Ein Bild, das Grafiken, Schrift, Text, Logo enthält.

Automatisch generierte Beschreibung

Diplomarbeit in höherer Abteilung für Informatik

Green Crypto Mining

## Überschussbasierte Crypto-Mining Software für eine Photovoltaik-Anlage und Speicher mit InfluxDB und MongoDB

Diplomarbeitsnummer:

Autoren:

Alessandro Davare

Sarah Hagenhofer

Betreuer:

Dipl.-Ing. Raimund Eigner

Prof. Mag. Robert Gröbl

Ausgeführt im Jahr 2023/24

Vorwort

Die Aufgabe dieses Projektes besteht darin, eine Software für unseren Auftraggeber Raimund Eigner zu entwickeln, welche Daten einer geplanten PV-Anlage mit dazugehörigem Stromspeicher erfasst und anhand der ausgelesenen Daten eine automatische Steuerung der für uns bereitgestellten Mining Asics der Marke Whatsminer und Antminer bietet. Ebenfalls soll es eine Möglichkeit geben, die Rechner manuell zu konfigurieren und die Hashrate von jeden einzelnen sowie von allen Geräten gleichzeitig verändern zu können

…

Abstracts

The goal of this project is the development of a software for our partner Raimund Eigner, which records data from a planned PV system with associated electricity storage and based on the data read, offers automatic control of the mining asics from Whatsminer and Antminer provided to us. here should also be an option to configure the miners manually and to be able to change the hashrate of each individual and all devices at the same time…

# Kurzfassung/Aufgabenstellung

## 5AHINF – Reife und Diplomprüfung 2023/24

|  |  |  |
| --- | --- | --- |
| Thema | Entwicklung einer Software zum überschussbasierten Crypto-Mining für eine Photovoltaik-Anlage mit Speicher | |
| Aufgabenstellung (Kurzfassung) | Ziel ist die Planung und Programmierung einer Software, welche auf einem handelsüblichen RasPi zur Steuerung von Crypto-Mining-Hardware umgesetzt werden soll. Das System soll Daten aus einer Photovoltaik-Anlage zur optimier­ten Steuerung nutzen. Einerseits die Momentan Produktionsleistung der An­lage sowie die Momentan Kapazität des Speichers.  Die intelligente und adaptive Steuerung der Hashrate der Mining-Hardware ba­siert auf momentan produzierter und gespeicherter Energie, der aktuellen Ta­ges- und Jahreszeit sowie die momentan betrieblich benötigte Leistung (saiso­nal starke Schwankungen). Dadurch soll lediglich Überschussenergie sowie nicht benötigte Speicherkapazität des Systems (über Nacht) für Mining zur Ver­fügung gestellt werden. | |
| Kandidaten / Kandidatinnen | | Betreuer / Betreuerin |
| Sarah Hagenhofer | | Dipl.-Ing. Raimund Eigner  Prof. Mag. Robert Gröbl |
| Alessandro Davare | | Dipl.-Ing. Raimund Eigner  Prof. Mag. Robert Gröbl |

|  |
| --- |
| Externe Kooperationspartner |
| Firma / Institution: - |
| Betreuer / Kontaktperson: Raimund Eigner |
| Schriftliche Kooperationsvereinbarung liegt vor: / |
| **Budget:** 10.000 € |
| Bedeckung durch: Dipl.-Ing. Raimund Eigner |
| **Geplante Verwertung der Ergebnisse:**  Inbetriebnahme der Software |

Erklärung

Die unterfertigten Kandidaten haben gemäß § 34 (3) SchUG in Verbindung mit § 22

(1) Zi. 3 lit. b der Verordnung Uber die abschließenden Prüfungen in den

berufsbildenden mittleren und höheren Schulen, BGBl. II Nr. 70 vom 24.02.2000

(Prüfungsordnung BMHS), die Ausarbeitung einer Diplomarbeit mit der umseitig

angeführten Aufgabenstellung gewählt.

**Die Kandidaten / Kandidatinnen nehmen zur Kenntnis, dass die Dip­lomarbeit in eigenständiger Weise und außerhalb des Unterrichtes zu bearbeiten und anzufertigen ist, wobei Ergebnisse des Unterrich­tes mit einbezogen werden können.**

**Die Abgabe der vollständigen Diplomarbeit hat bis spätestens**

**05.04.2024**

**beim zuständigen Betreuer zu erfolgen.**

**Die Kandidaten nehmen weiters zur Kenntnis, dass gemäß § 9 (6) der Prüfungsordnung BMHS nur der Schulleiter bis spätestens Ende des vorletzten Semesters den Abbruch einer Diplomarbeit an­ordnen kann, wenn diese aus nicht beim Prüfungskandidaten (bei den Prüfungskandidaten) gelegenen Gründen nicht fertiggestellt werden kann.**

|  |  |
| --- | --- |
| Kanditaten / Kandidatinnen | Unterschrift |
| Alessandro Davare |  |
| Sarah Hagenhofer |  |

# Inhaltsverzeichnis

[Überschussbasierte Crypto-Mining Software für eine Photovoltaik-Anlage und Speicher mit InfluxDB und MongoDB 1](#_Toc159306466)

[Kurzfassung/Aufgabenstellung 3](#_Toc159306467)

[5AHINF – Reife und Diplomprüfung 2023/24 3](#_Toc159306468)

[Thema 3](#_Toc159306469)

[Entwicklung einer Software zum überschussbasierten Crypto-Mining für eine Photovoltaik-Anlage mit Speicher 3](#_Toc159306470)

[Aufgabenstellung 3](#_Toc159306471)

[(Kurzfassung) 3](#_Toc159306472)

[Kandidaten / Kandidatinnen 3](#_Toc159306473)

[Betreuer / Betreuerin 3](#_Toc159306474)

[Externe Kooperationspartner 4](#_Toc159306475)

[Inhaltsverzeichnis 6](#_Toc159306476)

[1. [ Planung ] 13](#_Toc159306477)

[1.1.1. Aufgabenstellung 13](#_Toc159306478)

[1.1.2. Projektfindung 13](#_Toc159306479)

[1.1.3. Ausgangslage 13](#_Toc159306480)

[1.1.4. Ziele 14](#_Toc159306481)

[1.1.5. Funktionale Anforderungen 14](#_Toc159306482)

[Muss-Ziele 14](#_Toc159306483)

[ Recherche: Hersteller APIs der einzelnen Komponenten 14](#_Toc159306484)

[ Kommunikationsfähigkeit: RasPi-GCMC-Controller / Mining-Hardware / PV-Controller / Stromspeicher-Controller 14](#_Toc159306485)

[ GCMC Maintenance-App mobile und Browser Webapp 14](#_Toc159306486)

[Soll-Ziele 15](#_Toc159306487)

[ Adaptive Logik für prädikative Stromproduktion und Verbrauch auf Basis historischer Daten 15](#_Toc159306488)

[ Einbindung von ROI (return of investment) Daten in die App auf Basis von Crypto-Kursen und Strompreisen 15](#_Toc159306489)

[1.1.6. Nichtfunktionale Anforderungen 15](#_Toc159306490)

[1.2. Projektorganisation 15](#_Toc159306491)

[1.2.1. Rollenverteilung 15](#_Toc159306492)

[1.2.2. Aufgabenverteilung 16](#_Toc159306493)

[Alessandro Davare 16](#_Toc159306494)

