Restschnittstelle für einen Kameraroboter

Individuelle Praktische Arbeit

Titelbild

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

# Änderungsnachweis

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Beschreibung |
| 0.1 | 07.03.2022 | Maurice Meier | Teil 1: Umfeld und Ablauf, Tagesjournal |
| 0.2 | 09.03.2022 | Maurice Meier | Teil 2: Informieren, Tagesjournal |
| 0.3 | 10.03.2022 | Maurice Meier | Teil 2: Planen, Tagesjournal |
| 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 1: Tagesjournal  Teil 2: Realisieren |
| 0.7 | 17.03.2022 | Maurice Meier | Teil 2: Realisieren |
|  |  |  |  |

EINTRAGEN!!!!

Inhaltsverzeichnis

[1. Änderungsnachweis 1](#_Toc98312232)

[2. Kurzfassung 5](#_Toc98312233)

[2.1. Ausgangssituation 5](#_Toc98312234)

[2.2. Umsetzung 5](#_Toc98312235)

[2.3. Ergebnis 5](#_Toc98312236)

[3. Projektorganisation 6](#_Toc98312237)

[3.1. Involvierte Personen 6](#_Toc98312238)

[3.2. Projektmethode¨ 6](#_Toc98312239)

[3.3. Dokumentationsaufbau 6](#_Toc98312240)

[3.4. Datensicherung 7](#_Toc98312241)

[4. Aufgabenstellung 8](#_Toc98312242)

[4.1. Titel der Arbeit 8](#_Toc98312243)

[4.2. Ausgangslage 8](#_Toc98312244)

[4.3. Detaillierte Aufgabenstellung 8](#_Toc98312245)

[4.4. Mittel und Methoden 8](#_Toc98312246)

[4.5. Vorkenntnisse 9](#_Toc98312247)

[4.6. Vorarbeiten 9](#_Toc98312248)

[5. Zeitplan 10](#_Toc98312249)

[6. Arbeitsjournal 11](#_Toc98312250)

[6.1. Montag, 06.03.2022 / Tag 1 11](#_Toc98312251)

[6.1.1. Probleme/Fragen & Ergriffene Massnahmen 11](#_Toc98312252)

[6.1.2. Bemerkungen 11](#_Toc98312253)

[6.2. Mittwoch, 08.03.2022 / Tag 2 12](#_Toc98312254)

[6.2.1. Probleme/Fragen & Ergriffene Massnahmen 12](#_Toc98312255)

[6.2.2. Bemerkungen 12](#_Toc98312256)

[6.3. Donnerstag, 08.03.2022 / Tag 3 13](#_Toc98312257)

[6.3.1. Probleme/Fragen & Ergriffene Massnahmen 13](#_Toc98312258)

[6.3.2. Bemerkungen 13](#_Toc98312259)

[6.4. Freitag, 11.03.2022 / Tag 4 14](#_Toc98312260)

[6.4.1. Probleme/Fragen & Ergriffene Massnahmen 14](#_Toc98312261)

[6.4.2. Bemerkungen 14](#_Toc98312262)

[6.5. Montag, 14.03.2022 / Tag 5 15](#_Toc98312263)

[6.5.1. Probleme/Fragen & Ergriffene Massnahmen 15](#_Toc98312264)

[6.5.2. Bemerkungen 15](#_Toc98312265)

[6.6. Mittwoch, 16.03.2022 / Tag 6 16](#_Toc98312266)

[6.6.1. Probleme/Fragen & Ergriffene Massnahmen 16](#_Toc98312267)

[6.6.2. Bemerkungen 16](#_Toc98312268)

[7. Informieren 17](#_Toc98312269)

[7.1. Anforderungsanalyse 17](#_Toc98312270)

[7.2. Systemaufbau 18](#_Toc98312271)

[7.2.1. Camrobot 19](#_Toc98312272)

[7.3. Api Framework 19](#_Toc98312273)

[7.3.1. Django Rest 19](#_Toc98312274)

