Restschnittstelle für einen Kameraroboter

Individuelle Praktische Arbeit

Ein Bild, das Text, drinnen, Boden, grün enthält.

Automatisch generierte Beschreibung

**Lernender**: Maurice Meier  
**Hauptexperte**: André Lichtsteiner  
**Berufsbildner**: Janusz Szymanski  
**Lehrbetrieb**: Fachhochschule für Technik FHNW  
**Startdatum**: 07.03.2022  
**Abgabedatum**:25.03.2022

# Änderungsnachweis

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Beschreibung |
| 0.1 | 07.03.2022 | Maurice Meier | Teil 1: Umfeld und Ablauf, |
| 0.2 | 09.03.2022 | Maurice Meier | Teil 2: Informieren |
| 0.3 | 10.03.2022 | Maurice Meier | Teil 2: Planen, Entscheiden |
| 0.4 | 11.03.2022 | Maurice Meier | Teil 1: Tagesjournal |
| 0.5 | 14.03.2022 | Maurice Meier | Teil 1: Tagesjournal |
| 0.6 | 16.03.2022 | Maurice Meier | Teil 2: Realisieren |
| 0.7 | 17.03.2022 | Maurice Meier | Teil 2: Realisieren |
| 0.8 | 18.03.2022 | Maurice Meier | Teil 2: Realisieren |
| 0.9 | 21.03.2022 | Maurice Meier | Teil 2: Kontrollieren |
| 1.0 | 23.03.2022 | Maurice Meier | Teil 2: Kontrollieren |
| 1.1 | 24.03.2022 | Maurice Meier | Teil 2: Auswerten, Verzeichnisse |
| 1.2 | 25.03.2022 | Maurice Meier | Korrektur |

Inhaltsverzeichnis

[1. Änderungsnachweis 1](#_Toc99100252)

[2. Projektorganisation 6](#_Toc99100253)

[2.1. Involvierte Personen 6](#_Toc99100254)

[2.2. Projektmethode¨ 6](#_Toc99100255)

[2.3. Dokumentationsaufbau 7](#_Toc99100256)

[2.4. Datensicherung 7](#_Toc99100257)

[3. Aufgabenstellung 8](#_Toc99100258)

[3.1. Titel der Arbeit 8](#_Toc99100259)

[3.2. Ausgangslage 8](#_Toc99100260)

[3.3. Detaillierte Aufgabenstellung 8](#_Toc99100261)

[3.4. Mittel und Methoden 9](#_Toc99100262)

[3.5. Vorkenntnisse 9](#_Toc99100263)

[3.6. Vorarbeiten 9](#_Toc99100264)

[4. Zeitplan 0](#_Toc99100265)

[5. Arbeitsjournal 0](#_Toc99100266)

[5.1. Montag, 07.03.2022 / Tag 1 0](#_Toc99100267)

[5.1.1. Probleme/Fragen & Ergriffene Massnahmen 0](#_Toc99100268)

[5.1.2. Bemerkungen 0](#_Toc99100269)

[5.2. Mittwoch, 09.03.2022 / Tag 2 1](#_Toc99100270)

[5.2.1. Probleme/Fragen & Ergriffene Massnahmen 1](#_Toc99100271)

[5.2.2. Bemerkungen 1](#_Toc99100272)

[5.3. Donnerstag, 10.03.2022 / Tag 3 2](#_Toc99100273)

[5.3.1. Probleme/Fragen & Ergriffene Massnahmen 2](#_Toc99100274)

[5.3.2. Bemerkungen 2](#_Toc99100275)

[5.4. Freitag, 11.03.2022 / Tag 4 3](#_Toc99100276)

[5.4.1. Probleme/Fragen & Ergriffene Massnahmen 3](#_Toc99100277)

[5.4.2. Bemerkungen 3](#_Toc99100278)

[5.5. Montag, 14.03.2022 / Tag 5 4](#_Toc99100279)

[5.5.1. Probleme/Fragen & Ergriffene Massnahmen 4](#_Toc99100280)

[5.5.2. Bemerkungen 4](#_Toc99100281)

[5.6. Mittwoch, 16.03.2022 / Tag 6 5](#_Toc99100282)

[5.6.1. Probleme/Fragen & Ergriffene Massnahmen 5](#_Toc99100283)

[5.6.2. Bemerkungen 5](#_Toc99100284)

[5.7. Donnerstag, 17.03.2022 / Tag 7 6](#_Toc99100285)

[5.7.1. Probleme/Fragen & Ergriffene Massnahmen 6](#_Toc99100286)

[5.7.2. Bemerkungen 6](#_Toc99100287)

[5.8. Freitag, 18.03.2022 / Tag 8 7](#_Toc99100288)

[5.8.1. Probleme/Fragen & Ergriffene Massnahmen 7](#_Toc99100289)

[5.8.2. Bemerkungen 7](#_Toc99100290)

[5.9. Montag, 21.03.2022 / Tag 9 8](#_Toc99100291)

[5.9.1. Probleme/Fragen & Ergriffene Massnahmen 8](#_Toc99100292)

[5.9.2. Bemerkungen 8](#_Toc99100293)

[5.10. Mittwoch, 23.03.2022 / Tag 10 9](#_Toc99100294)

[5.10.1. Probleme/Fragen & Ergriffene Massnahmen 9](#_Toc99100295)

[5.10.2. Bemerkung 9](#_Toc99100296)

[5.11. Donnerstag, 24.11.2022 / Tag 11 10](#_Toc99100297)

[5.11.1. Probleme/Fragen & Ergriffene Massnahmen 10](#_Toc99100298)

[5.11.2. Bemerkungen 11](#_Toc99100299)

[5.12. Freitag, 25.03.2022 / Tag 12 12](#_Toc99100300)

[5.12.1. Probleme/Fragen & Ergriffene Massnahmen 12](#_Toc99100301)

[5.12.2. Bemerkungen 12](#_Toc99100302)

[6. Kurzfassung 13](#_Toc99100303)

[6.1. Ausgangssituation 13](#_Toc99100304)

[6.1.1. Ist 13](#_Toc99100305)

[6.1.2. Soll 13](#_Toc99100306)

[6.2. Umsetzung 13](#_Toc99100307)

[6.3. Ergebnis 13](#_Toc99100308)

[7. Informieren 14](#_Toc99100309)

[7.1. Anforderungsanalyse 14](#_Toc99100310)

[7.1. Use Case 15](#_Toc99100311)

[7.2. Systemaufbau 16](#_Toc99100312)

[7.2.1. Cambot 17](#_Toc99100313)

[7.2.1. Django Rest 18](#_Toc99100314)

[7.2.2. Flask Restful 18](#_Toc99100315)

[7.2.3. Fast API 18](#_Toc99100316)

[7.3. Reverse Proxy 18](#_Toc99100317)

[7.3.1. NGINX VS Apache 18](#_Toc99100318)

[7.4. Python Libraries 19](#_Toc99100319)

[7.4.1. Pyserial 19](#_Toc99100320)

[7.4.1. Pyrealsense2 19](#_Toc99100321)

[8. Planen 20](#_Toc99100322)

[8.1. Cambotmanager 20](#_Toc99100323)

[8.1.1. Cambot-Handler 20](#_Toc99100324)

[8.2. Reverse Proxy 20](#_Toc99100325)

[8.2.1. Autorisierung 20](#_Toc99100326)

[8.3. UI 21](#_Toc99100327)

[9. Entscheiden 22](#_Toc99100328)

[9.1. Technologie für den Reverse Proxy 22](#_Toc99100329)

[9.2. Technologie für API 22](#_Toc99100330)

[9.3. Aufbau des Cambotmanagers 22](#_Toc99100331)

[10. Realisieren 23](#_Toc99100332)

[10.1. Schnittstellen 23](#_Toc99100333)

[10.2. Cambotmanager 23](#_Toc99100334)

[10.2.1. Benötigte Python Libraries 23](#_Toc99100335)

[10.2.2. Models 24](#_Toc99100336)

[10.2.3. Manager 25](#_Toc99100337)

[10.2.4. Storage-Handler 26](#_Toc99100338)

[10.2.5. Cambot-Handler 26](#_Toc99100339)

[10.2.6. Camara 27](#_Toc99100340)

[10.2.7. Robot 27](#_Toc99100341)

[10.3. Reverse Proxy 28](#_Toc99100342)

[10.3.1. Konfiguration 28](#_Toc99100343)

[10.3.2. Autorisation 29](#_Toc99100344)

[10.4. UI 31](#_Toc99100345)

[11. Kontrollieren 32](#_Toc99100346)

[11.1. Unit Tests 32](#_Toc99100347)

[11.2. Testfallspezifikationen 33](#_Toc99100348)

[11.2.1. Rest API 33](#_Toc99100349)

[11.2.2. UI 34](#_Toc99100350)

[11.2.3. Datenstruktur 37](#_Toc99100351)

[11.2.4. Roboter 39](#_Toc99100352)

[11.3. Testprotokolle 40](#_Toc99100353)

[11.3.1. Testfall Protokoll TP-01 40](#_Toc99100354)

[12. Auswerten 41](#_Toc99100355)

[12.1. Fazit 41](#_Toc99100356)

[12.2. Persönliches Fazit 41](#_Toc99100357)

[12.3. Mögliche Erweiterungen 41](#_Toc99100358)

[13. Quellenverzeichnis 42](#_Toc99100359)

[13.1. Glossar 42](#_Toc99100360)

[13.2. Internetquellen 43](#_Toc99100361)

[13.3. Abbildungsverzeichnis 43](#_Toc99100362)

[13.4. Anhang 43](#_Toc99100363)

Teil 1: Obligatorisches Kapitel

# Projektorganisation

## Involvierte Personen

**Hauptexperte:**

André Lichtsteiner  
+41 79 469 33 67¨  
andre.lichtsteiner@gmx.ch

**Nebenexperte:**

Daniel Wilhelm

**Verantwortliche Fachkraft:**

Martin Gwerder  
Bahnhofstrasse 6  
5210 Windisch  
5.2B15  
+41 65 202 76 81

**Kandidat:**

Maurice Meier  
Bahnhofstrasse 6  
5210 Windisch  
5.2B06  
+41 79 152 52 74

## Projektmethode¨

Für die Durchführung dieses Projekt habe ich die Projektmethode IPERKA gewählt. IPERKA besteht aus den insgesamt 6 Phasen Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Auswerten. Aus deren Anfangsbuchstaben das Akronym IPERKA gebildet wurde. Durch diese Methode erhält man einen klaren Ablauf und Struktur über das gesamte Projekt hinweg. Ausserdem habe ich diese Methode bereits in Projekten in der Schule verwendet.

