

Magic Mirror

Florian Vogel

11. März 2019

Inhaltsverzeichnis

1	Einführung	4
2	Zielsetzung	5
3	Ausgangslage	6
3.1	Bewertungskriterien Opensource Projekt	6
3.1.1	Modularität und Erweiterbarkeit	6
3.1.2	Einarbeitungszeit	7
3.1.3	Hardwareeinsatz	7
3.2	Designentscheid	7
4	Aufbau	8
4.1	Auswahl Hardware	8
4.1.1	Bildschirm	9
4.1.2	Hauptrechner	9
4.1.3	Infrarot Präsenzmelder	10
4.1.4	Hilfsrechner	10
4.1.5	WLAN Router	11
4.1.6	Laptop	11
4.1.7	Handy	11
4.2	Auswahl Software	11
4.2.1	Raspian auf Raspberry Pi	12
4.2.2	MagicMirror ²	12
4.2.3	MQTT Broker	12
4.2.4	Node-Red	12
4.2.5	Android Applikation	12
4.2.6	ESP-IDF Programm	12
5	Magic Mirror auf dem Raspberry Pi	13
5.1	Aufsetzen von MagicMirror ²	13
5.2	Integration Node-Red und MQTT	14
5.3	Implementation eigenes MagicMirror ² Modul	15
6	Applikation für Präsenzdetektion	15
7	Android Applikation für Notizen	15

Glossar

Bash Befehls Sprache im UNIX Umfeld. 13

DVI Digital visual interface, auf deutsch Digitale visuelle Schnittstelle zum übertragen von Videodaten. 9

HDMI High definition multimedia interface, auf deutsch hochauflösende Multimedia-Schnittstelle. 9

LCD Liquid crystal display, auf deutsch Flüssigkristallanzeige. 9

MQTT Message Queuing Telemetry Transport, ein offenes Nachrichtenprotokoll um zwischen Maschinen Nachrichten auszutauschen. 2, 12, 14, 15

Node-Red Tool um verschieden Hardware zusammen zu verknüpfen. 2, 12, 14

Opensource Software die von Dritten eingesehen, gelesen und bearbeitet werden kann. 2, 6, 7, 9, 13

pm2 Es handelt sich um einen node.js Prozess Manager. 13

SD Karte Secure Digital Memory card, auf deutsch sichere digitale Speicherkarte. 7, 13

SSID Service set identifier, Name eines WLAN Netzwerks. 11

URL Uniform resource locator, Identifikation und Lokalisation einer bestimmten Ressource. 14

VGA Video graphics array, Schnittstelle für Bildübertragung. 9

WLAN Wireless local area network, auf deutsch drahtloses lokales Netzwerk. 2, 9–12

1 Einführung

Ein Magic Mirror ist ein optisch ansprechendes Anzeigegerät. Es handelt sich um ein Spiegel mit integriertem Bildschirm, wobei es sich bei dem Spiegel um einen sogenannten Spionspiegel handelt. Er ist von einer Seite möglichst reflektierend und von der anderen Seite möglichst durchlässig. Mit dem verbauten Bildschirm ergeben sich beinahe unbegrenzte Möglichkeiten um Informationen zu präsentieren und diese ansprechend darzustellen. Dadurch passt ein Magic Mirror mit passendem Design gut in einen Wohnbereich.

Nun, was soll denn auf solch einem Spiegel angezeigt werden? Natürlich gibt es einige Klassiker, wie zum Beispiel die aktuelle Zeit. Die Möglichkeiten lassen aber viel mehr zu. Es ist beispielsweise auch denkbar jeweils den nächsten Zug von Bern nach Zürich auf dem Magic Mirror anzuzeigen.

Die Interessen für Informationen werden sich mit Sicherheit ändern über die Zeit. Das bedingt eine Konfigurationsmöglichkeit für den Benutzer des Spiegels, mit welcher er anzeigende Informationen ändern kann. Diese genannte Modularität zu erreichen ist eines der Ziele in dieser Semesterarbeit. Weiter soll am Ende ein funktionierender Prototyp eines Magic Mirrors vollendet sein, welcher im Heimbereich eingesetzt wird.

2 Zielsetzung

Das erste und am höchsten gewichtete Ziel ist das Erstellen eines fertigen Prototypes. Dies wird in folgende Teilschritte unterteilt.