[Sarah Hagenhofer 16](#_Toc159306495)

[1.2.3. Zeitplan 16](#_Toc159306496)

[2. Projektrecherche 16](#_Toc159306497)

[2.1. Betriebssysteme (Davare) 16](#_Toc159306498)

[2.1.1. Windows 10/11 (Hagenhofer) 17](#_Toc159306499)

[2.1.2. Linux (Davare) 17](#_Toc159306500)

[2.1.2.1. EndeavourOS „Galileo“ (Davare) 18](#_Toc159306501)

[2.1.3. Android (Hagenhofer) 18](#_Toc159306502)

[2.2. Anwendungen (Davare) 19](#_Toc159306503)

[2.2.1. Programmierumgebungen (Davare) 19](#_Toc159306504)

[2.2.1.1. Visual Studio (Davare) 19](#_Toc159306505)

[2.2.1.2. Visual Studio Code 20](#_Toc159306506)

[2.2.1.3. Jetbrains PyCharm 20](#_Toc159306507)

[2.2.2. Microsoft Office Produkte 20](#_Toc159306508)

[2.2.2.1. Microsoft Word (Hagenhofer) 20](#_Toc159306509)

[2.2.2.2. Microsoft Powerpoint (Hagenhofer) 20](#_Toc159306510)

[2.2.2.3. Microsoft Excel (Hagenhofer) 21](#_Toc159306511)

[2.2.2.4. Microsoft OneDrive 21](#_Toc159306512)

[2.2.3. Weitere Anwendungen 21](#_Toc159306513)

[2.2.3.1. Figma (Hagenhofer) 21](#_Toc159306514)

[2.2.3.2. Draw.io (Hagenhofer) 21](#_Toc159306515)

[2.2.3.3. Adobe Photoshop (Hagenhofer) 22](#_Toc159306516)

[2.2.3.4. MongoDB Compass (Hagenhofer) 22](#_Toc159306517)

[2.2.3.5. Postman (Hagenhofer) 22](#_Toc159306518)

[2.2.3.6. Discord (Hagenhofer) 22](#_Toc159306519)

[2.2.3.7. Google Chrome (Hagenhofer) 22](#_Toc159306520)

[2.2.3.8. Microsoft Edge 22](#_Toc159306521)

[2.2.3.9. OpenVPN 23](#_Toc159306522)

[2.2.3.10. Google Suchmaschine 23](#_Toc159306523)

[2.2.3.11. Oracle VM Virtual Box 23](#_Toc159306524)

[2.2.4. Projekt Verwaltung 23](#_Toc159306525)

[2.2.4.1. Git 23](#_Toc159306526)

[2.2.4.2. GitHub 23](#_Toc159306527)

[2.3. Programmiersprachen 24](#_Toc159306528)

[2.3.1. Python 24](#_Toc159306529)

[2.3.2. C# 24](#_Toc159306530)

[2.3.3. JavaScript 24](#_Toc159306531)

[2.4. Datenbanken (Davare) 24](#_Toc159306532)

[2.4.1. MongoDB 25](#_Toc159306533)

[2.4.2. InfluxDB 25](#_Toc159306534)

[2.5. Frameworks 25](#_Toc159306535)

[2.5.1. .NET 25](#_Toc159306536)

[2.5.2. Blazor 25](#_Toc159306537)

[2.5.3. MudBlazor 25](#_Toc159306538)

[2.5.4. Swagger 26](#_Toc159306539)

[2.6. Nuget’s 26](#_Toc159306540)

[2.6.1. Newtonsoft.Json 26](#_Toc159306541)

[2.6.2. MongoDB Driver 26](#_Toc159306542)

[2.6.3. Asp.Net.Cors 26](#_Toc159306543)

[2.6.4. InfluxDB Client 26](#_Toc159306544)

[2.7. Systemkomponenten 26](#_Toc159306545)

[2.7.1. Stromkomponenten 26](#_Toc159306546)

[2.7.2. Miner 26](#_Toc159306547)

[2.7.2.1. Whatsminer 26](#_Toc159306548)

[2.7.2.2. Antminer 26](#_Toc159306549)

[2.7.3. Weitere Komponenten 27](#_Toc159306550)

[2.7.3.1. OpenVPN Verbindung 27](#_Toc159306551)

[2.7.4. Revolution Pi 27](#_Toc159306552)

[3. Umsetzung 27](#_Toc159306553)

[3.1.2. [ Einleitung ] 28](#_Toc159306554)

[3.1.3. Motivation und Zielsetzung 29](#_Toc159306555)

[3.1.4. Arbeitsdokumentation 30](#_Toc159306556)

[Recherche 30](#_Toc159306557)

[ PV-Paneele / Hagenhofer 30](#_Toc159306558)

[ Speicher / Hagenhofer 30](#_Toc159306559)

[ Miner APIs / Davare 33](#_Toc159306560)

[ Bitmain Whatsminer 33](#_Toc159306561)

[3.1.5. Zusammenfassung 34](#_Toc159306562)

[3.1.6. Literaturverzeichnis 35](#_Toc159306563)

[3.1.7. Anhänge 36](#_Toc159306564)

[3.1.8. [ Selbstständigkeitserklärung ] 37](#_Toc159306565)

**NOTES AN UNS WÄHREND WIR SCHREIBEN:**

* **Umbrüche, Schriften etc. vor Abgabe noch einmal korrigieren**
* **Überschriften 1. Ordnung Arial 16, fett, Abstand** 🡪 **12 Pt. links (0,42 cm) und 5 Pt. rechts (0,18 cm)**
* **Überschriften 2. Ordnung Arial 14 fett, Abstand 🡪 12 Pt. links und 5 Pt. rechts**
* **Standard-Schrift: Arial 12, Blocksatz, Zeilenabstand 1.5,**

[ Abbildungsverzeichnis ]

# [ Planung ]

# Aufgabenstellung

# Projektfindung

Das Interesse an Kryptowährungen jeglicher Art als Geldanlage oder generell als sichere sowie verschlüsselte digitale Währung ist heute auf einem All-Time-High und immer mehr und mehr Menschen wollen sich damit auseinandersetzen, genauso wie unser Auftraggeber Raimund Eigner.

Die Idee hinter dieser Diplomarbeit ist die alternative Verwertung von PV-produzierten Überstrom. Nachdem die Einspeisetarife wieder gefallen sind und die Crypto-Adoption im Vormarsch ist, kann diese alternative Verwertung von Überstrom durchaus als ein interessantes Pilotprojekt gesehen werden.

# Ausgangslage

Der Auftraggeber plant derzeit eine PV-Anlage für einen bestehenden Ferienhauskomplex. Der Strombedarf des Betriebes schwankt saisonal sehr stark, wobei die PV-Anlage den Großteil des nötigen Strombedarfs decken soll. Die Anlage liefert eine Engpassleistung von ca. 70 kWp und wird über einen Stromspeicher von ca. 60 Ah verfügen.

In den Wintermonaten zwischen Dezember und März wird ein zusätzlicher Stromzukauf aufgrund der wenigen Sonnenstunden und des niedrigen Sonnenstandes notwendig sein. In diesen Monaten beträgt der Verbrauch im Schnitt 140 kWh. In den Zwischensaisonen im Frühjahr und im Herbst sink der tägliche Verbrauch auf ca. 11 kWh, was zu einer Überkapazität von geschätzten 340 kWh pro Tag führt. Der Verbrauch in der Sommersaison beträgt 130 kWh. Mit den Überkapazitäten soll in erster Instanz der Stromspeicher gefüllt werden und folglich die Restkapazität für Mining-ASICs zur Verfügung stehen.

