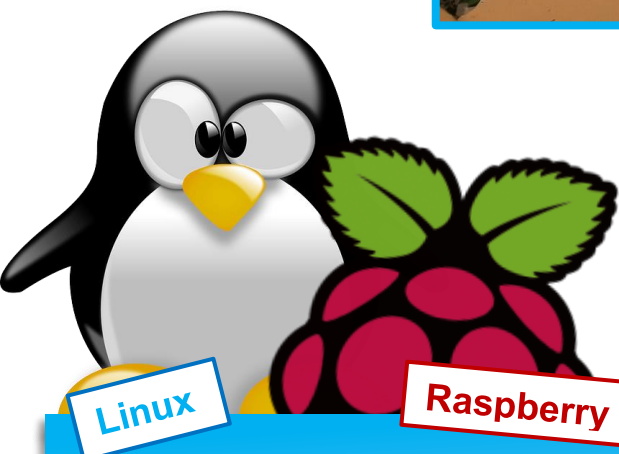


IDPA Projekt 2021: Poolsteuerung

GBS St. Gallen

Verfasser: Jens Aerni, Jun Heule und Marc Hagmann
Schule: GBS St.Gallen
Klasse: BMTL4a
Betreuungsperson: Herr Kappler Sandro
Betreuungsperson: Herr Müller Patrick
Abgabetermin: 23. Februar 2021



Inhaltsverzeichnis

1	Abstract	3
2	Einleitung.....	4
3	Überblick.....	5
3.1	Filterpumpe (Aqua Vario Plus).....	5
3.2	UV-C Filter (conZero UV-C Entkeimung 75W)	6
3.3	Wärmepumpe (conZero Super Efficient)	6
4	Pflichtenheft.....	7
4.1.1	Was muss die Steuerung können?	7
4.1.2	Was soll die Steuerung können?	7
5	Evaluation der Steuerung	8
5.1	Steuerung mit einem Raspberry Pi	8
5.1.1	Erklärung Raspberry Pi.....	10
5.1.2	Vorteile, Nachteile und Bedenken.....	12
5.2	Steuerung von Siemens.....	13
5.2.1	Erklärung Siemens	14
5.2.2	Vorteile, Nachteile und Bedenken.....	15
5.3	Entscheid für den "Raspy-Pool"	16
5.3.1	Vergleich Siemenssteuerung und Raspberry Pi.....	17
6	Bestellung der Bauteile	18
7	Raspberry Pi 4 aufsetzen (ohne Bildschirm)	19
8	Software Entwicklung.....	22
8.1	Applikationsstruktur	24
8.2	Autostart	25
8.2.1	Entry.desktop.....	25
8.2.2	MyApp.sh	25
8.3	Main.py	26
8.4	open.sh.....	26
8.5	close.sh	27

8.6	pH-Sensor.sh.....	28
8.7	ftdi.py.....	29
9	GUI (graphical user interface)	30
9.1.1	Meldefenster.....	30
9.1.2	Modus.....	31
9.1.3	Start / Stop	31
9.1.4	Home.....	31
9.1.5	Status / Handfunktionen	31
9.1.6	Parameter.....	32
10	Aufbau Schaltschrank	33
10.1	Schema zeichnen	34
10.2	Mechanischer Aufbau	35
10.3	Raspberry Display	36
10.4	Elektrische Verdrahtung.....	37
10.4.1	Übersicht Schaltschrank	38
10.5	Inbetriebnahme.....	39
11	Schluss / Fazit.....	40
11.1.1	Fazit Jens	40
11.1.2	Fazit Marc.....	41
11.1.3	Fazit Jun.....	41
12	Quellenverzeichnis.....	44
12.1	RaspyPool Projekt	44
12.2	Anhang	44
12.3	Verwendete Programm / Webseiten	45
12.4	Bildverzeichnis.....	45
12.5	Datenblätter der Technikbox.....	46

1 Abstract

Wir taten uns schwer, die passende Projektarbeit für uns zu finden. Als Automatiker, im vierten und somit letzten Lehrjahr, haben wir alle drei ein gewisses Grundwissen von Technik. Die Projektarbeit sollte etwas Sinnvolles sein und uns auch herausfordern. Wir wollten eine neue, innovative und selbständige Arbeit machen – ein Prototyp. Diese Arbeit fanden wir bei Heule's neuem Pool im Garten. Dies ist ein runder Stahlwandpool, günstig, versenkt und ohne Betonierung.



Abbildung 1: Heule's Traumpool

Das Ziel unserer IDPA war, die komplette Poolsteuerung zu entwickeln.

Die Steuerung soll in einem kleinen Gartenhäuschen untergebracht werden. Einerseits soll sie gut funktionieren, andererseits wollen wir die Ausgaben dafür möglichst klein halten. Zuerst haben wir unser Projekt geplant und alle Bauteile bestellt. Im zweiten Schritt bauten wir die Steuerung zusammen. Dann haben wir eine Software mit Python entwickelt und in Betrieb genommen. Zusätzlich versuchten wir, einen pH-Sensor in unsere Steuerung zu integrieren. Wir wussten nicht, ob uns das gelingen würde. In dieser Arbeit erfahren sie mehr darüber. Einige Stolpersteine mussten wir aus dem Weg räumen. Nach 22 Wochen intensiver Arbeit, konnten wir unsere Poolsteuerung abschliessen.

2 Einleitung

Anfang dieses Jahres wurde unser Leben durch die Corona Pandemie auf den Kopf gestellt. Wir wurden angewiesen, Zuhause zu bleiben, Kontakte zu vermeiden und Home-Office zu machen. Die viele Zeit Daheim und auch Kurzarbeit während des Lock-Downs liessen uns Projekte in und ums Haus verwirklichen. Die geplanten Strandferien waren nicht möglich. Das gesparte Feriengeld wurde in Haus und Garten investiert. Der Sommer war lang und heiss, wie auch schon in den Jahren zuvor.

Ein Pool war nun Familie Heule's grosser Traum, vor allem Mamas. Wir haben schnell gemerkt, dass viele Leute dieselbe Idee hatten. Die Poolbauer konnten nicht mehr liefern. Nun Monate später, planen wir den Pool für diesen Frühling 2021. Schön, praktisch und günstig soll er sein. Unser ConZero Stahlwand-Pool kommt aus Deutschland. Er wird einen Durchmesser von 4m haben und 1.35m tief sein. Zum Pool bestellen wir eine Technikbox mit integrierter Pumpe, einem UV-C Filter und einer Wärmepumpe. Das komplette Angebot wird CHF 6970.-- kosten. Beim Hersteller könnte auch eine passende Poolsteuerung bestellt werden. Diese kostet ca. CHF 400.--, ist jedoch nur eine Schützensteuerung mit Zeitrelais. Darauf haben wir verzichtet.

Ich als Automatikerlernender hätte gerne eine hochwertige, saubere, kompakte und moderne schweizer Lösung für die Steuerung. Ziel unserer IDPA ist es, diese als Team auszuarbeiten. Um unsere Steuerung zu planen, werden wir ein Pflichtenheft schreiben. Wir werden verschiedene Steuerungen miteinander vergleichen. Auch unkonventionelle Ideen beziehen wir mit ein, welche nicht dem industriellen Standard entsprechen. Anschliessend vergleichen wir die Steuerungen und zeigen die Vor- und Nachteile auf. Wir wählen unseren Favoriten aus und bestellen diesen. Im Nächsten Schritt programmieren wir die Steuerung und den Touchscreen. Für die elektrischen Komponenten zeichnen wir ein Elektroschema. Zum Schluss bauen wir die Steuerung zusammen, verdrahten diese und nehmen sie in Betrieb. Zusätzlich möchten wir versuchen eine pH-Sonde in die Steuerung zu integrieren.

Unser Ziel ist es, eine hochwertige, saubere, moderne und günstige Steuerung für unseren "Lock-Down Pool" zu bauen. Uns ist zudem aufgefallen, dass es keinen Markt für universell einsetzbare Poolsteuerungen gibt. Deshalb braucht man nebst dem Gärtner für den Pool, auch einen Elektriker für die Steuerung. Vielleicht könnten wir unser fixfertiges, universelles System auch einem Gärtner verkaufen? Wäre doch toll für den Käufer, wenn er vom Gärtner den kompletten Pool, mit Steuerung, günstig erwerben könnte.

3 Überblick

Zuerst mussten wir uns einen Überblick über das bestellte Poolsystem verschaffen. Dazu haben wir die Datenblätter der conZero Technikbox studiert. Die Datenblätter sind als Anhang beigefügt. Für die Planung unserer Steuerung waren vor allem die elektrischen Daten relevant. Diese sind unten nochmals zusammengefasst.

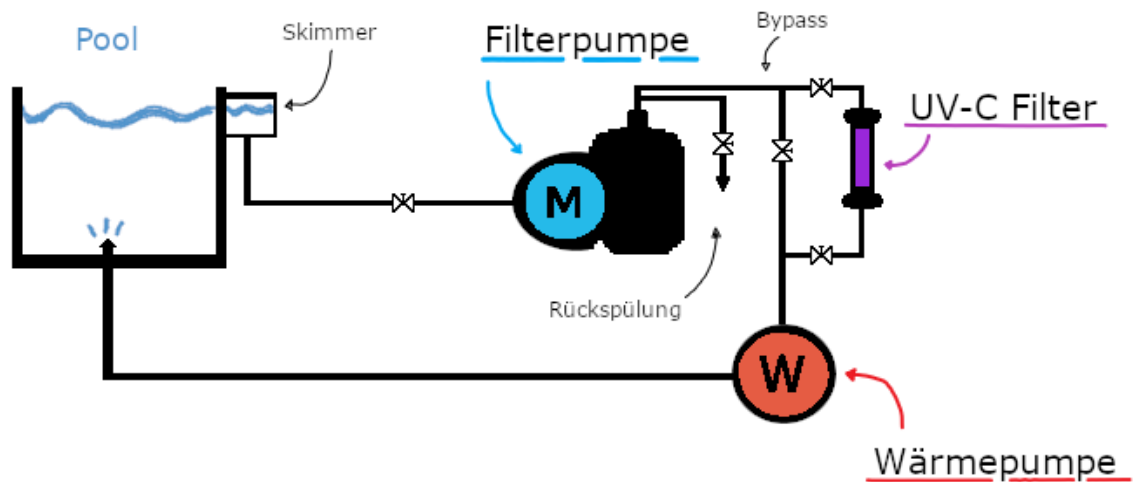


Abbildung 2: Ablaufschema unseres Poolsystems

3.1 Filterpumpe (Aqua Vario Plus)

Die Filterpumpe ist die erste Stufe unseres Aufbereitungssystems. Sie pumpt Wasser durch das System. An ihr wird ein Sandfilter angeschlossen, welcher die Schmutzpartikel aus dem Wasser filtert. Die Pumpendrehzahl kann über einen integrierten Frequenzumformer verstellt werden.

Zusammenfassung der wichtigsten elektrischen Daten:	
Spannung:	230 VAC (1Ph)
Nennstrom:	3 A
Leistung:	0.65 kW
max. Drehzahl:	2850 U/min

3.2 UV-C Filter (conZero UV-C Entkeimung 75W)

Der UV-C Filter bildet die zweite Stufe unseres Aufbereitungssystems. Der Filter darf nur einschalten, wenn Wasser durch das System fließt. Ansonsten wird die Lampe nicht mehr ausreichend gekühlt und überhitzt. Deshalb kann der Filter auch über einen Bypass überbrückt werden. Der Filter desinfiziert das Beckenwasser auf umweltfreundliche Weise.

Die UV-C Strahlung neutralisiert Bakterien, Viren und andere Mikroorganismen und stoppt deren Vermehrung.

Zusammenfassung der wichtigsten elektrischen Daten:	
Spannung:	230 VAC (1Ph) / 50Hz
Nennstrom	0.57 A
Leistung:	75 W
Lebensdauer:	9'000 Stunden

3.3 Wärmepumpe (conZero Super Efficient)

Die Wärmepumpe ist die letzte Stufe unseres Aufbereitungssystems. Sie besitzt einen Wärmetauscher aus Titan und heizt das Beckenwasser mit der Wärmeenergie der Umgebungsluft. Nach Hersteller kann ein COP¹ von 11,8 erreicht werden. Das bedeutet, die Wärmepumpe leistet bis zu 11,8-mal mehr Wärmeenergie, als elektrische Energie verbraucht wird.