[7.3.2. Flask Restful 19](#_Toc98312275)

[7.3.3. Fast API 20](#_Toc98312276)

[7.4. Reverse Proxy 20](#_Toc98312277)

[7.4.1. NGINX VS Apache 20](#_Toc98312278)

[8. Planen 21](#_Toc98312279)

[8.1. Cambotmanager 21](#_Toc98312280)

[8.2. Reverse Proxy 21](#_Toc98312281)

[8.3. UI 21](#_Toc98312282)

[9. Entscheiden 21](#_Toc98312283)

[10. Realisieren 21](#_Toc98312284)

[11. Kontrollieren 22](#_Toc98312285)

[11.1. Unit Tests 22](#_Toc98312286)

[11.2. Testfallspezifikationen 22](#_Toc98312287)

[11.2.1. Getesteter Teil 22](#_Toc98312288)

[11.3. Testprotokolle 23](#_Toc98312289)

[11.3.1. Testfall Protokoll TP-01 23](#_Toc98312290)

[12. Auswerten 24](#_Toc98312291)

[12.1. Mögliche Erweiterungen 24](#_Toc98312292)

[12.2. Fazit 24](#_Toc98312293)

[12.3. Persönliches Fazit 24](#_Toc98312294)

[13. Quellenverzeichnis 24](#_Toc98312295)

[13.1. Glossar 24](#_Toc98312296)

[13.2. Internetquellen 24](#_Toc98312297)

[13.3. Bildquellen 24](#_Toc98312298)

AKKTUALISIEREN

Teil 1: Obligatorisches Kapitel

# Kurzfassung

## Ausgangssituation

## Umsetzung

## Ergebnis

# Projektorganisation

## Involvierte Personen

**Hauptexperte:**

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

**Nebenexperte:**

Jonathan Teig  
Rolltreppen Weg 3  
1235 Nullhausen   
+41 14 411 44 11

**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¨

Gewählte methode und grund

Für die Durchführung dieses Projekt wurde 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 wurde diese Methode bereits in Projekten in der Schule vom Kandidaten verwendet. (Ich person ??)

## Dokumentationsaufbau

Struktur der Der Doku wie viele Teile was ist in diesen Teilen

Die Dokumentation ist insgesamt in 2 Teile gegliedert. Teil 1 Obligatorisches Kapitel beinhaltet allgemeine Informationen zur Arbeit sowie eine Kurzfassung, Auflistung aller beteiligten, Die Aufgabenstellung und 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

Wie wird das Projekt gesichert und wann

Die Datensicherung wird mithilfe von Täglichen commits auf ein Github repo so wie Das Speichern des Projekts auf einem zusätzlichen USB-Stick damit immer ein backup mit der momentan aktuellsten Version vorhanden ist.

# Aufgabenstellung

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

## Titel der Arbeit

Restschnittstelle 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 RestAPI 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 Calls, die ein Zipfile retournieren haben ein Zipfile, das Strukturiert ist wie folgt: <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 eine kleine 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

PK ORG

## 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-Files 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

EINFÜGEN

# Arbeitsjournal

## Montag, 06.03.2022 / Tag 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 2h | 2h | Zeitplan | Zeitplan erstellen | Zeitplan wurde erstellt |
| 2h | 2h | Informieren | Anforderungsanalyse | Anforderungen wurden aus der Aufgabenstellung abgeleitet. |
| 20 min | 10min | Datensicherung | GitHub erstellen | Github repo wurde erstellt. |
| 2h | 2h | Informieren | System Aufbau festhalten | Das System wurde analysiert und als Diagramm festgehalten unter ([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 wurde geführt |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden Kurz vor 17:00 auf GitHub gepushed 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 Eigene .ini Datei Pro Bild oder Pro Zipfile? | **||** |
|  |  |

### Bemerkungen

Martin Gwerder erteilte uns den Auftrag für die IPA bereits am FR. Abend (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 2 Teile im 1.Teil der Dokumentation.