- Einfache Anzeigeelemente auf dem Spiegel, wie beispielsweise die Uhrzeit, das Wetter oder einen Kalender.
- Energiesparmodus, dabei wird der Bildschirm über einen externen Infrarotsensor ein- und ausgeschaltet.
- Modul auf welches über mobile Applikation zugegriffen werden kann.
- Eine Android Applikation um auf obengenanntes Modul zugreifen zu können.
- Sprachsteuerung welche vom Google Assistant gemacht wird.

Die genannte Punkte werden schrittweise umgesetzt. Somit hat der letzte Punkt die niedrigste Priorität und wird als optional erachtet.

3 Ausgangslage

Die Idee des Magic Mirror ist bekannt. Es gibt bereits zahlreiche Versionen davon online zum Nachbau. Es gibt Vorschläge für das Spiegelglas, den Bildschirm, die Recheneinheit, sogar wie der Rahmen des Spiegels aufgebaut werden kann. Deshalb ist es grundsätzlich einmal notwendig, in diesem Dschungel von Ideen eine gute Zusammensetzung zu finden.

Sehr prominent tritt dabei ein Opensource Projekt auf, welches sich MagicMirror² nennt [2]. Es ist das einzig verfügbare Projekt, welches modular aufgebaut ist und zudem von einer grossen Community unterstützt wird. Deshalb muss als erstes der Einsatz dieses Projekts abgewogen werden.

3.1 Bewertungskriterien Opensource Projekt

Um zu evaluieren, ob mit dem Opensource Projekt Mein Standort MagicMirror² weitergearbeitet werden kann oder ein komplett eigenständiger Ansatz gewählt werden muss, werden folgende Bewertungskriterien genauer analysiert:

- Modularität und Erweiterbarkeit
- Einarbeitungszeit
- Hardwareeinsatz

Die folgenden drei Unterkapitel behandeln kurz diese Punkte jeweils für das Opensource Projekt, sowie die komplette Eigenentwicklung. Dabei werden Bewertungen angegeben zwischen 1-10, wobei 10 sehr gut ist.

3.1.1 Modularität und Erweiterbarkeit

Die Startseite des Opensource Projekts wirbt bereits mit grosser Modularität. Wie gross diese ist, zeigt sich durch die Liste der bereits implementierten Erweiterungen, welche auch auf dem Magic Mirror angezeigt werden können. Bereits mehrere hundert solcher Zusatzmodule können eingebunden werden [3].

Die Erweiterbarkeit ist ebenfalls gut, da die Einbindung der einzelnen Module über eine zentrale Stelle (eine Konfigurationsdatei) gemacht werden kann. Bewertung: 8 → keine Informationen darüber, wie gut Module getestet wurden

Bei einer Eigenentwicklung muss von Beginn klar die Modularität und Erweiterbarkeit berücksichtigt werden. Dies ist gut machbar, da es beim Designen noch keine weiteren Abhängigkeiten gibt. Jedoch ist dies mit Zusatzaufwand verbunden.

Bewertung: 5 → die Modularität zu gewährleisten bedeutet aber ein grosser Mehraufwand

3.1.2 Einarbeitungszeit

Das Opensource Projekt rühmt sich damit, auch für Leute ohne nennenswerten technischen Hintergrund einsetzbar zu sein. Diese Aussage bestätigt sich, denn innerhalb einer Stunde ist es möglich, das Projekt auf einem vorbereiteten Raspberry Pi (Raspian auf SD Karte) lauffähig zu haben und mit der Grundkonfiguration zu betreiben. Deshalb wird die Einarbeitungszeit als klein eingeschätzt.

Bewertung: 8 → da Einarbeitung in Modulbearbeitung noch dazukommt

Bei der Eigenentwicklung verhält sich die Einarbeitungszeit unterschiedlich je nach verwendeter Methode. Was jedoch sicherlich stark ins Gewicht fällt ist die Konzeptfindung, welche vom Vorwissen abhängt. Dadurch kann die Einarbeitungszeit minimiert werden, jedoch steigt damit sicherlich die Zeit der Konzeptphase.

Bewertung: 6 → Konzeptphase kann sehr lange dauern

3.1.3 Hardwareeinsatz

Die verwendete Hardware des Opensource Projekts ist nicht definiert. Das Projekt wurde adaptiert um auf Smartphones und Tablets zu laufen, ist aber auch auf einem Raspberry Pi 2 oder Raspberry Pi 3 verwendbar. Diese Portabilität ist optimal, da in dieser Semesterarbeit eine low-cost Hardwareplattform verwendet werden soll.

