

Höhere technische Schule fur Informationstechnologie  
Shkolla e mesme profesionale private për teknologji informacioni  
Österreichiche Schule Peter Mahringer  
Shkolla Austriake Shkodër

## **Gesichtsregistrierung und Gesichtserkennung**

**Diplomarbeit Nr. 17.06**

Klasse 5A, Schuljahr 2019/20



Ausgeführt von: Aron Terzeta  
Rei Hoxha  
Egli Hasmegaj  
Jordi Zmiani

Projektbetreuer1: Matthias Maurer  
Projektbetreuer2: Dominik Stocklasser  
Projektbetreuer3: Andreas Kucher

Shkoder, 17. Januar 2020

# Eidesstattliche Erklärung

Wir versichern, dass wir die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt haben. Wir haben uns keiner anderen als der im beigefügten Quellenverzeichnis angegebenen Hilfsmittel bedient. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht.

Rei Hoxha

Ort, Datum  
Unterschrift  
Egli Hasmegaj

Ort, Datum  
Unterschrift  

Approbation Datum u. Unterschrift	PrüferIn	IT-Koordinator/Direktion
--------------------------------------	----------	--------------------------

Sämtliche in dieser Diplomarbeit verwendeten personenbezogenen Bezeichnungen sind geschlechtsneutral zu verstehen.

# Kurzfassung

Vom Auftraggeber wird ein System gefordert, das Gesichter erkennt, um einen kontrollierten Zugang in der Schule zu ermöglichen und die Sicherheit der Schule wird dadurch erhöht. Alle Gesichter sollen von den aktuellen Schülern und Lehrer erkannt werden. Es soll auch zwischen einer reellen Person und einem Foto den Unterschied berücksichtigt.

Wir sind für diese Idee entschieden, weil Sicherheit heute hoch interessant und relevant ist. Es geht hier um einen kontrollierten Zugang in Institutionen mittels Gesichtserkennung zu ermöglichen, da Gesichter ja eindeutig für jede Person sind. Die größten Herausforderungen und Voraussetzungen des Projekts befinden sich in dem Planungsprozess. Eine andere Voraussetzung ist das Gebrauch von zwei Kameras, damit der Unterschied zwischen einer reellen Person und einem Foto berücksichtigt wird.

# Abstract

This paper represents the face detection and recognition system that enables the detection of a human face and is able to identify it. It is thought to improve the security system of an institution while controlling the access of certain locations, rooms. Further it achieves the goal of differentiating between a real person and a photo being identified.

Ky punim paraqet sistemin e zbulimit dhe njohjes së fytyrës që mundëson zbulimin e një fytyre njerëzore dhe është në gjendje ta identifikojë atë. Mendohet se përmirëson sistemin e sigurisë së një institucionit ndërsa kontrollon hyrjen në disa lokacione, dhoma. Më tej ajo arrin qëllimin e diferencimit midis një personi të vërtetë dhe një fotografie që identifikohet.

Dieses Dokument stellt das Gesichtserkennungs- und -erkennungssystem dar, mit dem ein menschliches Gesicht erkannt und identifiziert werden kann. Es wird angenommen, dass es das Sicherheitssystem einer Institution verbessert und gleichzeitig den Zugang zu bestimmten Orten und Räumen kontrolliert. Ferner wird das Ziel erreicht, zwischen einer realen Person und einem identifizierten Foto zu unterscheiden.

# Danksagung

# Inhaltsverzeichnis

<b>1</b>	<b>Allgemeines</b>	<b>1</b>
<b>2</b>	<b>Planung</b>	<b>2</b>
2.1	Projektziele . . . . .	2
2.1.1	Ziele . . . . .	2
2.1.2	Nicht Ziele . . . . .	3
2.1.3	Optionale Ziele . . . . .	3
2.2	Projektplanung . . . . .	3
2.3	Projektmanagementmethode . . . . .	4
<b>3</b>	<b>Gesichtsregistrierung - Aron</b>	<b>6</b>
3.1	Umsetzung . . . . .	6
3.1.1	Allgemein . . . . .	6
3.1.2	Technische Lösung . . . . .	7
3.1.3	Herausforderungen . . . . .	12
3.1.4	Qualitätssicherung und Controlling . . . . .	14
3.2	Ergebnisse . . . . .	15
3.2.1	Implementierung . . . . .	15
<b>4</b>	<b>Umsetzung - Jordi</b>	<b>17</b>
4.1	Allgemeine Beschreibung . . . . .	17
4.2	Technische Lösungen . . . . .	20
4.2.1	Datenbank . . . . .	20
4.3	Herausforderungen . . . . .	23
4.3.1	Datenbank - MariaDB . . . . .	23
4.3.2	Python Scripting . . . . .	23
4.3.3	OpenCV . . . . .	24
4.4	Qualitätssicherung . . . . .	24
<b>5</b>	<b>Ergebnisse - Jordi</b>	<b>26</b>
5.1	Datenbank . . . . .	26

5.2 Tieferkennung . . . . .	26
<b>6 Bildverarbeitung - Egli</b>	<b>27</b>
6.1 Allgemeines . . . . .	27
6.1.1 Entwicklungsumgebung und Technologien . . . . .	27
6.1.2 Frameworks und Bibliotheken . . . . .	28
6.2 Technische Lösungen . . . . .	30
6.2.1 Lösungsweg - Structed Software design . . . . .	31
6.2.2 Face Detection . . . . .	32
6.2.3 Face Detect and crop . . . . .	35
6.2.4 Facial Landmarks Extraction . . . . .	36
6.3 Herausforderungen, Probleme, und wie wurden sie gelöst . . . . .	38
6.4 Qualitätssicherung, Controlling . . . . .	39
6.5 Ergebnise . . . . .	39
<b>7 Gesichtserkennung - Rei</b>	<b>41</b>
7.1 Allgemeines . . . . .	41
7.2 Technische Lösung . . . . .	42
7.2.1 Hardware und Aufbau . . . . .	42
7.2.2 Software . . . . .	45
7.3 Herausforderungen, Probleme, und wie wurden sie gelöst . . . . .	54
7.4 Projektmanagement und Controlling . . . . .	54
7.5 Ergebnisse . . . . .	56
<b>8 Planung vs Realisierung</b>	<b>57</b>
<b>9 Evaluierung und Resümee</b>	<b>60</b>
9.1 Wertschöpfung und Lessons Learned . . . . .	60

# Kapitel 1

## Allgemeines

Unten werden die Idee, das Thema und die Aufgabenstellung dieser Diplomarbeit verfasst.

Die Idee, ein System zu entwickeln dass Gesichter erkennt und registriert, ist daraus entstanden wegen folgenden Grunds. Es wurde vom Auftraggeber dieses System gefordert, um einen kontrollierten Zugang in der Schule zu ermöglichen. Es ist gedacht, die Sicherheit der Schule dadurch zu erhöhen und die Überwachung effizienter machen. Hauptziel ist es, alle Gesichter von den aktuellen Schülern und Lehrer zu registrieren und zu erkennen. Das System sollte auch den Unterschied zwischen einer reellen Person und einem Foto berücksichtigen. Es ist auch gefordert, dass die betreffende Person keine Maske, Brille oder Hüte bei der Gesichtserkennung trägt. Die Erkennung von Gesicht erfolgt auch nicht beim Bewegen von Person. Das Team besteht aus Aron Terzeta, Egli Hasmegaj, Rei Hoxha und Jordi Zmiani. Aufgaben sind wie folgend geteilt.

- Aron beschäftigt sich hauptsächlich mit der Gesichtsregistrierungsteil und Tiefeinschärfe des Bildes herauszuholen.
  - Egli kümmert sich um die wichtigsten Gesicht Daten zu extrahieren(Größe und Form der Augenhöhlen, Nase, Wangenknochen und Kiefer). Position/ Verhältnisse der Hauptmerkmale relativ zueinander herausholen. Aufbereitung der Daten für Abgleich.
  - Rei: User-Gesichtsdaten von Bildverarbeitung-Funktion holen, Vergleichen von Gesichtsdaten, System aufbauen.
  - Jordi: Datenbankdesign: Eine DB einrichten, Entwurf der Struktur der DB, DB in MySQL implementieren, Zugriffsberechtigungen festlegen, Error-checking.
- Wir sind dafür hoch motiviert, dieses Projekt richtig umzusetzen.

# Kapitel 2

## Planung

Dieses Kapitel beschreibt im Detail wie die Diplomarbeit gestaltet und abgegrenzt ist. Die Abgrenzung der Arbeit ist entscheidend wegen der hohen Komplexität des Projektes. Sie erfolgt durch Ziele, nicht-Ziele und optionale Ziele. Das ist im Unterkapitel 2.1 genau verfasst. Weiter folgt die Planung im Kapitel 2.3. Es werden hier das Lösungskonzept und die Projektmanagement erklärt. Es wird nun spezifiziert welche Projektmanagementmethode eingesetzt wurde.

### 2.1 Projektziele

Ziele, nicht Ziele und optionale Ziele

#### 2.1.1 Ziele

Ziele sind wesentlich für jedes Projekt. Deshalb wurden die Ziele dieses Projekts in drei Kategorien geteilt. In der ersten Kategorie gehören Ziele, die unbedingt erfüllt werden müssen. Anderfalls würde das Projekt scheitern.

1. Live vs. Foto unterscheiden. (3-dimensionale Erkennung an Gesicht machen. Tiefe messen damit zwischen einer Person und einem Foto differiert wird.)
2. Gesichts-Schlüsselpunkt-Extraktion, um ein Gesicht zu identifizieren.
3. Größe und Form der Augenhöhlen, Nase, Wangenknochen und Kiefer analysieren.
4. Position/Verhältnisse der Hauptmerkmale relativ zueinander herausholen.
5. Bilderdaten in Vektoren umwandeln mithilfe eines Algorithmus.
6. Abstimmung (Vergleichen mit den anderen Fotos in der Datenbank, um zu sehen, ob die Person schon registriert wurde).
7. Max. 500 Personen in einer Datenbank speichern.
8. 10 Tests, jeder Test in einer anderen Raumkondition, um alle Betriebskonditionen zu testen.

9. Datenbankdesign
10. Error checking
11. Safe Mode (eine Batterie, Back-ups in einem lokalen Server)
12. Min. Arbeitsvorbereitung (Min. Gesichtsdetektionszeit)
13. Admin account (Register-Rechte nur für Schüler und Lehrer eingeben)

### 2.1.2 Nicht Ziele

Hier sind die Nicht-Ziele definiert, damit das Projekt begrenzt ist und damit nichts gemacht wird, was nicht angefordert war.

1. Mehr als ein Gesicht gleichzeitig erkennen.
2. Maske, Brille, Hüte tragen.
3. Gesicht in Bewegung erkennen.
4. Person ins Profil oder andere Position sein.
5. Thermische Kamera einsetzen.

### 2.1.3 Optionale Ziele

Hier gehören Ziele, die optional sind. Das heißt sie sind nicht zwingend und wurden eingesetzt nur nachdem alle wichtigen und primären Ziele erfüllt sind.

1. Öffnung der Haustüren oder jeder anderen Tür mit Gesichtserkennung.
2. LCD-Display Implementation.
3. Integration in dem Infotainment-System.
4. Licht neben der Kamera (Night Vision implementieren damit die Erkennung/Registrierung auch dann funktioniert, wenn es dunkel ist.)

## 2.2 Projektplanung

Unsere Big Picture ist unser erstes grobes Design, das die Lösungsskizze des Projekts beschreibt. Es gibt bestimmte Gründe, warum Big Picture und Structed Design verwendet wurden, um die Software zu beschreiben. Diese Methode ermöglicht eine sehr gute Darstellung und Beschreibung des Lösungswegs. Ist schnell und leicht zu machbar. Alles ist klar sichtbar und nicht kompliziert. Big Picture und Structed Design folgt das Top-Down Prinzip, das heißt die Funktionen werden hierarchisch zerlegt (Jede Funktion wird in die folgenden Ebenen detaillierter beschreibt).

