Smarthome Dokumentation

M426

Tillo Razzai, Lucas Lange, Danilo Jakob, David Chin, Stefan Ninkovic

2020

Inhalt

[Definition 2](#_Toc52401910)

[Aufgabe 2](#_Toc52401911)

[Ziele 2](#_Toc52401912)

[Projekt-Zusammenfassung 2](#_Toc52401913)

[Backend 2](#_Toc52401914)

[Projekte 2](#_Toc52401915)

[Common.Data 2](#_Toc52401916)

[Business.Api 2](#_Toc52401917)

[Business.Authentication 2](#_Toc52401918)

[Communication.Web 2](#_Toc52401919)

[Communication.Database 2](#_Toc52401920)

[Communication.Raspberry 2](#_Toc52401921)

[Authentifizierung 3](#_Toc52401922)

[Repositorien 3](#_Toc52401923)

[Raspberry Kommunikation 4](#_Toc52401924)

[Frontend 5](#_Toc52401925)

[Dashboard 5](#_Toc52401926)

[Design 5](#_Toc52401927)

[Marketing 6](#_Toc52401928)

[Glossar 7](#_Toc52401929)

[Quellen 7](#_Toc52401930)

# Definition

## Aufgabe

Im Modul 426 lernen wir alles zum Thema SCRUM und einer der Leistungsbeurteilungen, die wir im Rahmen dieses Moduls haben ist mittels SCRUM ein anspruchsvolles Projekt zu bewältigen.

## Ziele

Die Generellen Ziele sind unteranderem, dass wir erste Erfahrungen mit SCRUM sammeln und lernen, wie wir besser im Team kommunizieren können.

## Projekt-Zusammenfassung

Wir haben uns dazu entschieden eine Applikation zu realisieren welche es ermöglicht, Smart-Gadgets wie z.B. eine Lampe. Wenn die Lampe nun verbunden ist, wird man zunächst nur im Stande sein das Gerät einzuschalten bzw. Auszuschalten. Diese Interaktion haben wir vor über einer Website zu realisieren. Zu der eigentlichen Applikation wird es noch eine Marketing-Website geben, welche kurz und bündig erklärt wofür das Programm existiert etc.

# Backend

Das Backend wird grundsätzlich mit der Programmiersprache C-Sharp realisiert und beinhaltet alle wichtigen Funktionen wie z.B. die Interaktion mit den Smart-Gadgets.

## Projekte

Hier wird erklärt welche Projekte das Backend beinhaltet.

### Common.Data

Alle Daten-Klassen werden hier abgespeichert ganz egal welche Funktion es hat.

### Business.Api

Alle verschiedenen Business-Projekte können gewisse Schnittstelle liefern. Diese Schnittstellen werden hier abgespeichert damit alle Projekte (abgesehen von [Common.Data](#_Common.Data)) dieses Projekt referenzieren können.

### Business.Authentication

Dieses Projekt liefert wichtige Klassen für die Authentifizierung eines Benutzers.

### Communication.Web

Die Website, welche es ermöglicht sich einzuloggen, die Geräte zu bedienen etc. wird hier festgehalten. Dieses Projekt beinhaltet auch alle Bootstrap-Dateien

### Communication.Database

Damit die Applikation mit der Mongo-DB Datenbank interagieren kann beinhaltet Communication.Database wichtige Klassen.