Bewertung: 8 → da es auf den meisten gängigen Plattformen läuft, aber nicht auf allen (z.b. Probleme beim Raspberry Pi 1)

Bei einer Eigenentwicklung ist der Einsatz der Hardware frei und kann eingeplant werden. Eine low-cost Hardwareplattform zu verwenden stellt kein Problem dar.

Bewertung: 10 → Projekt kann auf Hardware angepasst werden

3.2 Designentscheid

Anhand der Bewertung aus den vorangehenden Kapiteln fällt der Entscheid klar zu Gunsten vom Opensource Projekt aus. Das bedeutet, dass als Plattform für die Software das MagicMirror² Projekt verwendet wird. Von dieser Grundlage aus, werden die nötigen Anpassungen und Erweiterungen gemacht in den einzelnen Modulen.

4 Aufbau

Unter einem Kapitel mit dem Namen Aufbau können mehrere Erwartungshaltungen zutreffen. Zum einen sicherlich der Aufbau des Magic Mirrors aus Sicht der verwendeten Materialien und Baugruppen. Zum anderen, und das ist für diese Semesterarbeit das Relevante, der Aufbau der Software und deren Zusammenspiel mit der Hardware. Um jedoch das erstgenannte nicht komplett aussenvor zu lassen, hier ein Sinnbild wie der Magic Mirror aufgebaut wird. Darauf zu sehen ist der benötigte Bildschirm, das Spionglas und ein Rahmen.



Abbildung 1: Bauelemente für den Magic Mirror [2]

4.1 Auswahl Hardware

Die Auswahl der Hardware beinhaltet mehrere Einzelteile. Die folgenden Unterkapitel gehen auf die verschiedenen Teile ein. Zur Veranschaulichung jedoch vorweg eine Übersicht über die Komponenten.

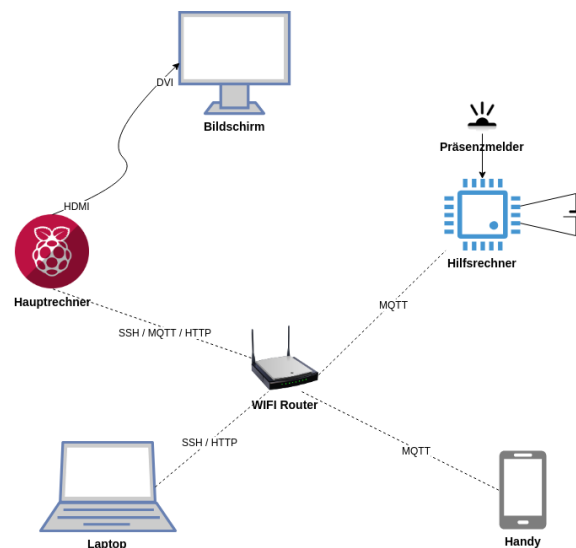


Abbildung 2: Hardware Komponentenübersicht

4.1.1 Bildschirm

Der Bildschirm ist für den Magic Mirror natürlich zentral. Jedoch ist bei der Auswahl nicht besonders viel zu beachten. Bei dieser Semesterarbeit wird ein 24 Zoll LCD Monitor verwendet, der einen DVI Anschluss hat und noch kein HDMI unterstützt. Die Auswahl fällt aufgrund des Projektbudgets auf diesen Monitor. Er war gratis verfügbar und ist nicht weiter in Gebrauch. Allfällige Inkompatibilitäten können durch Adapterkabel (HDMI → DVI) gelöst werden. Der Bildschirm wird durch ein Netzgerät mit 230 Volt betrieben.



Abbildung 3: 24"Monitor von Magic

4.1.2 Hauptrechner

Wichtig ist für den Hauptrechner, eine einfache Anbindung an das Heimnetzwerk zu ermöglichen. Weiter muss direkt eine Video Schnittstelle wie HDMI, DVI oder VGA vorhanden sein. Dies funktioniert alles mit einem Raspberry Pi 3 sehr gut, die Netzwerkverbindung sogar auch kabellos mittels Onboard-WLAN Chip. Weiter wird das Opensource Projekt von einer grossen Community auf dem Raspberry Pi eingesetzt, was ein grosser Pluspunkt im Falle von Fragen mit dem Projekt ist. Ein weiterer Pluspunkt des Raspberry Pi's ist die Verfügbarkeit und die tiefen Kosten. Diese Gründe machen ein Raspberry Pi 3 Model B zum Hauptrechner in dieser Semesterarbeit.