Structured Design und Big Picture haben keine Begrenzung. Dort können eindeutig alle Funktionen, Schnittstellen, Signalen und Daten beschrieben werden, sodass von allem leicht zu verstehen ist. Sehen Sie auf Abb. 2.1

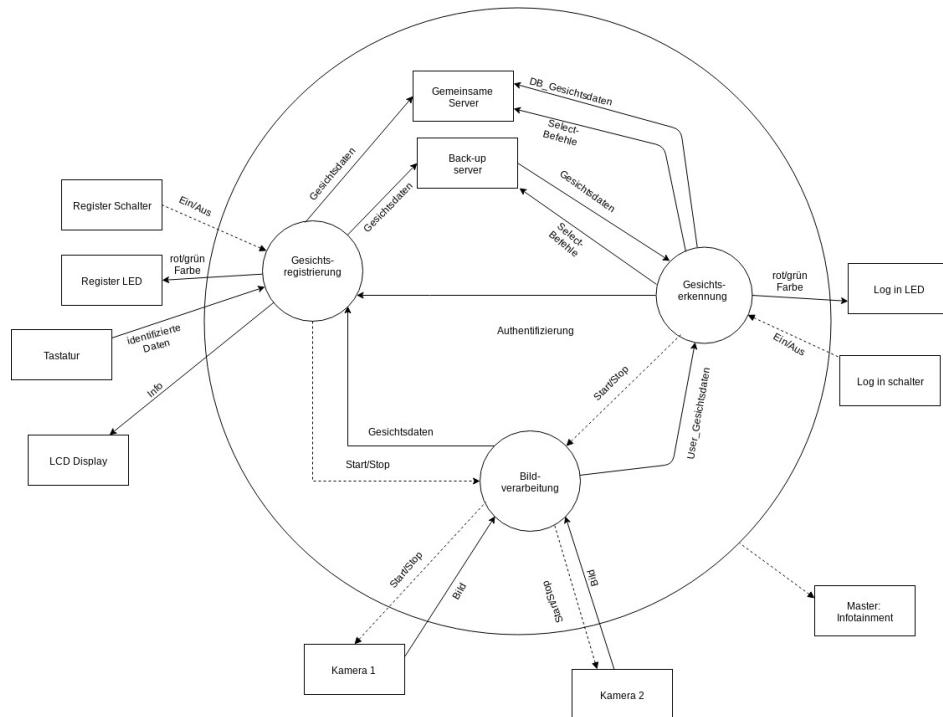


Abbildung 2.1: Big picture

## 2.3 Projektmanagementmethode

Als Projektplanmethode haben wir Scrum, eine agile Methode, gewählt, weil es die Möglichkeit bietet, komplexe Projekte mit einem kleinen Personenkreis zu verwalten. Scrum ist ideal für Software- bzw. Hardware-Entwicklungsteams, weil das Team während des Projekts verschiedene Änderungen an seinem Plan vornehmen muss. Aus diesem Grund ist es besser, tägliche Zielvorgaben zu haben und in einem kurzen Zeitraum von 1 bis 4 Wochen so genannte Sprints durchzuführen, bei denen das Ziel am Ende dieser Springs ein Prototyp ist. Verschiedene Prototypen herzustellen und am Ende den richtigen auszuwählen, ist die beste Wahl für die Projektmanagementmethode zur Gesichtserkennung. Es gibt auch tägliche Pläne, in denen sich das Team zusammensetzt und entscheidet, was die Ziele für den Tag sind und was sie tun müssen. Sehen Sie auf Abb. 2.2

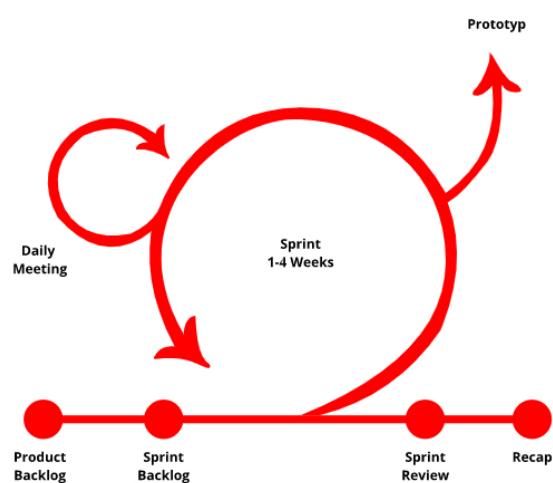


Abbildung 2.2: Scrum

# Kapitel 3

## Gesichtsregistrierung - Aron

### 3.1 Umsetzung

#### 3.1.1 Allgemein

Sicherheit ist heutzutage hoch interessant und relevant in mehreren technischen und nicht technischen Bereichen. Sicherheit ist eigentlich relativ, niemand weißt, ob er sicher ist oder nicht. Aber es gibt Systeme, bzw. Geräte, Personen usw., die Sicherheit garantieren. Das System, das entwickelt wird, hat mit Gesichtern der Personen zu tun. Alle wissen, dass das Gesicht für jede Person anders ist. Jede Person wird mit ihrem Gesicht identifiziert, weil es einzig ist. Das Gesicht hat Daten, die von verschiedenen Algorithmen herausgeholt werden können, und sie für Realisierung der Überprüfung und Identifizierung der Personen verwenden zu können.

Das System ist in zwei Teile geteilt. Es gibt den Registrierungsteil und den Erkennungsteil. Bei dem Registrierungsteil wird die komplette Registrierung der Schüler und Schülerinnen, der Lehrer und Lehrerinnen. Es werden verschiedene Pakete gebraucht werden wie z.B. python-MySQLdb, numpy, build-essential, cmake, git, libgtk2.0-dev pkg-config, libavcodec-dev, libavformat-dev, libswscale-dev. Das Paket „python-MySQL db“ ist eine Schnittstelle, das nur für Python gültig ist, und die Verbindung zwischen Python (Scripts, Programme) und dem Datenbank Management System ermöglicht. Dient als Verbinde zwischen Python und der Datenbank, die in MySQL integriert ist. Wir brauchen es, um Statements (Select,Insert,Update) in der Datenbank ausführen zu können.

Das andere Paket namens „git“ ist für das System nicht notwendig, aber git könnte als eine Backup-Strategie verwendet, wenn das System abstürzt. Git ist ein Versionsverwaltungssystem, das verschiedene Versionen bzw. Commits auf einem Github-Server speichert. Auf dem Github-Server gibt dann verschiedene Versionen des Systems und die Daten werden von einem bestimmten Commit dann zurückgeholt. Es wird meistens bei der Implementierung-Phase verwendet, um die Veränderungen der Source-Code, wann geändert hat, wer geändert hat, deutlich zu sehen. Cmake Paket ist ein Paket, das gebraucht wird, wenn das System mit OpenCV-Framework arbeitet wird. Wenn das OpenCV-Framework in C++ Skripten verwendet wird, dann wird cmake so geschrieben und das build-Verzeichnis so kompiliert, damit die C++ Skripten das OpenCV-

Framework verwenden können. Das gleiche passiert auch, wenn z.B. Python statt C++ verwendet wird. Die andere Pakete wie z.B. libgtk2.0-dev pkg-config, libavcodec-dev, libavformat-dev, libswscale-dev, sind nötige Paketen, damit das OpenCV-Framework eigentlich verwendet kann.

### 3.1.2 Technische Lösung

Technologien, die ich für die Implementierung verwendet habe sind:

- Linux als Betriebssystem [**Linux**]  
**Betriebssystem**] Linux ist das verwendeste Betriebssystem der Welt. Es ist eine open-source Software. Linux ist flexibel, man kann die einzelnen Modulen wegnnehmen, ohne dass das Betriebssystem abstürzt. Der Benutzer kann auch die Kernkomponenten wählen wie z.B. welches System-Grafiken angezeigt werden, bzw. die ganze Komponenten der Benutzeroberfläche. Warum ich Linux gewählt habe, gibt es verschiedene Gründe. Linux ist für eingebettete Systeme sehr geeignet. Es ist sicher gegen Schadprogrammen, Viren, Trojanern. Linux ist einfacher. Vorher war ein kompliziertes System, jetzt seit den Bemühungen der Ubuntu-Fondationen und der Ubuntu-Distribution ist jetzt sehr einfach verwendbar.
- python "Python ist eine Programmiersprache, die 1991 veröffentlicht wurde. Python besitzt eine einfache Lesbarkeit und eine eindeutige Syntax. Python lässt sich leicht erlernen und unter UNIX, Linux, Windows und Mac OS verwenden." [**python**] Warum ich python gewählt habe, gibt es verschiedenen Gründe. Python hat weniger Schlüsselwörter, reduziert den Syntax auf das Wesentliche und optimiert die Sprache. Ein Programm, das in python geschrieben ist, ist vom Betriebssystem unabhängig. Das bedeutet, sie können in allen Betriebssystemen interpretiert werden. Python hat auch eine gute Lesbarkeit.

Das System besteht aus verschiedenen Terminatoren. Der wichtigste Terminator ist der "Register-Schalter". Er initialisiert das ganze Programm. Schalter in Technik ist nichts anders, nur ein Konnektor oder mit anderen Wörtern, eine Brücke. Wenn diese Brücke geöffnet ist, dann bekommt das System keinen Input und gibt keinen Output zurück. Wenn der Schalter gedrückt wird, bekommt das System einen Input, transformiert und gibt dann einen Output. Das System ist sehr einfach verwendbar.

Das Register-LED dient als einen Anzeiger. Wenn mit dem System etwas schiefgeht, z.B. nicht richtige Inputdaten, dann wird mit einer bestimmten Farbe eingeleuchtet, mit rot. Wenn etwas passt, dann wird mit grün eingeleuchtet. Eigentlich das normale LED hat nur eine Farbe, aber es wird ein spezielles LED verwendet, namens RGB LED. RGB LED hat drei Grundfarben, rot, grün, blau, und mit diesen drei Farben kann man alle Farben erstellen. Es könnte auch zwei LEDs geben, rot und grün, aber es ist effektiver, ein RGB LED.

Eine spezielle Eigenschaft des Systems ist die Verwendung einer Tastatur. Es wird verwendet, weil die einzelnen Personen ihren Namen, bzw. Email schreiben werden. Die

andere spezielle Eigenschaft ist die Verwendung eines LCD-Screens. Da werden z.B. Errors gezeichnet, die Daten, die in Log gespeichert sind usw. Es ist leicht auch für den Benutzer zu sehen, dass z.B. ein Problem mit dem System hat, damit er nicht vor der Kamera 1 Stunde warten muss, damit er weißt, dass die Registrierungs-Phase nicht fertiggemacht wurde, usw. Eigentlich die Hauptfunktion des LCD-Screens ist, alles was der Benutzer mit der Tastatur schreibt, da gezeigt zu lassen. Warum es so geplant ist? Das Problem steht daran, wenn der Benutzer seine Email schreibt, dann kann er Fehler machen, weil er nicht sieht, was er schreibt. Und bei der Gesichtserkennung muss er noch einmal seine Email schreiben, aber werden nicht übereinstimmen, weil bei der Registrierung falsch getippt hat. Um es zu vermeiden, wird das LCD-Screen verwendet, damit der Benutzer sehen kann, was er schreibt.

Damit die Personen mit ihren Infos irgendwo zu speichern, wird einen Server gebraucht. In diesem Server läuft ein Datenbank Management System, in dem eine Datenbank erstellt ist. Die Datenbank ist so konfiguriert, damit die Person mit ihren Infos gespeichert werden können. Um die Verbindung zwischen System und Server zu ermöglichen, wird das Paket „python-mysqldb“ verwendet. Dies Paket ist vorher erklärt.

Anschließend gibt es ein Backup-Server. Die Daten werden parallel bei Server sowie bei Backup-Server gespeichert, damit die Daten noch gesichert sind, wenn der Server ein Problem hat. Die Verwendung des Backup-Servers ist zustande gekommen, weil das System 24/7 arbeiten muss, und wenn der Hauptserver Maintenance oder Probleme hat, der Backup-Server arbeiten kann. Auf dem Abb. 3.1 können Sie in einem technischen Weg besser sehen, wie der Gesichtsregistrierung-Teil arbeitet.

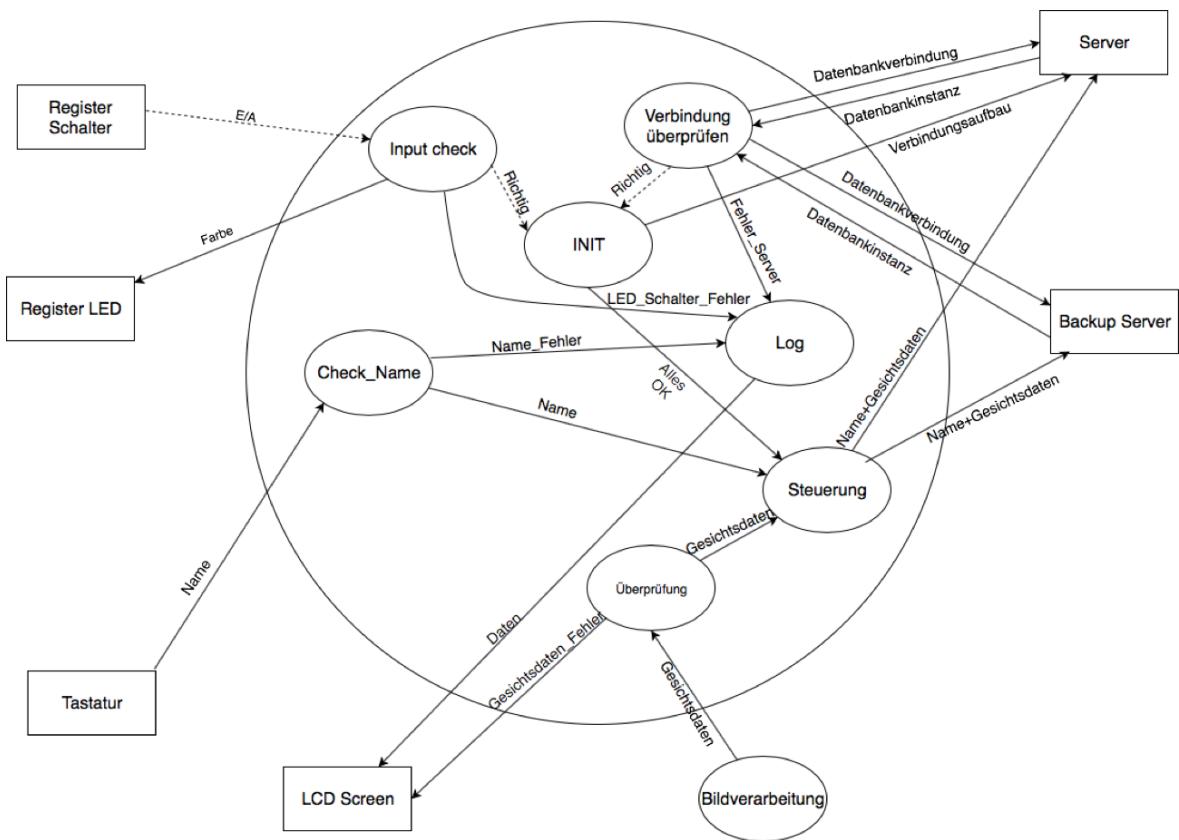


Abbildung 3.1: Structed Software Design bzw. erste Ebene

Um Schalter und LEDs im System verwenden zu können, brauchen wir ein spezielles Paket namens „RPi.GPIO“. Dieses Paket macht möglich, das Raspberry PI mit dem HW (LED und Schalter) verbinden zu können. Dafür werden GPIOs verwendet. Der Schalter hat 3 Beine. Ein wird mit 5V verbunden, das andere mit Ground und das andere ist für Daten. Dies dann wird mit einem GPIO-Port in Raspberry PI verbunden. Das gleiche ist auch für LED, damit wir es von Raspberry PI kontrollieren zu können, wird mit einem GPIO-Port verbunden. Jetzt mithilfe dieser GPIO-Ports bekommt das System zurück, wenn der Schalter gedrückt wird. Wert „1“ ist der Schalter gedrückt und werden dann die verschiedenen Skripten aufgerufen.