Zusammenfassung der wichtigsten elektrischen Daten:	
Spannung:	230 VAC (1Ph) / 50Hz
Nennstrom:	9.5 A
Nennfehlerstrom: → Hersteller verlangt FI-Schutzschalter	30 mA
Scheinleistung elektrisch: ($S = U \cdot I$)	ca. 2.2 kW
Heizleistung:	9.0 kW

¹ COP (engl. Coefficient of Performance)

4 Pflichtenheft

Um eine Übersicht zu haben, was unsere Steuerung können muss, haben wir ein Pflichtenheft erstellt. Das Pflichtenheft ist in zwei Hauptkriterien unterteilt: Muss-Kriterien und Soll-Kriterien. Muss-Kriterien sind zwingend zu erfüllen, sie bilden die Mindestanforderungen. Soll-Kriterien sind Wunschkriterien, diese müssen nicht zwingend erfüllt werden, sind aber erwünscht. Soll-Kriterien können aber auch so ausgelegt werden, dass sie in Zukunft nachgerüstet werden könnten.

4.1.1 Was muss die Steuerung können?

- ☒ einfache Bedienung
- ☒ programmierbare Steuerung
 - Filterpumpe
 - UV-C Filter
 - Wärmepumpe
- ☒ manuell und Zeitgesteuert einschalten
- ☒ installierbar durch Laien

4.1.2 Was soll die Steuerung können?

- ☒ Visualisierung mit Touchscreen
- ☒ pH-Wert Messung



Diese Wünsche sind in der Planung berücksichtigt und könnten nachgerüstet werden.




Weil uns diese Funktionen kosten und nicht viel nützen, haben wir momentan darauf verzichtet.

- ☐ LED-Beleuchtung einschalten
- ☐ Verbindung mit Smartphone
- ☐ Temperatursensor Wasser
- ☐ Durchflusswächter
- ☐ Betriebsstundenzähler

5 Evaluation der Steuerung

5.1 Steuerung mit einem Raspberry Pi

Komponenten	Kosten	Info
Raspberry Pi4 Starterkit: <ul style="list-style-type: none"> ▪ Raspberry Pi4 (4GB) ▪ Gehäuse ▪ Raspberry Netzteil ▪ Speicherkarte 16GB 	CHF 129.— 	Probleme bei Tests: Mit den GPIO Pins (3.3V) ist es schwierig, die Last von normalen Relais zu schalten. Es mussten spezielle Elektronikrelais mit einem Pull-Down Widerstand verwendet werden. Externe Netzteile können Probleme machen. Keine Erfahrungen mit dem Raspberry Netzteil. Zur Programmierung und Visualisierung eignen sich Python oder Node-Red.
Raspberry Pi 7" Touchscreen: <ul style="list-style-type: none"> ▪ 7" Touchscreen ▪ Basisplatine 	CHF 80.--	Die Basisplatine und das Raspberry werden auf die Rückseite des Touchscreens montiert.
Touchscreen Gehäuse: <ul style="list-style-type: none"> ▪ schützt den Bildschirm ▪ gleichzeitig Hülle für Raspberry 	CHF 26.— 	Ersetzt das Gehäuse aus dem Starterkit. Fälschlicherweise Gehäuse für Raspberry Pi 3 bestellt! Musste darum von Hand angepasst werden.
Relaismodul: <ul style="list-style-type: none"> ▪ Relaismodul (250V / 10A) 	CHF 20.--	Schaltet den Hauptstrom mit den 3.3V vom Raspberry Pi.

Komponenten	Kosten	Info
pH-Sensor von Atlas Scientific:		
▪ EZO pH Circuit	CHF 40.--	Mikrocontroller für die pH-Sonde.
▪ EZO Carrier USB Board	CHF 24.99	Platine für die Montage des Mikrocontrollers und Anschluss der pH-Sonde. Muss mit den richtigen Pins des Raspberry verbunden werden. (Siehe Wiring-Diagramm)
		Neue Lösung über USB!
▪ Consumer pH-Sonde	CHF 40.--	pH-Sonde nimmt den pH-Wert auf.
▪ Versand	CHF 40. -- 	Versand ist sehr teuer. Ich habe keine günstigere Variante / Händler gefunden! PH-Sensor von Seeeds-Studio war überall ausverkauft.
Temperatursensor von Atlas Scientific:		
▪ EZO Temperatur Circuit	CHF 27.99	Mikrocontroller für den PT-1000 Sensor.
▪ EZO Carrier USB Board	CHF 24.99 	Platine für die Montage des Mikrocontrollers und Anschluss des PT-1000. Anschluss über USB möglich.
▪ PT-1000 Probe	CHF 19.99	PT-1000 nimmt die Temperatur auf.
Total Basissteuerung:	CHF 256.--	Raspberry Pi 4 Steuerung mit Relaismodul und 7" Touchscreen.
Erweiterung mit pH-Sensor	CHF 144.99	
Erweiterung mit Temperatursensor	CHF 72.97	

5.1.1 Erklärung Raspberry Pi

Das Raspberry Pi wurde von der Pi Foundation entwickelt, um jungen Menschen den Erwerb von Programmier- und Hardware-Kenntnissen zu erleichtern. Entsprechend niedrig ist der Preis angesetzt. Das Raspberry Pi 4 ist ein Computer, mit dem Format einer Kreditkarte. Der Speicherplatz kann für einen Mehrpreis zwischen minimal 1GB bis maximal 8GB gewählt werden. Die Platine wird mit 5 Volt über eine USB-C Schnittstelle gespiesen. Auf vielen Internet Foren sind Probleme mit der Spannungsversorgung durch externe Netzteile beschrieben. Aus diesem Grund haben wir das Raspberry Netzteil mit 3.5 Ampere bestellt, welches einwandfrei funktionieren sollte, laut Bewertungen. Das Board kann entweder über Wireless oder die Gigabit-Ethernet Schnittstelle in ein Netzwerk eingefügt werden. Auch eine Bluetooth Verbindung ist möglich. Das Raspian Betriebssystem ist eine, speziell für das Raspberry entwickelte, Linux Variante. Das Betriebssystem wird über eine SD-Karte auf den Controller übertragen. Der SD-Kartenslot befindet sich auf der Unterseite des Raspberry. Für Audiogeräte steht eine 3.5mm Klinken Schnittstelle zur Verfügung. Das Board ist mit vier USB und zwei micro-HDMI Schnittstellen ausgestattet. Für das Raspberry gibt es auch eine Kamera, welche über den CSI² Kamera Port angeschlossen werden kann. Unseren Touchscreen können wir am DSI³ Port anschliessen.

Anders als herkömmliche PCs besitzt das Raspberry eine frei programmierbare GPIO⁴ Schnittstelle mit Ein- und Ausgängen. Die GPIO Pins können Softwaremässig als Ein- oder Ausgänge konfiguriert werden. Über diese Schnittstelle können Signale mit max. 3.3 Volt eingelesen oder geschaltet werden. In Unserem Fall brauchen wir die GPIO Pins, um die Relais und somit den Hauptstrom zu schalten. Auf den Bildern unten ist die Raspberry Platine und ein Pin-Out der GPIO Pins abgebildet.

² CSI (engl. Camera Serial Interface)

³ DSI (engl. Display Serial Interface)

⁴ GPIO (engl. General Purpose Input Output)

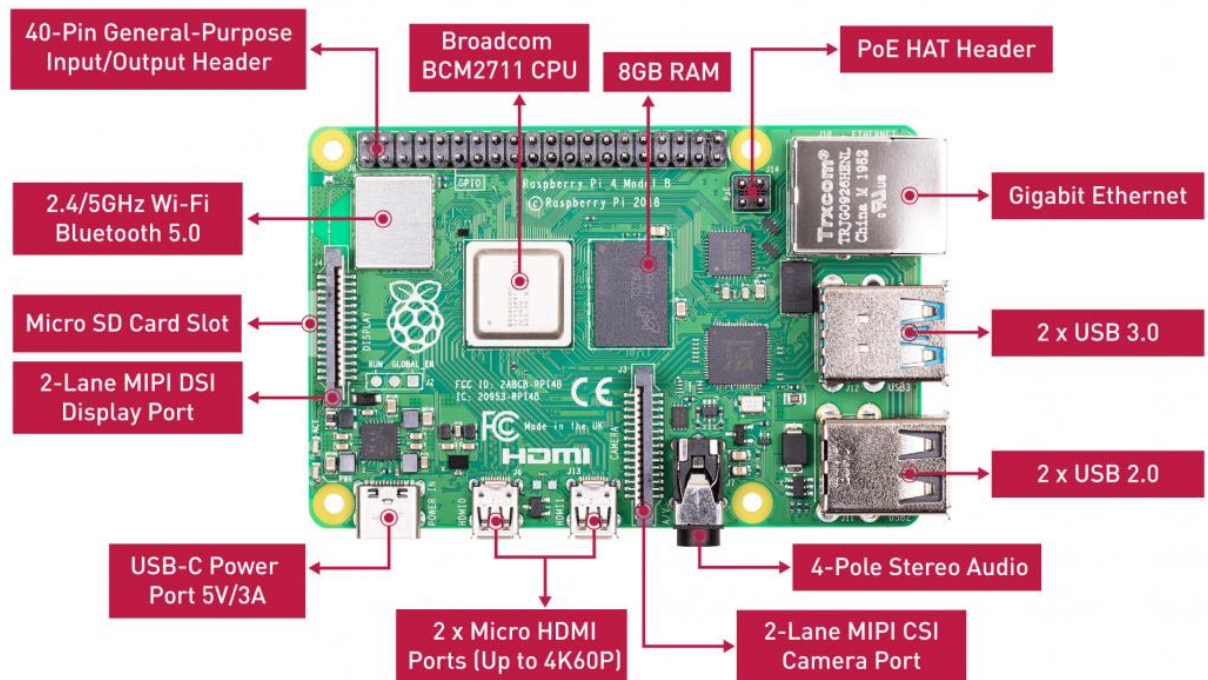


Abbildung 3: Raspberry Pi 4 Platinen Layout

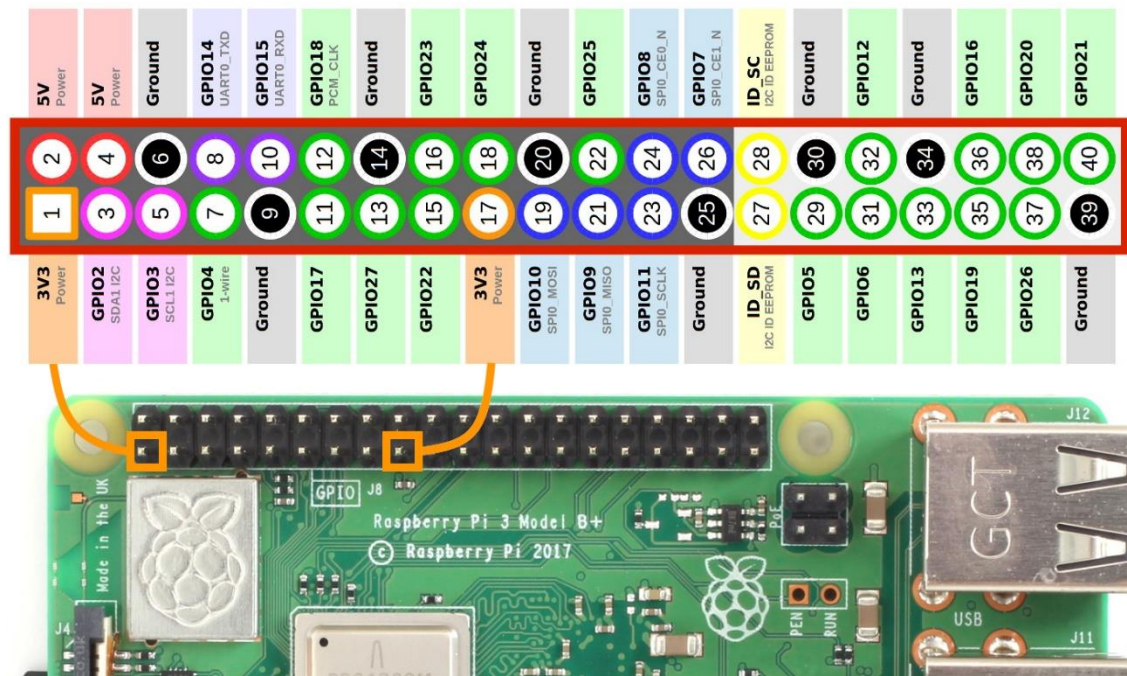


Abbildung 4: Raspberry Pi 4 PinOut GPIO Header

5.1.2 Vorteile, Nachteile und Bedenken

Folgende Gedanken haben wir uns in der Planungsphase gemacht:

Das Raspberry Pi 4 bietet mit dem flexiblen Board und den variantenreichen Schnittstellen sehr viele Möglichkeiten. Es ist sehr gut für kleine Elektronikprojekte geeignet. Die Programmierung und Visualisierung ist mit Python oder Node-Red (*Erkl. Programmiersprachen*) sicher möglich. Der Hauptstromkreis von 230 Volt muss mit den GPIO Ausgängen des Raspberry's geschaltet werden. Dafür braucht man Relais. Bei vorhergehenden Tests hatten wir Probleme, mit den 3.3 Volt der GPIO Ausgängen die Last eines Relais zu schalten. Vielleicht wäre ein Relaismodul mit vorangeschalteten Transistoren die Lösung? Dieses Modul ist sehr billig und bietet vier ansteuerbare Relais mit Wechslerkontakten. Da keiner von uns Erfahrung mit diesem Relaismodul hat, würden wir es testen müssen. Der Ausgangsstrom aller Ausgänge darf 50 mA nicht überschreiten. Die Anschaffungskosten für ein Netzteil sind im Preis des Raspberry inbegriffen.