Abbildung 4: Raspberry Pi 3 Model B

4.1.3 Infrarot Präsenzmelder

Wie in der Zielsetzung erwähnt, soll der Magic Mirror möglichst Energieeffizient sein. Der grösste Energieverbrauch liegt sicherlich beim Betreiben des Monitors. Deshalb soll dieser automatisch ausschalten, wenn niemand zu Hause ist. Diese Detektion kann mit einem Infrarot Präsenzmelder gemacht werden. Aus Kostengründen wird der HC-SR501 eingesetzt. Dieser bietet die Möglichkeit, über ein Potentiometer die Sensitivität einzustellen um eine optimale Anpassung an den Einsatzraum zu gewährleisten. Der Melder funktioniert ab einer Speisespannung von 3.3V. Bei einer Detektion wird ein Puls am Ausgangspin generiert.

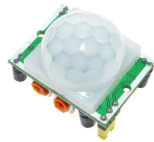


Abbildung 5: Infrarot Präsenzmelder HC-SR501

4.1.4 Hilfsrechner

Um den erwähnten Puls vom Infrarot Präsenzmelder zu verarbeiten und einen Befehl an den Magic Mirror zu senden, muss noch eine weitere Recheneinheit vorhanden sein. Natürlich könnte auch das Raspberry Pi diese Aufgabe übernehmen. Jedoch müsste dann der Präsenzmelder direkt über ein Kabel verbunden sein mit dem Magic Mirror. Dies ist nicht wünschenswert, da der Präsenzmelder entfernt vom Magic Mirror aufgestellt soll, um Bewegungen im Raum zu erkennen bevor der Betrachter den Magic Mirror sieht. Damit bleibt genügend Zeit um den Bildschirm zu starten und der Betrachter findet den aktiven Magic Mirror vor.

Diese Aufgabe der Verarbeitung von einem Puls erfordert keine aufwändigen Berechnungen. Wichtig ist aber, dass der Hilfsrechner eine Möglichkeit hat ins Netzwerk zu kommen um mit dem Magic Mirror zu kommunizieren. Der ESP32 erfüllt diese Anforderung sehr gut, da er einen WLAN Chip integriert. Weiter ist das Board sehr kostengünstig erwerbbar.

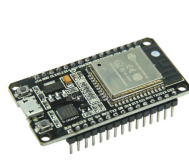


Abbildung 6: ESP32 Board mit Microcontroller und WLAN

4.1.5 WLAN Router

Die Kommunikation soll in einem Heimnetz funktionieren, wobei eine Verbindung zum Internet bestehen muss, um alle Funktionalitäten des Magic Mirrors ausschöpfen zu können. Da beispielsweise der Infrarot Präsenzmelder nicht kabelgebunden installiert werden soll, muss ein wireless Netz vorhanden sein, sprich WLAN. Es kann auch ein Hotspot von einem mobilen Gerät erstellt werden um das Netz herzustellen. Der Magic Mirror, wie auch der Infrarot Präsenzmelder verbinden sich jedoch auf eine fix eingestellte SSID.

4.1.6 Laptop

Der Laptop im System wird verwendet um Konfigurationen des Magic Mirror zu ändern. Der Vorgang ist so, dass eine SSH Verbindung zum Raspberry Pi aufgebaut wird, wo die Konfiguration angepasst werden kann.

4.1.7 Handy

Eine Android Applikation wird gebaut, um mit dem Magic Mirror interagieren zu können. Mindestens die Version Android Kitkat 4.4 wird dafür benötigt. Dies ist bei Geräten neuer als 2013 standardmässig gegeben und dadurch wird eine grosse Unterstützung erreicht.

4.2 Auswahl Software

Auf verschiedenen Teilen im System läuft unterschiedliche Software. Die Eigenentwicklungen oder Erweiterungen an bestehender Software sind im Bild unten aufgeführt. Die einzelnen Softwareteile sind in den folgenden Kapiteln detaillierter beschrieben und erläutert. Hier ist bloss eine Übersicht gegeben.