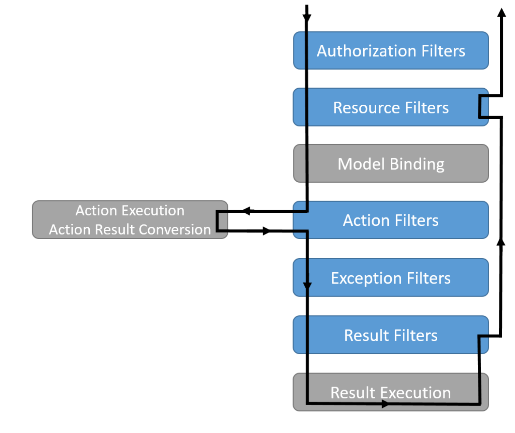
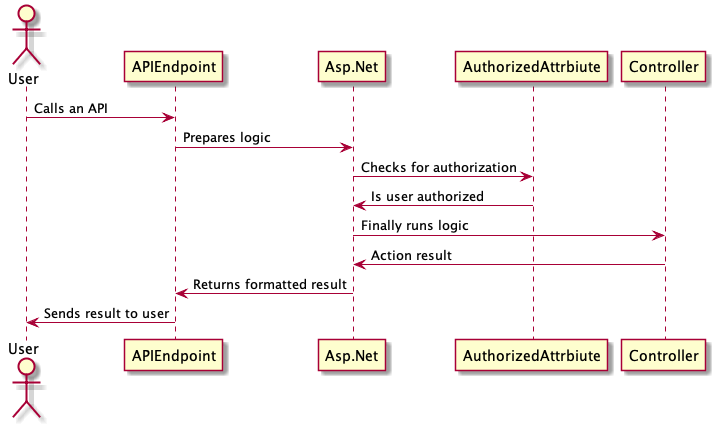
### Communication.Raspberry

Der Raspberry liefert geeignete Schnittstellen, wo man gewisse Pins einstecken kann. Diese Pins verbinden in unserem Fall die verschiedenen Smart-Gadgets mit dem Backend. Die Logik für die Kommunikation wird in diesem Projekt behandelt.

## Authentifizierung

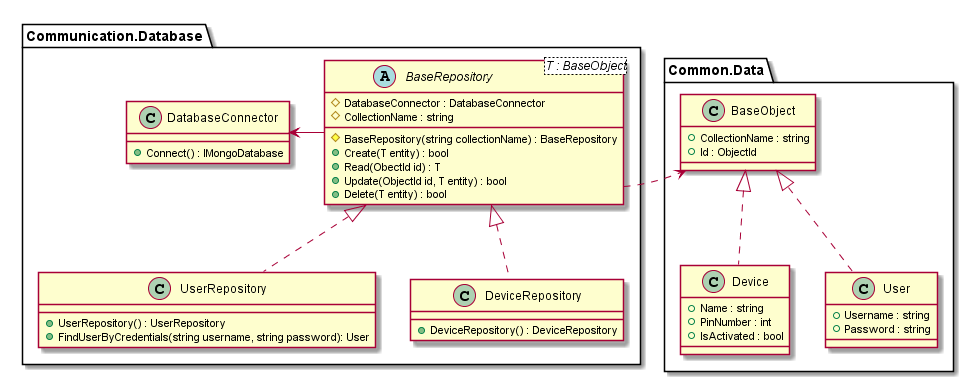
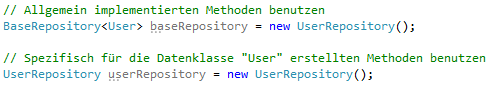
Im Backend gibt es zwei verschiedene Arten an Controllern. Controller, welche für anonyme Benutzer zugänglich sind und Controller, welche nur für authentifizierte ansprechbar sind. Damit nicht in jedem einzelnen Controller welche kritische Logik beinhaltet ein If-Statement vorhanden ist, wurde ein Attribut erstellt welcher vor jedem Request vom User überprüft, ob der jeweilige Controller dieses Attribut beinhaltet oder nicht. Die Authentifizierungs-Logik ist im [Business.Authentication](#_Business.Authentication) auffindbar.





## Repositorien

Sogenannte Repositorien sind dafür zuständig einen geordneten Zugang zu der Datenbank zu erstellen. Die Repositorien verbinden sich mit der Datenbank, ändern diese ab bzw. lesen etwas heraus, und schliessen die Verbindung zum Schluss. Für jede Datenklasse in [Common.Data](#_Common.Data) gibt es ein dazugehöriges Repository welche nur diese Datenklasse alleine behandelt. Alle Repositorien erben von einem ***BaseRepository*** welche grundsätzlich allgemeine Logik für alle Datenklassen implementiert.