## Dokumentationsaufbau

Die Dokumentation ist insgesamt in 2 Teile gegliedert. Teil 1, Obligatorisches Kapitel, beinhaltet allgemeine Informationen zur Arbeit, Auflistung aller Beteiligten, Die Aufgabenstellung und der Zeitplan mit Tagesjournalen.

Teil 2, Projekt-Dokumentation, ist nach IPERKA strukturiert und aufgebaut. Jede Phase wird von einem Kapitel abgedeckt und darin die Tätigkeiten während der Phase dokumentiert und festgehalten. Am Ende befindet sich das Glossar so wie die Quellenverzeichnisse.

## Datensicherung

Die Datensicherung wird mithilfe von täglichen Commits auf ein GitHub Repository sowie das Speichern des Projekts auf einem zusätzlichen USB-Stick gemacht. Dies dient dazu, dass im Falle von Komplikationen mit GitHub, immer ein Backup mit der momentan aktuellen Version vorhanden ist. ([GitHub](https://github.com/Red8Bee/Cambotmanager-IPA-Maurice-Meier))

Ein Bild, das Text, Monitor, schwarz, Bildschirm enthält.

Automatisch generierte Beschreibung

Abbildung 1 Commit History

# Aufgabenstellung

Bei diesen Angaben halte ich mich and die eingetragenen Informationen des Fachverantwortlichen auf PKOrg. Kleinere Änderungen wurden nur vorgenommen, um Rechtschreibfehler zu korrigieren oder die Formatierung anzupassen.

## Titel der Arbeit

REST-Schnittstelle für einen Kameraroboter

## Ausgangslage

CamBot ist ein für diese Arbeit neu entwickelter Roboter. Er ist in der Lage eine Kamera, um einen 3D Drucker frei und repetierbar zu positionieren. Dieser Roboter soll in OctoPrint einbindbar sein und 3D-taugliche Daten sowie Hyperlapses (Zeitraffer mit Kamerabewegung) liefern, die mittels der in der Intel Realsense d435 und Meshroom (Fotogrammmetrie) gewonnen werden. Diese Arbeit wird parallel mit der Arbeit von Semjon Buzdin durchgeführt. Weil Semjon die 3D-Rekonstruktion macht und Maurice die Kontrolle des Roboters wird für die Arbeit eine API vorgegeben.

## Detaillierte Aufgabenstellung

Ziel dieser Arbeit ist es CamBot zu steuern. Dazu wird ein Dienst namens "cambotmanager" geschrieben. Der Dienst läuft auf einem Raspberry PI mit Raspian (oder Ubuntu) und stellt für die Ansteuerung des CamBots eine API auf der Basis von REST nach aussen zur Verfügung.

Die REST API ist spezifiziert auf Swagger unter der URL <https://app.swaggerhub.com/apis/mgwerder/cambot/0.0.1>.

Der Roboter ist mittels G-Codes ansteuerbar (GRBL-Basis; https://github.com/grbl/grbl/wiki).   
Der Roboter hat 3 Achsen. Die A-Achse ist die Rotationsachse um dem Roboter um den 3D-Drucker zu steuern (Kreis; ca. 270°; X-Achse auf GRBL; 1mm entspricht einem Grad). Die Y-Achse steuert den Roboter in der Höhe (Linearer Spindelantrieb; ca. 40cm; Y-Achse auf GRBL). Die B-Achse steuert den Kamera-Tilt des Roboters (Rotation; ca. -90°-90°; Z-Achse auf GRBL; 1mm entspricht einem Grad).

Ziel ist es die oben spezifizierte API zu realisieren (Swagger-Link) und zu schützen.

Alle Zip-Dateien, die von der API retourniert werden, sollten folgende Struktur haben: <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini and images (png or jpg) Im File Metadata sind mindestens folgende Informationen enthalten:  
 - Zeit des Snapshots   
- Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben)   
- Genaue Position des Roboters  
 - Status am Ende des Vorganges (OK oder Fehlercode und Text)

Zusätzlich ist ein kleines UI vorhanden (Web) welches es erlaubt (ebenfalls Passwortgeschützt) den Roboter manuell zu steuern und seine Werte anzuzeigen.

Die vom Roboter abgeholten Bilder werden mit den Metadaten in der EXIF-Struktur angereichert

## Mittel und Methoden

* Maurice steht während der ganzen Zeit der Roboter vollumfänglich zur Verfügung.
* Gefährdete Bauteile (z.B. Motorentreiber oder Endschalter) sind in kleiner Anzahl vorrätig.

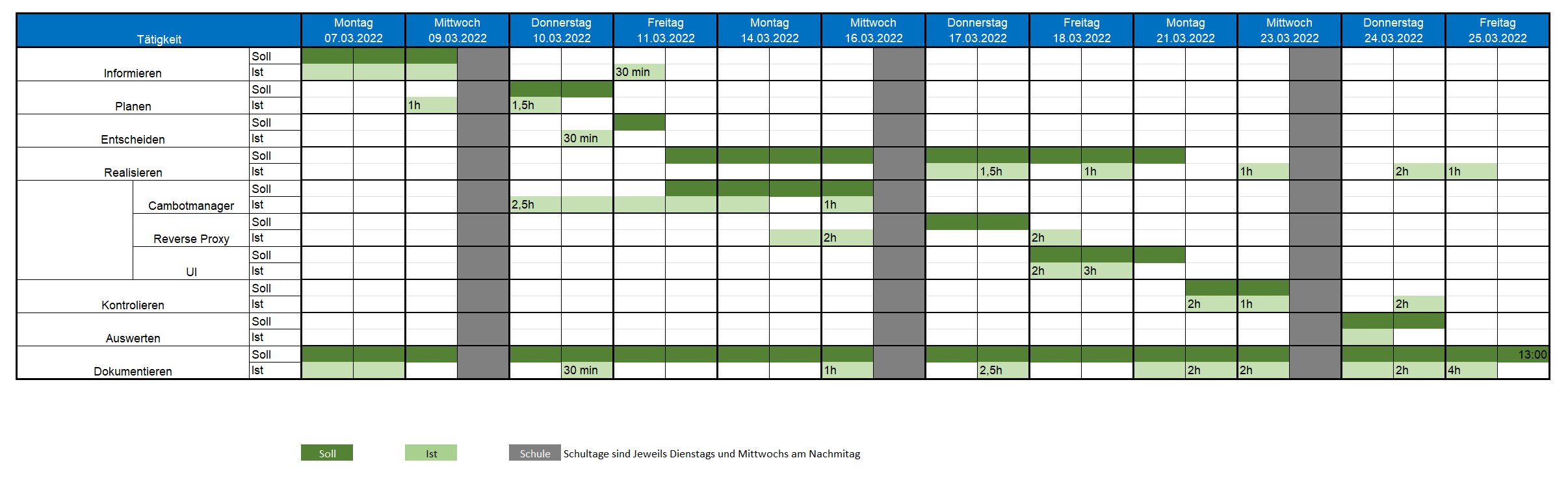
## Vorkenntnisse

* Maurice hat ca. ein Monat im Voraus Zugriff auf den Roboter und kann bereits mit ihm arbeiten
* Er hat im Monat vor der IPA mit Python gearbeitet (1.5d pro Woche) und vorher auch schon damit Erfahrungen gesammelt (Namentlich: Flask, Zip-Dateien und RestAPI).
* Er kennt den Raspberry-PI

## Vorarbeiten

* Der Kandidat hat eine kurze Ausbildung mit Zustandsmaschinen vor der Arbeit gehabt.
* Der Kandidat hat bereits einen Reverse-Proxy realisiert und darüber eine Benutzerauthentifizierung gemacht auf der Basis von Apache2.
* Der Kandidat hat bereits den Roboter mittels Python angesteuert.
* Der Kandidat hat bereits eine sehr einfache RestAPI auf Python realisiert.
* Der Kandidat hat bereits Dateihandling geübt (Files erstellen, verschieben, löschen)
* Der Kandidat hat bereits Daten in die EXIF-Informationen geschrieben.

# Zeitplan



# Arbeitsjournal

## Montag, 07.03.2022 / Tag 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h | 2h | Zeitplan | Zeitplan erstellen | Ich habe den Zeitplan erstellt |
| 2h | 2h | Informieren | Anforderungsanalyse | Die Anforderungen habe ich aus der Aufgabenstellung abgeleitet. |
| 20 min | 10min | Datensicherung | GitHub erstellen | GitHub Repository erstellt. |
| 2h | 2h | Informieren | System Aufbau festhalten | Ich habe das System analysiert und als Diagramm dargestellt ([Informieren/Systemaufbau](#_Systemaufbau)) |
| 1h 20 min | 2h | Dokumentation | Titel und Teil 1 | Titel und Layout sowie ein Grossteil von Teil 1 Fertig. |
| 15 min | 10 min | Tages Journal | Häutiges Journal einfüllen | Journal geführt |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Ich habe die Daten kurz vor 17:00 auf GitHub hochgeladen und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
| Was muss alles im UI möglich sein? | Allfällige Fragen wurden festgehalten, damit diese Matin Gwerder gestellt werden können. |
| Eine .ini Datei pro Bild oder pro Zip-Datei? | **||** |
|  |  |

### Bemerkungen

Martin Gwerder erteilte uns den Auftrag für die IPA bereits am Freitagabend (04.03.2022) da er am ersten Tag verhindert war.  
  