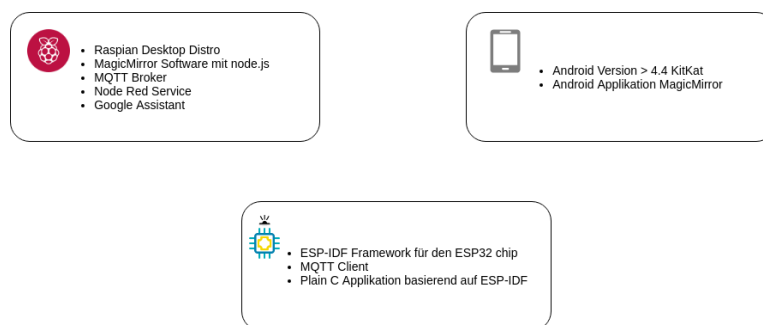


Abbildung 7: System Software Overview

4.2.1 Raspian auf Raspberry Pi

Der einfachste Weg um einen Desktop als Ausgabe zu haben, ist die Raspian Desktop Distribution für das Raspberry Pi zu verwenden. In diesem Projekt wird die aktuellste Version von Raspian Stretch vom 13. November 2018 verwendet, welche die Kernelversion 4.14 beinhaltet.

4.2.2 MagicMirror²

Das MagicMirror² kann direkt von GIT geklont werden. Die Möglichkeit besteht, dies direkt vom Raspberry Pi zu machen. Um aber das Projekt nicht direkt auf dem Raspberry Pi konfigurieren zu müssen besteht die Variante mittels Kopiervorgängen über das Netzwerk einzelne Konfigurationsinhalte auf dem Laptop anzupassen und anschliessend auf das Raspberry Pi zu kopieren. Durch die grosse Community des Projekts ergibt sich ungefähr monatlich ein neues Stablerelease, was einfach mit *git pull* upgedatet werden kann.

4.2.3 MQTT Broker

Auf dem Raspberry Pi wird der wohl bekannteste MQTT Broker verwendet. Namentlich der Mosquitto Broker. Die Installation ist sehr einfach und kann über den üblichen Paketmanager abgehandelt werden.

4.2.4 Node-Red

Node-Red wird verwendet, um den externen Infrarot Präsenzmelder in das System zu integrieren und auf dessen publizierte Nachrichten zu agieren. Der Dienst ist erreichbar im Netzwerk unter dem Port 1880. Das heisst mit *http://magicmirror:1880* wird der Dienst erreicht und dessen Verhalten kann geändert werden. Dabei ist "magicmirror" der Netzwerkname vom Raspberry Pi.

4.2.5 Android Applikation

Die Entwicklung einer Android Applikation ist am besten unterstützt mit dem Android-Studio. Dabei engt man die Unterstützung auf Android Geräte ein, was jedoch in diesem Projekt kein Problem ist.

4.2.6 ESP-IDF Programm

Um die Funktionalitäten vom ESP32 Microcontroller zu benutzen ist ein Framework sehr hilfreich. *Espressif IoT Development Framework* [1] ist das meistbenutzte und bietet neben Benutzung der IO Pins auch direkte Netzwerkinteraktion über WLAN, sogar das MQTT Protokoll ist im Framework implementiert und bereit zur Verwendung.

5 Magic Mirror auf dem Raspberry Pi

Wie bereits beschrieben, wird die Opensource Software von MagicMirror² als Grundlage verwendet. Deshalb wird hier als erstes der Ablauf beschrieben um diese Software auf dem Raspberry Pi aufzusetzen. Dabei wird davon ausgegangen, dass das Raspberry Pi bereits mit einer bootbaren SD Karte versehen ist.

5.1 Aufsetzen von MagicMirror²

Es gibt ein vorbereitetes Bash Skript, mit dem die Installation automatisch ausgeführt wird. Der Befehl dafür ist:

```
pi@magicmirror bash -c "$(curl -sL https://raw.githubusercontent.com/MichMich/MagicMirror/master/installers/raspberry.sh)"
```

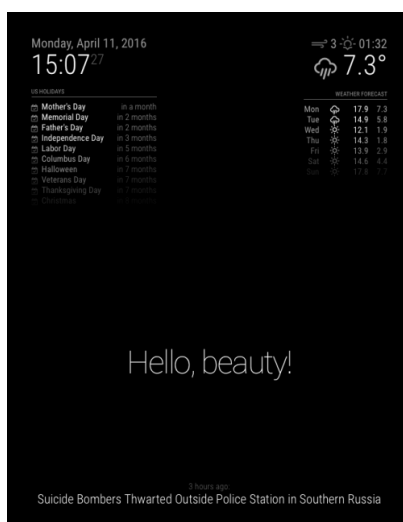
Die Installation erstellt im `/home/pi` einen Ordner mit dem Namen *Magic-Mirror*, welcher das ganze Projekt beinhaltet.