# Ziele

Das Projektziel ist es, über einen Raspberry Pi 4 den Controller umzusetzen, welcher folglich im Headless-Mode (ohne Maus, Tastatur, Bildschirm) im 24/7 Betrieb läuft und über entsprechende APIs der Hersteller (PV-Steuermodul sowie Batterie-Speicher-Steuermodul) die nötigen Daten bzgl. Verfügbarer Leistung – sowie über die APIs der Mining-ASICs die nötigen Steuerbefehle bzgl. Anpassung der Hashrate – auswertet bzw. absetzt

Der Zugriff soll über eine Mobile-App sowie über den Browser auf eine Maintenance Page zur Einstellung und zum Monitoring des Systems umgesetzt werden. Es sollen auch Status-Meldungen von den Minern als auch der PV-Anlage sowie des Stromspeichers angezeigt werden

# Funktionale Anforderungen

## Muss-Ziele

Diese Ziele müssen nach Fertigstellung des Projektes erfüllt sein, damit es als gelungen betrachtet werden kann.

### Recherche: Hersteller APIs der einzelnen Komponenten

Es muss eine umfangreiche Recherche über die verfügbaren APIs der Hersteller durchgeführt und dokumentiert werden.

### Kommunikationsfähigkeit: RasPi-GCMC-Controller / Mining-Hardware / PV-Controller / Stromspeicher-Controller

Es muss der Softwarecontroller mit den nötigen Interfaces zu den Systemkomponenten voll funktionsfähig auf dem RasPi umgesetzt werden.

### GCMC Maintenance-App mobile und Browser Webapp

Es muss eine App mit responsivem Design für mobile Endgeräte und Browser umgesetzt werden, über welche diverse Einstellungen vorgenommen- und das Monitoring des Systems dargestellt werden kann (Verbrauchskurve, geminte Crypto Tokens, Produktionsleistung, Ladestand Stromspeicher, etc.)

## Soll-Ziele

Diese Ziele sind optional, an ihnen wird nur gearbeitet, wenn nach der Bearbeitung der Muss-Ziele noch Zeit zur Verfügung steht.

### Adaptive Logik für prädikative Stromproduktion und Verbrauch auf Basis historischer Daten

Es soll eine Logik integriert werden, welche auf Basis von Jahreszeit (Produktionsleistung aufgrund des Sonnenstandes und der Sonnenstunden) und saisonaler Auslastung (Verbrauchsdaten im saisonalen Betrieb) einen Stellwert für eine optimale Wahl der Hashrates definiert. Es werden daher historische Daten verwendet um eine Vorhersage (predictive algorithm) über Produktion und Verbrauch treffen zu können.

### Einbindung von ROI (return of investment) Daten in die App auf Basis von Crypto-Kursen und Strompreisen

Es soll eine Rentabilitätsdarstellung in der App umgesetzt werden.

# Nichtfunktionale Anforderungen

…

# Projektorganisation

# Rollenverteilung

Auftraggeber Dipl.-Ing. Raimund Eigner

Hauptbetreuer Dipl.-Ing. Raimund Eigner

Nebenbetreuer Prof. Mag. Robert Gröbl

Projektleiter Alessandro Davare

Projektteam Alessandro Davare, Sarah Hagenhofer

# Aufgabenverteilung

## Alessandro Davare

* Recherche und Dokumentation: Mining Asics bzw. deren APIs
* Aufsetzung und Konfiguration Raspberry Pi
* Miner Controller
* Webapp für Monitoring sowie Maintenance Page
* Möglichkeit mehrere Asics hinzuzufügen
* Einstellungsseite User
* Return of Investment (Soll Ziel)

## Sarah Hagenhofer

* Recherche und Dokumentation: PV-Anlage und Speicher
* PV Controller
* Speicher Controller
* Mobile App für Monitoring sowie Maintenance Page
* Vergangene Wetterdaten in Datenbank (Soll Ziel)
* Vorhersage für „beste“ Einstellungen anhand Wetterdaten (Soll Ziel)

# Zeitplan

## Recherche

Die ausführliche Recherche sowie die Dokumentation zu den Schnittstellen der einzelnen Komponenten dieser Arbeit ist der erste und grundlegende Schritt dieser Arbeit.

## Vorarbeit

Als Vorarbeit wird das Design und das Implementieren der Datenbankmodelle sowie das Installieren von benötigten Anwendungen bezeichnet. Ebenso zählt hier das Einrichten des Raspberry PI’s.

## Implementierung Apps

Der erste große Schritt ist das Planen, das Designen und Implementieren der Mobile sowie der Webapp. Dieser Schritt gilt als erfolgreich, wenn alle Funktionen mit der angebundenen Datenbank funktionieren.

## Implementierung Controller

Der zweite große Schritt ist das Implementieren der Lokalen Software zur Steuerung der Miner. Hier sollen die Hersteller APIs benutzt werden, um Logs der Miner auszulesen und die Miner mit gespeicherten Daten der Datenbank zu steuern. Ebenso sollen Stromdaten der PV-Anlage sowie dem Stromspeicher ausgelesen werden und in die Datenbank gespeichert werden. Wenn die Geräte erfolgreich angesteuert werden können, gilt dieser Punkt als erledigt.

## Deployment

Der letzte Schritt dieser Arbeit wird das Aufsetzen des fertigen Projektes auf einem Industrieklassen Raspberry PI. Sollte unsere Software völlig funktional auf dem PI laufen ist das Projekt inklusiver aller Muss Ziele abgeschlossen.

## (Soll-Ziel) Return-of-Investment (ROI) Rechner

Es soll durch eine ROI-Berechnung mit aktuellen Crypto Kursen sowie einem gegebenen Strompreis erfolgen. Hier soll der Nutzer sehen, wie lange er Minen müsste bis er Profit erreicht. Dies ist jedoch eines der Soll-Ziele und wird erst umgesetzt, wenn das Projekt abgeschlossen ist und genug Zeit verbleibt.

## (Soll-Ziel) Predictive Algorithm

Es sollen Wetterdaten vom vergangenen Jahr ausgelesen und dazu benutzt werden, eine vorrausichtlich Ideale Einstellung für die Miner einzustellen. Dies ist jedoch das zweite Soll-Ziele und wird erst umgesetzt, wenn das Projekt abgeschlossen ist und genug Zeit verbleibt.

# Projektrecherche

# Betriebssysteme (Davare)

Ein Betriebssystem bringt die grundlegenden Funktionalitäten in die unterschiedlichsten Geräte, sodass die Hardware überhaupt laufen kann und wir Menschen weitere Funktionalitäten in die Maschinen bringen können. Diese Funktionalitäten gehen von Ressourcenverwaltung, also der Nutzung eingebauter Hardware wie Speicherplatz, RAM-Speicher, CPU-Cache-Speicher, etc. zu einer Benutzeroberfläche bzw. einem einfachen Terminal, um mit der Maschine interagieren zu können, sowie einer Dateiverwaltung um Dateien geordnet zu speichern.