Für unsere Automatisierungsanwendung standen wir dem Raspberry eher skeptisch gegenüber. Wir befürchteten, dass mit der Last der Relais, dem LCD-Touchscreen und allfälligen Erweiterungen durch Sensoren und LEDs das Raspberry an seine Leistungsgrenze kommen könnte. Das Raspberry wird nur über USB-C mit 5 Volt und maximal 3.5 Ampère gespiesen, das entspricht der Leistung eines Handy-Ladegerätes. Wegen der offenen Platinen hatten wir auch Bedenken, die Steuerung "im Freien" einzusetzen. Allenfalls könnte die Platine durch Luftfeuchtigkeit, Temperaturschwankungen oder Witterung, Schaden nehmen. Auf Seiten der Software sahen wir viele Vorteile. Das Raspberry ist ein kleiner Computer und bietet unzählige Möglichkeiten. Die Programmierung ist sicher möglich. Das Raspberry ist WLAN fähig und kann so einfach ins Heimnetzwerk eingefügt werden, auch wenn die Steuerung im Garten eingesetzt wird.

5.2 Steuerung von Siemens

Komponenten	Kosten	Info
Siemens Logo	CHF 100.--	Steuerung hat 8DI / 4DO. Je nach Konfiguration sind auch analoge Ein- und Ausgänge möglich.
Siemens Logo Erweiterungsmodul	CHF 50.--	Erweiterungsmodul für die Siemens Logo mit 4DI / 4DO.
Siemens CPU 1212C	CHF 139.--	Steuerung mit 6DI, 4DO und 2AI.
Siemens CPU 1214C	CHF 250.--	Steuerung mit 14DI, 10DO und 2AI. Kann für 230V Stromversorgung bestellt werden.
Siemens HMI KTP 400	CHF 250.--	4" Touchscreen. Günstigstes Eingabegerät von Siemens.
Total (günstigste Variante Logo)	CHF 350.--	Diese Variante beinhaltet keine Relais und kein Netzteil.
+ Zusätzliches 24VDC Netzteil		
+ Zusätzliche Relais		

5.2.1 Erklärung Siemens

Siemens ist in der Industrieautomation marktführend. Simatic Controller sind Steuerungen, welche für die Automatisierung von Maschinen und Anlagen eingesetzt werden. Die meisten CPUs⁵ werden mit 24 Volt versorgt. Jeder Controller besitzt eine gewisse Anzahl von digitalen Ein- und Ausgängen. Diese werden ebenfalls mit 24 Volt betrieben. Werden mehr Ein-, Ausgänge benötigt, können die Controller einfach mit einem Erweiterungsmodul erweitert werden. Die Steuerung muss im TIA-Portal programmiert werden. Das ist die Programmieroberfläche von Siemens. Das TIA-Portal kann nur mit einer Lizenz verwendet werden. Da wir im Betrieb Siemenssteuerungen verwenden, steht uns diese kostenlos zur Verfügung. Die Controller sind weder WLAN noch Bluetooth fähig, könnten jedoch über die Ethernet-Schnittstelle mittels Kabel ins Heimnetzwerk eingebunden werden. Die Ethernet Schnittstelle muss aber auch vom Touchscreen verwendet werden. Somit müssten wir unsere Steuerung mit einem Switch erweitern. Die Controller sind eher für raue Bedingungen entwickelt, und so haben wir keinerlei Bedenken für den Einsatz im "Freien".

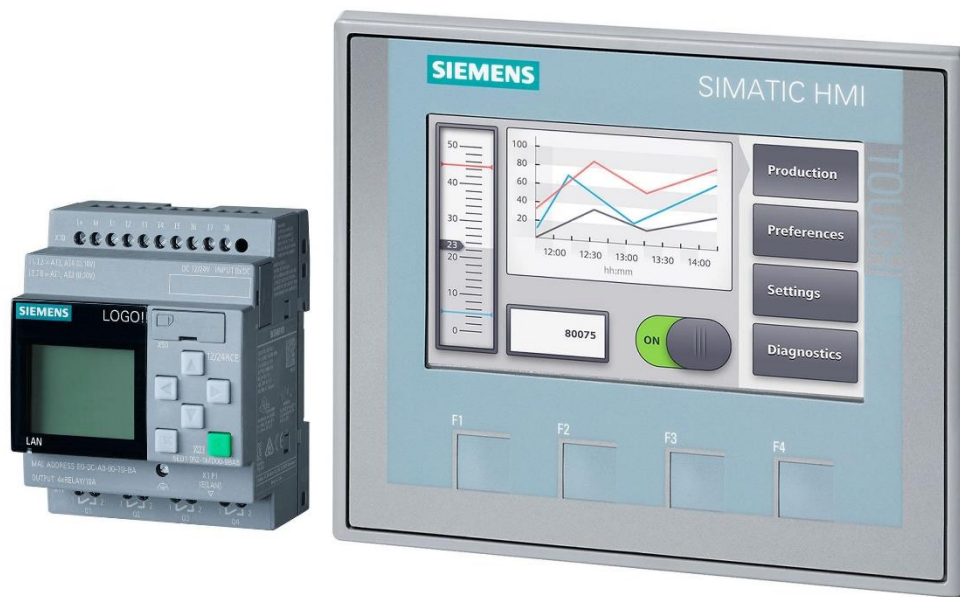


Abbildung 5: Siemens Logo und KTP400

⁵ CPU (engl. central processing unit)

5.2.2 Vorteile, Nachteile und Bedenken

Die Siemens Steuerung ist leistungsmässig sicher sehr gut für unser Projekt geeignet. Das Schalten der Leistung über Relais wird keine Probleme machen. Die Programmierung im TIA-Portal ist sehr einfach. Siemens Visualisierungen sind sehr teuer. Mit dem KTP 400 haben wir die günstigste Variante angeschaut. Preislich ist sie zudem die einzige Variante, die für unser Projekt in Frage kommt. Das KTP 400 wirkt nicht sehr edel und ist auch nur 4" gross. Das HMI⁶ besitzt aber 4 Buttons, welche verwendet werden können. Eine Verbindung ins Heimnetzwerk ist eher schwer realisierbar, da die Steuerung nicht Wireless fähig ist. Für diese Variante würden wir noch ein 24V Netzteil und mindestens vier Relais benötigen. Diese sind in der Kostenrechnung der Siemenssteuerung nicht berücksichtigt. Ein weiterer Nachteil sehen wir in der begrenzten Anzahl von Ausgängen. Der billigste Controller (Logo) besitzt leider nur vier Ausgänge. Die Anzahl der Eingänge kann vernachlässigt werden, da wir für die Eingabe eine Visualisierung verwenden und keine weiteren Sensoren benötigen.

⁶ HMI (enlg. Human Machine Interface)

5.3 Entscheid für den "Raspy-Pool"

Wir haben uns für eine Steuerung mit dem Raspberry Pi 4 entschieden. Ausschlaggebend für diesen Entscheid waren die vielen Möglichkeiten zum guten Preis.

Eine reine Steuerung mit der Siemens Logo, ist etwa in der gleichen Preisklasse wie ein Raspberry Pi. Die Siemenssteuerung benötigt aber zusätzlich ein 24 Volt Netzteil und mehrere Relais, welche nicht in die Kostenrechnung mit einbezogen sind. Die teure Visualisierung von Siemens wirkt eher billig. Die Programmierung ist jedoch sehr einfach. Mit dem günstigen Touchscreen für das Raspberry Pi haben wir keinerlei Erfahrungen. Die Programmierung könnte mit Python möglich sein, ist aber sicher aufwendiger als mit dem Siemenssystem. Die Siemenssteuerung hat keine analogen Eingänge zum Einlesen von Sensoren und nur vier Ausgänge. Das Raspberry Pi ist WLAN und Bluetooth fähig und verfügt über 20 GPIO Pins, welche wir selbst als Ausgang oder Eingang definieren können. Das Raspberry ist aber nur eine Platine, und die dazugehörigen Komponenten sind billig gemacht. Daher wird es eine Herausforderung, das Raspberry in eine saubere Lösung zu integrieren. Es soll keine Bastlerlösung werden.

Die Software wird sicher der Knackpunkt des ganzen Projekts. Mit dem selbständigen programmieren einer Visualisierung haben wir keine Erfahrungen. Ein Programm mit der Programmiersprache Python müsste möglich sein. Die Programmierung darf nicht unterschätzt werden. Diese wird sicherlich viel Zeit beanspruchen.



Mit unserem Projekt wollen wir beweisen, dass mit einer einfachen Lösung ein sauberes Steuersystem entwickelt werden kann. Es soll eine günstige, saubere Alternative zum industriellen Siemenssystem bieten. Keine Bastlerlösung. Uns ist bewusst, dass die Hardware günstiger ist, aber die Entwicklung und Programmierung einen massiven Mehraufwand fordern wird.

5.3.1 Vergleich Siemenssteuerung und Raspberry Pi

	Raspberry Pi	Siemens Logo
Steuerung:	Entwicklungs-Platine (Computer)	Steuerungssystem
Betriebssystem:	Linux OS	Siemens TIA
Bildschirm:	günstiges Zubehör (sieht sehr edel aus)	teures Eingabesystem (wirkt sehr billig)
Ein-, Ausgänge:	40 GPIO Pins → 3,3 Volt	8 Ein-, 4 Ausgänge → 24 Volt
Netzteil:	im Kit vorhanden	24VDC Netzteil muss zusätzlich verbaut werden.
Programmierung:	Grundsätzlich jede Programmiersprache für Computer möglich Grosser Aufwand! Muss alles selbst entwickelt werden!	TIA-Portal (Lizenzpflichtig) Kleiner Aufwand.
Verbindung:	WLAN, Bluetooth oder Gigabit-Ethernet	Profinet, Gigabit-Ethernet
Zuverlässigkeit:	keine Erfahrung / Angaben	läuft immer

6 Bestellung der Bauteile

Die folgende Tabelle zeigt alle Bauteile, welche für unsere Steuerung benötigt werden. Um unseren Prototypen möglichst günstig zu halten, haben wir ein paar **Komponenten aus dem Elektroschrott** wiederverwertet.