Der Start verlief wie geplant. Ich konnte nahezu alle Dinge, welche ich mir vorgenommen habe, fertig stellen bis auf zwei Teile im ersten Teil der Dokumentation.

## Mittwoch, 09.03.2022 / Tag 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1.5h | ~1.5h | Experten Besuch | Experten Besuch um 08:30-10:00 | Alle offenen Fragen geklärt.  Individuelle Bewertungskriterien definiert.  Präsentation Datum festgelegt |
| 2h | 1.5h | Informieren | Informieren fertig stellen | Ich wurde mit dem Informieren fertig |
| WnZ. | 1h | Planen | Mit Planen beginnen | Da noch Zeit übrig war, konnte ich bereits mit dem Planen beginnen. |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Die Daten habe ich auf den USB kopiert und auf GitHub hochgeladen. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
| Fragen von Montag, 06.03.2022 sind im Tagesjournal. | Beim Expertengespräch mit Martin Gwerder geklärt |
| Termin mit Martin Gwerder  Anforderungsanalyse, Entscheidungen, aufkommende Fragen | Termin mit Martin Gwerder am Donnerstag, 10.03.2022 15-16 Uhr |
|  |  |

### Bemerkungen

An diesem Tag hatte ich das erste Expertengespräch zusammen mit Herr Lichtsteiner und Martin Gwerder. Im Verlauf dieser Besprechung konnte ich alle meine offenen Fragen klären so wie Unklarheiten bei der Auftragsgebung und den Individuellen Bewertungskriterien klären.

#### Infos aus dem Gespräch

* Präsentation: 19.04.2022 um 13:30
* Zeitplan segregieren

## Donnerstag, 10.03.2022 / Tag 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h | 1 h | Planen | Cambot Manager Aufbau planen | Möglicher Aufbau geplant |
| 30 min | 30 min | Planen | UI aussehen Planen | Ich habe ein Mockup für die Webseite erstellt. |
| 30 min | 30 min | Entscheiden | Termin mit Martin Gwerder, Besprechen von getroffenen Entscheidungen, Anforderungsanalyse und offenen Fragen | Ich konnte die Anforderungen zusammen mit Martin Gwerder anschauen |
| 5h | 6h | Realisieren | Aufbau der API von Swagger übernehmen, Datamodels erstellen, | API-Calls und Datamodels von SwaggerHub implementiert.  Mit Ansteuerung von Roboter und Kamera begonnen |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden auf GitHub hochgeladen und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| - |  |

### Bemerkungen

Ich habe die Anforderungen mit Martin Gwerder besprochen. Es sollen noch kleinere Anpassungen vorgenommen und Anforderungen an den Roboter hinzugefügt werden. Danach soll ich es per E-Mail mit Martin Gwerder absprechen.

## Freitag, 11.03.2022 / Tag 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 15 min | 30min | Informieren | Anforderungsanalyse überarbeiten | Anforderungen angepasst und an Martin Gwerder gesendet |
| 7.75h | 7.5h | Realisieren, Cambotmanager | Cambotmanager so gut wie fertig | Ich konnte die Hauptfunktionalität fertigstellen |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Datenträger sichern | Daten wurden auf GitHub hochgeladen und auf dem USB-Datenträger gespeichert. |
|  |  |  |  |  |
|  |  |  |  |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Der Roboter fährt auf die Endschalter auf, obwohl diese nicht ausgelöst wurden. | Mail an Martin Gwerder |

### Bemerkungen

An diesem Tag konnte ich die Hauptfunktionalität des Cambotmanager fertigstellen. Es müssen noch ein paar kleinere Arbeiten vorgenommen werden, allerdings können bis auf das Ansteuern des Roboters alle Funktionalitäten genutzt werden. Das Storage Handling ist noch nicht implementiert.

## Montag, 14.03.2022 / Tag 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 6h | 4h | Realisieren, Cambotmanager | Storage Handling und Aufräumen | Ich konnte den Storage-Handler fertig realisieren. |
| 2h | 4h | Realisieren, Reverse Proxy | Den Reverse Proxy installieren und mit dem Einrichten beginnen | Der Reverse Proxy wurde installiert |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Datenträger sichern | Daten wurden kurz vor 17:00 auf GitHub hochgeladen und auf USB-Datenträger gespeichert. |
|  |  |  |  |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Raspberry hatte eine falsche Uhrzeit, weshalb ich Docker nicht installieren konnte, | Über Google fand ich heraus, wie ich die Zeit manuell umstelle |

### Bemerkungen

Der Code für den Cambotmanager ist so weit fertig und funktioniert. Aufgrund des Problems mit dem Roboter muss ich die Konfiguration des diesigen noch etwas anpassen. Martin Gwerder meinte ich liesse die Motoren zu schnell drehen.

## Mittwoch, 16.03.2022 / Tag 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 4h | 2h | Realisieren, Reverse Proxy | Reverse Proxy fertig einrichten und von aussen ansteuern | Reverse Proxy ist bereit zum fertig aufsetzen. Problem mit dem Erreichen des Proxys |
| - | 1h | Realisieren, Cambotmanager | Code aufräumen und Roboter abstrahieren | Roboter wurde abstrahiert und Code aufgeräumt |
| - | 1h | Dokumentieren | Realisieren auf den Neusten Stand Bringen | Realisieren wurde erweitert |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Datenträger sichern | Daten wurden kurz vor 17:00 auf GitHub hochgeladen und auf USB-Datenträger gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Um den Reverse Proxy anzusteuern, muss auf dem Router der Port 80 weitergeleitet werden. Dies kann ich an der FHNW nicht selbst machen. | Ich konnte Martin Gwerder darauf ansprechen. Das Problem war, dass ich die vergebene URL nicht in der Host-Datei eingetragen habe und der Computer sie so nicht auflösen konnte |
|  |  |

### Bemerkungen

Heute konnte ich das erste Mal über den Reverse Proxy auf eine Flask Applikation zugriffen.

## Donnerstag, 17.03.2022 / Tag 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1.5h | 5.5h | Realisieren | Cambotmanager auf Raspberry laufen lassen. | Nicht ganz erreicht durch zeit intensives konfigurieren |
| 2.5h | 2.5h | Dokumentieren | Realisieren über Cambotmanager und Reverse Proxy | erreicht |
| 1h | 1h | Realisieren | Roboter konfigurieren und mit Code testen | Da ich den Roboter falsch eingestellt habe und so in eine Endlosschlaufe manövrierte konnte ich das Ansteuern nicht testen. |
| 4h | - | Realisieren, UI | Mit UI beginnen | Konnte noch nicht beginnen |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Datenträger sichern | Daten wurden kurz vor 17:00 auf GitHub hochgeladen und auf USB-Datenträger gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Einige Libraries funktionieren nur auf Python Version 3.7. Probleme mit dem Herabstufen der Python Version | Nach einer Google Recherche und Fragen bei Schulkollegen konnte ich die Version ändern |
| Pyrealsense2 muss über Umwege heruntergeladen werden aufgrund der Kernel Architektur des Raspberry Pi’s | Google |

### Bemerkungen

Da ich viel Zeit benötigt habe, um die Version von Python herabzustufen und erfolglos Pyrealsense2 installieren versuchte, konnte ich noch nicht mit dem UI beginnen. Dies ist allerdings kein Problem, da das UI erst für den 18.03 eingeplant ist. Ausserdem konnte ich den Code mit dem Roboter nicht testen, da ich diesen falsch konfiguriert habe, was ich selbst nicht mehr beheben konnte. Weshalb ich ein Mail an Martin Gwerder sendete

## Freitag, 18.03.2022 / Tag 8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1h | 1g | Expertengespräch | 2. Expertengespräch mit Herr Lichtsteiner | Das Meeting fand stat. Ich konnte meine Frage an Herrn Lichtsteiner stellen |
| 6h | 6h | Realisieren, UI | UI Fertig und auf Raspberry | Ich konnte das UI fertigstellen und über die API ansteuern. |
| 2h | 2h | Realisieren | Raspberry fertig aufsetzen | Der Raspberry wurde fertig aufgesetzt. |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Datenträger sichern | Alles wurde auf GitHub hochgeladen und auf dem USB-Datenträger gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Der Raspberry startete nicht mehr richtig und musste neu aufgesetzt werden.  Beim Aufstarten zeigt er lediglich einen orangenen Bildschirm | Raspberry Pi neu aufgesetzt und Proxy Manager so wie Flask App erneut installiert. |
| Falsche Konfiguration auf dem Roboter. Er befindet sich in einer Endlosschlaufe. | Ich habe ein Mail an Martin Gwerder gesendet. |

### Bemerkungen

Leider startete der Raspberry Pi nicht mehr korrekt. So musste ich ihn neu aufsetzen. Dies dauerte glücklicher weise nicht mehr so lange wie beim ersten Mal.

Ich stellte beim Expertengespräch die Frage, ob es möglich sei, 3 Lehrlinge eines Tieferen Lehrjahrs als Raum Füller zur Präsentation einzuladen. Diese vertrauten Gesichter soll etwas meine Nervosität zügeln.

#### Infos aus dem Experten Gespräch

* Signierungszeitpunkt ist der Abgabe Zeitpunkt.
* Tages Journal in Ich Form Umschreiben.

## Montag, 21.03.2022 / Tag 9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 6h | 6h | Dokumentieren | Teil Realisieren fertig dokumentiert | Ich konnte den Teil Realisieren fertig Dokumentieren |
| 2h | 2h | Kontrollieren | Teil Kontrollieren Testfälle Schreiben | Ich konnte alle Testfälle Fertig schreiben. |
|  |  |  |  |  |
|  |  |  |  |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| - | - |

### Bemerkungen

Heute konnte ich das Realisieren Fertig Dokumentieren und Schrieb die Testfälle. So kann ich am Mittwoch mit dem Testen beginnen.

