it wirecloud sh -c "echo yes | python manage.py collectstatic"***

Zusätzlich muss in der Datei „settings.py“ im Ordner „[…]DockerInfrastructure\Docker\  
wirecloud\_config“ die IP-Adresse des Keyrock-Servers angepasst werden. Dazu muss in der Zeile **FIWARE\_IDM\_SERVER = "http://192.168.1.139:5000"** die Adresse durch die IP des Hosts im lokalen Netzwerk ersetzt werden. Hierfür kann die IP des virtuellen Docker-Netzwerkadapters hergenommen werden:  
Der Port muss hierbei gleichbleiben.

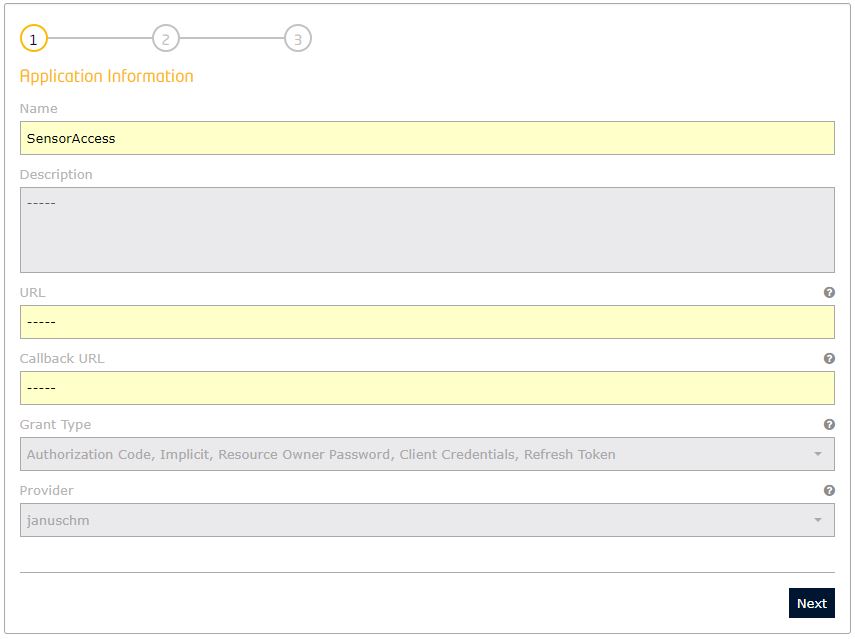
## Einrichtung der Security-Komponenten

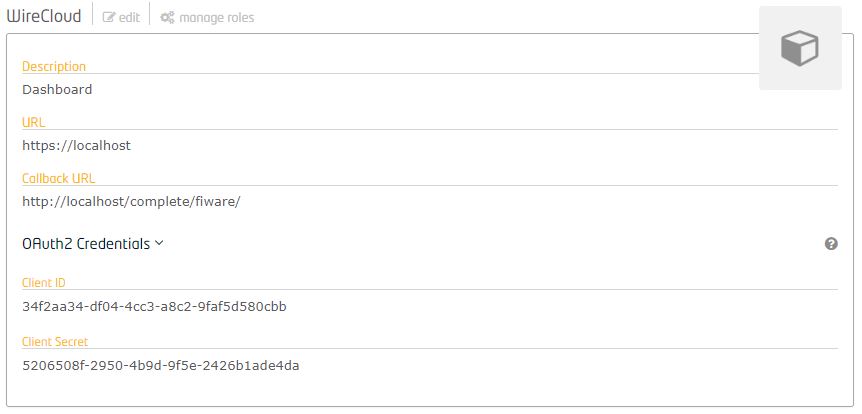
Im ersten Schritt muss ein Benutzeraccount im Keyrock-Portal angelegt werden. Die Identity-Manager-Komponente ist im Browser unter [**http://localhost:5000**](http://localhost:5000) erreichbar. Um einen neuen Benutzer anzulegen, muss in der Benutzeroberfläche der Button **Sign up** angeklickt werden. Im nun erschienenen Formular müssen die Benutzerdaten inklusive einer gültigen Emailadresse eingetragen werden:

Nach Bestätigung der Eingabe wird ein Email mit einem Aktivierungslink vom Keyrock-Server (Absenderadresse: [fiwaregraz@gmail.com](mailto:fiwaregraz@gmail.com)) an die angegebene Mailadresse des Benutzers gesendet. Nach Anklicken des Aktivierungslinks kann der neue Benutzer im Keyrock-Portal verwendet werden.  
Wichtig: Der Aktivierungslink ist nur auf dem lokalen Host einsetzbar, da der Keyrock-Server nicht aus dem Internet erreichbar ist.