Anzahl	Artikel	Artikel Nr.	Lieferant	Kosten
1	Hauptschalter Schneider 25A /3P	550811801	EM	CHF 18.70
1	Hager LS MCN001 (UVC-Filter)	805610004	EM	CHF 12.35
1	Hager LS MCN003 (Pumpe)	805614004	EM	CHF 12.35
2	Hager LS MCN010 (Heizung)	805618004	EM	CHF 17.72
1	Hager EB-Steckdose T13	663046034	EM	CHF 14.55
1	EB Steckdose T13	663076179	EM	CHF 5.95
1	Raspberry Pi4 (8GB) Kit mit Gehäuse, Netzteil, Micro SD	Link	Digitec	CHF 129.00
1	Raspberry Pi 7" Touchscreen	Link	Digitec	CHF 76.20
1	Touchscreen Gehäuse	Link	Digitec	CHF 26.00
	Falsch bestellt! Gehäuse für Raspberry Pi 3.			
1	Sertronics Relais Modul	Link	Digitec	CHF 15.00
1	pH-Sensor von Atlas Scientific <ul style="list-style-type: none"> EZO pH Circuit EZO Carrier USB Board Consumer pH-Probe Versandkosten 	Link Link Link	Atlas- Scientific	CHF 40.00 CHF 24.99 CHF 40.00 CHF 40.00
1	Rittal Verteilerschrank	824730333	EM	CHF 55.00
Summe Steuerung mit Neuteilen				CHF 527.81
Summe Prototyp mit Bauteilen aus Elektroschrott				CHF 420.19

7 Raspberry Pi 4 aufsetzen (ohne Bildschirm)

Wir haben das Raspberry Pi 4 Starterkit für CHF 129.-- bei Digitec bestellt. Das Kit beinhaltet ein Raspberry Pi 4 mit 4GB Speicherplatz, das Raspberry Netzteil mit 3.5 Ampère, eine 16GB Speicherkarte und ein Gehäuse. Das Raspbian OS Betriebssystem ist auf der Speicherkarte vorinstalliert. Bei uns hat das vorinstallierte Betriebssystem jedoch nicht funktioniert. Deshalb mussten wir das Raspberry Pi neu aufsetzen.

Um das Raspberry aufzusetzen benötigt man folgenden Programme / Dateien.

- [Raspbian OS](#)
- [Balena Etcher](#)
- Putty (vorinstalliert auf Windows 10)
- Remote Desktop (vorinstalliert auf Windows 10)

1. Vorformatierte Speicherkarte löschen

Zuerst benötigt man eine leere Speicherkarte mit mindestens 3,012 GB Speicherplatz. Falls die Speicherkarte aus dem Kit verwendet wird, müssen die bestehenden Dateien gelöscht werden.

2. Debian Raspberry Pi OS herunterladen

Das Linux Betriebssystem kann [hier](#) heruntergeladen werden. Die heruntergeladene ZIP Datei muss entpackt werden.

3. Betriebssystem auf die Speicherkarte übertragen

Das heruntergeladene Betriebssystem muss auf die SD-Karte "gebrannt" werden. Dazu benötigt man ein Installer Programm. Wir haben Balena-Etcher verwendet.

Im nachhinein haben wir herausgefunden, dass Raspberry, ein eigenes Programm anbietet. Auch der Raspberry Pi Imager kann anstelle von Balena-Etcher verwendet werden.

In Balena Etcher muss die Raspberry Pi OS Datei und das entsprechende Laufwerk (SD-Karte) ausgewählt werden. Mit Flash! wird das Betriebssystem auf die SD-Karte gebrannt.

4. SSH Datei erstellen

Wir wollen von unserem PC auf den Desktop des Raspberry Pi's zugreifen. Dazu müssen wir dem Betriebssystem eine SSH Datei hinzufügen. Wir öffnen die SD-Karte mit dem draufgebrannten Betriebssystem. Hier fügen wir ein neues Textdokument, mit dem Namen SSH ein. Die .txt Endung muss entfernt werden.

Wird die .txt Endung nicht angezeigt, kann diese unter *Ansicht > Dateinamenserweiterungen* eingeblendet werden.

5. Speicherkarte ins Raspberry einfügen

Die Speicherkarte ist fertig formatiert und kann ins Raspberry eingefügt werden. Um mögliche Fehler zu vermeiden, sollte das Netzteil erst danach eingesteckt werden.

6. IP-Adresse auf dem Router nachschauen

Je nach Internetanbieter muss hier unterschiedlich vorgegangen werden. Jeder Internetrouter besitzt eine Übersichtsseite. Auf dieser werden alle Geräte im Netzwerk aufgelistet. Hier sollte nun auch unser Raspberry mit der IP-Adresse zu finden sein.

Da unser Heimnetzwerk Passwortgeschützt ist, konnte sich unser Raspberry nicht automatisch mit dem Netzwerk verbinden. Wir haben das Raspberry dann einfach mittels Lan-Kabel an den Router angeschlossen.

7. Putty

Mit Putty kann auf die Konsole im Raspberry Pi zugegriffen werden. Hier muss die im vorhergehenden Schritt ermittelte IP-Adresse eingegeben werden. Der Port 22 darf nicht verändert werden.

Die Raspberry Konsole öffnet sich und fragt die Login Daten ab. Standardmässig lauten diese:

```
Benutzer: pi  
Passwort: raspberry
```

Jetzt haben wir vollen Zugriff auf die Konsole des Raspberry. Die Konsole ist aber nicht sehr übersichtlich.

8. Xrdp⁷ auf dem Raspberry installieren

Ein Computer nur mit der Konsole zu bedienen ist schwer, und muss geübt sein. Um Zugriff auf den Desktop zu erhalten, richten wir eine Remote Verbindung ein.

Dazu müssen wir Xrdp auf dem Raspberry installieren. Wir geben folgenden Befehl in die Putty Konsole ein.

```
sudo apt-get install xrdp
```

Xrdp wird nun auf dem Raspberry installiert. Putty kann beendet werden.

9. Windows Remote Desktop

Das Programm Remote Desktop ist auf Windows vorinstalliert. Wir müssen wieder die IP-Adresse unseres Raspberry Pi's eingeben. Zum Schluss werden wir nochmals nach Passwort und Benutzer gefragt. Das war's!

Jetzt können wir das Raspberry Pi wie jeden anderen Computer verwenden.

Falls bei Schritt 5, die Netzwerkverbindung mittels Lan Kabel hergestellt werden musste, kann man sich jetzt im Wlan anmelden. Danach kann die Kabelverbindung entfernt werden.

10. Raspberry Updaten

Zum Schluss setzen wir unser Raspberry auf den neuesten Stand. Dazu öffnet man die Konsole und gibt nacheinander folgende Befehle ein:

```
sudo apt-get update  
sudo apt-get dist-upgrade
```

Jetzt haben wir Zugriff auf unser Raspberry, unabhängig, wo es sich in unserem WLAN befindet.

⁷ Xrdp Remote Desktop Protocol

8 Software Entwicklung

Wir haben uns entschieden, dass ich, Jun, den Software-Entwicklungs-Part in unserem Projekt übernehme, da ich schon Erfahrung mit Python sammeln konnte.

Die Programmierung der GUI⁸ (engl. graphical user interface) hat sehr viel Zeit beansprucht. Ich musste mir alle Skills Schritt für Schritt beibringen. Dabei habe ich beim Ausprobieren sehr viele Erkenntnisse gewonnen. Den Umfang des Projekts habe ich unterschätzt. Am Anfang dachte ich: "Ich programmiere einfach schnell etwas – das wird schon funktionieren." Gegen den Schluss aber, stellten sich mir immer mehr Steine in den Weg. Deshalb musste ich das Python "Main" Programm mit Linux Bash Shells erweitern, um in die Prozesse auf dem Raspberry Pi's einzugreifen. Die ganzen Verknüpfungen machen das Programm sehr umfangreich. Um den pH-Sensor über USB zu integrieren, konnte ich ein Python2 Programm, vom Sensorhersteller Atlas Scientific, auf GitHub, herunterladen. Das Programm musste ich aber noch für unsere Anwendung umprogrammieren. Ich wollte ein persönliches und systemerklärendes Home Bild verwenden. Dazu habe ich mit GIMP selbständig ein Bild gepixelt.

Die Inbetriebnahme auf dem Raspberry Pi habe ich ebenfalls unterschätzt. Um eine stabile Funktion des Programms zu gewährleisten, musste ich noch viel Zeit ins Linux System investieren. Unter anderem habe ich einen Autostart eingefügt, welcher beim Aufstarten des Raspberry Pi's automatisch mein Python Skript ausführt. Dazu musste ich viele verschiedene Ansätze ausprobieren. Ich habe mich unter anderem auch einmal aus dem Raspberry ausgeschlossen und musste dieses neu aufsetzen. Um die Parameter über den Touchscreen einzugeben, habe ich eine On-Screen Tastatur eingefügt. Dazu habe ich folgende Anleitung verwendet: [Anleitung On-Screen Keyboard](#).

Alle meine Skripte habe ich in eine Ordnerstruktur "Raspy-Pool" eingefügt. Dieser Ordner musste unter `/home/pi` auf dem Raspberry Pi abgespeichert werden.

Wegen der GPIO Pins des Raspberry's, ist die Funktion aber nur auf diesem, und mit passender Hardware, gewährleistet. Das Python3 Skript kann auf allen PCs mit installierter Python Programmiersprache ausgeführt werden. Alle möglichen Fehler habe ich abgefangen.

⁸ GUI (engl. graphical user interface)

Als Programmieroberfläche habe ich die [IDE PyCharm](#) von JetBrains verwendet. Diese kann ich wärmstens weiterempfehlen. In PyCharm habe ich [GitHub](#) integriert. GitHub ist ein VCS⁹ und speichert Entwicklungsstände. Somit waren wir in unserem 3er Team auch immer alle auf dem aktuellen Stand. Wie GitHub in PyCharm integriert wird, habe ich in einem Erklärvideo beschrieben. Dieses ist als Anhang angefügt.

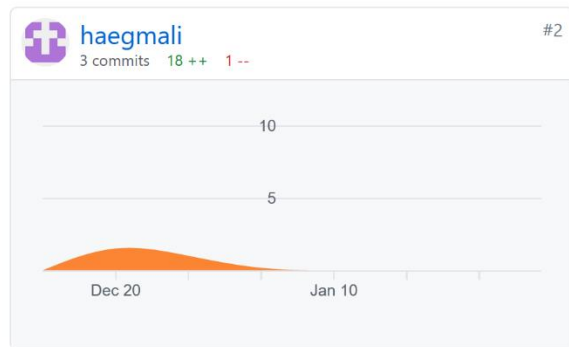
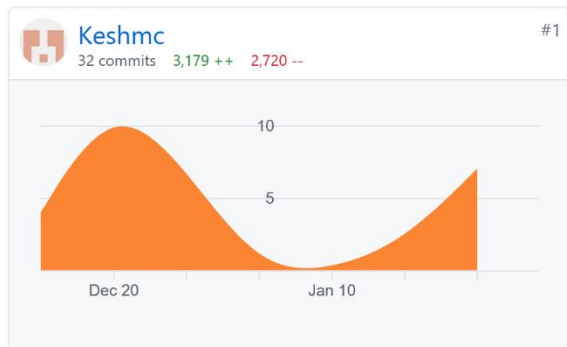
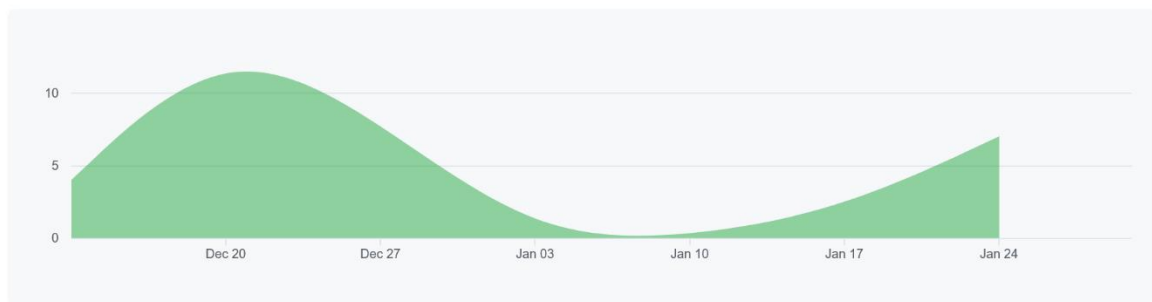
Um den Rahmen dieser Arbeit nicht zu sprengen, geh ich nicht weiter auf die Details der Programmierung ein. Hier werden die Applikationsstruktur und die einzelnen Skripte noch kurz erklärt. Die Programme und alle Daten zum Projekt sind auf GitHub verfügbar.