## Mittwoch, 23.03.2022 / Tag 10

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1h | 1h | Kontrollieren | Testfälle ausführen | Ich konnte alle Testfälle bis auf diese welche mit dem Roboter zutuen haben ausführen |
| 2h | 2h | Dokumentieren | Korrekturen und kleine Verbesserungen | Ich konnte Korrekturen und Verbesserungen im Dokument vornehmen |
| 1h | 1h | Realisieren | Roboter mit dem Code testen | Der Roboter ist laut Martin Gwerder Funktional. Das ansteuern funktionierte allerdings nicht. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Der Roboter lässt sich über keine Methode ansteuern. Er wirft den Fehler das er auf die Endschalter auffährt diese werden jedoch nie ausgelöst | Ich habe ein Mail mit der Problematik an Martin Gwerder gesendet. |

### Bemerkung

Martin Gwerder meldete mir am Fr 18.03 das der Roboter funktioniere. Als ich ihn heute allerdings wieder einmal Testen wollte funktionierte er nicht. Mir war es auf keine weise möglich den Roboter anzusteuern. Ob ich dies über den Universal-GCode-Sender gemacht habe oder über meinen Code. Nur das Homing funktionierte einwandfrei. Ich sendete ein Mail an Martin Gwerder in welchem ich die Problematik schilderte. Ich gab meine Dokumentation Martin Brenn. Er kondolierte sie nach Layout und Rechtschreibefehlern, welche er mir mit Kommentaren markierte.

## Donnerstag, 24.11.2022 / Tag 11

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h | 2h | Auswertung | Mögliche Erweiterungen und Fazit | Ich habe ein Fazit zur Arbeit sowie ein persönliches Fazit zum Ablauf der Arbeit gezogen. Ausserdem habe ich das Projekt nach möglichen Erweiterungen analysiert. |
| 2h | 2h | Dokumentieren | Layout kontrollieren, Rechtschreibung | Ich konnte den Teil 1 Korrigieren. |
| 2h | 2h | Realisieren | Roboter mit dem Code ansteuern | Martin Gwerder konnte Heute morgen endlich den Roboter Flicken. Der code Funktioniert hat durch die Eigenheiten von GRBL allerdings noch einige Macken. Diese werden vermutlich nichtmehr behoben werden können |
|  |  |  |  |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Sendet man den Homing Befehl an den Roboter so meldet dieser erst Ok, sobald er dies Fertig ausgeführt hat. Bei Bewegungsbefehlen tut er dies allerdings direkt ohne, dass der Roboter an der Position angekommen ist. | Leider fehlt mir etwas die Zeit, um dieses Problem zu beheben. Ich habe mich auf die Dokumentation konzentriert. Jedoch konnte ich einige Dinge ausprobieren wie einen Zweiten Befehl loszuschicken, welcher keine Bewegung auslöst, jedoch erst ausgeführt wird, wenn der andere Beendet ist. Leider funktionierte dies nicht. Falls am Fr morgen noch Zeit bleibt widme ich mich diesem Problem nochmals. |
| Janusz Szymanski verunsicherte mich etwas mit der Deadline. Er meinte sie sei bereits um 11:30 | Ich habe bei Herr Lichtsteiner nahgefragt um sicherzugehen. Es kahm die Rückmeldung das wie bei den Expertengesprächen aufgeschrieben die Deadline um 13:00 endet. |

### Bemerkungen

Martin Gwerder konnte heute Morgen den Roboter Reparieren. Ich habe den Code zum Ansteuern des Roboters, soweit ich konnte, angepasst. Jedoch fehlt mir die Zeit, um mich mit der Ausführungsbestätigung von GRBL auseinander zu setzen. Ich gab meine Dokumentation Jonas Suter. Er Kontrollierte sie nach Rechtschreibefehler und unsinnigen Sätzen. Ausserdem gab mir Janusz Szymanski einige Tipps zu meinem Persönlichen Fazit. Und dem Zeitplan. So Passte ich die Vorspalte angepasst so das nicht nur Das Datum sondern auch Der Wochentag darin vermerkt ist.

## Freitag, 25.03.2022 / Tag 12

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h |  | Dokumentieren | Anhang Bereit machen. | Der Anhang ist bereit |
| 2h |  | Dokumentieren | GitHub Abgabebereit machen | Github wurde alles auf dem Main Branch gemerged und das Readme mit Benutzerdokumentation und Benutzeranleitung hochgeladen. |
| 1h |  | Realisieren | Roboter Steuerung fertig machen |  |
| 30 Min |  | Abgeben | Arbeit Abgeben um 12:00 |  |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| - Zu wenig Zeit für Test | -Da der Roboter nun funktioniert, müsste ich nochmals testen. Für dies reichte allerdings die Zeit nichtmehr aus. |

### Bemerkungen

Ich schaffte es Heute morgen den Code für das Ansteuern des Roboters fertig zu stellen. Über Umwege gab Martin Gwerder mir den Tipp anstatt eines zweiten Bewegungsbefehles den Befehl «$G» zu verwenden. Dieser Befehl gibt eine andere Antwort zurück. So kann ich diesen Befehl direkt nach dem Bewegungsbefehles losschicken. Da die Befehle nacheinander ausgeführt werden, weiss ich das die Bewegung fertig ist so bald die spezielle Antwort des zweiten Befehles zurück kommt.

Teil 2: Projekt-Dokumentation

# Kurzfassung

## Ausgangssituation

### Ist

Martin Gwerder entwickelte den Roboter Cambot. Er soll über einem 3D-Drucker hängen. Der Cambot verfügt über 3 Achsen, mit welchen die Kamera positioniert werden kann. Die A-Achse ist die Rotationsachse, welche die Kamera, um den 3d-Drucker rotieret und entspricht der X-Achse in GRBL. Ein Lineare Spindelantrieb denkt die Y-Achse ab. Die B-Achse kontrolliert die Neigung der Kamera und kann in GRBL über die Z-Achse angesteuert werden. Momentan kann der Roboter nur über den Universal G-Code Sender angesteuert werden.

### Soll

Ziel dieser Arbeit ist es nun, den Cambot über eine nach Swagger Definition implementierte API ansteuerbar zu machen. Das ganze System soll von einem Reverse Proxy über autorisierte IP-Adressen und Benutzer mit Passwort geschützt sein. Ausserdem soll ein UI vorhanden sein, in welchem der Status des Roboters sowie das Inventar ersichtlich sein sollen. Ausserdem soll über das UI der Roboter in seinen Home Zustand zurückgesetzt werden und die Zip-Dateien der Inventory-Items aus dem Inventar heruntergeladen werden

## Umsetzung

Diese Arbeit wurde nach der Projektmethode IPERKA umgesetzt. Dies bedeutet das in einem Ersten Schritt das System analysiert und Informationen über Python API-Frameworks wie Flask und Fast API, Ausserdem sammelte ich Informationen über Möglichkeiten einen Reverse Proxy auf einem Raspberry zu realisieren. Die dabei gewonnen Erkenntnisse wurden genutzt, um Konzepte zu erstellen, aus welchen sich dann die Planung herauskristallisierte. In der anschliessenden Realisierungsphase wurde das Erarbeitete praktisch umgesetzt und die einzelnen Teile zusammengefügt. Anschliessend führte ich die davor erarbeiteten Testfälle aus und hielt das Resultat der Arbeit in einem Fazit fest.

## Ergebnis

Im Rahmen dieser Arbeit wurde eine REST API und ein UI erarbeitet. Über das REST API ist es möglich, Hyperlapses in Auftrag zu geben, welche dann auf dem Roboter abgefahren und fotografiert werden. Ausserdem ist über das API auch der Status des Roboters sowie das Inventar ersichtlich. Im UI erhält man Einsicht ins Inventar und kann die Snapshots der Inventory-Items als Zip-Datei herunterladen. Hier ist zudem auch der Status des Roboters ersichtlich und er kann zurückgesetzt werden.

# Informieren

## Anforderungsanalyse

Die Anforderungen wurden aus der Aufgabenstellung im PKOrg abgeleitet und nach den verschiedenen Bestandteilen des Projekts sortiert wie Reverse Proxy oder REST API.  
Jede Anforderung erhielt eine eindeutige Bezeichnung, welche sich nachfolgendem Schema richten:

* ANF-RA-#: Anforderungen an die Rest API
* ANF-UI-#: Anforderungen an das UI
* ANF-D-#: Anforderungen an die Datenstruktur
* ANF-R-#: Anforderungen an den Roboter

|  |  |
| --- | --- |
| Anforderung | Beschreibung |
| ANF-RA-1 | Die Rest API wird von einem Zugriffsschutz mit Basic oder Digest Authentifikation, so wie aufgrund der autorisierten IP-Adressen via Reverse-Proxy |
| ANF-RA-2 | Die Rest API ist nach Spezifikation auf Swagger implementiert |
| ANF-UI-1 | Die UI muss geschützt werden |
| ANF-UI-2 | Über die UI muss der momentane Status ersichtlich sein |
| ANF-UI-3 | Der Roboter soll über das UI zurückgesetzt werden können. |
| ANF-UI-4 | Das Aktuelle Inventar soll im UI aufgelistet sein |
| ANF-UI-5 | Elemente aus dem Inventar sollen als Zip, über das UI heruntergeladen werden können. |
| ANF-D-1 | Die Bilder werden von der API als Zip-Datei retourniert |
| ANF-D-2 | Alle Zip-Dateien, die von der API retourniert werden, sollten folgende Struktur haben: <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini and images (png or jpg) |
| ANF-D-3 | Die Bilder sind mit Metadaten in der EXIF-Struktur angereichert |
| ANF-D-4 | Im Zip File ist eine metadata.ini Datei vorhanden |
| ANF-D-5 | Im File metadata.ini oder in den Exif daten, sind mindestens folgende Informationen enthalten: - Zeit des Snapshots - Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben) - Genaue Position des Roboters - Status am Ende des Vorganges (OK oder Fehlercode und Text) |
| ANF-R-1 | Wenn der Roboter auf einen Endschalter auffährt, wird dies im UI richtig dargestellt. |
| ANF-R-2 | Ein spezifischer Test item mit 3 Positionen soll in einer Minute abgeschlossen werden |
|  |  |
|  |  |