Schritte:

1. Am Beginn des Skripts diese Zeile schreiben: ”#!/usr/bin/python”. Es gibt zwei Gründe, warum diese Zeile geschreibt wird. Der erste Grund ist, dass dieses Programm mit einem Python-interpreter ausgeführt wird, der zweite ist, Verwendung des Programmsuchpfads, um es zu finden.

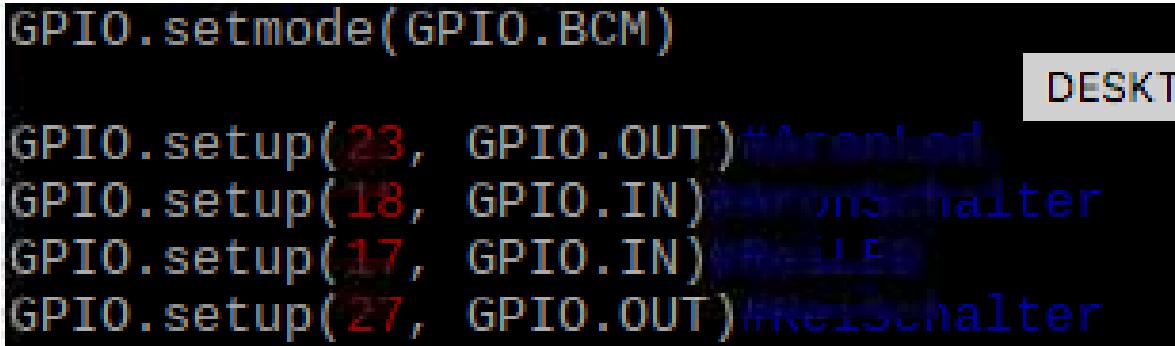
2. Alle Paketen importieren. Sehen Sie auf Abb. 3.2

```
import RPi.GPIO as GPIO
import time
import os
import subprocess
import sys
```

Abbildung 3.2: Packages zu importieren im Main-Skript

- RPi.GPIO ist ein Paket, das verwendet wird, um Zugriff auf die sogennanten GPIO-Ports zu haben. Vorher habe ich erwähnt, wenn wir Zugriff auf die HW-Komponenten haben wollen, die mit Raspberry PI verbunden sind, brauchen wir die GPIO-Ports. Jetzt, um diese GPIO-Ports in Python zu verwenden, brauchen wir das sogennante Paket "RPi.GPIO". Es gibt verschiedene Pakete, die einen Zugriff zwischen GPIO-Ports und python ermöglichen, wie z.B. rpi.gpio, GPIOZero usw. Ich habe rpi.gpio Paket verwendet, weil es leicht verständlich, programmierfreundlich und einfach zu verwenden ist. [rpigpio]
- time ist ein Paket in Python. Von diesem Paket habe ich nur die Funktion 'sleep' verwendet. Diese Funktion pausiert das python-Programm. [timepackage]
- os ist das wichtigste Paket in unserem Skript. Es erlaubt mir, dass ich in einem Python-Skript andere Skripten aufrufen kann. Ist egal, in welcher Programmiersprache diese Skripten geschrieben sind. Es gibt auch verschiedene Methoden, wie man verschiedene Skripten in einem Python-Skript aufrufen kann. Man macht mit dem subprocess Paket, eine Main-Funktion in dem Skript machen und hier die verschiedenen Funktionen des anderen Skripts aufrufen. Ich habe os Paket verwenden, weil es für unseres System am besten ist. [ospackage]
- subprocess Paket dient zur Verbindung zwischen verschiedenen Prozessen, in meinem Fall, ein Prozess heißt, ein Aufruf eines Skriptes, aber wird nicht im Skript verwendet.
- sys Paket verwende ich, um Console Parameter zu geben. Das bedeutet, wenn ich einen Skript aufrufe, z.B. login.py dann nach dem login.py gebe ich einen Parameter mit login.py <parameter >

3. GPIO-Ports direction einrichten. Direction für LED ist 'out', weil das LED als ein Output für unseres System dient. Direction für Schalter ist 'in', weil der Schalter als ein Input für unseres System dient. Auf Abb. 3.3 ist auch der Code in python.



```

GPIO.setmode(GPIO.BCM)

GPIO.setup(23, GPIO.OUT) # Admin
GPIO.setup(18, GPIO.IN) # Schalter
GPIO.setup(17, GPIO.IN) # Admin
GPIO.setup(27, GPIO.OUT) # Weischalter
    
```

Abbildung 3.3: GPIO-Ports Konfiguration

Es gibt verschiedene Betriebsarten für GPIO wie BCM und Board. Ich verwende BCM (Broadcom Pin Number), weil ich das Paket RPi.GPIO verwende. Mit diesem Paket darf ich nur die Betriebssart 'BCM'. [gpiomode] Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Admin da physisch sein muss. Die Überprüfung, ob der Admin da ist oder nicht, wird mit einem Vergleich von zwei Bildern gemacht. Ein Bild von Admin ist gespeichert, das andere wird gemacht, indem ich das Skript, das Bild macht, aufrufe, und dann vergleiche ich mit einem anderen Skript diese beiden Bilder. Es gibt 'matched' zurück, wenn die Gesichte bei den beiden Bildern gestimmt haben und 'not matched' wenn die Gesichte nicht gestimmt haben.

4. Dann kommt der Teil "Input check". Hier dann verwende ich die Methode 'input'. Die Methode befindet sich im Paket 'RPi.GPIO' und gibt entweder true oder false zurück. Im Verzeichnis '/sys/class/gpio/gpio<GPIO-PORt<' gibt es zwei Dateien, value and direction. Direction für die Port des Schalters ist IN und für die Port des LEDs ist OUT. Mit der Methode 'input' hole ich das Wert (value) der Schalter-Port. Wenn der Schalter gedrückt wird, das Wert wird '1' und 1 representiert 'true'. Das bedeutet, Input-methode liefert 'true' zurück und das Programm läuft weiter.

Nachdem der Schalter gedrückt wird, wird ein Skript aufgerufen. Dieses Skript dient zur Registrierung der Person in der Datenbank. Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Admin da physisch sein muss. Die Überprüfung, ob der Admin da ist oder nicht, wird mit einem Vergleich von zwei Bildern gemacht. Ein Bild von Admin ist gespeichert, das andere wird gemacht, indem ich das Skript, das Bild macht, aufrufe, und dann vergleiche ich mit einem anderen Skript diese beiden Bilder. Es gibt 'matched' zurück, wenn die Gesichte bei den beiden Bildern gestimmt haben und

'not matched' wenn die Gesichte nicht gestimmt haben. Wenn 'matched', dann können die Personen registrieren. Diese Person wird nach ihrem Vornamen, Nachnamen, Email und Rolle. Die Rolle schreibt der Admin. 1 ist für Admin, 2 für Schüler und 3 für Lehrer und Lehrerinnen. Die E-Mail speichere ich dann in einer Variable und diese Variable übergebe ich dann bei einem anderen Skript. Dieses Skript dann erstellt mit der Kamera eine Verbindung und macht ein Bild. Der Name des Bildes ist gleich mit der Email der Person. Es ist so gewählt, weil es leichter für das Einfügen der Daten in der Datenbank ist und bei der Speicherung des Paths des Bildes in der Datenbank mit dem gleichen Namen wie E-Mail einfacher ist.

Anschließend, wenn die Person in der Tabelle 'person' gespeichert ist, hole ich ID dieser Person und füge dann diese ID mit der E-Mail plus Path des Bildes in der Tabelle 'info'.

```
def insertPath (mycursor):
    mycursor . execute ( " select idP from person where email=%s ; " % ( variable )
    myresult=mycursor . fetchall ( )
    for x in myresult :
        var1=x [ 0 ]
        mycursor . execute ( " insert into info ( imagePath , idP ) values ('%s' , %s ) ; " % ( variable , var1 ) )
```

### 3.1.3 Herausforderungen

Eigentlich, das Projekt für mich hatte viele Herausforderungen. Die Gründe dafür sind, weil es ein ziemlich großes Projekt ist, wir haben neue Technologien verwendet, die wir vorher nie verwendet haben. Keine Erfahrung z.B. mit OpenCV, Python und viele verschiedene Dinge, die ich später erwähnen werde. Ich habe von diesen Herausforderungen und Problemen viel gelernt. Einerseits bin ich froh, andererseits bin ich sauer, weil das Datum des Ende des Projekts ist verzögert. Die Herausforderungen waren:

- opencv zu installieren. Das war eigentlich die größte Herausforderung. Es hat mir 3 Woche gedauert, bis ich es installiert habe.
- Beginn des Projektes. Immer der Beginn eines Projektes ist schwierig. Die Koordination im Team war sehr schwierig. Ich, als Projektleiter, musste allen sagen, wie sie arbeiten sollen, wo sie die Dateien finden können usw. Das war eine richtige große Herausforderung
- Git repository, Einrichtung von git. Manche von den Teammitgliedern wussten sehr wenig von git und ich musste sie erklären. Manche gab es merge conflicts, weil sie pull gemacht haben, ohne dass Sie die Änderungen committed haben. Ich sollte alle diese lösen, weil ich mehr Erfahrung mit git hatte.
- Python als Programmiersprache. Wir wollten vorher mit C++ es machen, aber es war sehr schwierig, OpenCV in Visual Studio zu installieren. Manche von uns wollten in Windows arbeiten und der einzige Weg war, mit Visual Studio zu arbeiten. Es hat nicht gegangen, deshalb sind wir in python umgewandelt.

Wir haben python gewählt, weil opencv in python sehr einfach installierbar war. Mit python hatten wir keine große Erfahrung. Das Maximum, was ich mit python gemacht habe ist, eine Verbindung mit der Datenbank und Statements da schicken (Insert,Select,Update,Delete). Alle andere Wissen sollte ich selbst von Bücher, Internet, Tutorials lernen. Die große Herausforderung hier war, die richtigen Quellen zu finden.

- Verwendung der Kamera und verbinden sie mit python. Ich wusste nicht, welche Funktionen man verwendet, um die Verbindung mit der Kamera zu erstellen.
- Bei der älteren Version von Raspbian heißt das Paket, das python mit Datenbank Management System(MySQL) verbindet, 'python-mysqldb' und jetzt heißt es 'python-mariadb'. Ich wusste das nicht und hat mir einbisschen Zeit genommen.
- Abhängigkeiten zwischen einzelnen Arbeitsteilen. Die Aufgaben sind so geteilt, dass sie Abhängigkeiten zwischen einander legen. Das hat dann zu einer Verspätung der Projektabgabe, weil jeder Teammitglieder aufeinander warten sollten.
- Was ich geplant habe, hat nicht gut funktioniert. Die großen Teile meiner Planung haben gepasst, nur wenige Kleinigkeiten musste ich ändern. Das Problem war, ich konnte nicht, dass diese Kleinigkeiten nicht passen. Sie sind erst in der Implementierungsphase angezeigt.

Ich habe diese Lösungen für die Herausforderungen vorgenommen:

1. Ich habe viel Tutorials geschaut, Websites gesehen, wie opencv in Raspberry PI installiert werden kann. Ich habe viele verschiedene Methoden probiert und mit keinem guten Ergebnis. Endlich nach vielen Proben ist es gegangen. Es ist installiert, und habe ich dann verschiedene Skripte in python gemacht, um es zu probieren. Manche von Skripten sind gegangen, manche nicht. Jetzt war eine kleine Herausforderung für mich gewesen, dass ich diese manche Skripte, die nicht ausgegangen sind, zu verbessern. Anschließend habe ich herausgefunden, dass das Problem bei dem Kompilieren von opencv war (cmake). Ich habe es noch einmal vom Beginn kompiliert. Jetzt ist alles in Ordnung, alle Skripte arbeiten, keinen Fehler mehr, der mit opencv Paket zu tun hat.
2. Eine Treffung mit meiner Gruppe vor dem Beginn des Projektes war notwendig. Ich hab sie gesagt und erklärt, in welchen Verzeichnise sollten sie arbeiten, die Struktur der Dokumentation, welcher Kommunikationskanal verwenden wir, um Probleme, Herausforderungen usw. zu besprechen usw. Jede Person hatte dann ihre Vorschläge, um das so und so zu lösen, und diese Treffung hat eigentlich zu viel gedauert, bis alle verstanden hatten, wie, wo,was, wann machen sollen. Aber auch nach der Treffung gab es zwischendurch Missverständisse bzw. Probleme mit der Kommunikation, nicht im richtigen Verzeichnis gearbeitet usw.