Dec 13, 2020 – Jan 30, 2021

Contributions: Commits ▾

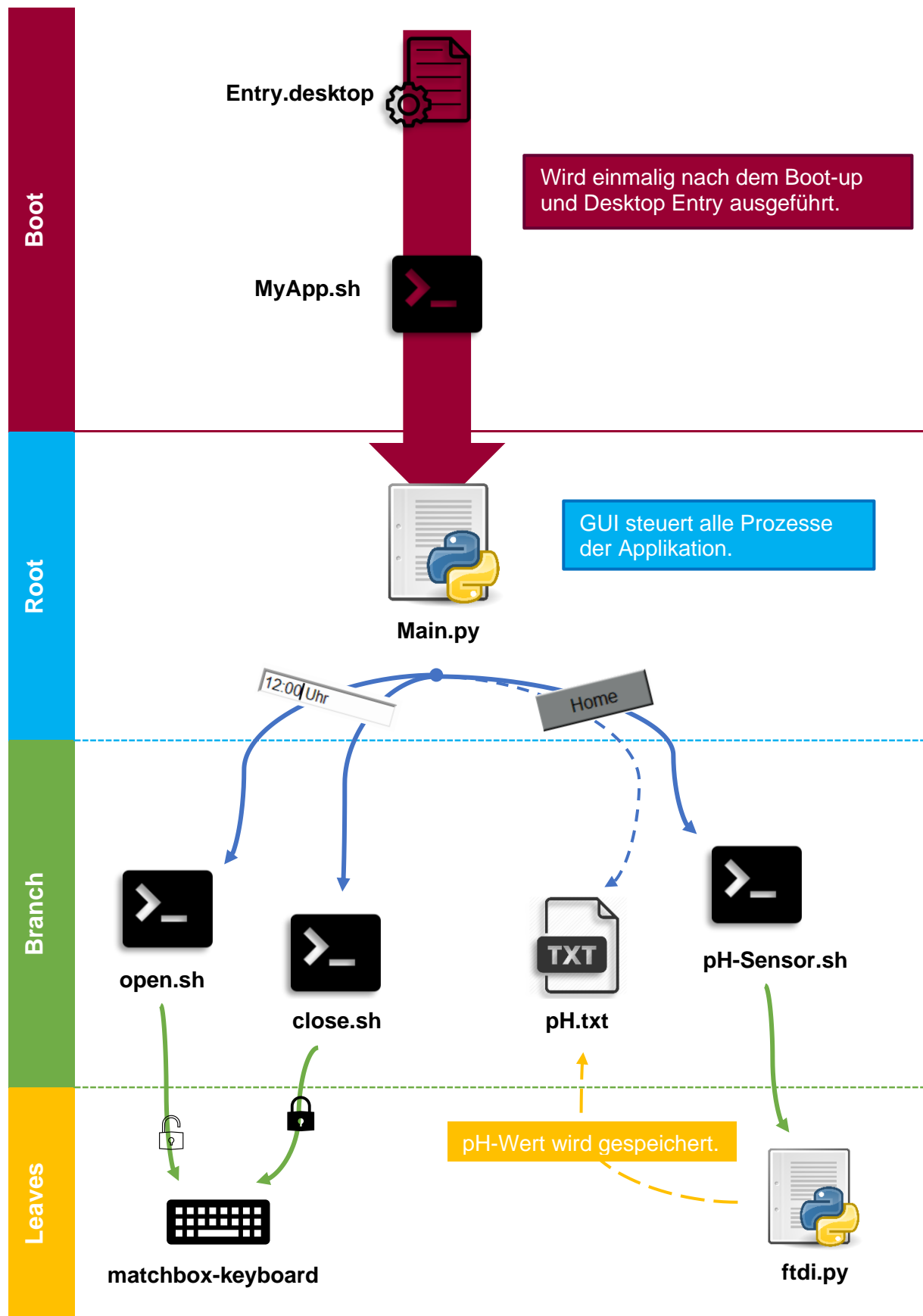
Contributions to master, excluding merge commits



Screenshot 1: Einblick in den Entwicklungsverlauf auf GitHub

⁹ VCS (engl. version control system)

8.1 Applikationsstruktur



8.2 Autostart

8.2.1 Entry.desktop

Speicherort Raspberry: /etc/xdg/autostart

Programm: .desktop

Funktion:

Die .desktop Datei im Autostart Ordner wird beim Öffnen des Desktops ausgeführt und ruft das MyApp.sh Skript auf.

```
1 [Desktop Entry]
2 Exec=sh /usr/bin/MyApp.sh
3 Terminal=false
4
```

Screenshot 2: komplettes Entry.desktop Skript

8.2.2 MyApp.sh

Speicherort Raspberry: /usr/bin

Sprache: Bash Shell

Funktion:

Dieses Shell wird vom Entry.desktop aufgerufen. Nach einer Pause von 3 Sekunden wird das Python3 Main.py Programm ausgeführt.

```
1 #!/bin/sh
2
3 # Shell Script wird bei Desktop Entry aufgerufen
4 # Python3 RaspyPool wird ausgeführt
5
6 (sleep 3s && sudo python3 /home/pi/RaspyPool/Main.py) &
7
```

Screenshot 3: komplettes MyApp.sh Bash Shell

8.3 Main.py

Speicherort Raspberry: /home/pi/RaspyPool

Sprache: Python 3.9

Funktion:

Das Main.py Skript ist der Stamm des RaspyPool Programms, sozusagen der Root. Es wird vom Autostart, nach dem Aufstarten des Raspberry Pi's ausgeführt. Die Visualisierung und Steuerung werden in diesem Programm abgehandelt. Ausschliesslich dieses Skripts hat als Root die Möglichkeit, die unten beschriebenen Bash Skripte auszuführen.

```

1  # imports
2  from tkinter import *
3  from PIL import ImageTk, Image
4  from subprocess import call
5  from gpiozero import DigitalOutputDevice
6  import time
7

```

Screenshot 4: verwendete Python Bibliotheken im Main.py

8.4 open.sh

Speicherort Raspberry: /home/pi/RaspyPool

Sprache: Bash Shell

Funktion:

Dieses Shell wird vom Main.py, beim Eintippen in die Eingabefelder, ausgeführt. Es ruft das Programm matchbox-keyboard und somit eine On-Screen Tastatur auf.

```

1  #!/bin/bash
2
3  #Shell öffnet matchbox-keyboard"
4  #Wird vom Python-Programm Raspy-Pool aufgerufen"
5  #Safe = /home/pi/Desktop/RaspyPool"
6
7  PID="$(pidof matchbox-keyboard)"
8  if [ "$PID" != "" ]; then
9      :
10     else
11         matchbox-keyboard &
12     fi
13

```

Screenshot 5: k-omplettes open.sh Bash Shell

8.5 close.sh

Speicherort Raspberry: /home/pi/RaspyPool

Sprache: Bash Shell

Funktion:

Dieses Shell wird beim Rausklicken aus den Ausgabefeldern ausgeführt. Die On-Screen Tastatur matchbox-keyboard wird wieder geschlossen.

```
1  ▶  #!/bin/bash
2
3      #Shell schliesst matchbox-keyboard wenn dieses offen ist
4      #Wird vom Python-Programm Raspy-Pool aufgerufen
5      #Safe = /home/pi/Desktop/RaspyPool
6
7      PID="$(pidof matchbox-keyboard)"
8      if [ "$PID" != "" ]; then
9          kill $PID
10         fi
```

Screenshot 6: komplettes close.sh Bash Shell

8.6 pH-Sensor.sh

Speicherort Raspberry: /home/pi/RaspyPool

Sprache: Bash Shell

Funktion:

Bei diesem Shell musste ich etwas tiefer in die Trickkiste greifen. Das ftdi.py von Atlas Scientific, welches den pH-Wert aufnimmt, ist in Python2 programmiert. Ich führe das ftdi.py jedes Mal neu aus, wenn die Home Taste gedrückt wird. Wird das ftdi.py aber mehrfach ausgeführt, kommt es zu Fehlern, und die Verbindung zum pH-Sensor kann verloren werden. Um das zu verhindern, überprüfe ich die PID¹⁰ von python2, welche sich von der PID von python3 unterscheidet. So weiss ich, ob das ftdi.py bereits ausgeführt wird oder nicht.

```

1  #!/bin/bash
2
3  #Shell öffnet Python2 Programm von Atlas Scientific und fragt pH-Wert ab"
4  #Wird vom Python Programm Raspy-Pool aufgerufen"
5  #Safe = /home/pi/Desktop/RaspyPool"
6
7  PID="$(pidof python)"
8
9  if [ "$PID" != "" ]; then
10
11     echo "#SH pH-Sensor -- Prozess bereits aktiv"
12
13 else
14     cd /home/pi/RaspyPool/pH-Sensor || exit
15     sudo python ftdi.py &
16 fi
17

```

Screenshot 7: Komplettes pH-Sensor.sh Bash Shell

¹⁰ PID (engl. Process identifier/ deutsch. Prozesskennung)

8.7 ftdi.py

Entwickler: Atlas Scientific (pH-Sensor Hersteller)

Speicherort Raspberry: /home/pi/RaspyPool/pH-Sensor

Sprache: Python 2.7

Funktion:

Damit die Wasserqualität des Pools überprüft werden kann, haben wir uns entschieden, einen pH-Sensor in die Steuerung zu integrieren. Diesen Teil der Arbeit hat Jens übernommen. Für unsere Steuerung setzten wir den pH-Sensor von Atlas Scientific ein. Dieser kann mit Hilfe von drei verschiedenen Kommunikationsprotokollen (USB, I2C und UART) mit dem Raspberry Pi kommunizieren. Wir haben uns für die vermeintlich einfachste Variante über USB entschieden und den Adapter dafür gleich mit gekauft.

Die erste Einrichtung des pH-Sensors an meinem Raspberry hat dann auch gut funktioniert. Nach einigen Minuten suchen, habe ich auf der Website des Herstellers einen «Sample Code» gefunden. Als ich diesen nach einiger Zeit auf dem Raspberry heruntergeladen hatte, konnte ich ihn im Terminal des Raspberry's ausführen. Mit ein paar weiteren Eingaben bekam ich dann auch einen pH-Wert heraus. Allerdings musste für jeden weiteren pH-Wert wieder eine Tastatureingabe gemacht werden, was für unsere Verwendung nicht viel Sinn machte.

Also habe ich den Sample Code auf meinem Raspberry Pi gesucht und im Phyton geöffnet. Nun habe ich die Stellen wo Tastatureigaben gemacht werden mussten gesucht und eine automatische Eingabe der gewünschten Daten programmiert.

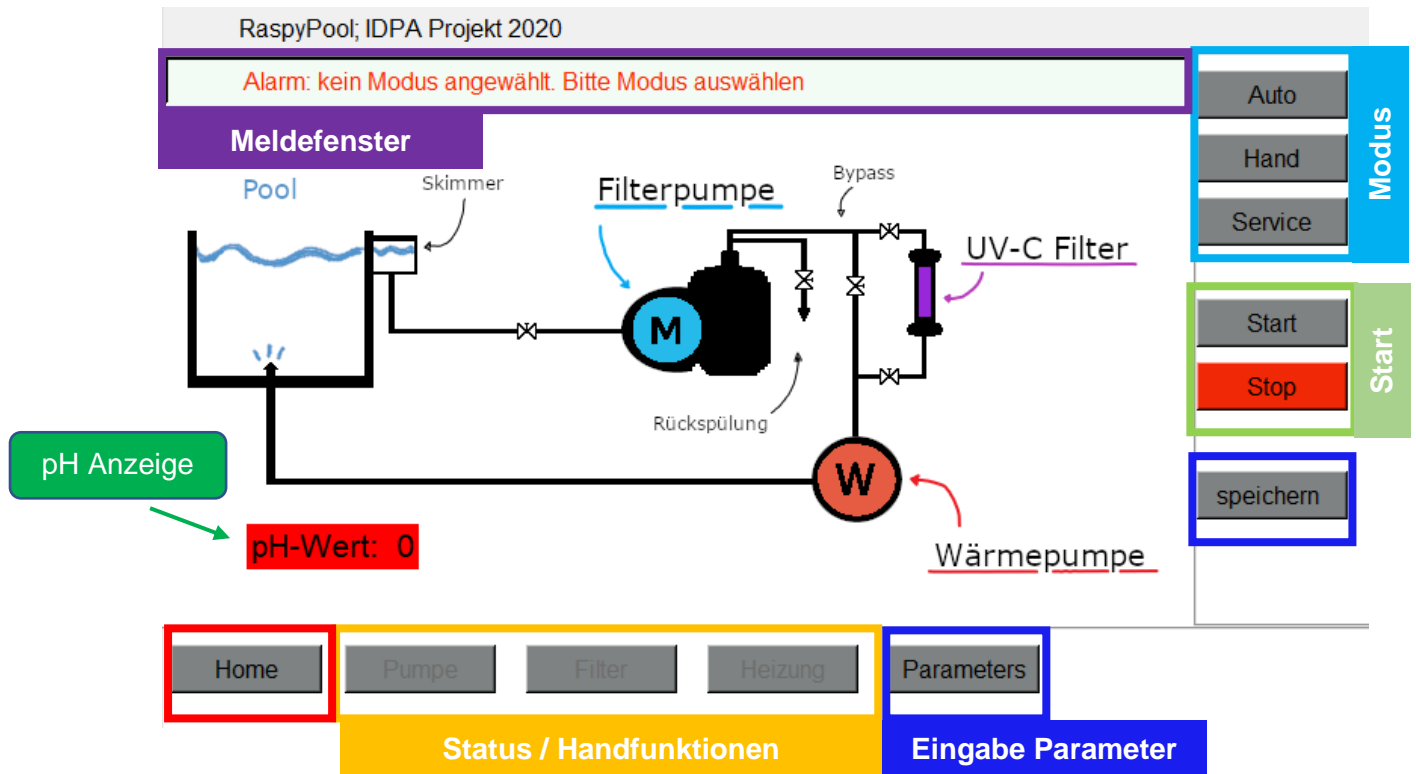
Weil dies an einer Stelle im Programm nicht funktioniert hat, habe ich dann Jun um Hilfe gefragt. Zusammen haben wir es geschafft auch diese Stelle automatisch zu überspringen und so den pH-Wert ohne Tastatureingabe herausbekommen.