[https://www.ionos.at/digitalguide/server/knowhow/was-ist-ein-betriebssystem/]

Bei unserem Projekt ist es notwendig mehrere Betriebssysteme zu nutzen, da unter Windows 10 bzw. Windows 11 programmiert, auf einer Linux Distribution für Raspberry PI (Raspbian) deployed und durch Apps auf mobilen Android Geräten angesehen wird.

# Windows 10/11 (Hagenhofer)

Windows 10 und Windows 11 stammen vom US-Unternehmen Microsoft und sind die neusten Mitglieder der Windows-Reihe, letztere ist die aktuell neuste Version der Reihe. Von insgesamt drei Geräten, auf denen wir hauptsächlich am Projekt arbeiten, laufen zwei auf Windows 10, während eines bereits auf Windows 11 upgegradet wurde.

…

# Linux (Davare)

Im Gegensatz zu Windows spricht man, wenn man über Linux redet ein Betriebssystem an, sondern Betriebssysteme, die auf dem Linux „Kernel“ basieren. Der Kernel ist in diesem Fall das Herzstück einer jeden Linux Distribution und bietet die Hauptfunktionen des Systems. Linux ist durch die die Open Source Entwicklung des Linux „Kernels“ mit einer riesigen Anzahl von freiwilligen Programmierern bekannt, da es dieses Betriebssystem sehr sicher sowie Personalisierbar macht. Theoretisch ist es so möglich, sich seine optimale Distribution je nach Nutzen, Design, und Oberfläche zu bauen.

[https://www.redhat.com/de/topics/linux/what-is-linux]

Eine Linux Distribution (abgekürzt Distro.) ist ein richtiges Betriebssystem, dass von einem Menschen benutzt werden kann. Jede Distro wurde rund um den Linux Kernel gebaut und je nach Zielgruppe, deren Bedürfnisse und Zweck entwickelt. Grundlegend benutzt Linux ein Command Line Interface (Kurz. CLI), um Befehle entgegenzunehmen, wobei es auch Distros gibt, welche ein „Graphical User Interface“ (Kurz. GUI) anbieten.

[https://www.computerweekly.com/de/definition/Linux-Distribution]

Das GUI ist an diesem Punkt aber auch nicht einheitlich und wird durch sogenannte Desktop Environments geregelt. Das sind zusammengebundene Pakete von einfachen UI-Komponenten wie Icons, Tool Bars und auch meistens Window-Manager. Diese können wiederum auch wieder je nach Bedarf konfiguriert und installiert werden, wobei dies für diese Arbeit nicht relevant ist.

[https://wiki.archlinux.org/title/desktop\_environment]

Beispiele für bekannte Linux Distributionen sind:

* Debian: Relativ altes, aber noch sehr aktive Distro, welches als sehr stabil gilt. Sie wird für PCs, Server, Embedded Systems, etc. bis heute von Erfahreneren Menschen verwendet.
* Ubuntu: Ein auf Debian basiertes Betriebssystem, dass dafür entwickelt wurde Linux für die breite Masse zugänglicher zu machen. Wird ebenfalls für Personal Computer und Server Systeme benutzt.
* Kali Linux: Das „Hacker-Linux“ basiert wie Ubuntu ebenso auf Debian und wird hauptsächlich von Profis für Penetration Tests und digitale Forensik benutzt.
* Linux Mint: Auf Ubuntu basierend wurde „Mint“ damals als attraktive Alternative zu Microsoft XP entwickelt. Sie wurde extra so entwickelt, dass es Windows Nutzern leichter viel, auf Linux umzusteigen und mehr und mehr Menschen mit Linux vertraut zu machen.
* Arch Linux: An sich ist „Arch“ nur eine minimalistische Linux Distro, welche die nötigsten Tools mit sich bringt, aber es sticht mit seiner Möglichkeit von Individualisierung hervor. Der Nutzer kann sich das Betriebssystem genau so zusammenstellen, wie er es braucht und wie genau er es möchte.

[https://www.heise.de/tipps-tricks/Linux-Betriebssysteme-eine-Uebersicht-4119937.html

# EndeavourOS „Galileo“ (Davare)

EndeavourOS ist eine relativ neue Linux Distribution welche regelmäßige Updates bekommt. Sie basiert auf Arch Linux und erleichtert es dem Benutzer Arch zu installieren. Mit Galileo bekommt man ein bereits fertiges System geliefert, welches nach der Installation direkt benutzt werden kann, im Gegensatz zu Arch Linux.

[https://endeavouros.com/]

Dieses Betriebssystem wurde nach einer gescheiterten Arch Linux Installation ausgewählt, da es ein ansprechendes Design bzw. eine angenehme User Experience bietet, ohne viele Konfigurationen tätigen zu müssen.

# Android (Hagenhofer)

Android ist eines der gängigsten Betriebssysteme für mobile Geräte, es wird vom Unternehmen Google vertrieben. Zum Stand 2024 trägt Android unter Mobile-Betriebssystemen einen Marktanteil von über 70% [Demandsage, Zugriff: 14.02.2024]. Technisch basiert Android auf einem Linux-Kernel, die verwendete Laufzeitumgebung ist Android Runtime, kurz ART. Die meisten Anwendungen werden in den Sprachen Java, C oder auch C++ geschrieben, auch bietet Android selbst Ressourcen an, die bei der App-Entwicklung für Android-Geräte helfen [Android Developer, Zugriff: 14.02.2024; Wikipedia Android, Zugriff: 14.02.2024].

# Anwendungen (Davare)

Für die Umsetzung dieser Arbeit werden einige externe Softwares benutzt, um die genannten Ziele zu erreichen. Die folgenden Programme werden hauptsächlich für das Programmieren unserer Software, Planung bzw. Design der Software sowie Datenaustausch und Projektumsetzung benutzt.

# Programmierumgebungen (Davare)

Da die Hauptaufgabe dieses Projekts darin besteht, eine Anwendung zur Steuerung von sogenannten Mining-Asics zu programmieren, sind mehrere Anwendungen erforderlich, die für die Programmierung genutzt werden können. Integrated Development Environments (IDEs) bieten Entwicklern eine Vielzahl von Tools, die ihnen direkt oder indirekt bei der Programmierung helfen. Beispiele für solche Tools sind Texteditoren mit Code-Autovervollständigung und Syntaxhervorhebung, Compiler für die Code-Übersetzung, Projekt- und Dateiverwaltung sowie viele weitere nützliche Funktionen. Diese Tools erleichtern die Entwicklung und Optimierung des Programmcodes, was zu einer effizienteren Entwicklung und einem besseren Endprodukt führt.

[https://www.pi-informatik.berlin/pi-lexikon/softwareentwicklung/was-ist-eine-entwicklungsumgebung/]

In diesem Projekt werden IDEs wie Visual Studio, Visual Studio Code oder JetBrains PyCharm verwendet, um die benötigten Anwendungen zu entwickeln. Diese IDEs bieten eine umfassende Entwicklungsumgebung, die alle erforderlichen Tools und Funktionen bereitstellt, um die Entwicklungsziele zu erreichen.

# Visual Studio (Davare)

Die von Microsoft entwickelte Integrated Development Environment (IDE) ist die erste Anlaufstelle für die Umsetzung von C++, C#, J#, Visual Basic und .NET-Projekten. Im Gegensatz zu anderen bereits vorhandenen Programmierumgebungen bietet Visual Studio nicht nur einen Compiler, sondern auch Codeabschluss-Tools, grafische Designer und viele weitere Quality-of-Life (QoL)-Funktionen, die Entwickler dabei unterstützen, die Projektentwicklung so einfach wie möglich zu gestalten.