## Mittwoch, 08.03.2022 / 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 |  | Informieren | Informieren fertig stellen |  |
| WnZ. |  | Planen | Mit Planen beginnen |  |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden Kurz vor 17:00 auf GitHub gepushed und auf USB gespeichert. |

### 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  Anforderungs Analyse, Entscheidungen, aufkommende Fragen | Termin mit Martin Gwerder am Donnerstag 10.03.2022 15-16 Uhr |
|  |  |

### Bemerkungen

An diesem Tag war 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.

## Donnerstag, 08.03.2022 / 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 | Mockup erstellt |
| 30 min | 30 min | Entscheiden | Termin mit Martin Gwerder, Besprechen von getroffenen Entscheidungen, Anforderungsanalyse und offenen Fragen | Ich konnte die Anforderungen zusammen mit 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 Kurz vor 17:00 auf GitHub gepushed und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

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

### Bemerkungen

Die Anforderungen wurden mit Martin Gwerder besprochen. Es sollen noch kleinere Anpassungen vorgenommen werden und Anforderungen an den Roboter hinzugefügt. Danach 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 |  |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden kurz vor 17:00 auf GitHub gepushet und auf USB gespeichert. |
|  |  |  |  |  |
|  |  |  |  |  |

### Probleme/Fragen & Ergriffene Massnahmen

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

### Bemerkungen

## Montag, 14.03.2022 / Tag 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 6h | 4h | Realisieren, Cambotmanager | Storage Handling und Aufräumen | Alles erreicht, Mögliche kleine Verbesserungen |
| 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-Sichern | Daten wurden kurz vor 17:00 auf GitHub gepusht und auf USB 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 Manuel umstelle |

### Bemerkungen

Der Code für den Cambotmanager ist so weit fertig und funktioniert mehrheitlich. Aufgrund des Problems mit dem Roboter muss ich beim Ansteuern des diesigen wahrscheinlich noch etwas anpassen.

## 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 abstahieren | 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-Sichern | Daten wurden kurz vor 17:00 auf GitHub gepusht und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Um den Reverse Proxy anzusteuern, muss auf dem Router der Port 80 geforwardet werden. Dies kann ich in der FH nicht selbst machen. | Ich konnte martin Gwerder darauf ansprechen. Das Problem war es das ich die vergebene URL nicht im Hostefile eingetragen habe und der pc sie so nicht auflösen konnte |
|  |  |

### Bemerkungen

## Donnerstag, 17.03.2022 / Tag 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Soll | Ist | Tätigkeit | Geplant | Erreicht |
| 1.5h | 6.5h | Realisieren | Cambotmanager auf Raspberry laufen lassen. | Nicht ganz erreicht durch zeit intensives configurieren |
| 2.5h | 2.5h | Dokumentieren | Realisieren über Cambotmanager und Reverse Proxy | erreicht |
| 4h | - | Realisieren, Reverse Proxy und CambotI | Mit UI beginnen | Konnte nochtnicht beginnen |
| 5 min | 5 min | Datensicherung | Alles auf GitHub und USB-Sichern | Daten wurden kurz vor 17:00 auf GitHub gepusht und auf USB gespeichert. |

### Probleme/Fragen & Ergriffene Massnahmen

|  |  |
| --- | --- |
| Probleme/Fragen | Ergriffene Massnahmen |
|  |  |
|  |  |
| Einige Librarys funktionieren nur auf Python version 3.7. Probleme mit dem Downgraden | Nach einer Google rechseche und fragen bei Schuhlkollegen konnte ich die Version ändern |
| Pyrealsense2 muss über umwege herunter geladen werden aufgrund von kernel achitektur des raspberrys | Google |

Teil 2: Projekt-Dokumentation

# Informieren

## Anforderungsanalyse

Die Anforderungen wurden aus der Aufgabenstellung im PKOrg abgeleitet und nach den verschiedenen Bestandteilen des Projekts sortiert z.B. Reverse Proxy.   
Jede Anforderung erhielt eine eindeutige Bezeichnung, diese richten sich nach dem folgenden Schema