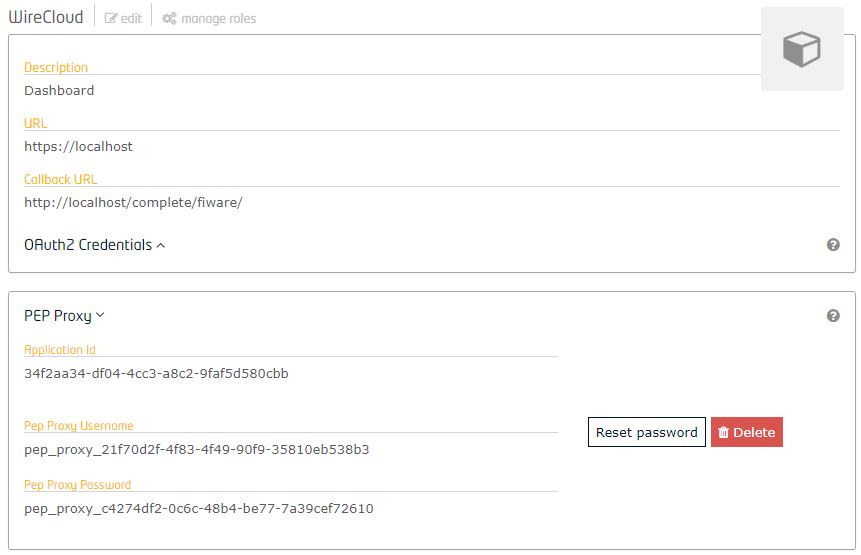
Als nächstes müssen im Keyrock-Portal zwei Applications angelegt werden: eine Application, die den Zugriff von Wirecloud auf die FIWARE-Komponenten IDAS, ORION und COMET regelt und eine zweite Application, welche die Übertragung der Sensorwerte absichert. Um eine neue Application anzulegen muss im Home-Screen des Portals bei der Applications-Liste auf **Register** geklickt werden. Für die Wirecloud-App muss das Registrierungsformular folgenderweise ausgefüllt werden:

Die Felder „Name“ und „Description“ können beliebig ausgefüllt werden, in „URL“ und „Callback URL“ müssen allerdings die Werte **https://localhost** und **http://localhost/complete/fiware/** unverändert übernommen werden. Die nachfolgenden Einrichtungsseiten können unverändert mit **Next** bzw. **Finish** bestätigt werden.

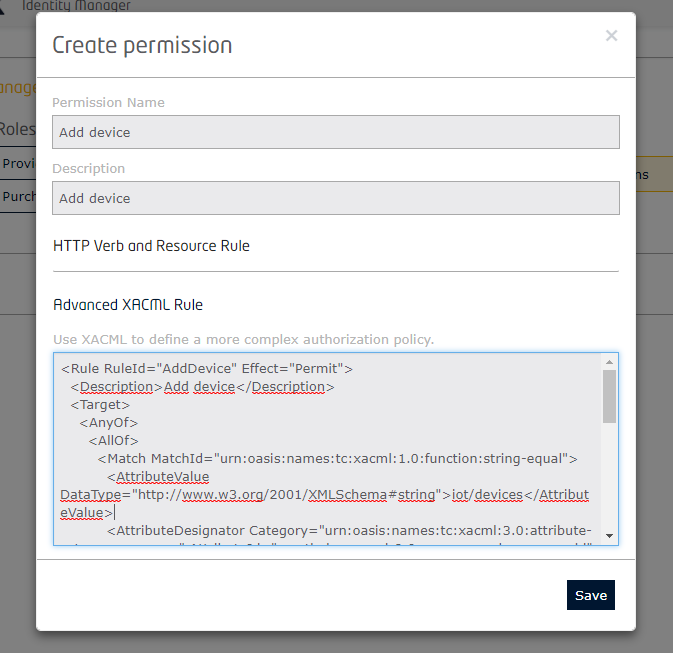
Für die zweite Application ist der Vorgang identisch, allerdings sind dabei die Felder „URL“ und „Callback URL“ irrelevant und können mit Platzhaltern befüllt werden:

Nachdem die Applications erfolgreich angelegt wurden, müssen die Credentials für das Wirecloud-Portal und die PEP-Proxies eingerichtet werden. Für die weitere Konfiguration des Wirecloud-Apps muss in der Applications-Liste auf dem Home-Screen der entsprechende Eintrag für Wirecloud angeklickt werden. Die für das Wirecloud-Portal benötigten Credentials sind unter dem Abschnitt „OAuth2 Credentials“ zu finden:

Die Werte für „Client ID“ und „Client Secret“ sind in der bereits zuvor bearbeiteten Datei „settings.py“ (im Ordner „[…]DockerInfrastructure\Docker\wirecloud\_config“) entsprechend bei den Variablen „SOCIAL\_AUTH\_FIWARE\_KEY“ und „SOCIAL\_AUTH\_FIWARE\_SECRET“ einzutragen.

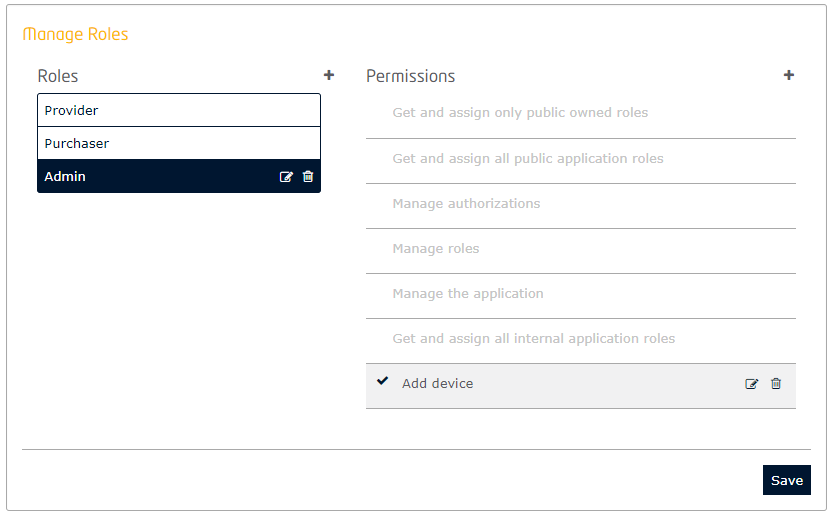
Um die Credentials für die PEP-Proxies zu erhalten, muss im Abschnitt „PEP Proxy“ auf **Register a new Proxy** geklickt werden. Die erzeugten Werte für „Application Id“, „Pep Proxy Username“ und „Pep Proxy Password“ (zu sehen in der nachfolgenden Abbildung) müssen in den „config.js“-Dateien in den Ordnern „[…]DockerInfrastructure\Docker\pep-orion\_config“ , „[…]DockerInfrastructure\Docker\pep-comet\_config“ und „[…]DockerInfrastructure\Docker\pep-idas-admin\_config“ für die Variablen „app\_id“, „username“ und „password“ eingetragen werden.

Da für die Schnittstelle, welche die gesendeten Sensorwerte empfängt, ein eigener PEP-Proxy benötigt wird, muss der zuvor durchgeführte Schritt mit der zweiten Application nochmals durchgeführt werden. Die so gewonnenen Credentials müssen nun in die Datei „config.js“ im Ordner „[…]DockerInfrastructure\Docker\pep-idas-sensor\_config“ nach demselben Schema wie zuvor eingetragen werden.