[https://visualstudio.microsoft.com/de/vs/]

Durch die Möglichkeit, zusätzliche Pakete (sog. Workloads) zu Visual Studio hinzuzufügen, eröffnen sich viele weitere Möglichkeiten, die mit der Standardversion nicht verfügbar sind. Dies umfasst unter anderem die Entwicklung für Azure, Python und Node.js, Desktop- oder Mobile-Entwicklung mit C++, sowie Spieleentwicklung mit Unity oder anderen ähnlichen Engines.

[https://learn.microsoft.com/en-us/visualstudio/install/modify-visual-studio?view=vs-2022]

In diesem Projekt wird die Software für die Umsetzung der Webanwendung in .NET MudBlazor verwendet.

# Visual Studio Code

…

# Jetbrains PyCharm

…

# Microsoft Office Produkte

# Microsoft Word (Hagenhofer)

Microsoft Word ist ein Textverarbeitungsprogramm und Teil des „Microsoft Office“-Pakets. Es bietet zahlreiche hilfreiche Funktionen und erlaubt einfaches Arbeiten mit Texten und Berichten. Auch für die gänzliche Dokumentation unserer Diplomarbeitsdokumentation haben wir Microsoft Word gewählt, hauptsächlich weil wir damit bereits jahrelang gearbeitet haben, aber auch aufgrund der anpassbaren und wiederverwendbaren Formatvorlagen und Möglichkeiten der Absatzgliederung und -nummerierung. Für das eigene Mitschreiben und Dokumentieren von Notizen und Ideen haben wir genauso das Programm verwendet.

# Microsoft Powerpoint (Hagenhofer)

Microsoft Powerpoint ist das Microsoft-Office-Produkt zur Erstellung von Präsentationen. Es erlaubt das Gestalten von verschiedenen Folien, sowie deren Übergänge, auch gibt es zahlreiche hilfreiche Funktionen und Textbearbeitungsmöglichkeiten. Während unserer Arbeit haben wir das Programm für alle Diplomarbeitspräsentationen verwendet.

# Microsoft Excel (Hagenhofer)

Mit Microsoft Excel lassen sich Tabellen erstellen und mit Daten befüllen. Man kann die Daten auch beispielsweise in Graphen darstellen, Rechenaktionen mit ihnen durchführen und vieles mehr. Grundsätzlich haben wir im Rahmen der Diplomarbeit nur für private Zwecke Microsoft Excel verwendet, beispielsweise für das eigene Projekttagebuch oder die Ideensammlung zu Beginn.

# Microsoft OneDrive

…

# Weitere Anwendungen

# Figma (Hagenhofer)

Das Programm Figma ermöglicht das einfache und vollständige Designen von Prototypen für verschiedene Apps. Geboten werden verschiedenste Tools, unter anderem um Bereiche zu füllen oder Steuerelemente einzufügen, mit denen der User später interagieren können soll, auch lassen sich beispielhaft Abläufe definieren. Online gibt es außerdem von Communitymitgliedern Designvorlagen und -ideen, an denen man sich inspirieren und ein Beispiel nehmen kann [Figma, Zugriff: 15.02.2024]. In unserer Planungsphase haben wir für Mobile-App und Web-App unsere ersten Design-Mock-Ups erstellt und damit grundlegende Dinge wie beispielsweise die Farbwahl oder den Aufbau festgelegt.

# Draw.io (Hagenhofer)

Draw.io, auch Diagrams.net genannt, ist ein Tool, mit dem verschiedene Diagramme für das eigene Projekt erstellt werden können [Wikipedia Diagrams.net, Zugriff: 15.02.2024]. Es gibt viele Vorlagen, um Komponenten und deren Verbindungen darzustellen, die verschiedenen Abschnitte können außerdem beispielsweise farblich gestaltet werden, was die Darstellung umso übersichtlicher macht. Für zahlreiche unserer Diplomarbeitspräsentationen haben wir das Programm für Ablauf-Diagramme und Aufbau-Darstellungen verwendet.

# Adobe Photoshop (Hagenhofer)

Das Bildbearbeitungsprogramm auf dem Hause Adobe bietet eine Vielzahl von Funktionen, mit denen man Bilder editieren kann. Beispielsweise lassen sich Filter anwenden, Bilder zuschneiden oder erweitern, Bereiche ausschneiden oder ersetzen, aber auch Bereiche retuschieren [Adobe Photoshop, Zugriff: 15.02.2024]. Für die kreative Gestaltung des Diplomarbeitsposters war das Programm ideal.

# MongoDB Compass (Hagenhofer)

MongoDB Compass ist ein Tool, um die Daten der eigenen MongoDB-Datenbanken grafisch anzuzeigen [MongoDB, Zugriff: 20.02.2024]. Einige Funktionen sind beispielsweise klassische Datenabfragen mit einer „Query“, das Hinzufügen oder Löschen von Daten, aber auch das Erstellen von „Pipelines“. Es ist ein sehr nützliches Tool zum Arbeiten mit MongoDB, das wir vor allem verwenden, um die Funktionalität unserer API’s zu kontrollieren.

# Postman (Hagenhofer)

Postman ist eine der führenden Anwendungen, wenn es um API-Testing geht – insgesamt wird es von 500.000 Unternehmen, darunter viele Top-Unternehmen, genutzt [Postman, Zugriff: 20.02.2024]. Bereits im Laufe des Unterrichts haben wir Postman mehrfach verwendet und Erfahrungen damit gemacht, deshalb binden wir es auch in unsere Diplomarbeit ein. Es spielt eine wichtige Rolle beim Testen der Funktionalität unserer API’s und stellt sicher, dass alle Endpoints ausreichend ausprobiert werden.

# Discord (Hagenhofer)

Discord ist ein Chatprogramm, das wir im Rahmen unserer Diplomarbeit für die private Kommunikation nutzen. In sogenannten „Servern“ können wir mögliche Links für Lösungsansätze, persönliche Dokumentation oder auch Screenshots, Bilder und Code-Ausschnitte miteinander teilen und sicher ablegen, sodass wir von mehreren Geräten freien Zugriff darauf haben. Besonders wichtige Abschnitte können auch angepinnt und somit besonders übersichtlich festgehalten werden.

# Google Chrome (Hagenhofer)

Google Chrome verwenden wir in unserem Projekt als Haupt-Web-Browser. Das Google-Produkt ist schnell, sicher und übersichtlich, zudem gibt es viele hilfreiche Funktionen und auch die Möglichkeit, Add-Ons über den Web Store zu ergänzen. Chrome läuft auf aktuelleren Versionen von Windows, Mac, Android und auch iOS und funktioniert damit auf einem Großteil der Geräte [Chrome, Zugriff: 20.02.2024].