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

|  |  |
| --- | --- |
| Anforderung | Beschreibung |
| ANF-RA-1 | Die RestAPI verfügt über einen Zugriffsschutz mit Basic oder Digest-Authentifikation via Reverse-Proxy. |
| ANF-RA-2 | Die Rest API verfügt über einen Zugriffsschutz aufgrund der autorisierten IP-Adressen. |
| ANF-RA-3 | Die Rest API ist nach Spezifikation auf Swagger implementiert |
| ANF-UI-1 | Die UI Muss mit einem Passwort geschützt werden |
| ANF-UI-2 | Über 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 über das UI heruntergeladen werden können. |
| ANF-D-1 | Die Bilder sind mit Metadaten in der EXIF-Struktur angereichert |
| ANF-D-2 | Die Bilder werden von der API als Zipfile retourniert |
| ANF-D-3 | Im Zip File ist eine metadata.init Datei vorhanden |
| ANF-D-4 | Im File metadata.init 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-D-5 | Die Bilder werden von der API als Zipfile retourniert |
| ANF-D-6 | Alle Calls, die ein Zipfile retournieren haben ein Zipfile, das Strukturiert ist wie folgt: <taskname> --- snapshots --- <iso8601-timestamp>--- metadata.ini and images (png or jpg) |
|  |  |

## Systemaufbau

Das System ist so wie im unten gezeigten Diagramm aufgebaut. Es soll aus einem Raspberry 3, 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 zugegriffen.

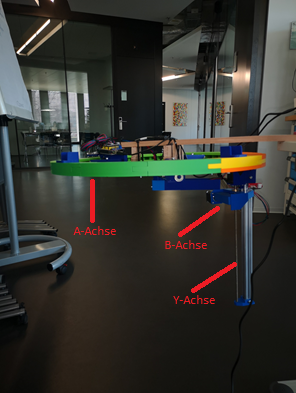
Diagram

Description automatically generated

### 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

## API-Framework

Während des Informierens sammelte ich auch Infos über verschiedene Frameworks um mit Python API’s zu erstellen. Die Infos habe ich von verschiedenen Webseiten beschafft, diese habe ich im Quellenverzeichnis festgehalten.

Infos über verschidene pyhon api frameworks/ info herkunft

### 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 einfaches Framework mitwelchem schnelle Rest-APIs erstellt werden. Genau wie Django besitzt auch Flask eine umfangreiche Dokumentation so wie eine Aktive Onlinecommunity. -Flask hat den zusätzlichen Vorteil

### 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 eine laufende API zu erstellen.

## Reverse Proxy

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

Infos über verschiedene Reverse Proxys für raspberry/ info herkunft

### NGINX VS Apache

NGINX und Apache sollen in etwa die gleiche Komplxität beim Einrichten besitzen. NGINX sei allerdings besitzt NGINX einen kleineren Memory Fussabdruck als Apache und ist somit leistungs freundlicher. Für einem Beitrib auf einem Raspberry wäre das von Vorteil da wir etwas begrentzt sind.

https://serverfault.com/questions/143238/nginx-vs-apache-as-reverse-proxy-which-one-to-choose

# Planen

## Cambotmanager

Der Cambotmanager soll

## Reverse Proxy

## UI

Das UI hat die Aufgabe den momentanen Status des Roboters anzuzeigen. Falls etwas schief gegangen ist, soll man ihn Zurücksetzen können. Ausserdem soll auf der Seite das momentane Inventar des Roboters angezeigt werden können und die einzelnen Items heruntergeladen werden.