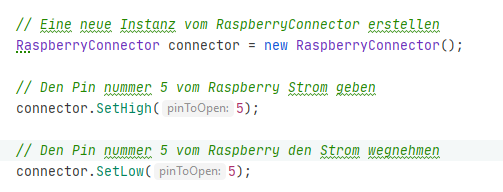
## Raspberry Kommunikation

Die Klasse ***RaspberryConnector*** wird von der Applikation benötigt, damit man die GPIO Pins, welche sich auf den Raspberry PI 3 befinden über die Web-Applikation ansteuern kann. Die Klasse findet man im Projekt [Communication.Raspberry](#_Communication.Raspberry).

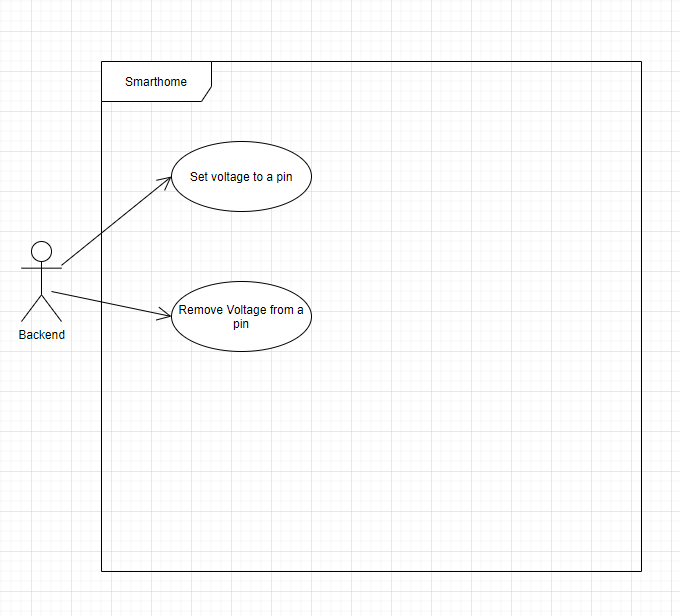
Dafür hat man folgende Methoden zu Verfügung:

* SetHigh – Man kann einem Pin-Strom geben
* SetLow – Man kann einem Pin den Strom wegnehmen

Diese ruft man wie folgt auf:



Der ***RaspberryConnector*** kann auch als Klassenattribut gesetzt werden und über den Konstruktor instanziiert werden.