Da dieses Programm aber in Python2 geschrieben war, unser Programm aber in Python3, mussten wir das Ganze noch ein bisschen anpassen. Wir haben uns entschieden, den pH-Wert in ein Textdokument zu schreiben und mit dem Main.py auszulesen.



9 GUI (graphical user interface)

Hier erkläre ich den Aufbau unseres Touchscreens und allgemein, wie die Anlage bedient werden muss.



9.1.1 Meldefenster

Im Meldefenster werden Meldungen (schwarz) und Alarmer (rot) angezeigt. Es wird Schritt um Schritt angezeigt wie die Anlage bedient werden muss. Bedienfehler können so auf ein Minimum reduziert werden.

9.1.2 Modus

Für die Anlage können drei verschiedene Modi ausgewählt werden. Um einen Modus zu starten, und die Komponenten einzuschalten muss die **Starttaste** betätigt werden

Auto

Im Automatikbetrieb schaltet die Anlage zu einem auswählbaren Zeitpunkt, für eine bestimmte Zeitdauer ein. Alle Zeitangaben können parametrisiert werden. Für den UVC-Filter kann eine Einschaltverzögerung eingestellt werden, da beim Ansaugen der Pumpe eventuell Luft angesaugt wird.

Hand

Im Handbetrieb kann die Anlage manuell bedient werden. Die Handfunktionen werden aktiviert und können betätigt werden.

Service

Der Servicebetrieb ist zur Reinigung des Pools gedacht. Pumpe und Heizgerät schalten dauerhaft ein.

9.1.3 Start / Stop

Mit der Starttaste wird ein Modus gestartet. Ist ein Modus aktiv, wird die Starttaste grün. Wird im Betrieb der Modus gewechselt, stoppt die Anlage. Um die Anlage sachgerecht abzuschalten, kann die Stopptaste gedrückt werden.

9.1.4 Home

Die Hometaste ruft das Home Bild auf. Auf diesem wird auch der pH-Wert angezeigt.

9.1.5 Status / Handfunktionen

Diese Schaltflächen werden als Statusanzeigen verwendet. Grün bedeutet eingeschaltet und Grau bedeutet ausgeschaltet.

Im Handmodus werden die Schaltflächen aktiviert. Dann können die einzelnen Komponenten von Hand angewählt werden (Schaltfläche wird orange). Mit Start werden die angewählten Elemente eingeschaltet.

9.1.6 Parameter

Im Parameterfeld können die Sollwerte für den Automatikbetrieb eingestellt werden.

Einschaltzeit:	Zeitpunkt, wann die Anlage einschaltet. <i>Format hh:mm Uhr.</i>
Einschaltverzögerung Filter:	Zeitverzögerung die der UVC-Filter nach dem Start der Anlage einschaltet. Zum Schutz für den Filter, falls die Pumpe zu Beginn Luft ansaugt. <i>Format min.</i>
Einschaltdauer:	Zeit, wie lange die Anlage eingeschaltet wird. <i>Format min.</i>

10 Aufbau Schaltschrank

Als wir die Komponenten für den Schaltschrank bestellten, fehlte uns noch ein geeigneter Schaltschrank, in welchen wir die Steuerung pflanzen konnten. Jährlich wird unglaublich viel Altmetall und Elektro Schrott weggeworfen. So schaute Jun in der SFS Altmetall Mulde nach und fand tatsächlich einen geeigneten Schaltschrank. Um den Schaltschrank so aussehen zu lassen, als hätten wir ihn neu gekauft, lackierte ihn Jun. Schliesslich traf die Lieferung mit den Komponenten ein und nun ging es darum, das geplante in die Realität umzusetzen! gemeinsam vereinbarten wir einen passenden Termin um den praktischen Teil soweit es geht fertigzustellen. Am 30. Dezember 2020 trafen wir uns in der SFS und teilten die geplante Arbeit in drei Teile auf.



Abbildung 6: Schaltschrank aus dem Elektroschrott



Abbildung 7: Bauteile (LS) aus dem Elektroschrott



Abbildung 8: neue Bauteile (Raspberry Pi4 Kit)

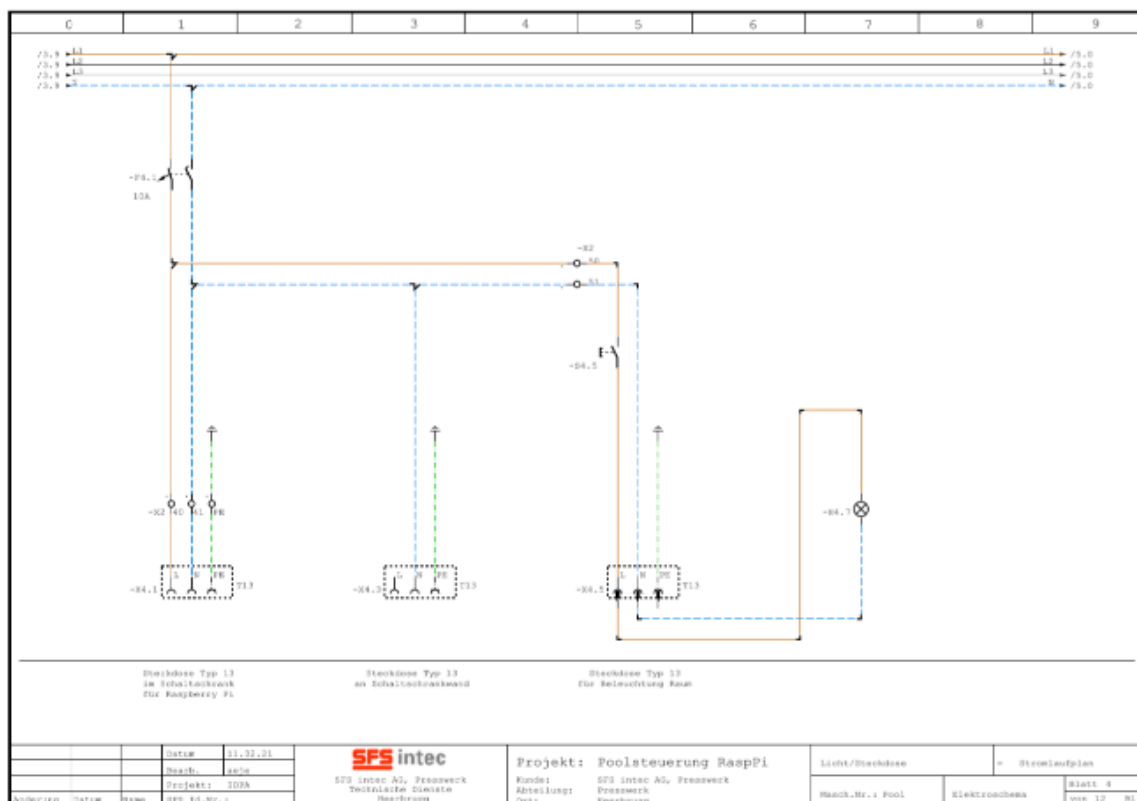
10.1 Schema zeichnen

Bereits eine Woche früher, nachdem wir uns überlegt hatten, was unsere Steuerung können muss, und welche Bauteile wir verwenden wollten, hatte ich begonnen das Elektroschema zu zeichnen. Dafür haben wir zuerst noch die Datenblätter der Pumpe, der Heizung und des UV-Filters studiert, um zu wissen, wie gross der jeweilige Stromverbrauch ist und dementsprechend die Sicherungen sein müssen.

Diese Aufgabe unseres Projekts hat Jens übernommen.

Das Schema habe ich mit dem Programm WSCAD Suite X gezeichnet, welches ich auch bei uns in der Firma verwenden. So konnte ich auch gleich Vorlagen aus unseren Bibliotheken brauchen, was das Zeichnen vereinfachte. Denn Aufbau des Schemas habe ich auch etwa so gemacht, wie wir das in der Firma machen.

Für den Raspberry Pi habe ich einfach eine Blackbox gezeichnet und die nötigen Anschlüsse, die wir verwenden, in dieser Blackbox hinzugefügt. Dasselbe habe ich auch für den PH-Sensor gemacht. Nach einer Absprache im Team, habe ich dann noch ein paar Sachen verbessert und ergänzt. So haben wir zum Beispiel die Klemmenbezeichnungen angepasst, und einen Lichtschalter mit Steckdose für eine Beleuchtung des kleinen "Schuppens" ergänzt. Auf der nächsten Seite sehen Sie einen Ausschnitt aus dem Schema. Dieses ist auch als Anhang beigelegt und auf GitHub verfügbar.



Screenshot 8: Ausschnitt aus dem Elektroschema

10.2 Mechanischer Aufbau

In einem Schaltschrank gibt es meistens eine logische Anordnung der Komponenten, um so einen besseren Überblick zu erreichen. Dies wollten wir ebenfalls erfüllen, obwohl unser Schaltschrank relativ klein ist. Marc hat den mechanischen Aufbau übernommen.

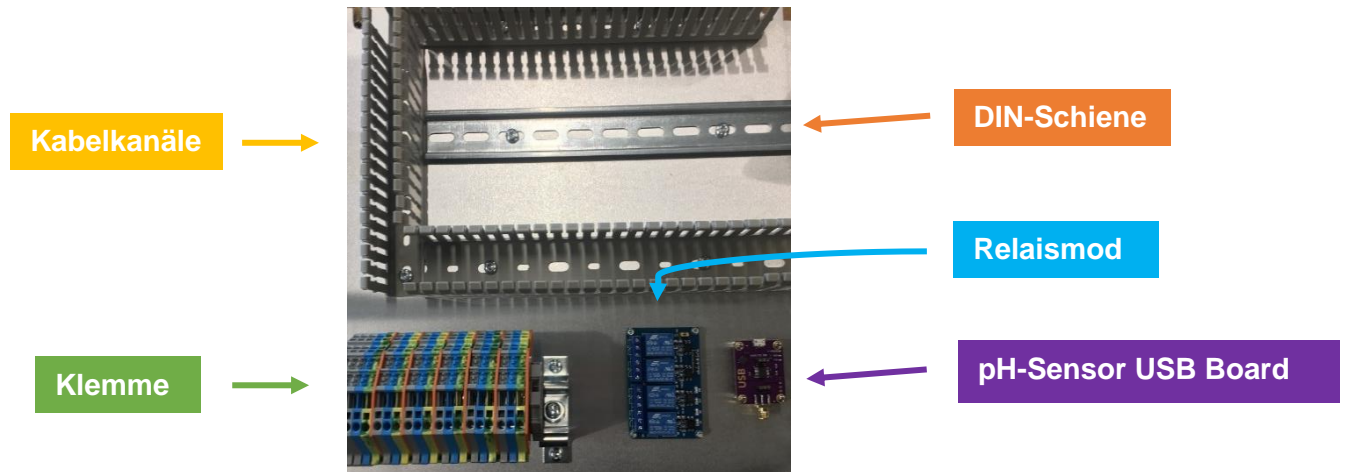


Abbildung 6: Aufbau des Schaltschranks

Um die geplante Anordnung von Jun zu überprüfen, platzierte ich alle Komponenten auf der richtigen Position. In Absprache mit Jun, kamen wir zum Entschluss, dass wir ein Bauteil versetzen müssen, da der Platz zu knapp wurde und somit auch die spätere Verdrahtung problematisch werden könnte. Ausserdem haben wir einen Hauptschalter montiert, mit welchem die gesamte Steuerung von aussen abgeschaltet werden kann. Dieser Hauptschalter befand sich an der Seitenwand, somit musste ich diese Masse ebenfalls miteinberechnen, damit am Ende nicht Kabelkanal und Hauptschalter kollidieren. Schliesslich hatte ich die definitive Anordnung der Komponenten festgelegt und konnte die Kabelkanäle und Schiene auf die richtige Länge zuschneiden und mit Hilfe eines Winkels genau ausrichten. Mit Filzstift kennzeichnete ich anschliessend die Löcher, welche ich danach mit dem richtigen Bohrer bohrte und mit einem Senkbohrer entgratete.