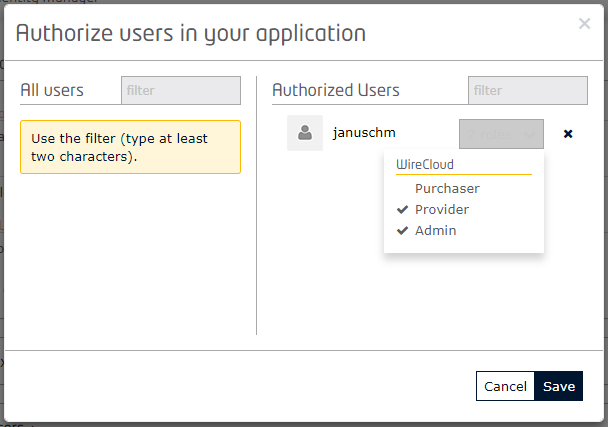
Nachdem nun die grundlegende Einrichtung der Security-Komponenten abgeschlossen ist, müssen noch entsprechende Berechtigungen für den zu Beginn angelegten Benutzer angelegt werden. In der Überblicksansicht der Wirecloud-Application (zu sehen in der vorherigen Grafik) muss der Button **manage roles** im oberen Bereich der Ansicht angeklickt werden. Durch einen Klick auf das Plus-Symbol im Permissions-Abschnitt kann nun eine neue Berechtigung angelegt werden. Im nun geöffneten Formular können Name und Beschreibung der Berechtigung sowie die entsprechende HTTP-Request-Methode und Ressource konfiguriert werden. Da die Einstellungen unter „HTTP Verb and Resource Rule“ für manche Berechtigungen nicht komplex genug sind, werden die benötigten Regeln in dieser Anleitung ausschließlich im XACML-Format über den Abschnitt „Advanced XACML Rule“ erstellt:

Mit einem Klick auf Save wird die jeweilige Berechtigung gespeichert. Die Dateien mit allen Regeln für die Wirecloud-App im XACML-Format befinden sich im Ordner „[…]\DockerInfrastructure\PermissionsXACML\WirecloudApp“.

Die erstellten Berechtigungen können nun beliebig mit Rollen verknüpft werden, die wiederum an Benutzer vergeben werden können. Eine neue Rolle kann durch einen Klick auf das Plus-Symbol im Roles-Abschnitt erstellt werden. Nach Vergabe eines Rollennamens und Speicherung der Rolle kann nach einem Klick auf den entsprechenden Eintrag ausgewählt werden, welche Berechtigungen die jeweilige Rolle haben soll:

Der Einfachheit halber werden in dieser Anleitung alle Berechtigungen der Rolle „Admin“ zugeordnet.

Nachdem die Zuteilung der Berechtigungen abgeschlossen ist und die Rolle gespeichert wurde, kann in der Überblicksansicht der Wirecloud-App die neue Rolle dem Benutzer zugeteilt werden. Dafür muss zuerst im Abschnitt „Authorized Users“ auf den Button **Authorize** geklickt werden. Im nun eingeblendeten Fenster können nun für alle autorisierten Benutzer entsprechend Rollen vergeben werden:



Die Einrichtung der Wirecloud-App ist damit abgeschlossen. Die Schritte zum Erstellen von Permissions, sowie zum Anlegen und Zuweisen von Rollen müssen nun noch analog für die zweite Application durchgeführt werden. Die Dateien mit den Regeln (in diesem Fall nur eine) für die zweite App im XACML-Format befinden sich im Ordner „[…]\DockerInfrastructure\PermissionsXACML\ SensorAccessApp“.

Sind all diese Konfigurationsschritte abgeschlossen, muss noch die Docker-Infrastruktur mit folgenden Befehlen (über die PowerShell-Konsole aus dem Ordner „[…]DockerInfrastructure\  
Docker“) neugestartet werden:

***docker-compose stop***

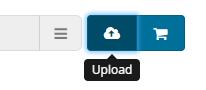
***docker-compose up***

## Einrichtung des Dashboards

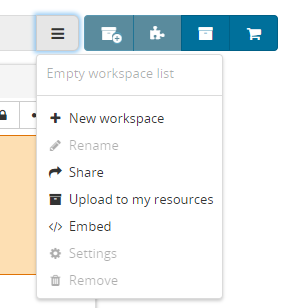
Nachdem nun die Basiskomponenten der FIWARE-Infrastruktur fertig konfiguriert sind, muss das Dashboard für die Verwaltung der Datenbankobjekte und die Darstellung der Sensorwerte eingerichtet werden. Dazu muss als Erstes das Wirecloud-Portal unter der Adresse [**https://localhost**](https://localhost) im Browser geöffnet werden. Nach einem Klick auf den Link **Sign in** in der rechten oberen Ecke kann der zuvor angelegte Benutzer über die Keyrock-Oberfläche eingeloggt werden. Bei der ersten Anmeldung muss noch die Kommunikation zwischen Wirecloud und Keyrock durch einen Klick auf **Authorize** autorisiert werden:

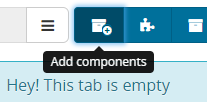
Nach der erfolgreichen Anmeldung wird der Workspace-Browser des jeweiligen Benutzers angezeigt. Über einen Klick auf den Button **My Resources** (siehe nachfolgende Grafik) gelangt man nun in die Übersicht der eigenen Ressourcen, wo die benötigten Widgets und Operatoren für das Dashboards hochgeladen werden können.