## Use Case

Um Zu verstehen welche Anfragen von wo aus kommen wurde ein Use Case Diagramm erstellt. Abgebildet werden die 2 Aktoren Cambotprocessor, wobei es sich um die IPA von Semjon Buzdin handelt und der Manuelle Zugriff über das UI

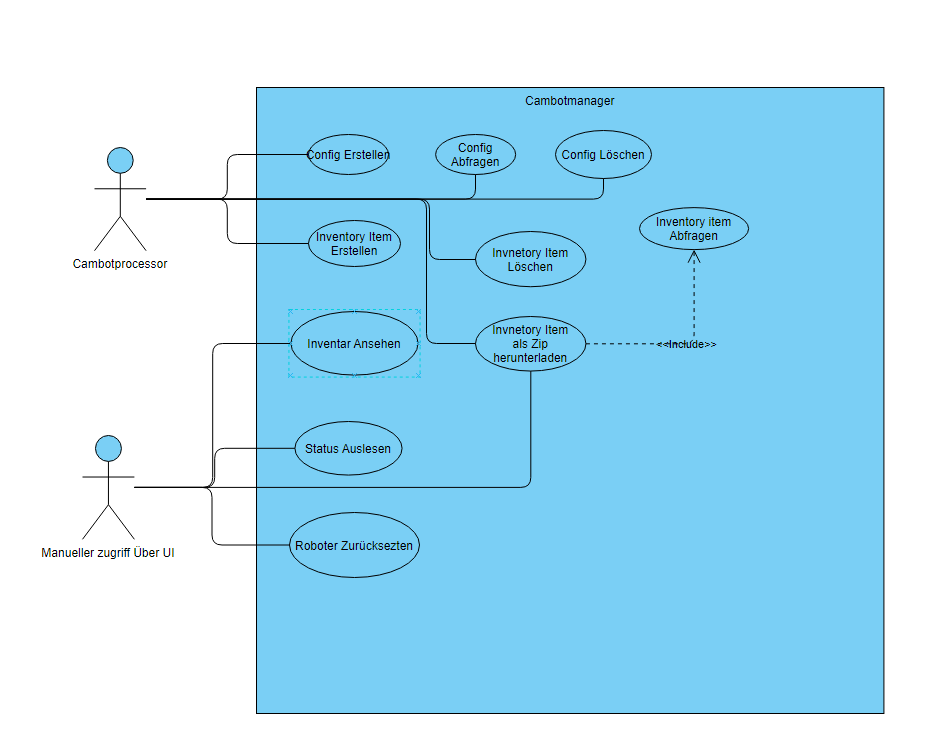


Abbildung Use Case

## Systemaufbau

Das System ist wie im unten gezeigten Diagramm aufgebaut. Es besteht aus einem Raspberry Pi 3b, auf welchem ein Reverse Proxy eingerichtet ist. Über diesen soll man auf den Cambotmanager oder das UI zugreifen können. Der Cambotmanager dient als Verbindung zum Cambot. Von aussen wird entweder über den Cambotprocessor welcher in der IPA von Semjon Buzdin erarbeitet wird oder manuell über das UI zugegriffen. Diagram

Description automatically generated

Abbildung System Diagramm

### Cambot

Der Cambot besitzt 3 Achsen, mit welchen die Kamera positioniert wird. Die A-Achse ist die Rotationsachse, um die Kamera um den 3d-Drucker zu rotieren und entspricht der X-Achse in GRBL. Ein Lineare Spindelantrieb denkt die Y-Achse ab. Die B-Achse kontrolliert die Neigung der Kamera und kann in GRBL über die Z-Achse angesteuert werden.

A picture containing floor, indoor

Description automatically generated

Abbildung Cambot

Ein Bild, das Text, Boden, drinnen enthält.

Automatisch generierte Beschreibung

Abbildung Cambot mit Achsen

Während des Informierens sammelte ich auch Infos über verschiedene Frameworks, um mit Python APIs zu erstellen. Die Informationen habe ich von verschiedenen Webseiten beschafft, welche im Quellenverzeichnis festgehalten sind.

### Django Rest

Mit Django REST können einfach REST-APIs erstellt werden. Django ist umfangreich dokumentiert und besitzt eine aktive Onlinecommunity. Django REST unterstützt auch mehrere integrierte Authentifizierungsrechtlinien.

### Flask Restful

Flask ist ein Framework, mit welchem REST-APIs schnell und einfach erstellt werden können. Genau wie Django besitzt auch Flask eine umfangreiche Dokumentation so wie eine aktive Onlinecommunity. Flask hat den zusätzlichen Vorteil das ich bereits damit gearbeitet habe und somit etwas vertraut bin.

### Fast API

Fast API ist eine der schnellsten API-Frameworks. Das Fast bezieht sich auf die Anzahl Queries pro Sekunde, sondern auch die benötigte Zeit, um ein laufendes API zu erstellen.

## Reverse Proxy

Um eine gute Entscheidung zu treffen informierte ich mich auch über verschiedene Möglichkeiten, einen Reverse Proxy auf einem Raspberry Pi zu erstellen.

### NGINX VS Apache

NGINX und Apache sollen in etwa die gleiche Komplexität beim Einrichten besitzen. Allerdings besitzt NGINX einen kleineren Memory Fussabdruck als Apache, was für einem Betrieb auf einem Raspberry von Vorteil wäre.

## Python Libraries

### Pyserial

Pyserial ist eine Library für Pyton. Sie vereinfacht das Senden von Strings über USB. Da die Kommunikation mit dem Roboter über eine USB-Schnittstelle mit G-Code vorgenommen werden muss ist dies sehr nützlich. Ich durfte mich bereits vor der IPA damit ausprobieren ([siehe Vorarbeiten](#_Vorarbeiten)), weshalb ich weiss, dass dies die beste Wahl ist.

### Pyrealsense2

Pyrealsense2 ist die Python Library, für die in diesem Projekt verwendete Intel Realsense D4325. Da es auch möglich sein soll Tiefen-Aufnahmen zu erstellen und dies nur mit dieser Library möglich ist.  
Jedoch könnte es sein, dass Pyrealsense2 nicht mit dem Raspberry kompatibel ist. Weshalb möglicherweise auf OpenCV zurückgegriffen werden muss.

# Planen

## Cambotmanager

Das Verwalten der Daten soll unabhängig vom Erstellen von Snapshots sein. Dazu soll im Cambotmanager ein Cambot-Handler erstellt werden. Dieser soll als Zustandsmaschine realisiert sein und immer wieder von einem Background Scheduler von Appscheduler aufgerufen werden. Dies soll das Erstellen von Snapshots von API-Calls abtrennen.

### Cambot-Handler

Der Cambot-Handler soll, sobald sich ein Inventory-Item im Inventar befindet, soll er dieses aus dem Inventar holen und die Snapshots für dieses Inventory-Item erstellen. Dazu fährt er die in der Konfiguration des Items festgelegten Positionen an und macht dann an diesen Stellen Fotos.

#### Aufbau

Der Ablauf im Cabot-Handler soll nach dem Statemachine Prinzip aufgebaut sein. So soll er mindestens über die Zustände

* Idle, in welchem auf das nächste Item gewartet wird.
* Ein Zustand in welchem die Snapshots ausgeführt werden und ein
* Ein Homing zustand damit der Roboter nach jedem ausgeführten Inventory-Item wieder in seine Ausgangsposition zurückfährt.

## Reverse Proxy

### Autorisierung

Das UI und das API müssen Geschütz werden. Dies soll direkt über den Reverse Proxy geschehen. Das heisst ein Zugriff soll nur zugelassen werden, wenn er von einer Autorisierten IP stammt oder mit einem Passwort und Benutzer bestätigt wurde. Der Domain Name soll «cambot» lauten und das UI soll über «cambot/» erreichbar sein.

## UI

Das UI hat die Aufgabe, den momentanen Status des Roboters anzuzeigen. Falls etwas schief gegangen ist, soll man den Roboter zurücksetzen können. Ausserdem soll auf der Seite das momentane Inventar des Roboters angezeigt werden können und die einzelnen Inventory-Items heruntergeladen werden. Realisiert werden soll das UI mit HTM, CSS und Javascript. Javascript wird unter anderem für die Kommunikation mit der API benötigt.

Graphical user interface, application

Description automatically generated

Abbildung UI Mockup

# Entscheiden

## Technologie für den Reverse Proxy

Um den Reverse Proxy zu realisieren habe ich mich für den NGINX Proxy Manager entschieden, da dieser ein UI besitzt welches das Konfigurieren des Reverse Proxys stark vereinfacht. Ausserdem besitzt NGINX eine gute Dokumentation und wird, wie ich es in einer Google Recherche herausfinden konnte, von vielen Personen empfohlen.

## Technologie für API

Um die API zu realisieren, entschied ich mich für [Flask](#_Flask_Restful). Der Grund dazu war, dass ich bereits mit Flask gearbeitet habe und damit vertraut bin. Damit ich nicht Zuviel Zeit benötige mich in ein neues Framework einzuarbeiten habe ich mich dafür entschieden.