Die Kabelkanäle und die Schiene befestigte ich mit selbstschneidenden Schrauben, somit musste ich auf der gesamten Platte keine Gewinde schneiden, sondern nur Löcher vorbohren. Um die verschiedenen Ausgänge jeweils mit der richtigen Stromstärke abzusichern, montierten wir mehrere Leitungsschutzschalter auf die Schiene. Da man das Relaismodul und die Platine des PH-Sensors nicht auf eine Schiene platzieren konnte, befestigten wir diese unterhalb der restlichen Steuerung. Die Ausgangsklemmen erhöhten wir, um somit das Verdrahten durch die Kabeldurchführung zu vereinfachen.

Somit war der mechanische Aufbau der Platte erledigt. Nun musste ich nur noch die Komponenten korrekt verbinden.

10.3 Raspberry Display

Für die Bedienung der ganzen Steuerung haben wir uns für ein Touch-Display von Raspberry entschieden. Zusammen mit dem Display haben wir auch ein Gehäuse für Display und Raspberry gekauft. Leider haben ich, Jens, erst beim Zusammenbauen bemerkt, dass dieses Gehäuse eigentlich für ein Raspberry Pi 3 ausgelegt ist. Wir haben allerdings ein Raspberry Pi 4, bei welchen die Anschlüsse gegenüber dem Pi 3 leicht angepasst wurden. Somit musste ich als erstes die Aussparungen mit einer kleinen Feile vergrössern, bis der Pi 4 ins Gehäuse passte.

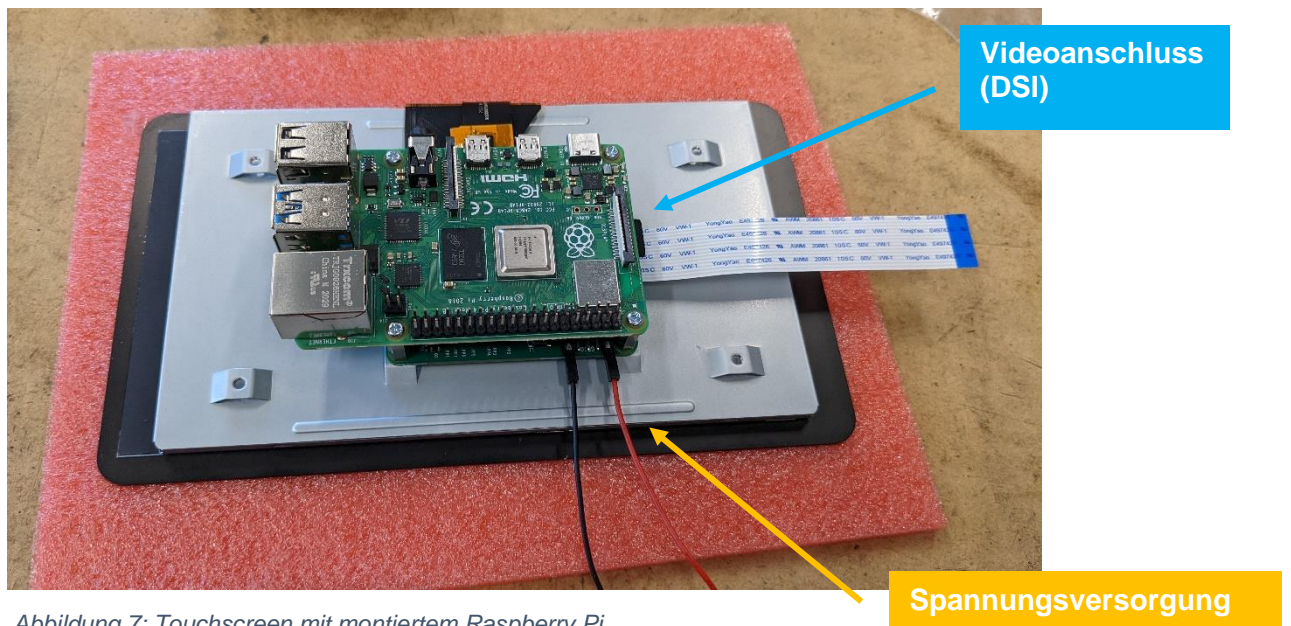


Abbildung 7: Touchscreen mit montiertem Raspberry Pi

Für die Montage am Schaltschrank habe ich als erstes die Aussparung, welche bereits im Schaltschrank vorhanden war, auf die richtige Grösse angepasst. Da das Gehäuse des Raspberry's nicht für die Montage an einer Wand gedacht war, musste ich dieses noch mit vier Löchern für die Befestigung an der Schaltschranktüre ausstatten. Schlussendlich habe ich das Raspberry hinten auf das Display montiert und in das Gehäuse geschraubt. Dann mussten Raspberry und Display nur noch mit zwei Jumperkabel für die 5V Stromversorgung und einem Flachbandkabel für Video- und Datensignale verbunden werden.

10.4 Elektrische Verdrahtung

Damit die verschiedenen Bauteile miteinander funktionieren können, müssen sie korrekt miteinander verdrahtet werden. Das hat Marc übernommen. Die Zuleitung zu unserem Schaltschrank ist 3x400V. Insgesamt brauchte ich vier Ausgänge. Somit legten ich die zwei Ausgänge, welche den meisten Strom benötigen jeweils auf einen Aussenleiter. Die 2 kleineren Ausgänge hängten ich zusammen auf den dritten Aussenleiter.

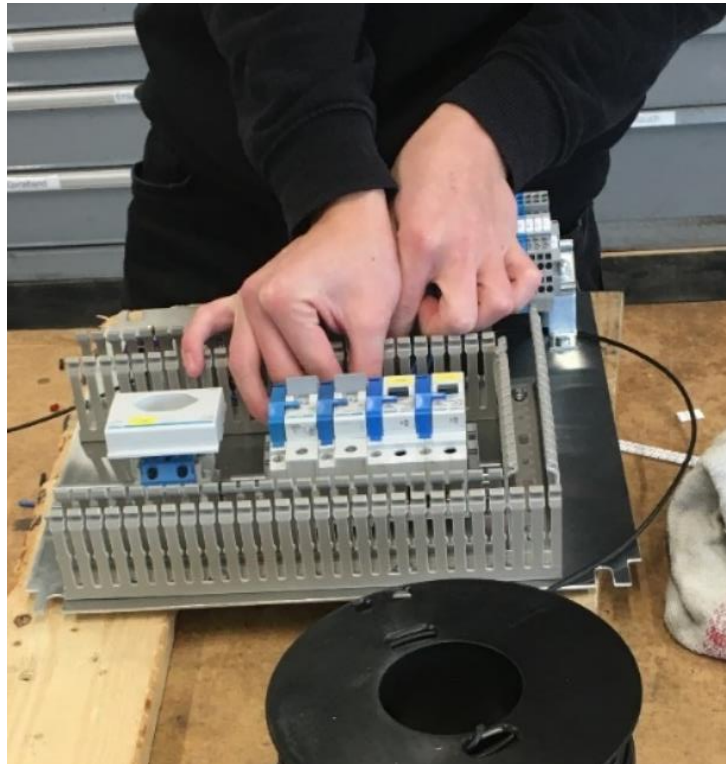
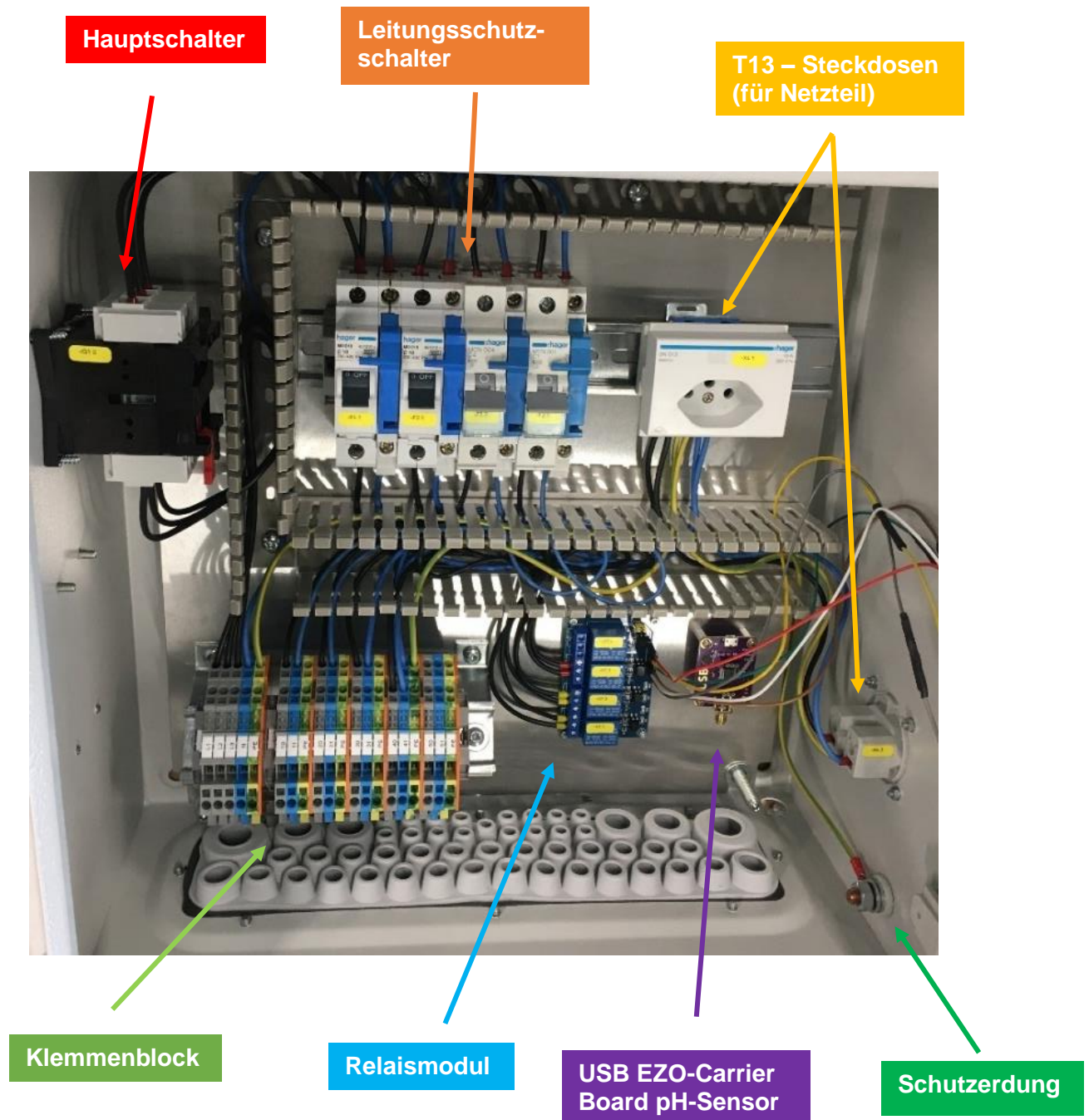


Abbildung 8: Marc beim Verdrahten

Vor der Absicherung benutzten ich Litzen mit einem Querschnitt von 1.5mm^2 in den Farben Schwarz, Hellblau und Grün-Gelb. Schliesslich wurden die Leitungen abgesichert und ich konnte mit einem kleineren Querschnitt von 1mm^2 fortfahren. Eine Leitung war für die Steckdose im inneren des Schaltschranks gedacht. Die restlichen drei Leiter verdrahtete ich weiter bis zum Relaismodul und schliesslich zu den Klemmen. Das Raspberry Display wurde von der internen Steckdose gespeisen, und mit Hilfe der Ausgangssignale des Raspberry Pi's konnten wir die Relaismodule ein- und ausschalten. Um den Menschen vor elektrostatischen Ladungen zu beschützen haben wir das gesamte Gehäuse mit der Erde verbunden. Der verdrahtete Schaltschrank ist auf der nächsten Seite ersichtlich.