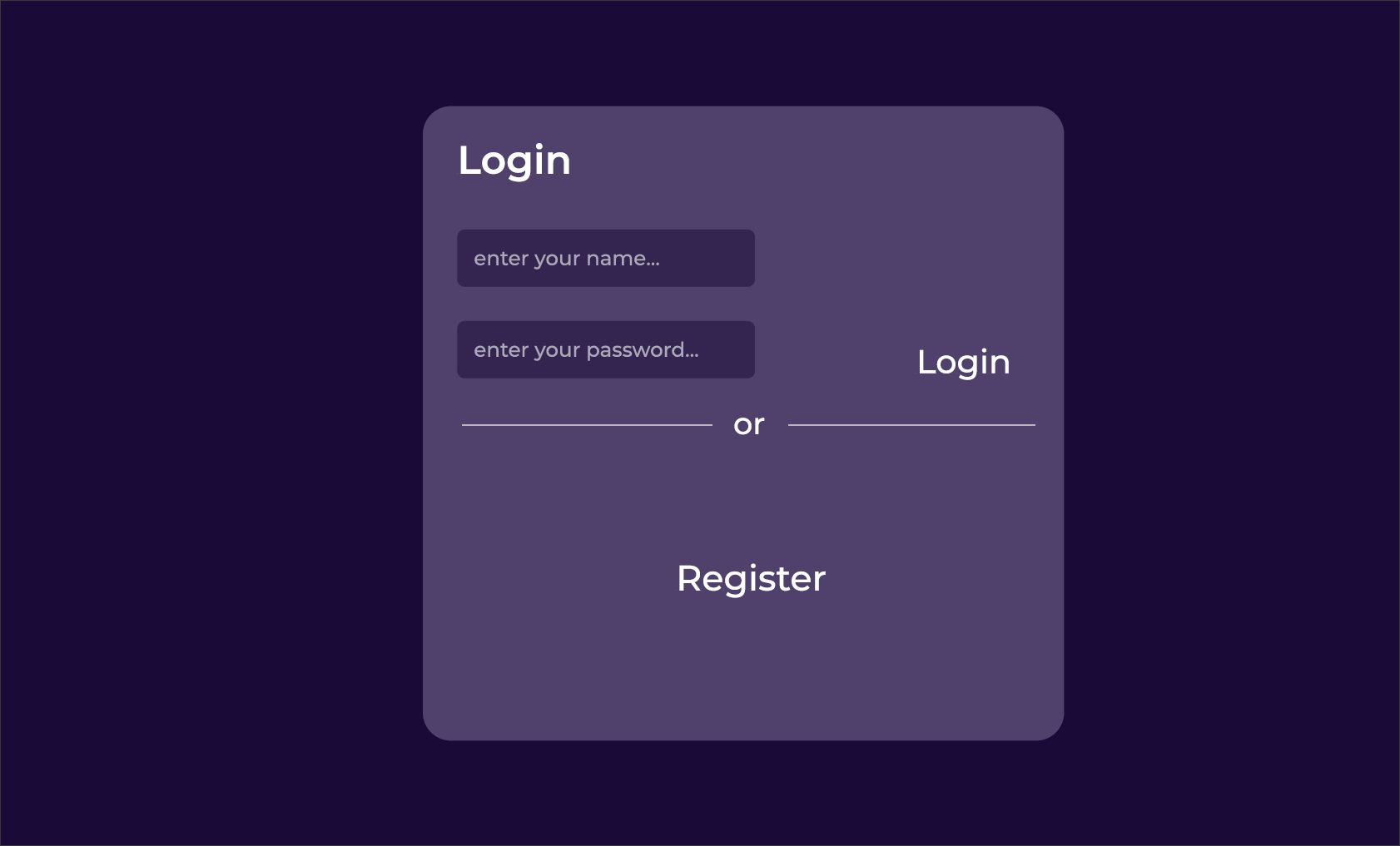
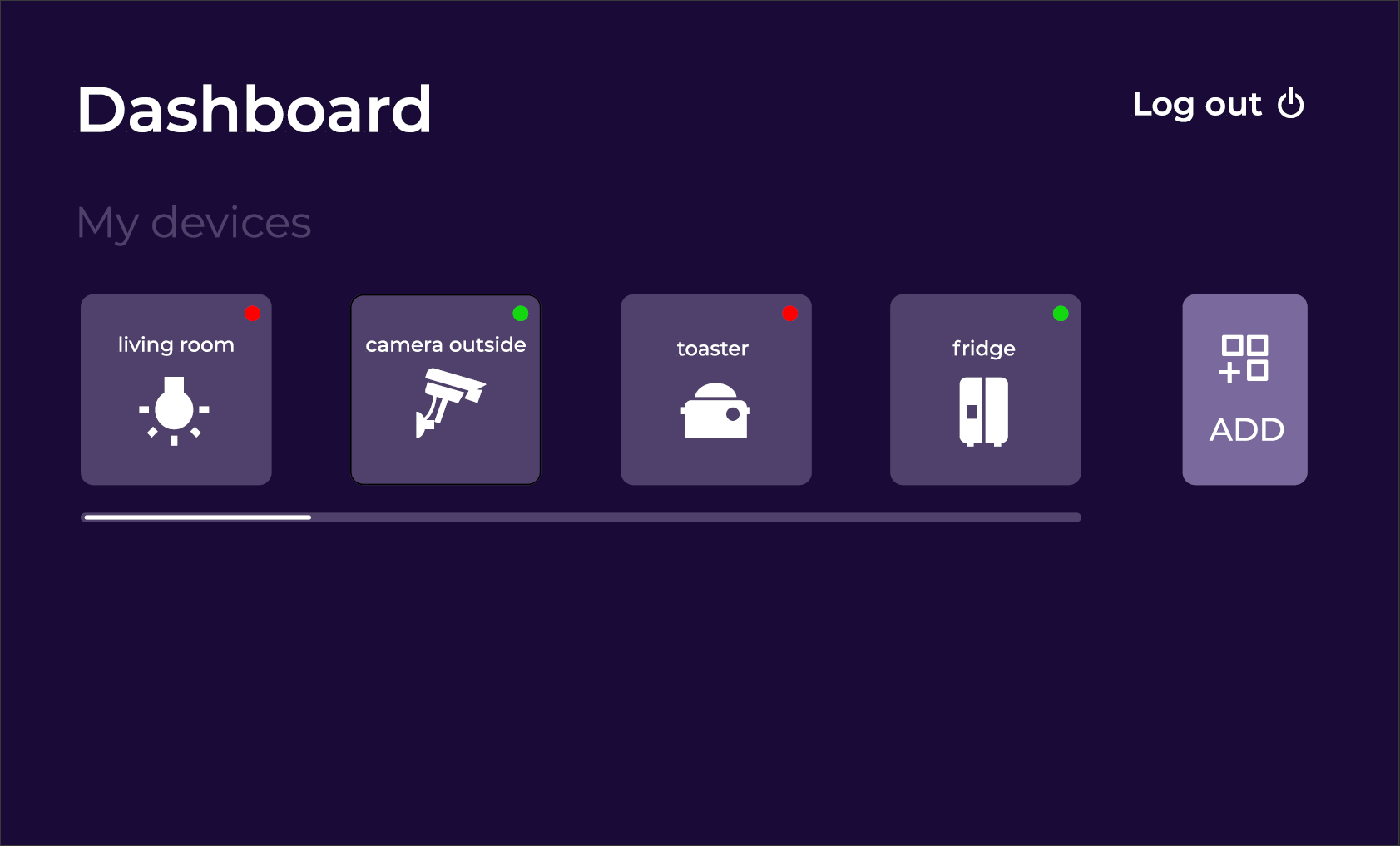