3. Ein Git-Repository erstellen und einzurichten war einfach. Ich hab es online in github.com erstellt. Einen Name eingegeben und dann als Collaborators die anderen Teammitglieder hinzugefügt. Strukturierter zu werden, habe ich dann verschiedene Branches eingelegt. Wie immer, gab es mit dem Befehl 'push' und 'pull' wieder Probleme. Das habe ich gelöst, in dem ich allen gesagt habe, dass, wenn sie in einem Github-Repository arbeiten möchten, dann bevor dem Beginn der Arbeit, müssen sie ein 'pull' machen, damit die Änderungen, die von anderen in dem Repository gemacht sind, mit deiner Version in Computer zu synchronisieren. Sie wissen nie, was die anderen in diesem Repository machen. Sie machen 'push', ohne zu sagen, dass sie ein 'push' gemacht haben. Das führt dann zu merge-Probleme usw.
4. Ich habe jedem Teammitglieder gesagt, er muss mindestens zwei Wochen an das Kennenlernen von python ausgeben. Tutorials anzusehen, Beispiele selbst zu probieren, die Quellen dafür selbst zu finden.
5. Für die Verbindung der Kamera mit OpenCV, gibt es einen Skript in der offiziellen Website-Dokumentation von OpenCV. Da habe ich alle Funktionen gesucht und gefunden, die ich brauchte, um die Kamera in Python verwenden zu können.
6. Damit wir die Abhängigkeiten zu vermeiden, habe ich gedacht, dass jeder Teammitglieder andere Aufgaben bekommt, als die, die in der Dokumentation stehen. Ich war gezwungen, diese Änderung zu machen, sonst würde das Projekt viel länger dauern.
7. Bei der Implementierung sind Kleinigkeiten herausgekommen, die bei der Planung nicht berücksichtigt waren. Die habe ich direkt in der Implementierung verbessert, ohne dass ich noch einmal die Planung mache. Aber ich habe diese Kleinigkeiten zur Kenntnis genommen, damit ich keinen solchen Fehler(Kleinigkeiten) mehr auf die Planungsphase machen werde.

### 3.1.4 Qualitätssicherung und Controlling

Ein Risiko ist meistens nur eine Einschätzung, was kostet einem Unternehmen, wenn die Projektziele nicht erreicht werden. Ich, als Projektleiter, sollte es machen. Eine Risikoanalyse zu planen ist sehr schwierig, weil es mit der Zukunft zutun hat. Zuerst muss ich an die Zukunft denken, welche Bauteile z.B. können zu Fehler kommen, welche Programme können schief gehen. Das bedeutet, ein Überblick in die Zukunft und einschätzen, was für Fehler und Risiken geben kann. Dann schätze ich die Wahrscheinlichkeit ihres Eintretens und am Ende die Maßnahmen. Da versteckt sich eine große Arbeit.**[Risikoanalyse]** Auf dem Abb. 3.4 können Sie die Risikoanalyse in Excel sehen. Es ist in Excel, weil es meistens mit Zahlen geht. Wahrscheinlichkeit, Kosten usw. alles sollen wir berechnen und Excel ist super an Berechnungen.

**Risikoanalyse: Gesichtsregistrierung und Gesichtserkennung**

Risikotyp	Nr.	Wahlsch.	Auswirk.	Ampel	Manager	Beschreibung	Behandlung und Kontrolle	Termin / Nächster Schritt
<b>Standardrisiken</b>								
Ressourcen	1	1	8	8	Rei Hoxha	Ausfall von Ressourcen	Ressourcen im Voraus sichern (Reserven an Mitarbeitern, an HW, an Zeit)	Projekt gleich abschließen
Planung	2	4	8	32	Aron Terzeta	Schlechte Planung (verschiedene Eintrittsfälle nicht berücksichtigt)	Nocheinmal mit der Planung beginnen	Den Projektantrag ablehnen
Technik	3	3	3	9	Egli Hasmeja	Nicht eindeutig Gesichtspunkte-Extraktion	Fertige Skripts aus dem Internet holen	Die Implementierung zu anderen Firmen zulassen
Staat	4	8	7	56	Aron Terzeta	Rechtliche Aspekte (Persönliche Informationen) + Datenschutz	Mit dem Staat vor der Implementierung sprechen	Geld zu Staat bezahlen, damit der Staat nicht in Schwierigkeiten uns bringt
Planung	5	2	4	8	Jordi Zmiani	Mehr Benutzer als geplant (Datenbankdesign)	ein skalierbares Design der Datenbank	nocheinmal Datenbank erstellen
Zeit	6	3	5	15	All	Spät mit Arbeit begonnen, z.B. die Installation von OpenCV sollte 4 Stunden aber ist 2 Tage	Schneller dann arbeiten oder vorher dem Kunden sagen, dass es ein bisschen spät das Produkt fertig ist.	Kunden sagen, dass entweder wartet bis zum Ende(eine große Verspätung) oder nimmt das nicht fertige Produkt zurück
Staat	7	4	0	0	Niemand	Wächter wird seinen Job verlieren, weil System viel Know-How braucht	-	-
Kommunikation	8	4	8	32	All	Wenn das Team keine gute Beziehung zwischen den Mitgliedern hat	Versuchen, einen gemeinsamen Weg und Sprache zu finden, wenn nicht, neue Teammitglieder	Team wechseln
Technik	9	3	3	9	Rei Hoxha	HW nicht genug, keine Know-How, wie man die speziellen Bauteile verwenden kann	Reserven, Bedienungsanleitungen lesen(auch in Internet suchen)	Experten fragen, Hilfe von Experten bekommen
Technik	10	5	5	25	All	Mangelnde Einfahrung	Tutorials sehen	Teile von Algorithmen und Skripts von Internet holen
Technik	11	3	3	9	Aron Terzeta	Nachträgliche Änderungswünsche des Systems	Vorher planen (Skalierbarkeit und Erweiterung)	Nocheinmal Planung
Technik	12	2	3	6	Aron Terzeta	Veränderung am kritischen Weg	Schnell den neuen kritischen Weg finden	Projekt ohne kritischen Weg(sehr gefährlich)
Technik	13	1	3	3	Aron Terzeta	fehlende Terminüberwachung	-	-
Zeit	14	2	2	4	All	Zeitprognose unterschätzen	Sagen, dass es eine Verspätung gibt. Der Kunde und der Chef muss es wissen	-
Zeit	15	3	4	12	Jordi Zmiani	Mangelnde Puffer in der Kalkulation	Mehr Stunde daran bis zum Ende arbeiten	Hilfe von außen bekommen
Ressourcen	16	1	1	1	Rei Hoxha	Mangelhafte Kontrolle der Projektkosten	Selbst dann den Mangel bezahlen	-
Ressourcen	17	2	4	8	Rei Hoxha	fehlende Ausrüstung	Direkt mit dem Projektleiter sprechen, und dann er entscheidet, ob es gekauft, ausgeliehen oder ... wird	-
Planung	18	0	1	0	Aron Terzeta	Geringe Personalkapazitäten	Entweder bleiben wir mit diesen Personalkapazität und das Produkt später fertig machen oder neue Personal einstellen, damit das Produkt in time fertig zu machen	-
Ressourcen	19	1	4	4	All	Ausfall einzelner Projektglieder	Ein anderer Projektglieder diese Arbeit machen	Einem neuen Team das Projekt einrichten

Abbildung 3.4: Risikoanalyse in Excel

## 3.2 Ergebnisse

Es sind 3 Monaten vergangen, seit ich angefangen habe zu arbeiten. Insgesamt habe ich 100 Stunden für die Diplomarbeit bis jetzt gearbeitet. Diese 100 Stunden haben dann ein Ergebnis gegeben. Weil mein Teil nicht viel Hardware hatte, nur ein LED, einen Schalter, Tastatur, war es nicht schwierig, die mit dem ganzem System zusammenzusetzen. Bis jetzt ist es gedacht, dass das System keine LCD Anzeige haben wird, weil ich nicht viel Zeit habe, um sie zu programmieren. Statt LCD Anzeige wird einen Bildschirm verwendet.

### 3.2.1 Implementierung

Nachdem eine große und gute Arbeit meinerseits, denke ich, dass das Produkt in der Inbetriebnahme-Phase sich befindet. Das bedeutet, bis jetzt gibt es einen Prototyp. Ich habe für diesen Prototyp die sogennante Kernfunktionen implementiert. Kernfunktionen sind Grundfunktionen bzw. wesentliche Funktionen. Ohne denen geht nichts. Die

Funktionen, die in diesem Prototyp implementiert sind, sind:

- Admin Account. Wenn eine Person registriert möchte, dann muss der Admin sich auf das System einloggen. Für diesen Prototyp gibt es nur ein Passwort, damit der Admin erkannt wird. Auf dem anderen Prototyp wird kein Passwort mehr geben, sondern die Überprüfung wird durch den Vergleich die Gesichter erfolgt.
- Die Person mit ihrem Vornamen, Nachnamen, Email und Role in der Datenbank speichern
- Ein Bild von dieser Person nehmen, das in der Datenbank speichern mit dem ID der Person.
- Wenn die Person erfolgreich in Datenbank gespeichert wird, wird das LED leuchtet.

# Kapitel 4

## Umsetzung - Jordi

### 4.1 Allgemeine Beschreibung

Ein wichtiger Teil jedes Projekt ist die Datenbank, und dass ist der Teil wo ich am meisten konzentriert bin. Andere Teilebereiche wo ich teilnehmen habe, sind bei der Extraktion der Gesichtspunkte, wo mein Job, den Tiefe von Bildern zu finden, damit ein 2D vs. 3D unterschied geben sollte, ist.

Technologie	Beschreibung	Lizenz
MySQL	Datenbankverwaltungssystem	Kostenlos
MariaDB	Datenbankverwaltungssystem	Kostenlos
Python	Objekte-orientierte Programmiersprache	Kostenlos
OpenCV	Bildverarbeitung Programmbibliothek	Kostenlos

Tabelle 4.1: Technologien

### MySQL

MySQL ist eine echte Multi-User, Multi-Treaded SQL Datenbank und wird von allen großen Providern oder auch Suchmaschinenbetreibern eingesetzt. MySQL ist eine CLi-



Abbildung 4.1: MySQL Logo  
[MySQLlogo]

ent/Server Implementierung, die aus einem Server-Dämon mysqld und vielen Client Programmen, sowie Bibliotheken für PERL, PHP/3, PHP/4 sowie ASP besteht. SQL ist eine standardisierte Datenbanksprache, die das Speichern, Updaten und den Zugriff auf Informationen erleichtert. Beispielsweise kann man Produktinformationen eines Kunden auf einem WWW-Server speichern und abrufen. MySQL ist äußerst schnell und flexibel genug, um sogar Bilder und Log-Dateien darin abzulegen. In der Praxis ist MySQL sehr viel schneller, als z.B. ORACLE oder INFORMIX.[[stepken1999mysql](#)]

MySQL hat ein breites Anwendungsspektrum und wird meistens in Verbindung mit PHP, Linux, Python usw. verwendet. Der Grund zu der Auswahl von MySQL steht bei der einfachen Bedienung und Verwaltung von Datenbank. Da es sich bei DBS um eines der am häufigsten verwendeten DBS handelt, gibt es eine Vielzahl von Tools, die zum Verwalten der Datenbank verwendet werden können.

## MariaDB

MariaDB ist aktuell die am schnellsten wachsende Open-Source-Datenbanklösung. Sie wird hauptsächlich von der MariaDB Corporation entwickelt und ist ein Fork von MySQL. Mittlerweile bietet das Datenbankverwaltungssystem mit seinen diversen kostenfreien Features vieles, was MySQL nicht oder nur kostenpflichtig zur Verfügung stellt (z.B. eine Speicher-Engine zur performanten Verarbeitung von riesigen Datenmengen; ein Datenbank-Proxy zur sicheren und hoch-verfügbarer Verwaltung skalierbarer Installationen u.v.m.). Im Gegensatz zu MySQL verfügt MariaDB jedoch nicht über einen eigenen Client wie die Workbench. Eine gute kostenfreie Alternative stellt HeidiSQL dar, jedoch verfügt diese über kein Dashboard, welches z.B. die Funktionsweise des Servers darstellt und damit Optimierungsentscheidungen erleichtert.[[MariaDB-Monitor](#)]



Abbildung 4.2: MariaDB Logo  
[[MariaDBlogo](#)]

Es ist hauptsächlich in Linux implementiert, da es das Standard-DBS ist. Es bietet eine einfache Verbindung mit Linux-Dateien wie Python, C ++ und vielen anderen Programmiersprachen. Da dies der Standard-Linux-DBS ist und unsere Software unter Linux mit Python-Skripten geschrieben wurde, ist MariaDB unsere Wahl für DBS.

## Python

Python ist eine Programmiersprache, die dank ihrer klaren Syntax und einfachen Lesbarkeit leicht zu erlernen ist und sich sehr vielseitig einsetzen lässt. Für die gängigen

Betriebssysteme ist Python frei verfügbar. Die üblichen Programmierparadigmen wie die objektorientierte oder funktionale Programmierung werden unterstützt.