## Aufbau des Cambotmanagers

Beim Aufbau des Cambotmanagers entschied ich mich früh dazu, die Applikation in mindestens drei Teile aufzuteilen: Der Manager, welcher alles überwacht und die Daten aufbereitet, der Storage-Handler, welcher für den Speicher zuständig ist und der Cambot-Handler, welcher das Steuern des Roboters übernimmt. Für diese Struktur habe ich mich entschieden da so eine klare Struktur mit einzelnen Zuständigkeiten entsteht.

# Realisieren

## Schnittstellen

Das Projekt verwendet zwei Serielle Schnittstellen über USB, um die Kamera und den Roboter anzusteuern. Ausserdem stellt das Projekt selbst eine REST-Schnittstelle zur Verfügung worüber der Cambotmanager angesteuert wird.

## Cambotmanager

Der Cambotmanager ist das Herz des ganzen Systems. Zusammengesetzt ist er aus der API, den Models, dem Manager, dem Storage-Handler, dem Cambot-Handler, der Kamera und dem Roboter. Die API kommuniziert dabei nur mit dem Manager.

### Benötigte Python Libraries

Um den Cambotmanager umzusetzen, wurden einige Python Libraries benötigt. Die Libraries so wie der Grund weshalb sie benötigt wurden, werden hier kurz aufgelistet.

|  |  |
| --- | --- |
| Library | Verwendungszweck |
| Pyserial | Pyserial wird benötigt, um Strings wie G-Code über USB zu senden |
| Pyrealsense2 | Pyrealsense2 ist die Library von Intel Realsense. Sie wurde verwendet, um Tiefen und RBG Bilder zu erstellen. Konnte nicht auf Raspberry installiert werden. |
| Opencv-python | Ist die OpenCV Library für Python. Sie wurde verwendet, um Fotos mit der Kamera zu schiessen. |
| APScheduler | Hier wird spezifisch nur der Background Scheduler benötigt. Dieser ruft den Cambot-Handler in einem Fixen Intervall auf. |
|  |  |

### Models

Um die Daten einfacher handhaben zu können wurden Models geschrieben. Diese Models orientieren sich zu einem grossen Teil an den auf Swagger dokumentierten Schemas. Es wurden nur Änderungen vorgenommen um zusätzliche Infos welche Für gewisse Funktonen benötigt werden hinzuzufügen. Dazu gehören zum einen das Inventory, in welchem alle Inventory-Items gespeichert werden. Im Inventory-Item wurden noch die Anzahl Tage bis zum Löschen und im Snapshot ein Boolean, ob es sich um den letzten Snapshot im Inventory-Item handelt, erweitert. Der Aufbau sieht wie folgt aus:

A diagram of a flowchart

Description automatically generated with low confidence

Abbildung Model Aufbau

#### Models zu Json

Um die Daten in der Richtigen Form zurückzugeben, wurde eine Json Umwandlungsmethode («toJSON») bei jedem Model hinzugefügt. Über diese Methode lässt sich jedes beliebige Model in das Json Format umwandeln.Text

Description automatically generated

Abbildung To JSON Methode

### Manager

Die Aufgabe des Managers ist es, die Daten aus den Models aufzuarbeiten, Zip-Dateien zu erstellen und diese an das API weiterzugeben.   
Angesteuert wird der Manager von der API. Es ist auch der einzige Weg, wie die API mit den Daten und dem Roboter Interagieren kann. Über den Manager kann auf das Inventar, den Cambot- sowie den Storage-Handler zugegriffen werden.

Text

Description automatically generated

Abbildung Manger Aufbau

### Storage-Handler

Der Storage-Handler überwacht den Speicherplatz und sorgt, falls nötig, für Ordnung. Hier sollen regelmässig alle Inventory-Items mit dem Status «scheduled\_delete» gelöscht werden.

A picture containing text

Description automatically generated

Abbildung Storage-Handler Aufbau

### Cambot-Handler

Der Cambot-Handler steuert den Roboter und die Cammera an und macht so die Snapshots. Er hohl sich ein Inventory-Item aus dem Inventar und führt die dazu gehörende Config aus. Es kann immer nur ein Inventory-Item gleichzeitig mit Snapchats befüllt werden.

Diagram

Description automatically generated

Abbildung Cambot-Handler Statemachine

#### Auflistung aller Zustände

##### Idle

In diesem Zustand passiert nichts. Hier wird gewartet, bis ein neues Inventory Item erstellt wurde. Es wird in den Nächsten zustand gewechselt, sobald die Länge der Liste «Todo» im Inventory grösser als null ist.

##### Get Inventory-Item

Hier wird das Inventory-Item und die Konfiguration aus dem Inventory geladen. Wenn dies funktioniert hat, geht es im nächsten Zustand weiter.

##### Run Config

Hier werden die Positionen, welche in der Konfiguration definiert sind, angefahren und die Fotos geschossen. Sobald der letzte Snapshot geschossen wurde, geht es in den nächsten Zustand über.

##### Return Inventory-Item

Hier wird das Inventory-Item im Inventory von der Liste «Todo» in die Liste «Done» verschoben. Ausserdem wird der Status des Items auf «done» gesetzt.

##### Home

Der Status des Roboters wird auf «homing» gesetzt und der Roboter wird in seine Anfangsposition zurückversetzt. Danach folgt wieder der «»Idle» Zustand.

### Camara

Die Camara Klasse macht, wie der Name vermuten lässt, Fotos über eine Intel Realsense D435 Kamera. Momentan sind nur RGB aufnahmen möglich, da für die Umsetzung mit Tiefen Bilder das Pyrealsense2 Package für Python benötigt wird, welches allerdings auf dem Raspberry nicht installierbar ist. Die Kamera gibt den Pfad zu den erstellten Bildern sowie der zum Speichern dieser Bilder benötigten Speicherplatzes zurück.

### Robot

In dieser Klasse passiert das Steuern des Roboters. Hier wird aus der anzufahrenden Position für den Roboter verständlicher G-Code generiert und an den Roboter gesendet. Sollte etwas schiefgehen, so wird der Roboter in die Anfangsposition zurückgesetzt.

#### Testrobot

Der Test Roboter ist eine Abstraktion der Roboter Klasse, um das Testen der API möglich zu machen, ohne dass der Raspberry am Roboter angeschlossen ist.

## Reverse Proxy

Der Reverse Proxy wurde nach der Anleitung auf der Seite von [NGINX Proxy Manager](https://nginxproxymanager.com/guide/#project-goal) installiert und aufgesetzt. Da dort alles sehr gut erklärt ist, wird hier lediglich die Konfiguration genauer erläutert.

### Konfiguration

Der Proxy Host wurde über das UI von Proxy Manager erstellt und konfiguriert. So wurde der Domain Name festgelegt, welcher auf die festgelegte IP weiterleiten soll. Ausserdem wurde die zuvor erstellte Access List als Zugriffschutz festgelegt.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

### Autorisation

Wie das Erstellen des Proxy Host, wurde auch die Autorisierung im UI von Proxy Manager konfiguriert. So wurde eine Access List mit 2 Benutzern und Blockierten IP-Adressen erstellt. So ist nun der Zugriff nur möglich, falls er von einer Autorisierten IP-Adresse stammt oder mit der Hilfe eines Benutzernamens und Passwort bestätigt wurde.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Access List Details

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Access List Authorizaztion

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Access List IP-Autorisation

## UI

Das UI besteht klassisch aus je einem HTML, CSS und Javascript(JS) File. Damit die Seite auf den meisten Geräten anschaulich und responsive ist, wurde die Webseite mit einem CSS Grid als Grundlage erstellt. Darin wurden die Inhalte, wie im Mockup in der Planung festgehalten, verteilt. Zu guter Letzt wurde das UI mit Javascript an die API angebunden. So werden die Inhalte wie Status, und das Inventar, sobald die Seite geöffnet wird, aus der API geladen. Die Buttons «Download» und «Reset» lösen ihrerseits Funktionen auf der API aus. So kann über den Download Button die Zip-Datei eines Inventory-Item heruntergeladen werden.