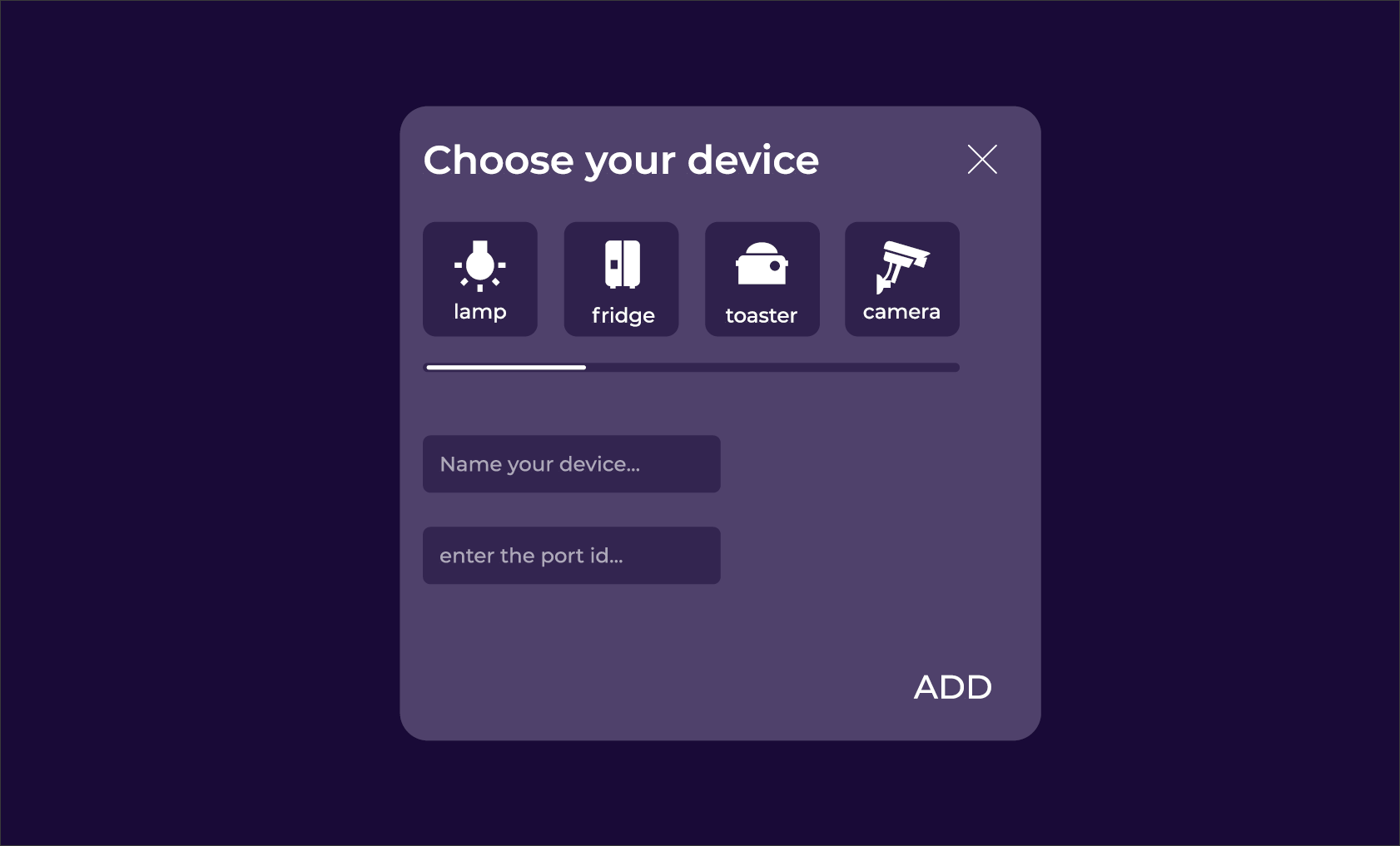