10.4.1 Übersicht Schaltschrank



Raspberry Pi und Touchscreen sind im Schaltschrankdeckel montiert und deshalb auf dem Bild nicht ersichtlich.

10.5 Inbetriebnahme

Schliesslich hatten wir unsere Steuerung soweit fertiggestellt und konnten sie nun auf allfällige Fehler oder Probleme testen. Bevor wir dies tun konnten mussten wir alle notwendigen Messungen durchführen. Als erstes führten wir eine Isolationsmessung durch, um somit einen allfälligen Kurzschluss unter den verschiedenen Leitern auszuschliessen. Ausserdem stellten wir mit einer Niederohmmessung sicher, dass wirklich alle Bauteile mit dem Schutzleiter verbunden sind. Mit ausgeschalteten Leitungsschutzschalter konnten wir nun die Steuerung an die Hauptstromversorgung schliessen. Wir schalteten ein Gerät nach dem anderen ein und prüften ob die vorgesehenen Spannungen noch vorhanden waren. Es war soweit alles in Ordnung, daher konnten wir nun das Display und die Ausgänge des Relaismoduls überprüfen. Als wir das Display einschalteten bemerkten wir, dass er um 180° falsch montiert war. Dieses Problem konnten wir auf Grund der Halterung nicht lösen, daher drehte Jun den Desktop in der Konfiguration des Raspberry's. Ausserdem wurden unsere Relaisausgänge nicht wunschgemäss geschaltet und wir standen erneut vor einem Problem. Als wir das Datenblatt des Relaisausgangmoduls nochmals genauer studierten, bemerkten wir, dass wir den entscheidenden V_{CC} Anschluss vergessen haben, welcher für das Schalten der einzelnen Kontakte notwendig ist.

Raspberry Pi4 und Display

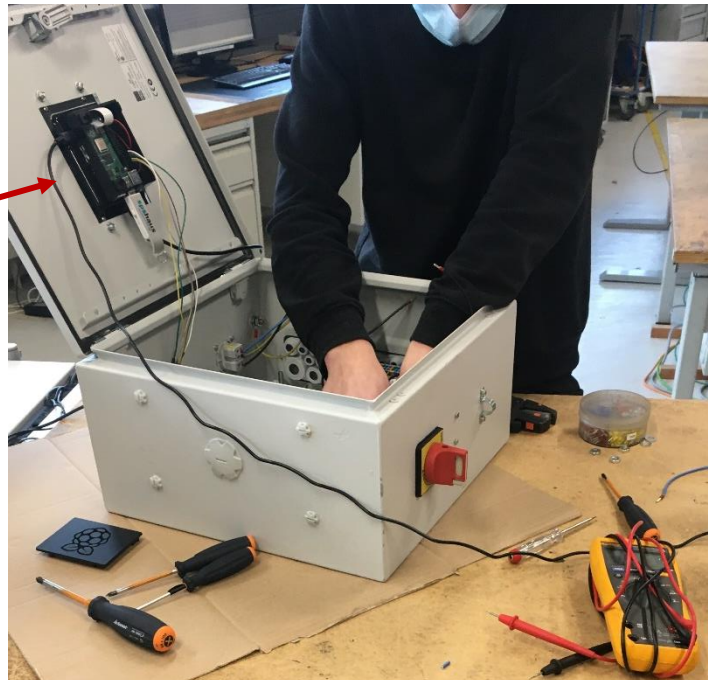


Abbildung 9: Inbetriebnahme und Prüfung der Steuerung

11 Schluss / Fazit

Unser Projektziel war es, eine hochwertige, saubere, moderne und günstige Steuerung für unseren "Lock-Down Pool" zu bauen.

Unsere Steuerung ist hochwertig, sogar mit Touchscreen und pH-Sensor versehen. Eine saubere und moderne Arbeit ist es geworden, weil wir die besten Lösungen gesucht und viele Stunden in die Entwicklung gesteckt haben. Alle Muss-Kriterien haben wir erfüllt. Die beiden angekreuzten Soll-Kriterien konnten wir ebenfalls erledigen, die übrigen könnten nachgerüstet werden. Mit der Verwendung von Elektroschrott (z.B. Schaltschrank und Sicherungen) haben wir eine umweltbewusste und günstige Steuerung gebaut. Hochmodernes und Ausschuss in einem Projekt zu vereinen, ist einfach cool.

Wir haben unser Projektziel erreicht. Das freut uns sehr!

Es freut uns auch ganz besonders, dass wir die völlig neue Herausforderung zusammen angenommen und so gut gemeistert haben! Zumal der Start nicht optimal war. Wir mussten gleich mit zwei grossen, komplett verschiedenen Projekten zur selben Zeit beginnen, die IDAF und die IDPA. Wir haben vorher überhaupt noch nie ein solches Projekt gemacht.

Anfangs hatten wir Angst, dass unsere Idee für die Poolsteuerung vielleicht nicht aufgehen könnte. Wir hatten keine Erfahrung und wussten nicht, ob unser Plan so umsetzbar ist. Wir haben es geschafft, auch dank guter gemeinsamer Teamarbeit. Es war sehr viel Arbeit. Stolpersteine und Unvorhergesehenes liessen uns fasst verzweifeln, und jede Menge Spannendes und viel Lehrreiches nehmen wir mit. Mit Ausdauer, Flexibilität und Teamarbeit ist aus einer Idee ein tolles Projekt geworden.

Geschafft!

11.1.1 Fazit Jens

Vor diesem Projekt hatte ich erst wenig über den Raspberry und seine Möglichkeiten gewusst. Auch die Programmiersprache ist anders als die, welche ich bis jetzt kannte. So habe ich auch gleich nach dem Start des Projektes einen eigenen Raspberry Pi gekauft, um die Programme jeweils gleich testen zu können. Und da man heutzutage im Internet sehr vieles zu Python anschauen und lernen kann, konnte ich innerhalb weniger Stunden die groben Grundlagen von Python lernen. Bei Problemen mit dem Programm habe ich oft lange nach einer Lösung gesucht, aber danach hat es umso mehr Freude gemacht, wenn es funktioniert hat.

Aus diesem Projekt konnte ich sicher sehr viel lernen für die Zukunft und es hat viel Spass gemacht diese Poolsteuerung zusammen zu realisieren. Deshalb bin ich mit dem Endergebnis des Projekts sehr zufrieden.

11.1.2 Fazit Marc

Das Endergebnis ist meiner Meinung sehr gelungen. Anfangs hatte ich noch nie etwas mit Python zu tun. Ein gewisses Grundwissen von programmieren hatte ich, da wir in der Berufsschule meistens S7 programmiert haben. Mithilfe von YouTube Videos versuchte ich, mich mit Python auseinanderzusetzen. Anfangs funktionierte dies sehr gut, doch mit der Zeit stellte ich fest, dass ich immer weniger vorankam. Da Jun bereits viel fortgeschrittener war als ich, übernahm er diese Aufgabe. Mittlerweile muss ich festgestellt, dass ich mit zu wenig Elan diese Aufgabe angegangen bin und werde dies, bei einem nächsten Projekt für meine Teamkameraden und auch für mich selbst, ändern. Trotz der Schwierigkeiten mit Python, konnte ich mit dem Aufbau des Schaltschranks und der Verdrahtung einiges zu dem Projekt beitragen.

11.1.3 Fazit Jun

Ich habe aus dieser Projektarbeit viel gelernt: Programmieren, Software entwickeln, ein kompletter Projektablauf, Teamarbeit bei Projekten, und eine grosse Dokumentation zu schreiben. Ich habe mehr über meine Stärken und meine Schwächen erfahren.

Ich habe herausgefunden, dass Unvorhergesehenes sehr viel Zeit in Anspruch nehmen kann. Dies muss ich bei der nächsten Projektarbeit berücksichtigen. Unser, anfangs "simples" Python Programm, entpuppte sich zum Schluss, als sehr umfangreich.

Ich weiss jetzt, dass ich Herausforderungen annehmen kann. Für unser Pool Projekt gibt es keine Vorlage. Die ganze Software musste ich selbst schreiben. Dabei habe ich viel ausprobiert. Bei Fehlern und Problemen habe ich mir auf Programmierforen (z.B. [Stackoverflow](#)) Vorlagen und Lösungsansätze gesucht. So habe ich während der Entwicklung immer mehr Erkenntnisse gewonnen. Und die Freude beim Gelingen war gross.

Ich habe gemerkt, dass es mir Spass macht, zu programmieren. In Zukunft und ausserhalb dieser Arbeit möchte ich das RaspyPool Programm weiterentwickeln.

Und zu guter Letzt:

So eine Arbeit ist eigentlich nie fertig, man muss sie für fertig erklären, wenn man nach Zeit und Umständen das Mögliche getan hat.

(~ Johann Wolfgang von Goethe)

Unser Projekt – RaspyPool Steuerung von Jens, Jun und Marc



Abbildung 10: unser fertiges Projekt



Wir danken Jens, Marc und Jun für ihre wertvolle Arbeit und die grossartige Poolsteuerung! Wir können es kaum erwarten, diese zu bedienen – auch Technikbanause Mama Heule und Poolsitter Opa Franz.

Herzliche Einladung zur Pooleinweihung.
Datum folgt.

Familie Heule

12 Quellenverzeichnis

12.1 RaspyPool Projekt

GitHub Repository

➤ <https://github.com/Keshmc/RaspyPool>

12.2 Anhang

Der Anhang wurde ins Word Dokument verknüpft. Leider kann dieser von einem PDF nicht geöffnet werden. Deshalb schicken wir Ihnen den Anhang separat mit.

IDPA Projektskizze



IDPA_Poolsteuerun
g_Projektskizze.doc

IDPA Journal und Zeitplan



IDPA_Poolsteuerun
g_Journal_und_Zeitp

IDPA Schema und Dispo



IDPA_Poolsteuerun
g_Schema.pdf

IDPA Erklärvideo GitHub in PyCharm integrieren



IDPA_Poolsteuerun
g_Tutorial_PyCharm

12.3 Verwendete Programm / Webseiten

python

- <https://www.python.org/>

Raspian OS

- <https://www.raspberrypi.org/software/raspberry-pi-desktop/>

Balena Etcher

- <https://www.balena.io/etcher/>

PyCharm by JetBrains

- <https://www.jetbrains.com/de-de/pycharm/>

GitHub

- <https://github.com/>

GitHub Projekt Atlas Scientific (pH-Sensor)

- <https://github.com/Atlas-Scientific/Raspberry-Pi-sample-code>

Anleitung On-Screen Keyboard Raspberry Pi

- <https://pimylifeup.com/raspberry-pi-on-screen-keyboard/>

12.4 Bildverzeichnis

Jedes Bild einer fremden Quelle ist mit dem Link versehen.

12.5 Datenblätter der Technikbox

Filterpumpe Aqua Vario Plus

- https://www.poolakademie.de/filteranlagen/filterpumpen/filterpumpe-serie-aqua-vario-plus-drehzahl-von-1-000-2-850-u-min_1118_8646

UVC-Filter 75W

- https://www.poolakademie.de/wasserpflege/uv-c-entkeimung/conzero-uv-c-edelstahl-75-w-entkeimung-mit-weniger-chemie_1206_9864

Wärmepumpe

- https://www.poolakademie.de/heizung/waermepumpen/waermepumpen-serie-conzeroturbosilent-mit-wifi-extrem-leise-und-sehr-effizient-durch-neueste-stufenlose-inverter-technologie-cop-bis-16-mit-grauem-aluminium-gehaeuse_1535_17729

Website von Atlas Scientific (pH-Sensor)

- https://atlas-scientific.com/?gclid=Cj0KCQiAx9mABhD0ARIsAEfpavQdE6HVVLZ8E5P1FWRfjtli96Pgr8NTIDPb_TMSZln1JtKslxaWsywaAjQCEALw_wcB