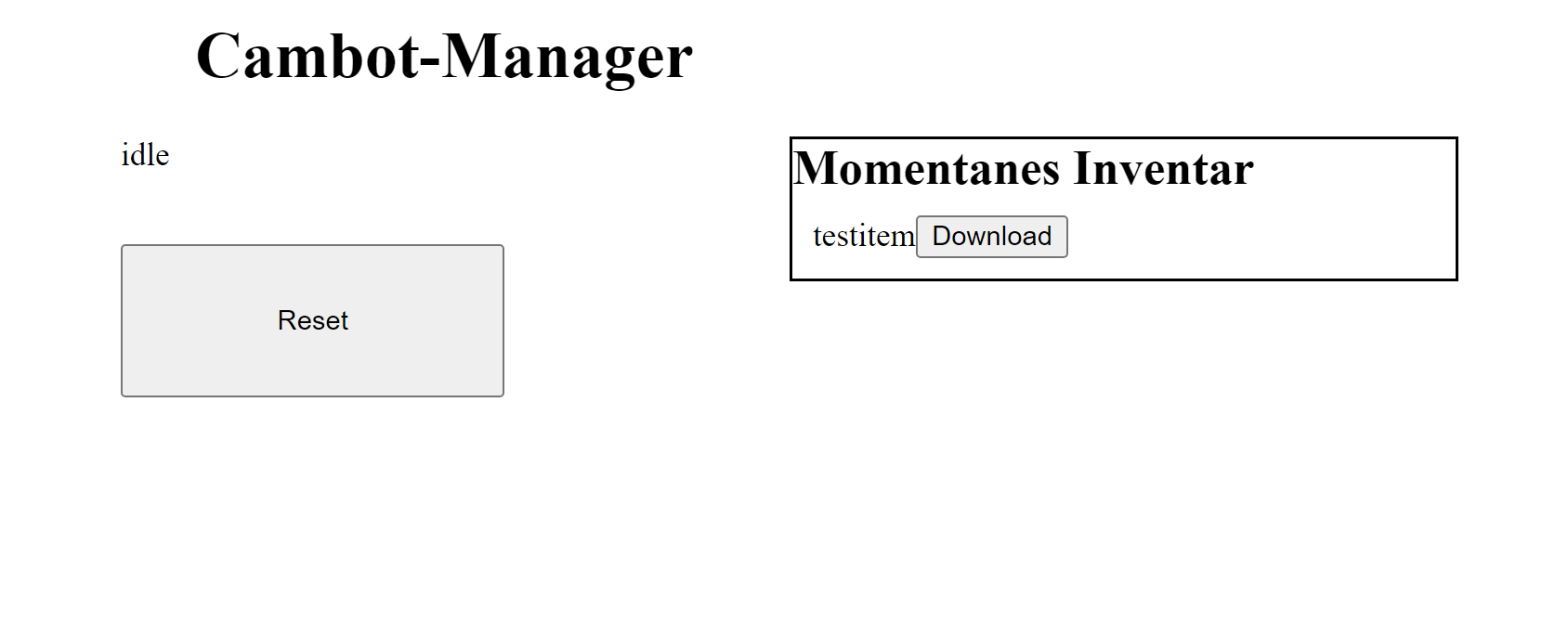


Abbildung UI

# Kontrollieren

Die Funktionalität des Projekts wurde mit Funktionstests so wie Unit Tests sichergestellt. Die Funktionstests wurden nach der Realisierungsphase durchgeführt und Ihre Ergebnisse in einem Testprotokoll festgehalten.

## Unit Tests

Mit Hilfe der Unit Tests wird vor allem die Anforderung ANF-RA-2 getestet. So überprüfe ich, dass alle in Swagger Dokumentierten Funktionen vorhanden sind und Funktionieren.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Storage Tests

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung Config Tests

## Testfallspezifikationen

In diesem Kapitel wurde für jede Anforderung mindestens 1. Testfall spezifiziert. Jeder Testfall ist eindeutig mit einer ID erkennbar welche sich nach dem Schema «Tf-GT-##». GT steht hier für «Getesteter Teil», so haben Testfälle für Anforderungen an die REST API eine ID in der Art von «Tf-RA-##».

### Rest API

#### Testfall Tf-RA-01

|  |  |
| --- | --- |
| Testfall | Tf-RA-01 |
| Anforderung | ANF-RA-1 |
| Kurzbeschreibung | Die API ist aufgrund von autorisierten IP-Adressen und Reverse Proxy geschützt. |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Der Roboter ist angeschlossen oder es ist der Test Roboter eingesetzt.  IP-Adresse des Pcs ist im Reverse Proxy autorisiert. |
| Eingabe | Erwartete Ausgabe |
| 1. Starten sie Postman 2. Erstellen sie folgenden Call in Posteman  cambot/inventory Methode: Get | Der Call sollte eine Liste mit den IDs aller Items zurückgeben. |

### UI

#### Testfall Tf-UI-01.1

|  |  |
| --- | --- |
| Testfall | Tf-UI-01.1 |
| Anforderung | ANF-UI-1 |
| Kurzbeschreibung | Die UI Muss geschützt werden. |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Die IP des PCs ist nicht autorisiert.  Tf-RA-01 |
| Eingabe | Erwartete Ausgabe |
| 1. Öffne «cambot/» in einem Browser | War der Test Tf-RA-01 erfolgreich so wird es auch dieser sein. Der Grund ist, das beide über den Selben Proxy Host angesteuert werden.  Ansonsten:  Die Webseite wird nicht geöffnet. Es wird nach einem Benutzer und Passwort gefragt. |

#### Testfall Tf-UI-01.2

|  |  |
| --- | --- |
| Testfall | Tf-UI-01.2 |
| Anforderung | ANF-UI-1 |
| Kurzbeschreibung | Die UI Muss geschützt werden. |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet.  Tf-RA-01 war dieser Test erfolgreich so wird es auch dieser sein. |
| Eingabe | Erwartete Ausgabe |
| 1. Statische IP auf PC festlegen 2. IP auf dem Reverse Proxy autorisieren. 3. Öffnen sie «cambot» in einem Browser | Die Webseite öffnet sich. Es wird nicht nach einem Passwort gefragt. |

#### Testfall Tf-UI-02

|  |  |
| --- | --- |
| Testfall | Tf-UI-02 |
| Anforderung | ANF-UI-2 |
| Kurzbeschreibung | Der Momentane Status muss über das UI ersichtlich sein |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Der Roboter ist angeschlossen. |
| Eingabe | Erwartete Ausgabe |
| 1. Öffnen sie «cambot/» in einem Browser 2. Falls die IP des PCs nicht autorisiert ist, Melden sie sich mit folgendem Benutzer an.  **Benutzer**: **Passwort**: | Die Webseite öffnet sich. Unter Dem Titel auf der linken Seite ist der momentane Status ersichtlich |

#### Testfall Tf-UI-03

|  |  |
| --- | --- |
| Testfall | Tf-UI-03 |
| Anforderung | ANF-UI-03 |
| Kurzbeschreibung | Im UI soll der Roboter zurückgesetzt werden können. |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Der Roboter ist angeschlossen oder es ist der Test Roboter eingesetzt. |
| Eingabe | Erwartete Ausgabe |
| 1. Öffnen sie «cambot/» in einem Browser 2. Falls die IP des PCs nicht autorisiert ist, Melden sie sich mit folgendem Benutzer an.  **Benutzer**: **Passwort**: 3. Klicken sie auf den Button resett | Der Roboter wird in seinen Home zustand zurückgesetzt. Und das Momentane Item beendet. |

#### Testfall Tf-UI-04

|  |  |
| --- | --- |
| Testfall | Tf-UI-04 |
| Anforderung | ANF-UI-4 |
| Kurzbeschreibung | Das Aktuelle Inventar soll im UI aufgelistet sein |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Der Roboter ist angeschlossen oder es ist der Test Roboter eingesetzt. |
| Eingabe | Erwartete Ausgabe |
| 1. Öffnen sie «cambot/» in einem Browser 2. Falls die IP des PCs nicht autorisiert ist, Melden sie sich mit folgendem Benutzer an.  **Benutzer**: **Passwort**: | Die Webseite öffnet sich. Auf der rechten Seite ist das komplette Inventar ersichtlich. |

#### Testfall Tf-UI-05

|  |  |
| --- | --- |
| Testfall | Tf-UI-05 |
| Anforderung | ANF-UI-5 |
| Kurzbeschreibung | Elemente aus dem Inventar sollen heruntergeladen werden können. |
| Voraussetzung | Die API und der Reverse Proxy sind auf dem Raspberry gestartet. Der Roboter ist angeschlossen oder es ist der Test Roboter eingesetzt. |
| Eingabe | Erwartete Ausgabe |
| 1. Erstellen sie eine Config 2. Erstellen sie ein Inventory Item 3. Öffnen sie «cambot» in einem Browser 4. Falls die IP des PCs nicht autorisiert ist, Melden sie sich mit folgendem Benutzer an.  **Benutzer**: **Passwort**: 5. Klicken sie auf den Dowload Button hinter dem gerade erstellten Item. | Es wird eine Zip Datei erstellt und zum Download bereitgestellt. |

### Datenstruktur

#### Testfall Tf-D-01

|  |  |
| --- | --- |
| Testfall | Tf-D-01 |
| Anforderung | ANF-D-1, ANF-D-2, ANF-D-3 |
| Kurzbeschreibung | Die Bilder werden von der API als Zip File retourniert und ist nach der Gewünschten Struktur aufgebaut. |
| Voraussetzung | Tf-UI-05 ausgeführt und erfolgreich. |
| Eingabe | Erwartete Ausgabe |
| 1. [Tf-UI-05](#_Testfall_Tf-UI-05) 2. Die bereitgestellte Datei herunterladen und Öffnen. | Die Heruntergeladene Datei ist eine Zip File. Darin beinhaltet sind eine  «Metadata.ini» Datei so wie ein Ordner «images» in welchem die Fotos abgelegt sind.  Alle Inhalte sind nach der Struktur  <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini und images (png or jpg) abgespeichert. |

#### Testfall Tf-D-02

|  |  |
| --- | --- |
| Testfall | Tf-D-02.1 |
| Anforderung | ANF-D-4, ANF-D-5 |
| Kurzbeschreibung | In der Datei «Metadata.ini» und in der EXIF Struktur sind mindestens folgende Informationen vermerkt.   * Zeit des Snapshots * Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben) * Genaue Position des Roboters * Status am Ende des Vorganges (OK oder Fehlercode und Text) |
| Voraussetzung | Tf-D-01 ausgeführt und erfolgreich. |
| Eingabe | Erwartete Ausgabe |
| 1. Tf-D-01 2. Öffnen sie die Heruntergeladene Zipdatei. 3. Begutachten sie die Struktur | In der Datei «Metadata.ini» sind folgende Informationen Aufgelistet.   * Status am Ende des Vorganges (OK oder Fehlercode und Text)   In der EXIF-Struktur sind folgende Daten vorhanden.   * Zeit des Snapshots * Filename und Typ des Snapshots (Achtung: Es könnte Snapshots mit Tiefen und RGB-Informationen gleichzeitig geben) * Genaue Position des Roboters |

### Roboter

#### Testfall Tf-R-01

|  |  |
| --- | --- |
| Testfall | Tf-R-01 |
| Anforderung | ANF-R-1 |
| Kurzbeschreibung | Wenn der Roboter auf einen Endschalter auffährt, wird dies im UI richtig dargestellt. Oder Abgefangen |
| Voraussetzung | API und Reverse Proxy auf Raspberry gestartet und am Cambot angeschlossen |
| Eingabe | Erwartete Ausgabe |
| 1. Erstellen sie eine Config mit folgender Position a: 30 b: 5 y: 5 2. Erstellen sie ein Item mit der davor erstellten Config 3. Warten sie bis der Cambot das Item ausführt | Der Test ist erfolgreich, wenn, die Schalter nicht getroffen werden oder es im UI unter Status angezeigt wird. |