# Frontend

## Dashboard

Das Frontend für das Dashboard wird grundsätzlich mit der Technologie Razor realisiert und beinhaltet alle wichtigen Komponenten, um das Programm genau zu benutzen.

### Design



## Marketing

Es wurde eine Marketing-Website erstellt, welche unser Produkt genauer erklärt und etwas Werbung erstellt. Die Website URL lautet: <https://www.iage-luc.ch/smarthome/>

# Glossar

|  |  |
| --- | --- |
| Name | Erklärung |
| Git | Git ist eine kostenlose und open source erstelltes version control system. |
| Projekte | Projekte sind in C-Sharp «Ordner», welche benutzt werden, um Klassen mit ähnlichem Ursprung oder Kontext nah beieinander zu halten. Zusätzlich noch kann man so gewisse Sichtbarkeit einschränken kann gegenüber den anderen Projekten. In Java gibt es ein ähnliches Konzept namens packages. |
| Dashboard | Wird auf der internen Website vom Programm erstellt und wird benutzt, um eine Schnittstelle von User und backend zu erstellen. |
| Backend | Ist das hintere Teil vom Programm. Die gesamte Logik findet hier stat. |
| Frontend | Ist das vordere Teil vom Programm. Das Frontend ist dazu da eine visuelle Schnittstelle für den User bereitzustellen. Mögliche Frontend-Technologien sind Desktop-programme, Websites, Handy-Apps etc. |
| Raspberry | Der Raspberry Pi ist ein Einplatinencomputer, der von der britischen Raspberry Pi Foundation entwickelt wurde. |

# Quellen

|  |  |
| --- | --- |
| Name | Link |
| Request-Filter | <https://docs.microsoft.com/en-us/aspnet/core/mvc/controllers/filters?view=aspnetcore-3.1> |
| Razor Cookie | <https://www.learnrazorpages.com/razor-pages/cookies> |
| Raspberry Wikipedia | <https://de.wikipedia.org/wiki/Raspberry_Pi> |
| Git | <https://git-scm.com/> |
| Git-Repository | <https://github.com/Team-Taucher/Smarthome> |