# Microsoft Edge (Davare)

Der vorinstallierte Webbrowser auf jedem Windows 10- oder Windows 11-Gerät stellt einen würdigen Nachfolger des legendären Internet Explorer dar. Da er nun auf dem Open-Source-Chromium-Framework von Google basiert, ist er erheblich schneller und sicherer als sein Vorgänger, der IE. Wie andere Browser bietet er Tools zur einfachen Navigation im Internet. Ein herausragendes Merkmal ist die integrierte KI Microsoft Copilot, die ähnlich wie andere bekannte KIs eine "Chat"-Funktion bietet. Der Unterschied besteht darin, dass sie durch das DALL-E-3-Modell auch in der Lage ist, Bilder zu generieren. Da der Browser von Microsoft entwickelt wurde, ist er perfekt auf Windows und andere Office-Produkte abgestimmt, weshalb eine reibungslose Zusammenarbeit und Interaktion mit den anderen Anwendungen entsteht.

[https://www.microsoft.com/de-de/edge?form=MA13FJ]

# OpenVPN (Davare)

OpenVPN ist eine freie VPN (Virtual Private Network) Software, die in diesem Projekt verwendet wird, um eine sichere Verbindung zwischen einem Zweitnetzwerk, in dem die Hardware des Ferienhauskomplexes eingerichtet ist, und unseren Endgeräten herzustellen. Diese Open-Source-Software zeichnet sich durch ihre plattformübergreifenden Releases aus und läuft auf jedem gängigen Betriebssystem, sodass sie auch problemlos auf Netzwerkgeräten wie Routern installiert werden kann, was bei uns der Fall ist. Um eine Verbindung herzustellen, wird eine Konfigurationsdatei benötigt, die wichtige Einträge wie die Ziel-IP, die Verschlüsselungsmethode, ein Zertifikat und den privaten Schlüssel enthält.

[https://openvpn.net/client/client-connect-vpn-for-windows/]

Der Auftraggeber hat uns die Konfigurationsdatei bereitgestellt und sich für diese Art der Datenübertragung entschieden, da sie einen hohen Sicherheitsstandard bietet und keine permanente Anwesenheit an den Geräten erforderlich ist, da der Netzwerkzugriff aus diesem Netzwerk eingeschränkt ist.

# Google Suchmaschine

…

# Oracle VM Virtual Box

…

# Projektverwaltung (Hagenhofer)

Im Laufe des Projekts ist es wichtig, das Zusammenarbeiten möglichst effizient und sicher zu gestalten. Dazu haben wir uns entschieden, einige Tools in unsere Arbeit mit einzubinden.

# Git (Hagenhofer)

Git ist ein Source-Control-System, das es einem erlaubt, durch verschiedene „Branches“ effektiv mit anderen Personen zusammenzuarbeiten. Neben einem Master-Branch gibt es mehrere lokale Branches, an denen man ganz unabhängig voneinander arbeiten und Änderungen vornehmen kann, dies bietet sichereres Arbeiten, da man sich keine Sorgen darum machen muss, Fortschritt zunichte zu machen oder zu verlieren. Außerdem gibt es die Möglichkeit, jederzeit zu älteren Versionen des Projekts zurückzukehren. [Git, Zugriff: 20.02.2024]

# GitHub (Hagenhofer & Davare)

GitHub ermöglicht es, die Softwareentwicklung durch sicheres Ablegen und ein Push- und Pull-System einfacher zu machen. Es wird getrennt voneinander in Lokal-Branches gearbeitet, bevor schließlich über die GitHub Desktop App die eigenen Änderungen und Ergänzungen zum Main-Branch hinzugefügt und für alle anderen zur Verfügung gestellt werden – dieses System ist sicher, effizient und vor allem denkbar einfach. GitHub ist ein sehr großes Netzwerk mit über 420 Millionen Repositories und über 100 Millionen Entwicklern, der Austausch mit anderen ist also dementsprechend einfach und effizient. Weiters gibt es die Möglichkeit, Projekte auf public zu stellen und damit der Öffentlichkeit zur Verfügung zu stellen. [Github, Zugriff: 20.02.2024]

Auf wichtige Begriffe im GitHub-Umfeld möchte ich genauer eingehen:

* Push: Die eigenen Änderungen werden dem Main-Branch zugefügt und dort übernommen.
* Commit: Man erfasst alle Änderungen seit dem letzten Commit, und gibt sie zur Weiterverarbeitung weiter (z.B. Push).
* Pull: Der Lokal-Branch abgeändert, sodass er alle Änderungen aus dem Main-Branch übernimmt.
* Merge: Der Lokal-Branch und der Main-Branch werden zusammengeführt, hierbei kann es möglicherweise auch zu Kollisionen kommen.
* Brach: Man kann sich ein Repo wie einen Baum vorstellen, es gibt den Stamm, welcher der Master bzw. Main Branch ist, und andere Branches sind die Äste des Baumes. Nebenbranches werden in der Regel dazu erzeugt, um neue experimentelle Funktionen zu entwickeln, ohne den funktionierenden Haupt Branch zu zerstören.
* History: Vergangene Pushes und Pulls werden aufgelistet.
* Lokales Repository: Lokal geklonter eigene Version vom Remote Repo.
* Clone: Das Remote Repo wird auf den lokalen Rechner heruntergeladen.
* Fetch: Daten werden vom Repo heruntergeladen, ohne zwingenden Merge.
* Remote Repository: Repository welches in der GitHub Cloud liegt und von dort gecloned werden kann. Wenn es ein öffentliches Repository ist, kann es jeder klonen.

# Programmiersprachen

…

# Python

…

# C#

…

# JavaScript

…

# Datenbanken (Davare)

In unserer digitalen Welt steht der Schutz unserer wertvollen Daten an erster Stelle. Um eine große Menge Daten sowohl sicher als auch effizient zu speichern bzw. sie abzurufen, werden sogenannte Datenbanken benutzt.   
  
Eine „Datenbank“ besteht aus zwei Teilen, einmal einem Datenbank Management System (DBMS) welche die eigentliche Verwaltung Software ist und der eigentlichen Datenbank selbst, was eine organisierte Sammlung von Daten ist  
.

Das DBMS ist die Schnittstelle zwischen der Abgelegten Daten der Datenbank sowie dem Endnutzer bzw. einer anderen Software, welche die Daten im Endeffekt braucht. Sie können sich ans ACID (Atomicity, Cosistency, Isolation, Durability) oder im deutschen AKID (Atomarität, Konsistenz, Isolation, Dauerhaftigkeit) halten, was für Verlässlichkeit und Konsistenz steht. [https://www.ionos.at/digitalguide/hosting/hosting-technik/datenbanken/]

Atomicity oder Atomarität bedeutet, dass eine Aktion in der Datenbank entweder vollständig oder gar nicht ausgeführt wird. Dadurch wird sichergestellt, dass Änderungen vollständig übernommen werden und dadurch keine ungewollten Fehler in die Datenbanken gelangen. Wenn eine Transaktion trotzdem fehlschlägt, werden einfach alle Änderungen ignoriert und die Datenbank in den Stand vor den Änderungen gebracht (Rollback).

Consistency oder Konsistenz bedeutet, dass eine Datenbank von einem konsistenten Zustand in einen anderen konsistenten Zustand gelangt. Der Zustand muss konsistent sein, da es sonst zu verschiedenen Ständen von Datenbanken kommen kann, wenn mehrere Clients gleichzeitig zugreifen.

Isolation bedeutet, dass eine Aktion nicht eine andere Aktion beeinflusst und eigenständig ausgeführt werden muss. Diese Sicherstellung wird meistens durch Sperrmechanismen garantiert und wirkt dadurch blockierend.