Bei Python handelt es sich um eine Programmiersprache mit einer klaren Syntax und guten Lesbarkeit. Sie gilt als leicht zu erlernen und ist in den gängigen Betriebssystemen interpretierbar. Python unterstützt mehrere Paradigmen der Programmierung wie die funktionale, objektorientierte oder aspektorientierte Programmierung und ist auch als Skriptsprache nutzbar. [pythonInfo] Die Sprache weist ein offenes, gemein-



Abbildung 4.3: Python Logo  
[Pythonlogo]

schaftsbasiertes Entwicklungsmodell auf, das durch die gemeinnützige Python Software Foundation gestützt wird, die de facto die Definition der Sprache in der Referenzumsetzung CPython pflegt.

Der Grund, warum die Software in Python geschrieben ist, ist, dass Python eine einfache und schnelle Möglichkeit bietet, Datenbankskripte und Gesichtserkennungsprogramme zu erstellen.

## OpenCV

OpenCV (Open Source Computer Library) ist eine Open-Source-Bibliothek für Computer Vision und Machine-Learning-Software. OpenCV wurde entwickelt, um eine gemeinsame Infrastruktur für Computer Vision-Anwendungen bereitzustellen und die Nutzung der maschinellen Wahrnehmung in den kommerziellen Produkten zu beschleunigen. Als BSD-lizenziertes Produkt macht OpenCV es Unternehmen leicht, den Code zu nutzen und zu ändern. Es verfügt über C++, Python, Java und MATLAB-Schnittstellen und unterstützt Windows, Linux, Android und Mac OS. [OpenCV1]



Abbildung 4.4: OpenCV Logo  
[OpenCVlogo]

Obwohl es in C ++ geschrieben wurde, bietet es viele Schnittstellen zu anderen Programmiersprachen wie Python, die in diesem Projekt implementiert werden. OpenCV ist eine plattformübergreifende Bibliothek und da es viele Deep-Learning-Algorithmen und Frameworks enthält, musste es unsere Wahl für die Objekterkennung sein.

## 4.2 Technische Lösungen

### 4.2.1 Datenbank

Wie bereits erwähnt, haben wir uns für MySQL und MariaDB als DBS entschieden, wobei meistens das zweite mehr zum Einsatz kommt. Es ist wichtig zu beachten, dass die Datenbank nicht über die Befehlszeile oder die MySQL/MariaDB-Konsole implementiert wurde, sondern mithilfe von Python-Skripten. In diesem Fall würde jeder gegebene Befehl in den Python-Dateien gespeichert, und wenn etwas schief ging, könnten die Skripte einfach verbessert und ein Fehler behoben werden.

Um eine Datenbank zur Gesichtserkennung und -identifikation zu entwerfen, mussten einige Nachforschungen angestellt werden, bei denen es hauptsächlich darum ging, Beispiele für vorhandene Datenbanken zu finden. Auf diese Weise könnten wir uns eine Vorstellung davon machen, wie die Datenbanken aussehen. Obwohl die meisten Datenbanken nicht kostenlos waren und einige Unterlagen unterschrieben werden mussten, um die vollständigen Details zu erhalten, war dies mehr als ausreichend, um einen Ausgangspunkt zu haben. Einige der gefundenen Datenbanken sind:

- 3D Mask Attack Dataset
- The AR Face Database, The Ohio State University
- Caltech Faces
- CAS-PEAL Face Database
- The Color FERET Database, USA

Auf den ersten Blick scheint es nicht sehr schwierig zu sein, eine Gesichtserkennungsdatenbank zu erstellen, aber da sich das Projekt entwickelt, müssen viele Änderungen vorgenommen werden, da sie an die Gesichtsdaten der Person weitergegeben werden müssen.

Der nächste Schritt bestand darin, in Python nachzuforschen, wie die Schnittstelle mit MariaDB verbunden wird. Da es von vielen Linux-Benutzern empfohlen wurde, Python als beste Wahl für die Verbindung zu MariaDB zu wählen, wurde der Programmierer aufgefordert, den Code mithilfe von Methoden und try and except-Anweisungen zu schreiben, um einige saubere und perfekte Skripte zu erstellen. Auf diese Weise wäre der Code sehr einfach zu analysieren und zu verbessern, aber am wichtigsten wäre es, Fehlermeldungen zu beseitigen.

## Beschreibung der Tabellen

Nach der anfangs Planung wurden drei Tabellen definiert, die folgenden Tabellen:

- Person
- Info
- Log

### Tabelle Person

Die erste Tabelle, Person, enthält alle User mit ihren grundlegenden Informationen, wie Vorname, Nachname, Rolle, Email. Die Spalte Rolle ist damit geeignet, damit ein Unterschied zwischen Personen geben sollte, weil es zwei Gruppen, die Admins und des normalen Users. Die Admin bekommen die Zahl 1 bei der Rolle, während die anderen Users 0. Bei den Spalten, wo der Datentyp varchar verwendet wurde, gibt es ein Grund wiese varchar und nicht char. Das ist so denn char den ganzen Platz reserviert, wo anderseits der Datentyp varchar nur die Lange des gespeicherten Strings reserviert, obwohl die Spalte ein varchar mit der Lange 50 sein kann.

Die Spalte idP ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primär Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

Die Spalte Nachname enthält der Nachname der Person, deshalb wird als Datentyp Varchar mit der Lange 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist. bekommt.

Die Spalte Email enthält der Email der Person, deshalb wird als Datentyp Varchar mit der Lange 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist. bekommt.

Die Spalte Rolle enthält der Rolle der Person, deshalb wird als Datentyp Char mit der Lange 1 gespeichert, weil Char zum Speichern von Strings geeignet ist. Um eine einfache Nummer zu speichern, ist hier auch den Datentyp int verwendbar.

### Tabelle Info

Die zweite Tabelle enthält die Bilder der Personen, die während der Zeit gemacht wurden, und auch die Grund Information der Bilder, wie Punkten und der Path von dem Bild. Die Spalten der folgenden Tabellen sind:

Die Spalte idF ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primär Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

Die Spalte idP ein fremder Schlüssel, der als Primär Schlüssel für die Person Tabelle geeignet ist. Diese fremden Schlüssel sind dazu, damit eine Verbindung zwischen diese zwei Tabellen geben kann. Diese zwei Tabelle haben eine 1 zu mehreren Beziehungen, wo eine Person mehrere Bilder haben kann. Das wurde so geplant, damit je mehr Bilder eine Person hat, desto sicher, besser und funktionierbar der Vergleich sein kann.

Jeder Punkt besteht aus einer X und Y Koordinate, und von jedem Bild gibt es einen Startpunkt, wo die anderen Punkten relativ zu diesem Punkt gespeichert werden. Für diese Punkte wird den Datentyp dezimal werden, weil die Koordinaten kommastellen Zahlen sind. Nur für den ersten Punkt werden genau die Koordinaten gespeichert, für die anderen zeigt die X- und Y- Koordinate einen Vektor, das heißt es wird der Distanz von diesem Startpunkt aufgenommen.

Die Spalte imagePath speichert die Path der Bilder und wird mit dem Datentyp varchar mit einer Länge von 50 gespeichert, weil Varchar zum Speichern von Strings geeignet ist. Die Bilder wurden als Path und nicht als ganze Image gespeichert, weil ob diese Bilder in der Datenbank gespeichert werden, wurde die Datenbank überfüllt mit Daten. Alternativ was medium blob aber dieser Datentyp brauch zu viel Speicherplatz von unserer Datenbank. Es konnte eine Overflow von der Datenbank geben.

## Log Tabelle

Die dritte Tabelle ist nichts anders als eine Tabelle, wo alle Änderungen in der Datenbank protokolliert werden. In diesen Tabellen sollten die Grund Informationen gespeichert werden müssen, wie z.B wer hat was, wann gemacht. Diese frage muss die Tabelle beantwortet.

Die Spalte idL ist eine eindeutige Spalte, die es zu der Identifizierung von der Datenzeile gilt, deshalb wird auch als Primär Schlüssel definiert. Es wird auch auto increment festgelegt, damit die nächste Zeile automatisch die nächste Zahl bekommt.

Die Spaltet User soll der Person speichert, wer die Änderungen durchgeführt hat. Hier wird der Datenbank User gespeichert, weil um etwas zu verändern soll die Person zuerst in der Datenbank einloggen. Die Spalte hat den Datentyp Varchar mit der Lange 50, weil die Funktion Current User, da eingetragen wird.

Die Spalte Date als der Name zeigt, kümmert um die Speicherung von der Zeit, wann eine SQL Statement ausgeführt wird. Die Spalte hat den Datentyp datetime, und da wird die Funktion now(), die die aktuelle Zeit darstellt, gespeichert.

Die Spalte Info, informiert den User, was für eine Änderung gemacht wurde, z.B neue User wurde eingelegt. Diese Spalte bekommt den Datentyp Varchar, weil Varchar zum Speichern von Strings geeignet ist. Die Werten in dieser Spalte werden durch Triggers definiert, also wenn etwas in der Datenbank passiert, kümmern die Triggers um die eintragen von Daten.

## 4.3 Herausforderungen

Jedes Projekt hat seine Herausforderungen, und wenn etwas schief geht sollten Alternativen gefunden werden. Auch in diesem Projekt gab es solche Probleme, die durch Hilfe von Tutorials und Internet Blogs gelöst werden.

### 4.3.1 Datenbank - MariaDB

Die meisten Problemen sind bei der Datenbank getroffen, das ist so passiert wegen der kleinen Unterschiede zwischen MySQL und MariaDB. Manchmal wurde mit dem Programmierregeln von MySQL, obwohl MariaDB in Verwendung war. Zwischen diese zwei Datenbankverwaltungssystem gibt es kleine Unterschiede, die zu Problemen führen. Den getroffenen Fehlern werden hinunter besprochen:

*ERROR 1452: Cannot add or update a child row: a foreign key constraint fails.*  
Das Problem hier war es, dass es keine Tabelle geändert werden konnte, wegen die Beziehungen von Tabellen. In Prinzip sollte zuerst das Kind Tabelle gelöscht werden, und dann die Eltern Tabelle, weil sonst wird eine Konflikt geben, wozu bei der Kind Tabelle einen Fremd Schlüssel definiert ist, der jetzt nicht mehr existiert.

*You have an error in your SQL syntax; check the manual that corresponds to your mariadb server version for the right syntax to use near .. At line 1*

Dieser Fehler wurde getroffen, wenn einen Trigger Erstellung Script ausgeführt wurde. Der Fehler hat hier beim Delimiter gestanden, weil wenn ein Trigger durch ein Script erstellt wird, braucht man keine Delimiter einsetzen, sondern nur wenn es manuell durch der Command Line sollte der Delimiter festgelegt. Hier kann man sehen, die Änderungen zwischen MariaDB und MySQL, in MySQL sollte dieser Code ohne Probleme ausgeführt werden.

*Mysql-Server is missing//* Dieser Fehler zeigte, dass keine MySQL Server auf dem Betriebssystem installiert war. Dieses Problem ist getroffen, wegen die Änderungen in Linux System. MySQL Server wurde nicht mehr supportiert, und jetzt war im Betrieb MariaDB, und statt mysql-server zu installieren, sollte mariadb-server durch den Befehl *sudo apt-get install mariadb-server* installiert.

*ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2 "No such file or directory")*

Dass war nur ein kleiner Fehler, weil wenn man in Kali Linux programmiert und gearbeitet hat, musste den MariaDB Server immer noch einmal durch den Befehl *service mariadb restart* gestartet werden. Im Vergleich zu Debian Betriebssystem, die auf RaspberryPi installiert war, musst den Server nicht gestartet werden, weil es automatisch im Betrieb war.

### 4.3.2 Python Scripting

Python ist beliebige Programmiersprachen von vielen Users, aber Python unterschiedet sich von anderen Programmiersprachen durch den Code Struktur. Die meisten Fehlern

wurde bei der Übertragung von den Parametern zwischen den verschiedenen Funktionen.

*TypeError: not all arguments converted during string formatting python*

Der folgende Code nimmt als Parameter die Daten von dem neuen User, damit der User in der Datenbank registriert wird. Aber wenn man nur die Methode zu probieren möchte, und gibt als Parameter nur der Name, als eine einfache Registrierung, dann wurde dieser Fehler getroffen.

```
insertvalues1(curs,'jordi','zmiani','jorzmi14@htl-shkoder.com',1)
```

Bei der ersten Möglichkeit sollten zwei Parameter gegeben, denn nur bei einem String Parameter wurde dieser Fehler generiert. Bei der zweiten Möglichkeit musst diese Parameter geparsst werden durch die Funktion format().

*TypeError: 'str' object is not callable (Python)*

Str in Python ist eine spezielle Methode, und diese Methode sollte nicht überschrieben werden, das heißt, dass es sollte keine Variable bzw. Funktion mit diesem Namen erstellt werden. Aber trotzdem wurde dieser Fehler generiert, weil wenn ein Parameter auf den falschen Datentyp geparsst wird, dann wird dieses Problem dargestellt.

### 4.3.3 OpenCV

OpenCV Bibliotheken können manchmal schwierig zu installiert werden, weil es so viele Möglichkeit zum Einrichten von OpenCV gibt, die aber von dem Betriebssystem abhängen. Am meisten Fallen wenn die Installation Fehler produziert, musste OpenCV von Beginn noch einmal eingerichtet werden.