Graphical user interface, application

Description automatically generated

# Entscheiden

Festhalten der getroffenen Entscheidungen so wie Kommunikation mit Arbeitgeber

# Realisieren

## Cambotmanager

Diesen Teil habe ich mich dazu entschieden etwas aufzuteilen damit das Projekt Übersichtlich bleibt. So besteht der Cambotmanager nun aus der API, welche die Calls verarbeitet und die antworten im richtigen Format zurückgibt. Das Handling der daten wird vom Manager vorgenommen. Dieser bereitet die daten für die API auf und überwacht die ganzen Geschehnisse. Der StorageHandler ist verantwortlich dafür das immer genug Speicherplatz vorhanden ist und Alte Item löscht.

### Models

Die Models dienen dazu die daten einfacher zu handhaben. So kann mit einer Klasse gearbeitet werden und man verliert nicht den Überblick wo was zu finden ist. Die Models sind wie folgt aufgebaut:

diagramm

### Manager

Der Manager ist dazu da, um die Daten an die API zu liefern und aufzuarbeiten. Hier werden die Zips erstellt und daten aus den einzelnen Klassen zur verfügung gestellt. Hier ist auch der Aktuelle Status zu finden. Ausserdem beinhaltet dieser das Inventar mit den ganzen gespeicherten Items.

Classen diagramm

### StorageHandler

Der StorageHandler überwacht den speicherplatz und sorgt falls nötig für Ordnung. Hier sollen regelmässig alle Items mit dem Status «scheduled\_delete» gelöscht werden.

Classen diagramm

### CambotHandler

Der CambotHandler steuert den Roboter und die Cammera an und macht so die Snapshots. Er hohl sich ein Item aus dem Inventar und führt die dazu gehörende Config aus. Es kann immer nur 1. Item gleichzeitig mit Snapchats befüllt werden.

Classen diagramm

#### Camera

Die Camera schiest Tiefen und RBG Fotos, mit hilfe einer «CAMNAME» und pyrealsense. Die Bilder werden direkt in den für den snapshot vorgesehen ordner gespeichert und ihr pfad zusammen mit der File grösse zurück gegeben.

#### Robot

Hier wird der Cambot mithilfe von G-Code angesteuert. Dazu wird G-Code über eine Serielle Schnittstelle an den Cambot gesendet, welcher dann diesen Ausführt

##### TestRobot

Der TestRobot ist eine Abstraktion des Robots. Er kann verwendet werden, um den Code zu testen, ohne den Roboter anzusteuern. Die Kamera muss allerdings angeschlossen sein.

## Reverse Proxy

Um den Reverse Proxy zu realisieren, entschied ich mich dazu den Proxy Manager von NGINX zu verwenden. Dazu wurden die Installation schritte auf der Seite von NGINX befolgt.

Nach der Installation wurde ein Proxy Host mit der IP des Raspberry und dem Port auf welchem sich die Flask Applikation befindet. Hier ist es nun der Port 8000 mit der IP 196.168.43.109

Graphical user interface, text, application, email

Description automatically generated

Danach muss der Domain Name so wie die IP im Host File oder auf dem DNS eingetragen werden, damit das Ganze auch aufgelöst wird, sobald man die Domain im Browser öffnet.

# Kontrollieren

## Unit Tests

## Testfallspezifikationen

### Getesteter Teil

#### Testfall Tf-GT-01

|  |  |
| --- | --- |
| Testfall | Tf-GT-01 |
| Anforderung |  |
| Kurzbeschreibung |  |
| Anforderung |  |
| Eingabe | Erwartete Ausgabe |
|  |  |

## Testprotokolle

### Testfall Protokoll TP-01

#### Kennung

#### Testumgebung

#### Durchgeführte Testfälle

|  |  |  |
| --- | --- | --- |
| Testfall | Ergebnis | Bemerk |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

#### Kommentar¨

#### Autogramm

# Auswerten

## Mögliche Erweiterungen

## Fazit

## Persönliches Fazit

# Quellenverzeichnis

## Glossar

## Internetquellen

* <https://geekflare.com/de/python-frameworks-for-apis/> (API Framework)
* https://singleboardbytes.com/1002/running-flask-nginx-raspberry-pi.htm

## Bildquellen