Mit einem Startup-Skript kann der Projekt einfach beim Booten gestartet werden. Das Skript hat folgenden Inhalt:

```
cd /MagicMirror
DISPLAY=:0 npm start
```

Es trägt den Namen *mm.sh* und wird mit dem Tool *pm2* zum Raspberry Bootprozess hinzugefügt.

Nach diesen kurzen Schritten wird bereits beim Starten des Raspberry Pi's automatisch der MagicMirror₂ gestartet mit der Default-Konfiguration, welche folgendermassen daherkommt:

Abbildung 8: Standardanzeige vom MagicMirror₂

5.2 Integration Node-Red und MQTT

Der MQTT Broker kann über den Paketmanager geholt werden. Folgender Befehl ist dafür nötig:

```
pi@magicmirror sudo apt-get install -y mosquitto mosquitto-clients
```

Danach ist er gestartet und kann von Node-Red verwendet werden. Erst muss aber auch Node-Red aufgesetzt werden. Dies kann auch automatisiert erfolgen mit dem Aufruf von:

```
pi@magicmirror bash <(curl -sL https://raw.githubusercontent.com/node-red/raspbian-deb-package/master/resources/update-nodejs-and-nodered)
```

Nach erfolgter Installation kann Node-Red zum Autostart hinzugefügt werden mit dem Befehl:

```
pi@magicmirror sudo systemctl enable nodered.service
```

Sobald Node-Red läuft, kann der grafische Designer erreicht werden über die URL <http://magicmirror:1880>. Dort wird schlussendlich die ganze Funktionalität, welche Node-Red übernimmt beschrieben. Folgendes wird gefordert:

- Subscriben an Topic von Infrarot Präsenzmelder.
- Nachricht im Topic prüfen.
- Trigger für Zeitmessung.
- Entscheiden ob Bildschirm an- oder ausgeschaltet werden soll.
- Entsprechendes Skript aufrufen.

Die geforderten Punkte können grafisch folgendermassen dargestellt werden in Node-Red: Der dargestellte Ablauf ist nicht komplex. Links ist mit dem

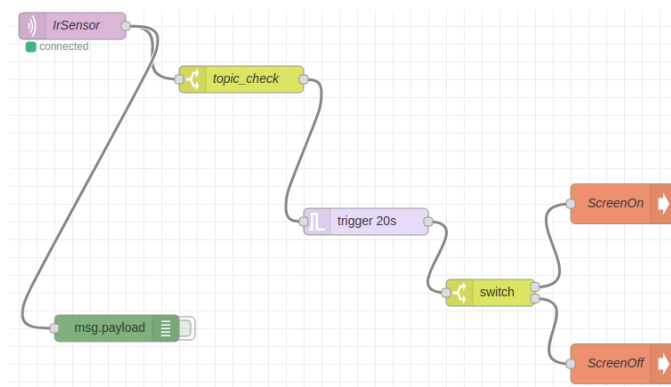


Abbildung 9: Node-Red grafische Implementation

IrSensor die Registration auf dem Topic */topic/magicmirror/irsensor* gemacht unter Localhost, da der MQTT auf demselben Gerät läuft. Mit dem Block *topic_check* wird geprüft, ob eine neue Nachricht mit dem Inhalt *irsensor on* erhalten wurde. Alles andere wird ignoriert. Falls eine solche Nachricht erhalten wurde, wird vom *trigger 20s* eine Nachricht mit dem Inhalt *on* an den *switch* gesendet, welcher dann *ScreenOn* ausführt. Falls nach 20 Sekunden keine weitere Nachricht mit *irsensor on* beim *trigger 20s* angelangt ist, sendet er automatisch ein offenes *switch*, welcher dann *ScreenOff* ausführt. *ScreenOn* führt folgendes Kommando aus um den Bildschirm einzuschalten:

```
tvservice -p && sudo chvt 6 && sudo chvt 7
```

Beim *ScreenOff* ist es der folgende Befehl um den Bildschirm auszuschalten:

```
tvservice -o
```

5.3 Implementation eigenes MagicMirror² Modul

6 Applikation für Präsenzdetektion

7 Android Applikation für Notizen

Literaturverzeichnis

- [1] Espressif. Espressif iot development framework. 2019. URL: <https://docs.espressif.com/projects/esp-idf/en/latest/>.
- [2] Teeuw Michael. Magicmirror opensource project. 2019. URL: <https://magicmirror.builders/>.
- [3] Various. Magicmirror modules. 2019. URL: <https://github.com/MichMich/MagicMirror/wiki/3rd-party-modules>.