*Unable to locate package libjasper-dev*

Diese Bibliotheken ermöglichen ein paar Funktionen von den OpenCV, das ist eine von vielen Bibliotheken, die eingerichtet werden sollten. In diesem Fall konnte diese Bibliothek nicht installiert werden, weil es die Repository geändert hat, und deshalb musste die Bibliotheken von einer älteren Version heruntergeladen werden. Mit den folgenden Befehlen wurden diese Bibliotheken installiert.

```
echo "deb http://us.archive.ubuntu.com/ubuntu/ yakkety universe sudo tee -a  
/etc/apt/sources.list  
sudo apt update  
sudo apt install libjasper1 libjasper-dev
```

## 4.4 Qualitätssicherung

Einer der wohl bekanntesten Berater/-innen, Lehrer/-innen und Autoren/Autorinnen (über 200 Veröffentlichungen) zum Thema Qualität ist der Amerikaner W. Edwards Deming. Deming entwickelte auf der Basis der allgemeinen Problemlösungsmethode den sogenannten Plan-Do-CheckAct-Zyklus (PDCA-Zyklus). Auf der Plan-Phase werden

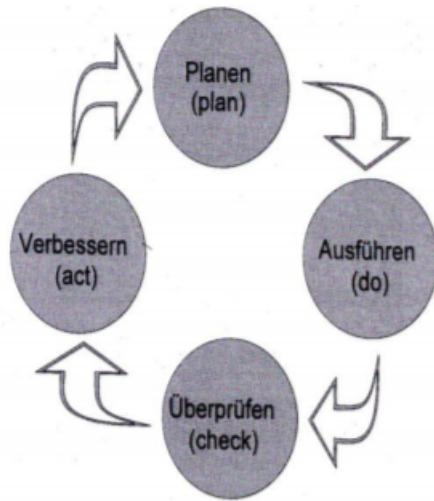


Abbildung 4.5: Plan-Do-Check-Act-Zyklus  
**[PDCA-Zyklus]**

Probleme betrachtet und Lösungsmaßnahmen erarbeitet. Die Do-Phase ist die Phase der Umsetzung bzw. Ausführung der zuvor gefundenen Lösungen. In der Check-Phase wird bewertet, ob die Maßnahmen zum Erfolg geführt haben. Innerhalb der nachfolgenden Act-Phase findet eine Standardisierung der erfolgreichen Maßnahmen statt, die fortan als Basis für weitere Verbesserungen dient.**[PDCA-Zyklus]**

Auf dem Bild 4.5 wird die PDCA Zyklus dargestellt, und es ist ganz sichtbar dass die Qualitätsmethode sehr gut für die Verbesserung von Software und Programmen ist. Deshalb wurde auch diese Methode gewählt, weil durch Error und Checking konnten die Fehler gefunden, analysiert und verbessert. Das passt wirklich gut beim Team, wenn sie keine Erfahrungen mit bestimmenden Programmiersprachen bzw. Softwareprogrammen haben.

Oben bei den Herausforderungen war klar, dass wenn Probleme getroffen wurden, wurden sie durch Hilfe von Tutorials bzw. Net Blogs gelöst. Durch verschiedene Zyklen beachtet man auch bei den Risiken, weil wenn etwas nicht funktionier, wird schnell das Problem gefunden und später wird eine andere Lösungswegen bestimmt.

# Kapitel 5

## Ergebnisse - Jordi

### 5.1 Datenbank

Für den Bereich Datenbank wurde ordentlich jede Woche programmiert, wo seit dem Anfang wurde es Recherche über verschiedenen Arten von Gesichtserkennung Datenbank, und später hat die Planung und Design von Datenbankentwurf begonnen. Es gab auch viele Änderungen abhängig von anderen Teilbereichen des Projektes. Die Datenbank musst immer geändert werden, und nach den Gesichtsdaten anpassen, damit alles in Ordnung ist. Die folgenden Punkten der Datenbank wurden realisiert:

- Datenbankdesign
- Erstellen von Tabellen
- Log Table
- Triggers
- Zugriffsrechte

Die Datenbank sollte fast fertig sein, nur kleine Anderungen konnten durchgeföhrt werden, wie fur die Backup von Daten.

### 5.2 Tieferkennung

Für den anderen Bereich Tieferkennung wurde gearbeitet aber nicht mit einem hohen Niveau wegen der Probleme beim Gesicht Erkennung, weil Tieferkennung ein letztes Ziel von dem Projekt ist. Hier wurde natürlich Recherche gemacht, und ein kleines stuck Programm geschrieben, nur zum Ausprobieren von Bibliothek und OpenCV.

# Kapitel 6

## Bildverarbeitung - Egli

### 6.1 Allgemeines

Diese Diplomarbeit besteht aus vielen unterschiedlichen Modulen, die um bestimmte Ziele bzw. Aufgaben zu lösen gut aufgeteilt sind.

Sehr wichtiger Teil dieses Projekts ist die Bildverarbeitungsteil.

In der folgenden Abschnitt der Ausarbeitung wird es genau erklärt und beschrieben wie diese Aufgabe gelöst wurde und was dafür verwendet war.

Für die Umsetzung wurden folgende Technologien gebraucht.

#### 6.1.1 Entwicklungsumgebung und Technologien

Die gewählte Entwicklungsumgebung war grundsätzlich eine virtuelle Maschine die Linux auf einem Windows System geboten hat.

Linux im Gegensatz von Windows ermöglicht volle Kontrolle über Updates und Upgrades und dadurch könnten komplexe Aufgaben einfacher erledigt werden.

Alles wird leicht und bequem durch Konsole eingegeben.

So wurden die Entwicklung, das Testen von den genutzten Algorithmen und anderen Technologien vielmals erleichtert. [[linx](#)]

Zusätzlich ist Raspberry Pi als Backup System verwendet worden das auch mit einem lauffähigen Linux Betriebssystem (Debian) funktionierte.

Raspberry pi ist zwar klein, aber funktioniert ganz gut wie ein normaler Rechner und ist kostengünstig. Für technische/elektronische Projekte wie das betreffende, ist es perfekt geeignet.

Für die Implementierung der Code wurde hauptsächlich die Programmiersprache Python verwendet.

Python ist eine allgemeine Programmiersprache auf hohem Niveau. Dies bedeutet dass es näher an menschlichen Sprachen und weiter von Maschinensprachen ist.

Also ist ein in Python geschriebener Code sehr leicht von einem Mensch zu lesen, zu verwalten und zu warten.

Es bietet zahlreiche Modulen durch seine große und robuste Standardbibliothek an, von denen man ruhig, abhängig vom Bedarf, auswählen kann. Sehr leicht kann es in der Dokumentation der Python-Standardbibliothek nachgesehen werden um sich besser mit den gezielten Funktionalitäten auszukennen.

#### [why python]

”Git ist ein freies und Open Source verteiltes Versionskontrollsysteem, das entwickelt wurde, um alles von kleinen bis zu sehr großen Projekten mit Geschwindigkeit und Effizienz abzuwickeln.” [Git1]

Die Versionierung der ganzen Software Änderungen erfolgte durch Git.

Es hat sich nützlich erweist indem die Arbeit dadurch zwischen die Projektmitgliedern sehr gut koordiniert und verwaltet wurde.

### 6.1.2 Frameworks und Bibliotheken

Schlüsselwort von unserem Projekt war die Framework bzw. die Bibliothek „OpenCV“.

OpenCV ist eine open-source Bibliothek für Computer Vision. Also, ganz allgemein kann sie als eine Bibliothek für Bildverarbeitung betrachtet werden.

Sie kümmert sich unter anderem um die Manipulation von Bildern, die Analyse von denen und um die daraus bestimmte Muster bzw. Objekte, für verschiedenen Zwecken einzusetzen. Ganz berühmte Anwendungsgebieten sind Gesichtserkennung und Stereo Vision.

Stereo Vision bedeutet die Extrahierung von Informationen aus einem Bild in einer 3-dimensionalen Ebene.

OpenCV wurde unter anderen Bibliotheken aufgrund seiner vielen guten Eigenschaften und seiner Flexibilität ausgewählt. Es umfasst hunderte von Computer-Vision-Algorithmen und besteht aus strukturierten Einheiten bzw. Module die als feststehende Bibliotheken implementiert sind.

Es ist Cross-Platform, in C/C++ geschrieben und unterstützt auch Python.

#### [opencv library]

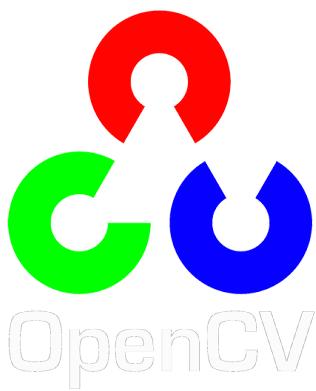


Abbildung 6.1: OpenCV logo

#### [OpencvLogo]

SciPy hat sich deswegen von Nutzen erweist, um gewisse mathematische Angaben zu lösen und bei zusätzliche Installationen von Bibliotheken zu helfen.

SSciPy ist eine kostenlose und Open-Source-Python-Bibliothek für wissenschaftliches und technisches Rechnen.”

[[2019arXiv190710121V-scipy](#)]

Dies sind einige der Kernpakete von SciPy die nutzbar für das Projekt waren:

NumPy legt die Basis für das wissenschaftliche Rechnen mit Python.

Es unterstützt große mehrdimensionale Arrays und Matrizen. Es enthält viele Mathematische Funktionen auf hoher Ebene um die Arrays zu bearbeiten.

NumPy kann also als leistungsfähiger mehrdimensionaler Behälter für generische Daten verwendet werden.

Es können beliebige Datentypen definiert werden.

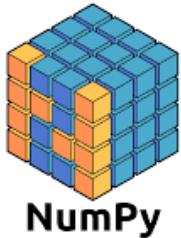


Abbildung 6.2: Numpy logo

[[Numpy](#)]

Dlib ist ein so genanntes Toolkit, die Algorithmen für Machine Learning und Tools zum Erstellen komplexer Software zur Lösung von der realen Welt getauchte Probleme, beinhaltet.

Es wird überall eingesetzt, in Robotik, Mobilgeräte und unter anderem in Computer Vision auch.[[dlib](#)]

Es wurde prinzipiell um bei der Extrahierung der so genannten „Facial Landmarks“ gebraucht.



Abbildung 6.3: dlib logo

[[dlib](#)]

## 6.2 Technische Lösungen

Bei der Gesichtsregistrierung und Gesichtserkennung Diplomarbeit war es die schwierigste und herausforderndste Aufgabe, sie sorgfältig zu planen, um die Effektivität der Arbeit zu steigen und das Endprodukt zufriedenstellend zu machen.

Wichtig war es die Ziele richtig zu setzen und sie gut abzugrenzen damit es in keine Lücken bei der Implementierung führte.

Aus diesem Grund musste die Arbeitsteilung gut geregelt werden, damit jedes Teammitglied an einem bestimmten Modul der Arbeit konzentrieren konnte.  
Es wurde auch das berücksichtigt, dass jedes Teammitglied das machte was ihm/ihr am besten gefällt und was ihm/ihr am leichtesten fällt.

Die Planung der betreffende Bereich der Projektarbeit hat durch die Methode „Structured Design“ erfolgt.

Structured Design ist eine Erweiterung von der „Big Picture“, die dazu dient, ein technisches System mit ihren Schnittstellen mit außen grob zu beschreiben.  
Es ist in verschiedenen Ebenen unterteilt, von außen beginnend.

Unten wird die erste Ebene der Structured Design von der Bildverarbeitungsteil dargestellt.

### 6.2.1 Lösungsweg - Structed Software design

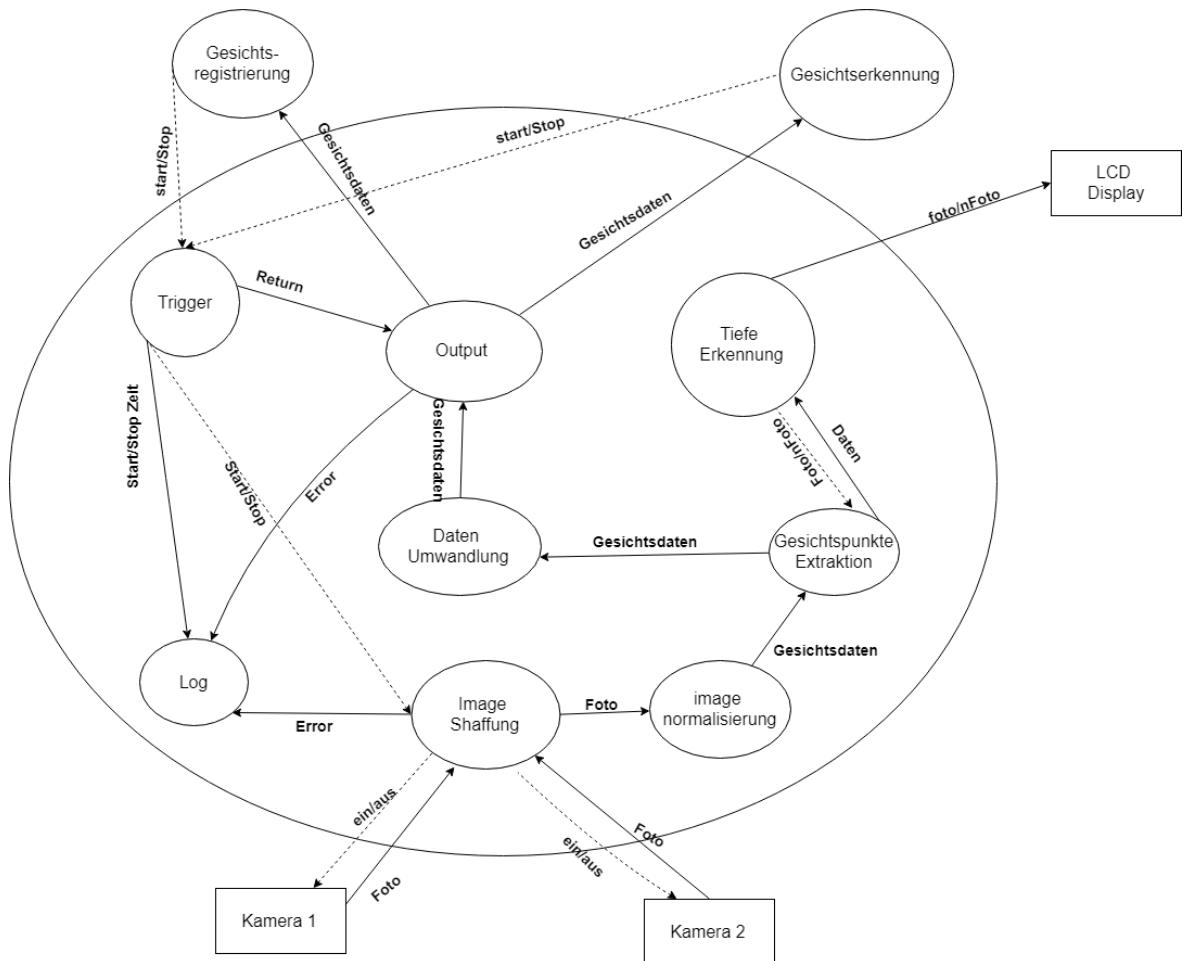


Abbildung 6.4: Bildverarbeitung Structured Design

Wie es in der Abb.6.4 sichtbar ist, ist die Bildverarbeitungsteil in diese Einheiten unterteilt:

1. Image Schaffung
  2. Image Normalisierung
  3. Gesichtsschlüsselpunkten Extrahierung
  4. Log
  5. Output
  6. Trigger

Die Schnittstellen von außen sind die Gesichtserkennungsmodul und die Gesichtsregistrierungsmodul.

Diese schicken eine Anforderung an den Trigger der dann die Image Schaffung Modul initialisiert. Unabhängig von welchen von diesen beiden der Bedarf hatte, macht die Image Schaffung aus der zwei Kameras ein Foto.

Implementierungsnahe wurde bis jetzt nur eine Kamera verwendet, da die Stereo Vision noch nicht funktional ist.

Nachdem das Foto gemacht wird folgt die Image Normalisierung bzw. die „Image Processing“.

Sei ein Bild ist zu klein wird die Größe angepasst, seien die Gesichter rotiert werden sie so gemacht dass sie gerade rotiert. Affines und perspektivisches Warping wird falls notwendig geführt. Farben werden nach Bedarf auch angepasst usw.

Danach erfolgt die Erkennung von Gesichtern, die Ausschneidung von denen und die Extrahierung der Gesichtsschlüsselpunkte.

Sie werden dann in Vektoren umgewandelt und zur Abgleich oder Registrierung je nach welche Signal der Trigger bekommt, geschickt.

Hinsicht: Bei der Umsetzung wurden bestimmte Bereiche mit einander verknüpft, was zu Folgendes führte: was bei der Planung steht, stimmt nicht völlig mit der Methoden zur Umsetzung überein. Weitere Unterschiede werden im Punkt 13b. genauer beschrieben.

### 6.2.2 Face Detection

Das erste Ereignis, dass erfüllt werden muss bei der Gesichtserkennung ist das Finden von Gesichtern, also die Feststellung, wo die Gesichter in das Bild befinden.

Dafür ist hier die „Haar Cascade Classifiers“ pre-trained Classifier gebraucht worden.

Gesichtsdetektion durch „Haar“ Merkmale ist eine sehr effektive Methode die von Paul Viola und Michael Jones entwickelt wurde indem sie Merkmale gruppiert haben und die Erkennung von diesen dadurch schneller gemacht.

Die Arbeit heißt „Rapid Object Detection using a Boosted Cascade of Simple Features“.

Es geht hier um Maschinelles Lernen, wie diese Kaskadenfunktionen durch viele negative(Bilder ohne Gesichter) und positive(Bilder mit Gesichter) Bilder trainiert wurden um Objekte zu erkennen.

In diesem Fall arbeiten wir selbstverständlich mit Gesicht Objekten. Dafür wurden die Haar Merkmale benutzt. Auf Abbildung 2 sind sie dargestellt. Jedes Haar Merkmal ist nichts anders als ein Wert, durch das Subtrahieren der Summe der Pixel unter dem weißen Rechteck von der Summe der Pixel unter dem schwarzen Rechteck, erhalten. Diese Summen berechnet man durch integrale Bilder, die zur Vereinfachung von den

Summenberechnungen dient.

[Viola01robustreal-time]

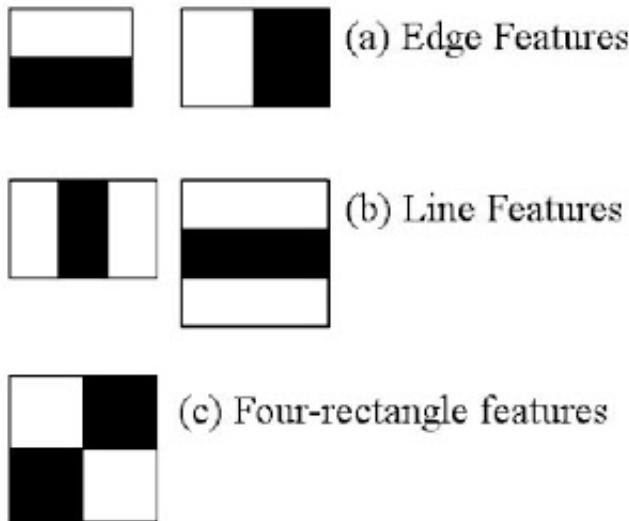


Abbildung 6.5: Haar Features

[Viola01robustreal-time]

Man muss aufpassen, da nicht alle Merkmale vom Nutzen sein können. Man kann es zum Beispiel deutlich auf Abbildung 3 sehen, wie die Nase viel heller als die Region von der Augen ist. Die Selektierung von den besten Merkmalen wird durch das Adaboost Algorithmus berechnet.

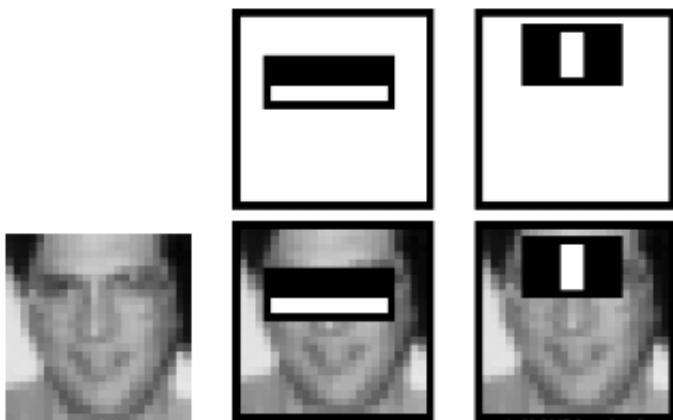


Abbildung 6.6: Haar Einsetzung

[Viola01robustreal-time]

Mit dieser Absicht setzen wir alle Funktionen auf alle Trainingsbilder ein. Für jedes Merkmal wird der beste Schwellenwert ermittelt, der die Gesichter in positive und

negative klassifiziert.

## Umsetzung in Code

Durch die Verwendung von classifiers die OpenCV bietet, setzen wir das um.

```
//haar cascade classifier laden
1         face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_def

2         gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

3         faces = face_cascade.detectMultiScale(gray, 1.1, 4)

4         for(x, y, w, h) in faces:
//parameter: image, startpunkt, endpoint(w..width, h..height), farbe, D
5             cv2.rectangle(image, (x, y), (x+w, y+h), (255,0,0), 2)

//opencv und numpy importieren
//Bild als input einfügen
//Bild einlesen
//Bild in „gray scale“ umwandeln wodurch die Rechenleistung reduziert wird. //(Da
es keine Farben zuhanden sind).
//Parametern: src(source image), dst(destination image), code(conversion code)
//Hier benutzt man die Cascade Classifier um Gesichter zu finden.
//Parameter:
//image, objects, scaleFactor = 1.1, minNeighbors = 3, flags = 0, minSize = new
cv.Size(0, 0), maxSize = new cv.Size(0, 0))
//Zeichenrahmen für das Gesicht zeichnen.
//Bild zeigen durch ‘image’ Fenster
//Press any key zum schließen vom Fenster
```

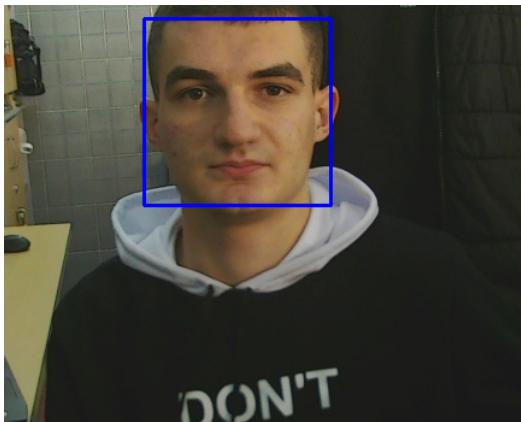


Abbildung 6.7: Box on face: Aron

Die Funktionalität des Code ist genau auf Abb.6.7 ersichtlich.

### 6.2.3 Face Detect and crop

Nachdem Gesichter gefunden werden, braucht man daraus eigene Bilder machen, die zur Erleichterung von der Extrahierung der Schlüsselpunkte helfen.

nrFace bestimmt die Anzahl von Gesichter von der cascade classifier gefunden. Nur wenn es größer als Null ist soll die Logik von der Crop Funktion weiterlaufen.

```

if nrFace > 0:
    for face in faces:
        for(x, y, w, h) in faces:
            1         r = max(w, h) /2
            2         centerx = x + w /2
            3         centery = y + h /2
            4         nx = int(centerx - r)
            5         ny = int(centery - r)
            6         nr = int(r * 2)
            7         faceimg = image[ny:ny+nr , nx:nx+nr]
    filenam = input(" Give new filename for cropped photo: \n")
    image2 = cv2.imwrite(filenam ,faceimg)
    elif nrFace <= 0:
        print(" no faces found")

```

Zeilen 1-6 berechnen das Zentrum von Bild neu und in Zeile 7 wird ein neues Image mit den berechneten Parametern angelegt.

Mit imwrite wird das Image gespeichert. Auf Abb.6.8 sieht man deutlich nur das Gesicht von Aron.



Abbildung 6.8: Aron Gesicht

#### 6.2.4 Facial Landmarks Extraction

Endlich ist es zu dem wichtigsten Punkt meines Teils der Diplomarbeit gekommen, die Gesichtsschlüsselpunkte Extraktion.

Dazu wurden die „Facial Landmarks“ von dlib verwendet.

Sie, Gesichtsmarkierungen, werden verwendet, um Bereiche des Gesichts zu finden und darzustellen, wie zum Beispiel: die Augen, die Augenbrauen, die Nase, den Mund, den Kiefer.

Wie macht man das? Wie erkennt man Landmarken?

Das ist eine Teilmenge des Problems der Formvorhersage.

Bei einem vorgegebenen Eingabebild (und normalerweise einer ROI<sup>1</sup>, die das interessierende Objekt angibt) versucht ein Formvorhersager, wichtige Punkte entlang der Form zu lokalisieren.

Dieser Formvorhersager die im dlib integriert ist wurde von Kazemi and Sullivan in ihrem Paper: One Millisecond Face Alignment with an Ensemble of Regression Trees entwickelt.

Dieser Methode werden viele Trainingsdaten hinzugefügt womit es ein Kombination von Regressionsbäumen trainiert wird um die Positionen der „Facial Landmarks“ direkt aus den Pixelintensitäten selbst zu beurteilen.

[Kazemi2014OneMF]

Dadurch werden unsere gewünschten 68 Gesichtsmarkierungen mit hohen Vorhersagbarkeit extrahiert und als x,y Koordinaten weitergegeben. Die Indizes der 68 Koordinaten sind in der Abb.6.9 dargestellt:

---

<sup>1</sup>Region of interest

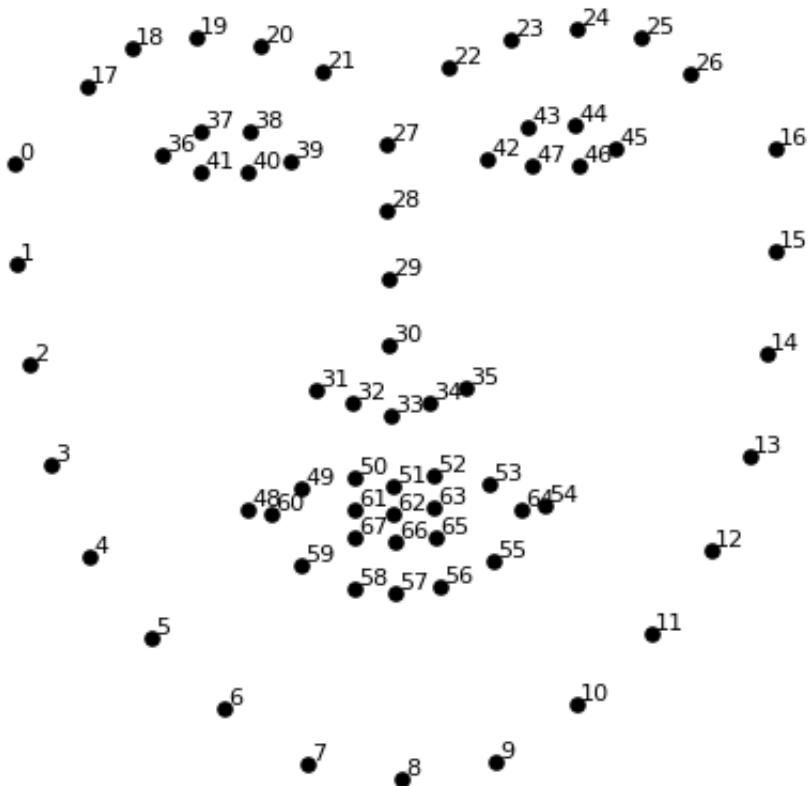


Abbildung 6.9: Facial Landmarks  
**[Kazemi2014OneMF]**

Nachdem wir der „Predictor“ geladen haben speichern wir es in einem Formobjekt die die 68(x,y) Koordinaten enthält.