Durability oder Dauerhaftigkeit beschreibt, dass Transaktionen der Datenbank dauerhaft gespeichert werden und auch bei Technischen Problemen erhalten bleiben.

[https://www.bigdata-insider.de/was-ist-acid-a-776182/]

Es gibt verschiedene Arten von Datenbanken, auf welche in den zu benutzenden Datenbanken eingegangen wird.

# MongoDB (Davare)

Mongo ist eine Dokumentenorientierte Datenbank…

…

# InfluxDB (Davare)

…

# Frameworks (Davare)

…

# .NET (Davare)

…

# Blazor (Davare)

Blazor ist ein von Microsoft entwickeltes Front-End Framework, dass einem helfen soll, eine Webapp schnell und einfach ohne JavaScript zu bauen, was aber nicht heißen muss, dass man es nicht kann. Blazor ist eine sogenanntes Single Page Application (SPA) Framework, was bedeutet, dass der Inhalt der Website dynamisch aktualisiert wird und nicht zu anderen Seiten gewechselt wird.   
[https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-apps]

[https://dotnet.microsoft.com/en-us/apps/aspnet/web-apps/blazor]

Der Entwickler gestaltet seine Seiten durch sogenannte Razor Pages, welche HTML (Hyper-Text Markup Language), CSS (Cascading Style Sheet) und C# Code benutzen. Es ist sowohl möglich, einen Clientseitigen Ansatz zu wählen sowie einem Serverseitigen Ansatz.

# MudBlazor (Davare)

MudBlazor ist eine Komponenten Bibliothek, welche auf Microsofts Blazor basiert. Sie wurde anfangs dafür entwickelt mehr Leute für Blazor zu begeistern, sodass mehr Blazor Seiten erzeugt werden. MudBlazor bietet dem Entwickler einige ansprechende Komponenten, welche ihm helfen sollen, eine Blazor Anwendung zu bauen.

Eine Komponenten Bibliothek ist eine Sammlung an Komponenten, welche verwendet werden können. Eine Komponente ist ein vorgefertigtes wiederverwendbares Element, welches auf einer Website verwendet werden kann. Ein Component kann alles sein, zum Beispiel eine Navigation Bar, ein Textfeld, ein Modal, eine Card, etc.. Bei unserer Webapp könnte es Beispielsweise so aussehen. Auf der linken Seite sieht man wo Components sein könnten und rechts siehr man wie es auf der Website aussieht.Ein Bild, das Text, Screenshot, Schrift, Diagramm enthält.

Automatisch generierte Beschreibung

[https://developer.mozilla.org/en-US/docs/Web/API/Web\_components]

[https://mudblazor.com/mud/introduction]

# Swagger (Davare)

Swagger ist ein Framework, dass es einem Entwickler vereinfacht, eine RESTful API zu Dokumentieren. Swagger generiert automatisch eine übersichtliche Website, auf der man interaktiv die erstellte API dokumentiert ist. Jeder Endpoint kann ausgeklappt werden, um weitere Details über ihn zu sehen wie z.B. die Parameter, welche übergeben werden und welche Response man erwarten kann.

[https://www.ionos.at/digitalguide/websites/web-entwicklung/was-ist-swagger/]Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

Automatisch generierte Beschreibung

# Nuget’s (Davare)

Nuget ist ein von Microsoft entwickelter Package-Installer für .Net, welcher es Entwickler erleichtert, nützlichen Code zu teilen bzw. Code von anderen als Paket zu installieren.

Die Folgenden Pakete wurden für unser Projekt verwendet.

# Newtonsoft.Json (Davare)

Newtonsoft ist ein JSON (De)-Serializer welcher dazu benutzt werden kann, ein Objekt in JSON-Form zu bringen bzw. einen JSON-String wieder in ein Objekt zu verwandeln. Visual Studio hätte einen eigenen Serializer eingebaut, aber wir haben uns für diesen entschieden, da wir mit diesem bereits Erfahrungen gesammelt haben.

# MongoDB Driver (Davare)

Das MongoDB-Driver Nuget ist ein grundlegender Baustein für die Entwicklung mit einer MongoDB Datenbank mit .Net. Es bietet die Grundfunktionen, um mit einer MongoDB Datenbank zu kommunizieren, egal ob Lokal oder in der Cloud mit MongoDB Altas.

# Asp.Net.Cors (Davare)

Eigentlich sollte eine Website keine Daten von einem anderen Server laden. Bei Cors (Cross-Origin Ressource Sharing) handelt es sich um ein Prinzip, dass eine Absprache von mehreren Servern erlaubt.

Dieses Nuget musste hinzugefügt werden, da es beim Benutzen der APIs bei den Apps zu Problemen mit CORS gekommen ist und wir durch dieses Nuget Einstellungen an Cors tätigen können.

# InfluxDB Client (Davare)

Gleich wie der MongoDB Driver ist das InfluxDB Client Nuget der grundlegende Stein um mit Influx Zeitreihendatenbanken zu Arbeiten. Es ermöglicht eine Verbindung zu einer InfluxDB Datenbank herzustellen, Daten hineinzuschreiben und Abfragen auszuführen.

# Systemkomponenten

…

# Stromkomponenten

# Huawei PV-Anlagen-Steuerung

# BYD-Stromspeicher

# Miner

…

# Whatsminer

…

# Antminer

…

# Weitere Komponenten

…

# OpenVPN Verbindung

…

# Revolution Pi

…

# Umsetzung

# [ Einleitung ]

# Motivation und Zielsetzung

# Arbeitsdokumentation

## Recherche

### PV-Anlage (Hagenhofer)

Für eine funktionierende PV-Anlage werden natürlich neben den Paneelen eine PV-Anlagen-Steuerung benötigt. Für unser Projekt war es zunächst unklar, ob hierfür mit der Marke „Fronius“ oder „Huawei“ gearbeitet werden wird.

Fronius ist ein seit 1945 existentes österreichisches Unternehmen, das unter anderem seit 1992 auf Solarenergie spezialisiert ist. Das Unternehmen ist innovativ, hochqualitativ und wurde von Dun & Bradstreet, dem größten Dienstleister für B2B-Wirtschaftsinformation weltweit, mit dem höchsten Sicherheitsfaktor ausgezeichnet [Fronius, Zugriff: 22.01.2024]. Zudem wurde Fronius mit zahlreichen Leistungen wie dem Österreichischen Staatspreis für Umwelt- und Energietechnologie (2007) und dem Intersolar Award (Symo GEN24 Plus, 2020) ausgezeichnet [Fronius, Zugriff: 22.01.2024].

Huawei ist ein ursprünglich chinesisches Unternehmen, das sich seit mehreren Jahren auch auf Solarenergie spezialisiert hat. Inzwischen ist es führend in mehreren verwandten Bereichen, unter anderem auch bei der Innovation von Energie-Speichersystem-Architekturen. Das Unternehmen hat zum aktuellen Stand 997 Mrd. kWh grüne Energie erzeugt und damit 431 Mio. Tonnen CO2 Emissionen reduziert [Huawei, Zugriff: 22.01.2024].

Schlussendlich wurde eine Huawei-Steuerung gewählt, mit der wir arbeiten. Zum aktuellen Stand wurde Anlagen-Steuerung noch nicht abgeschafft, weswegen wir diesen Teil „mocken“ werden müssen.