Durch einen Klick auf Upload (siehe nächste Abbildung) wird das Upload-Fenster geöffnet, von dem aus Ressourcen vom Host-Rechner hochgeladen werden können.



Die benötigten Dateien für das Dashboard befinden sich im Ordner „[…]DockerInfrastructure\Mashup“.

Nachdem der Upload der fünf Dateien erfolgreich abgeschlossen wurde, muss im Workspace-Browser über **New workspace** (siehe nachfolgende Grafik) ein neuer Workspace erstellt werden. Die Benennung des Workspace ist frei wählbar.

In der Basis-Ansicht des neu erstellten Workspace kann nun durch einen Klick auf **Add components** (zu sehen in nachfolgender Abbildung) das hochgeladene Mashup importiert werden.

Unter dem Reiter „Mashups“ muss nun das Mashup „Dashboard“ über das Plus-Symbol hinzugefügt werden. War der Vorgang erfolgreich, sollten am unteren Rand der Workspace-Ansicht neue Tabs dazugekommen sein („Karte“, „Diagramme“, etc.). Durch einen Klick auf dem jeweiligen Tab kann die entsprechende Dashboard-Ansicht angezeigt und mit den Widgets interagiert werden.

Für das Anlegen der benötigten Datenbankobjekte muss als nächstes auf den Tab „Batch Add“ gewechselt werden. Mit dem dort vorhandenen Widget können mehrere Datenbankobjekte gleichzeitig per JSON-Eingabe erstellt werden. Dafür muss nur die jeweilige JSON-Struktur in das leere Textfeld des Widgets kopiert und danach auf den Button **Send** geklickt werden. Die Dateien mit den Objekten im JSON-Format sind im Ordner „[…]DockerInfrastructure\DBObjectsJSON“ zu finden.  
Wichtig: Die Datenbankobjekte müssen immer in der Reihenfolge **Services – Entities – Devices – Subscriptions** erstellt werden. Wird diese Reihenfolge nicht eingehalten, werden unter Umständen fehlende Objekte intern automatisch generiert, wodurch es bei der Objekterstellung mit dem Batch-Add-Widget zu Komplikationen kommen kann.

Nach der Erstellung der Datenbankobjekte kann über die Widgets im Tab „Admin Tools“ überprüft werden, ob alle Objekte ordnungsgemäß erstellt wurden. Auch das Ändern und Löschen von Objekten sowie das Anlegen neuer Datenbankobjekte kann über diese Widgets durchgeführt werden.

## Erstellen von Sensorwerten

Um auch ohne echte Sensoren die FIWARE-Infrastruktur mit Sensorwerten versorgen zu können, werden künstliche Sensorwerte mithilfe eines Sensor-Mock-Scripts an den Server übertragen.

Dafür muss als erstes die Konfigurationsdatei des Skripts angepasst werden. In der Datei „config\_secure.ini“ im Ordner „[…]DockerInfrastructure\SensorMockScript“ müssen die Variablen „client\_id“ und „client\_secret“ mit den entsprechenden Werten aus dem Abschnitt „OAuth2 Credentials“ der Überblicksansicht von der Application für die Sensorschnittstelle versehen werden.

Nun kann ein PowerShell-Fenster im Ordner „[…]DockerInfrastructure\SensorMockScript“ geöffnet werden, aus dem mit dem folgenden Befehl Messwerte vom Sensor „Tram\_1“ an den Server übermittelt werden:

***python .\SendCombinedValuesSecure.py Dev\_Tram\_1 apimobile <user> <password> .\tram\_1-values.txt***

Die Platzhalter **<user>** und **<password>** müssen entsprechend durch die Mailadresse und das Passwort des angelegten (und in der Application autorisierten) Benutzers ersetzt werden. Das Skript sendet nun alle fünf Sekunden einen (in der Datei „tram\_1-values.txt“ hinterlegten) Messwert an den Server. Um das Skript zu stoppen, muss nur die Tastenkombination **Strg+C** getätigt werden.

Die Befehle für die restlichen Sensoren sind in der Datei „Commands.txt“ im selben Ordner zu finden.