#### Testfall Tf-R-02

|  |  |
| --- | --- |
| Testfall | Tf-R-02 |
| Anforderung | ANF-R-2 |
| Kurzbeschreibung | Ein spezifischer Test item mit 3 Positionen soll in einer Minute abgeschlossen werden |
| Voraussetzung | API und Reverse Proxy auf Raspberry gestartet und am Cambot angeschlossen |
| Eingabe | Erwartete Ausgabe |
| 1. Erstellen sie eine Config mit folgenden Positionen: [{"a": 1,"y": 1,"b": 1}, {"a": 1,"y": 2,"b": 5}, {"a": 3,"y": 2,"b": 1}] 2. Erstellen sie ein Item mit dieser Config 3. Stopen sie die Zeit sobald der Roboter anfängt sich zu bewegen. | Der Roboter fährt diese Positionen ab. Dabei braucht er nicht mehr als eine Minute |

## Testprotokolle

### Testfall Protokoll TP-01

#### Kennung

|  |  |
| --- | --- |
| Testprotokoll: | TP-01 |
| Durchführungszeitpunkt: | 23.03.2022 |
| Testperson: | Maurice Meier |
| Software: | Cambotmanager |

#### Testumgebung

|  |  |
| --- | --- |
| Gerät: | HP ZBook 15 G4 |
| Betriebssystem: | Windows 10 Pro for Workstations(21H2) |
| Python Version: |  |
| Postman Version: | V9.0.9 |
| Browser: | Chrome Version 99.0.4844.74 |

#### Durchgeführte Testfälle

|  |  |  |
| --- | --- | --- |
| Testfall | Ergebnis | Bemerkung |
| Tf-RA-01 | Ok |  |
| Tf-UI-01.1 | Ok |  |
| Tf-UI-01.2 | Ok |  |
| Tf-Ui-02 | Ok | Wurde über den PC getestet |
| Tf-UI-03 | Ok | Wurde über den PC getestet |
| Tf-UI-04 | Ok | Wurde über den PC getestet |
| Tf-UI-05 | Ok | Wurde über den PC getestet |
| Tf-D-01 | Ok | Wurde über den PC getestet |
| Tf-D-02 | Ok | Wurde über den PC getestet |
| Tf-R-01 | Fehlgeschlagen | Roboter ist defekt |
| Tf-R-02 | Fehlgeschlagen | Roboter ist defekt |

#### Kommentar

Alle Testfälle bis auf Tf-R-01 und Tf-R-02 Waren erfolgreich. Diese 2 funktionierten nicht da der Roboter defekt war.

#### Autogramm

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Abbildung 18 Autogramm

# Auswerten

## Fazit

Alle Anforderungen wurden mithilfe von Unit Tests oder Testfällen geprüft und in einem Testprotokoll festgehalten. Da alle Unit Tests durchgelaufen sind und, wie im Testprotokoll ersichtlich, alle Tests. Bis auf die Roboter Tests Funktionierten war das Projekt mehrheitlich erfolgreich. Da der Roboter, die meiste Zeit über defekt war, konnte ich den Code für diesen erst am Fr. Morgen fertig stellen. Konnte ich diese Testfälle nicht mehr testen.

## Persönliches Fazit

Die Durchführung der IPA war mit den von mir erdachten Lösungen erfolgreich. Bis auf drei Zeitintensive Schwierigkeiten, konnte ich alle Problemstellungen bewältigen. Die grossen drei Probleme bezogen sich vor allem auf den Roboter und den Raspberry Pi. Rückblickend denke ich, dass diese Schwierigkeiten davon rührten, dass ich meine Fachkenntnisse über diese Systeme erweitern muss. Ich bin zwar mit den Systemen vertraut, allerdings habe ich noch nicht so viel Erfahrung mit ihnen. So hatte ich die Schwierigkeit, dass der Raspberry beim Starten nur noch in einem Komplet orangen Bildschirm anzeigte, in welchem nichts gemacht werden konnte. So musste ich ihn neuaufsetzen und alles nochmals installieren. Dies ist zum Glück später nichtmehr aufgetreten verbrauchte allerdings einiges an Zeit.

Weitere Schwierigkeiten bereitete mir die Library von Intel Realsense. Hier hatte ich beim Informieren übersehen, dass dieses Package nur auf Prozessoren mit x Architektur läuft wie x86. Der Prozessor der Raspberry ist ein Arm. So musste ich auf OpenCV zurückgreifen.

Mit dem Roboter hatte ich auch einige Schwierigkeiten. In Vorarbeit zur IPA durfte ich mich mit dem Roboter bekannt machen und ihn bereits über Python ansteuern. Dies funktionierte damals auch. Jedoch machte ich im Verlauf des Projekts einen Fehler beim Konfigurieren wodurch eine Componente des Roboters durchschmorte. Dies wirkte sich stark auf meine Zeitplanung aus, so konnte ich den Roboter code erst am Morgen des 25.03 fertigstellen.

Trotz dieser Herausforderungen konnte ich mein am ende doch noch das Projekt fertigstellen.

## Mögliche Erweiterungen

Weiterführend könnte man sicherlich das UI erweitern. Es Wäre möglich Formulare hinzuzufügen, um das Erstellen von Inventory-Items und Configs zu vereinfachen. Dies ist momentan nur als API-Call möglich. Des Weiteren könnten im UI mehr Informationen ersichtlich sein.

# Quellenverzeichnis

## Glossar

|  |  |
| --- | --- |
| Begriff | Bedeutung |
| Access List | Die Access List ist die Konfigurationsdatei für den Zugriffschutz bei NGINX Proxy Manager |
| Cambot | Dies ist der Roboter, welcher von Martin Gwerder entwickelt wurde |
| Config | Die Config ist ein Model in welcher die Positionen für ein Inventory item gespeichert sind. Es kann mehrmals verwendet werden |
| CSS | CSS ist ein Stylsheet für HTML |
| Domain Name | Dies ist der Name, worüber auf die IP zugegriffen wird, |
| HTML | HTML Ist eine Auszeichnugssprache zum Strukturieren von virtuellen Dokumenten. Z.B. Webseiten. |
| Inventory | Hier werden die Erstellten inventory items gelagert |
| Inventory Item | Ein Model, um Bestellungen bei der API darzustellen. |
| Javascript | Ist eine Scriptsprache, welche sich gut für Webseiten eignet, da jeder Browser Javascript versteht. |
| JSON | Ein JSON ist einfaches Datenformat zum vereinfachten Darstellen von Informationen. |
| Proxy Host | Dies ist die Konfigurationsdatei für die Reverse Proxy Weiterleitung im NGINX Proxy Manager |
| Rest API | Eine Spezielle Form einer Programmierschnittstelle (API) |
| Snapshot | Dies ist ein Model, in welchem das Foto und zusätzliche Informationen gespeichert sind. |
| UI | Das User Interface |
| Zustandsmaschine | Ist eine Möglichkeit um verschiedene Schritte, welche nacheinander ablaufen müssen zu realisieren. |

## Internetquellen

* <https://geekflare.com/de/python-frameworks-for-apis/> [stand 25.03.2022]
* <https://singleboardbytes.com/1002/running-flask-nginx-raspberry-pi.htm>   
  [stand 25.03.2022]
* <https://serverfault.com/questions/143238/nginx-vs-apache-as-reverse-proxy-which-one-to-choose> [stand 25.03.2022]
* <https://nginxproxymanager.com/guide/#project-goal> [stand 25.03.2022]
* <https://apscheduler.readthedocs.io/en/3.x/userguide.html> [stand 25.03.2022]
* <https://github.com/IntelRealSense/librealsense/blob/master/doc/installation_raspbian.md>[stand 25.03.2022]
* <https://stackoverflow.com/questions/34588464/python-how-to-capture-image-from-webcam-on-click-using-opencv>[stand 25.03.2022]
* <https://www.precifast.de/cnc-programmierung-mit-g-code/>[stand 25.03.2022]
* <https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-uswgi-and-nginx-on-ubuntu-18-04> [stand 25.03.2022]
* <https://github.com/gnea/grbl/blob/master/doc/markdown/commands.md> [stand 25.03.2022]

## Abbildungsverzeichnis

[Abbildung 1 Commit History 7](#_Toc99100234)

[Abbildung 2 Use Case 15](#_Toc99100235)

[Abbildung 3 System Diagramm 16](#_Toc99100236)

[Abbildung 4 Cambot 17](#_Toc99100237)

[Abbildung 5Cambot mit Achsen 17](#_Toc99100238)

[Abbildung 6 UI Mockup 21](#_Toc99100239)

[Abbildung 7 Model Aufbau 24](#_Toc99100240)

[Abbildung 8 To JSON Methode 25](#_Toc99100241)

[Abbildung 9 Manger Aufbau 25](#_Toc99100242)

[Abbildung 10 Storage-Handler Aufbau 26](#_Toc99100243)

[Abbildung 11 Cambot-Handler Statemachine 26](#_Toc99100244)

[Abbildung 12 Access List Details 29](#_Toc99100245)

[Abbildung 13 Access List Authorizaztion 29](#_Toc99100246)

[Abbildung 14 Access List IP-Autorisation 30](#_Toc99100247)

[Abbildung 15 UI 31](#_Toc99100248)

[Abbildung 16 Storage Tests 32](#_Toc99100249)

[Abbildung 17 Config Tests 32](#_Toc99100250)

[Abbildung 18 Autogramm 40](#_Toc99100251)

## Anhang

Im Anhang befinden sich die Benutzeranleitung und Benutzerdokumentation welche auch als Readme auf [GitHub](https://github.com/Red8Bee/Cambotmanager-IPA-Maurice-Meier) zu finden ist.