### Speicher / Hagenhofer

Der Stromspeicher ist wohl das Herz einer hoch-funktionalen PV-Anlage. Wenn der Bedarf an Haushaltsstrom gedeckt ist, wird dieser Stromspeicher von dem überschüssigen PV-Strom aufgeladen wie ein Akku. Sobald der Bedarf des Hauses höher ist als der aktuell produzierte Strom, wird der Speicher wieder entladen.  
Obwohl PV-Anlagen grundsätzlich auch ohne Stromspeicher funktionieren, steigert er den Eigenverbrauch des eigenproduzierten Stroms enorm. Die Energiekosten werden dadurch gesenkt und weniger sonnige Zeiten überbrückt [Bundesverband Photovoltaic Austria, Zugriff: 18.12.2023].

Grundsätzlich sind zum aktuellen Stand zwei Batteriearten auf den Markt verbreitet. Sowohl Lithium-Ionen-Batterien, als auch Natrium-Ionen-Batterien werden zum Großteil verwendet. Beide Arten haben verschiedene Vorteile, die durchaus verglichen werden müssen, um die bessere Option für den eigenen Bedarf aussuchen zu können.

* Lithium-Ionen-Batterien:

Lithium-Ionen-Akkus werden schon seit einigen Jahren in elektronischen Gegenständen wie beispielsweise Smartphones verwendet. Sie haben einen hohen Wirkungsgrad und können sehr oft geladen werden, bevor sie ausgetauscht werden müssen. Außerdem sind sie immer noch voll im Trend der Forschung und werden fortlaufend modernisiert und verbessert. Ein potenzielles Problem stellt eine mögliche Überhitzung da, zu der es potentiell bei Speicherüberladung kommen kann. In modernen Batterien wird mögliche Überhitzung allerdings schon aktiv durch intelligente Ladesysteme vorgebeugt [Verbund AG, Zugriff: 18.12.2023], [Bundesverband Photovoltaic Austria, Zugriff: 18.12.2023].

* Natrium-Ionen-Batterien:  
  Natrium-Ionen-Batterien funktionieren grundsätzlich sehr ähnlich wie Lithium-Ionen-Batterien. Natrium, einer der Grundbestandteile, ist allerdings um einiges leichter anzuschaffen als Lithium, weswegen diese Art in der Anschaffung etwas umweltfreundlicher ist. Die Batterie stellt höhere Lade- und Entladeströme bereit, zusätzlich ist sie gegen Überhitzung und Explosion geschützt und deshalb sehr sicher. Ein Nachteil jedoch ist, dass Natrium-Ionen-Batterien recht groß und schwer sind, weswegen sie einen dementsprechend passenden Platz brauchen, um installiert zu werden [Bundesverband Photovoltaic Austria, Zugriff: 18.12.2023], [Energieheld Schweiz, Zugriff: 18.12.2023].

Im Rahmen unseres Projekts werden wir mit einem Stromspeicher der chinesischen Marke BYD arbeiten. Beim Vergleich des BYD-Speicherangebots fällt auf, dass alle verwendeten Batterien auf Lithium-Ionen-Batterien basieren.

### Miner APIs / Davare

### Bitmain Whatsminer

Die „Whatsminer“ Geräte des weltweit agierenden chinesischen Herstellers Bitmain benutzen eine einheitliche API mit der aktuellen Version 2.0.5 [AWS, Zugriff: 18.12.2023].

Bevor die API vollständig benutzt werden kann, müssen erst ein paar wenige Schritte befolgt werden. Per Default ist nämlich das Schreibrecht deaktiviert und es muss über das Hersteller Tool „WhatminerTool“ aktiviert werden. Hierfür ist es notwendig das Standardpasswort des Admins zu ändern und die API per Knopfdrück und Bestätigung zu aktivieren.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Abbildung 1: Aktivierung API in Whatminer Tool

Bei Erfolgreicher Aktivierung wird bei dem jeweiligen Miner unter „API“ der Status „ON“ angezeigt.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 2: API eingeschaltet

Nun sollte auf die API über die IP des Miners sowie dem Port 4028 zugegriffen werden können. Es ist zu beachten, dass wenn eine Verbindung zu dem Port aufgebaut wird und der Miner innerhalb von zehn Sekunden keine Daten erhält, dass die Verbindung automatisch geschlossen wird. Für die Write-Funktionen wird außerdem ein Token benötigt, welcher über den API-Endpoint „get\_token“ angefragt werden kann. Dieser hält für 30 Minuten und muss für weitere Anfragen nach Ablauf erneuert werden. Ein Miner kann gleichzeitig bis zu 16 IP-Clients gleichzeitig haben wobei jeder Client gleichzeitig bis zu 32 Tokens haben kann.

# Zusammenfassung

# Literaturverzeichnis

* [AWS, Zugriff 18.12.2023]:   
  <https://aws-microbt-com-bucket.s3.us-west-2.amazonaws.com/WhatsminerAPI%20V2.0.5.pdf>
* [Bundesverband Photovoltaic Austria, Zugriff: 18.12.2023]: <https://pvaustria.at/pv-speicher/>
* [Verbund AG, Zugriff: 18.12.2023]:  
  <https://www.verbund.com/de-at/privatkunden/themenwelten/photovoltaik/lithium-ionen-akku>
* [Energieheld Schweiz, Zugriff: 18.12.2023]: <https://www.energieheld.ch/solaranlagen/stromspeicher/salzspeicher#:~:text=Salzwasserspeicher%2C%20welche%20auf%20die%20Natrium,Kilowattstunden%20(kWh)%20Kapazit%C3%A4t%20angeboten>
* [Fronius, Zugriff: 22.01.2024]:   
  <https://www.fronius.com/de-at/austria/solarenergie/ueber-uns/warum-fronius>  
  <https://www.fronius.com/de-at/austria/solarenergie/ueber-uns/geschichte>
* [Huawei, Zugriff: 22.01.2024]:  
  <https://solar.huawei.com/at>
* [Android Developer, Zugriff: 14.02.2024]:

<https://developer.android.com/>

* [Demandsage, Zugriff: 14.02.2024]:

<https://www.demandsage.com/android-statistics/#:~:text=There%20are%203.3%20billion%20Android,share%20globally%20as%20of%202024>.

* [Wikipedia Android, Zugriff: 14.02.2024]:

<https://de.wikipedia.org/wiki/Android_(Betriebssystem)#Architektur>

* [Figma, Zugriff: 15.02.2024]:

<https://www.figma.com/community>

* [Wikipedia Diagrams.net, Zugriff: 15.02.2024]:

<https://de.wikipedia.org/wiki/Diagrams.net>

* [Adobe Photoshop, Zugriff: 15.02.2024]:

<https://www.adobe.com/at/products/photoshop/features.html>

* [MongoDB, Zugriff: 20.02.2024]:

<https://www.mongodb.com/docs/compass/current/#:~:text=MongoDB%20Compass%20is%20a%20powerful,View%20installation%20instructions>

* [Postman, Zugriff: 20.02.2024]:

<https://www.postman.com/company/about-postman/>

* [Chrome, Zugriff: 20.02.2024]:  
  <https://www.google.com/intl/de_at/chrome/>
* [Git, Zugriff: 20.02.2024]:  
  <https://git-scm.com/about>
* [Github, Zugriff: 20.02.2024]:  
  <https://github.com/about>

# Anhänge

# [ Selbstständigkeitserklärung ]