Es wird hier die resize Funktion nicht verwendet, weil es an Qualität verliert und es rechenintensiver ist, je größer das Eingabebild wird.

Jede Koordinate läuft in einer Schleife durch und entspricht dem spezifischen Gesichtsmerkmal im Bild.

Die Merkmale werden in einer numpy array gespeichert für den Zweck der Speicherung in der Datenbank.

Nur die Positionen von diesen Merkmalen alleine reichen bei Gesichtserkennung nicht. Deswegen wurden die relative Positionen bzw. Abstände zwischen Gesichtsmerkmalen als Vektoren genommen und betrachtet um die Qualität und die Präzision der Erkennung zu verbessern/verbessern.

\*Die Zahlen beziehen sich auf die oben genannten Indizes.

1. Rechtes Auge Höhe:  $V1 = 41 - 37$
2. Rechtes Auge Breite:  $V2 = 39 - 36$
3. Linkes Auge Höhe:  $V3 = 46-44$

4. Linkes Auge Breite: V4 = 45-42
5. Rechte Augenbraue Breite: V5 = 21 – 17
6. Linke Augenbraue Breite: V6 = 26 – 22
7. Rechte augenecke mit mittlerer rechter Augenbraue: V7 = 36 – 19
8. Linkes augenecke mit mittlerer linker Augenbraue: V8 = 45 – 24
9. Nasespitze mit Mundzentrum: V9 = 45 -24
10. Linke Augenecke mit linker Mund Ecke: V10 = 54 – 45
11. Rechte Augenecke mit rechter Mund ecke: V11 = 48 – 36
12. Nase Höhe: V12 = 33 – 27
13. Nase Breite: V13 = 35 – 31

### 6.3 Herausforderungen, Probleme, und wie wurden sie gelöst

Die größte Herausforderung lag bei der Überlegung von der Software und die Methoden die zum Extrahieren von den Gesichtsschlüsselpunkten dienten.

Es hat mir viel Zeit gedauert bis ich eine geeignete Lösung gefunden habe und das hat viel Stress gemacht.

Eine weitere Herausforderung war das Verknüpfen von den Entwicklungsumgebungen und die Kooperation zwischen den Teammitgliedern.

Die verwendete Systeme waren all zu unterschiedlich und es konnte keine Standardisierung zwischen denen geben. Also ist viel Zeit beim Installieren und Konfigurieren investiert worden. Ein Grund dafür ist die mangelte Erfahrung mit den neuen Technologien.

Viele Sachen waren im Beginn auch unklar wegen die Kommunikationslücken und auch die sehr grobe Planung hat nicht geholfen.

Lösungen:

Während der Arbeit habe ich viel recherchiert und mich genau über alles Mögliche informiert.

Die Kommunikation hat sich mit der Zeit viel verbessert und dadurch wurden auch die Unklarheiten abgeklärt.

## 6.4 Qualitätssicherung, Controlling

Die Qualität wurde durch verschiedene Methoden gesichert. Eine von denen war 5xWarum. „Fünf Warum (5x Warum) ist eine iterative Fragetechnik, mit der die Ursache-Wirkungs-Beziehungen untersucht werden, die einem bestimmten Problem zugrunde liegen.“ [fmea] Die aufgetauchten Probleme haben viele Ursachen, die nicht mit dem ersten Blick sichtbar sind. 5x warum hilft durch diese verschachtelten Ursachen das grundlegende Problem zu entdecken.

Eine Problemstellung: „Segmentation fault“ beim Finden von Keypoints mit FAST.“

1. Warum bekommt man überhaupt einen „Segmentation Fault“? Ein Segmentierungsfehler tritt auf, wenn ein Programm versucht, auf einen Speicherort zuzugreifen, auf den es nicht zugreifen darf, oder wenn versucht wird, auf einen Speicherort auf nicht zulässige Weise zuzugreifen (z. B. beim Versuch, an einen schreibgeschützten Speicherort zu schreiben, oder einen Teil des Betriebssystems zu überschreiben).
2. Warum versucht mein Program auf einen Speicherort zuzugreifen, auf den es nicht zugreifen darf?

Problem ergibt sich in dieser Zeile: “kp = fast.detect(image, None)”

Wahrscheinlich durften die Daten die fast.detect ergibt nicht in kp gespeichert werden.

3. Warum dürfen die Daten die fast.detect ergibt nicht in kp gespeichert werden ?  
Die Daten die fast.detect ergibt, dürfen nicht in kp gespeichert werden, weil kp kein Array ist.
4. Warum ist kp kein Array?  
Kp ist kein array, weil man es in Python manuell angeben muss.
5. Warum wurde es nicht manuell angegeben?  
Es wurde nicht manuell gegeben weil, die Methode fast.detect nicht gut untersucht/gearbeitet war. Man sollte mehr darüber in die Doku nachschauen. ODER hatte man beim Installation die benötigte Pakete nicht richtig installiert.

## 6.5 Ergebnise

Ich habe ungefähr 100 Stunden Zeit bis jetzt für die Diplomarbeit investiert und folgendes erreicht.

Face Detection wurde fertig implementiert.

Bilder sind halbwegs normalisiert worden.

Gesichtsschlüsselpunkte wurden extrahiert und die Merkmalvektoren wurden angegeben.

# Kapitel 7

## Gesichtserkennung - Rei

### 7.1 Allgemeines

Heutige Zeit ist die Technologie ein der wichtigsten Themen für den Welt. Mithilfe der Technologie hat die Menschheit wesentliche Fortschritte gemacht im Bereich der Lebensqualitätsverbesserung und Erleichterung. Mehr als jemals zuvor kommen intelligente Systeme im Einsatz, die um den Arbeitsaufwand und die Arbeitskomplexität zu reduzieren dienen. Diese Systeme werden in die wichtigsten Bereiche des Lebens verwendet, wie zum Beispiel: Menschlichen Beziehungen, Business, Transport, Banken, Medizin, Bildung und Kommunikation. Selbstverständlich gibt es zahlreiche Versuche von verschiedenen Menschen, die als Hauptziel den Zugriff auf diesen Systemen zu ermöglichen haben. Deshalb ist die Sicherheit in diesem Systemen eine der aktuellsten und wichtigsten Themen. Der unkontrollierte Zugriff auf solchen wichtigen und delikaten Systemen wurde zu unreparierbare Konsequenzen führen. Auf dieser Gründe muss es sehr genau und detailliert geplant werden, welche Sicherheit Strategien die klügsten bzw. die besten sind.

Der Hauptthema dieses Projekts ist genau die Sicherheit der Systeme, und zwar die Ermöglichung des kontrollierten Zugriffs von Menschen in verschiedenen Gebäude. Es wurde die Verantwortlichkeit übernommen, dass der Zugriff in der Österreichischen Schule "Peter Mahringer" überwiesen und kontrolliert ist. Zu diesem Zweck wird in diesem Projekt ein intelligentes System entwickelt, dass die Aufgabe hat, menschliche Gesichtern (Gesichtern von Schülern, Lehrern, Eltern usw.) zu registrieren und zu erkennen. Es wird höchstwahrscheinlich in dem Schultür integriert, weil vom dort werden alle unbedingt reinkommen. Dieser Projekt wird große Hilfe für die Schule sein, aufgrund der vom Kameras und komplexen Algorithmen angebotenen Möglichkeiten. Beispielweise wird es überprüft, ob der Person dass reinkommt ein Student oder Lehrer ist, oder wird es notiert, wer die Schule wann verlässt oder besucht. Hat ein Person kein Zugriff, wird er nicht reinzukommen erlaubt.

Um diesem Projekt zu realisieren, wird das Projekt in drei große und wichtige Teile zerlegt. Das erste Teil heißt Bildverarbeitung und hat die Aufgabe, Gesichtsbildern vom Kamera zu normalisieren und verarbeiten, damit sie in dem geeignete Format für die Gesichtsvergleichung sind. Das zweite Teil beschäftigt sich mit der Registrierung der Gesichter von Personen im Server. Das dritte Teil ist die Gesichtserkennung. Das

ist genau das Teil für dem ich verantwortlich bin. Hier geht es um die Erkennung der Gesichtern der Benutzern des Systems. Es wird durch eine Vergleichung überprüft, ob der Person vorher schon registriert worden ist oder nicht, und ob sein Gesichtsdaten schon im Server existieren oder nicht. Nur wenn die Vergleichsergebnisse positiv sind, darf der Person reinkommen. Dem Person werden die Ergebnisse durch Anzeigen kommuniziert. Dieses Teil des Projektes erfordert eine Arbeit mit Datenbanken, Gesichtsvergleichsalgorithmen und mit vielen System Tests. Teil meiner Aufgabe ist auch der Aufbau des Systems und alles was mit Hardware zu tun hat. Alle Hardware Komponenten werden in den folgenden Kapiteln klar, verständlich und deutlich erklärt. Eine grobe Skizze des Systems wurde sowie im Abb.7.1 zu sehen.

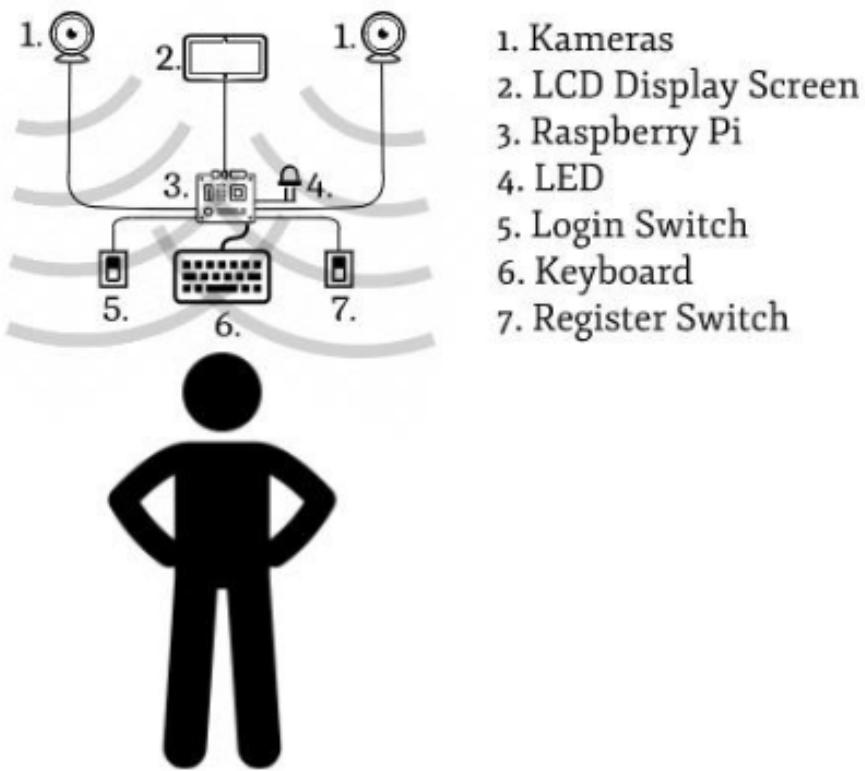


Abbildung 7.1: Grobe Skizze des Systems

## 7.2 Technische Lösung

### 7.2.1 Hardware und Aufbau

Als erstens wird der Aufbau des Systems beschrieben, zusammen mit allen verwendeten Komponenten, dass zu verwenden sind. Ohne die Hardware würde nichts funktionieren, weil die Software ohne die Hardware einfach nicht leben kann.

## Bauteile und HW-Komponente

Die Bauteile, die für dieses System verwendet geworden sind, sind die folgende:

### 1. *Steckboard bzw. Steckplatine*

Die Steckplatine ist der wichtigste Komponente der Hardware unseres Systems. Sie wird für die Entstehung der elektrischen Verbindung von verschiedenen elektrischen Bauteile, um elektrische Schaltungen zu bauen oder um verschiedene Tests und Experimenten zu machen. In dieser Steckplatine werden alle andere Komponenten platziert, damit die Verbindung erstellt werden kann und damit das System laufen kann.

### 2. *Kabeln bzw. Leitern*

Damit die verschiedene Komponenten, die in der Steckplatine platziert sind, miteinander verbunden sein können, braucht man unbedingt die Kabeln. Mithilfe von Kabeln können elektrische Impulse und Signale fließen, damit die Energie und die Information übertragen werden. Die verwendete Kabeln sind aus Kupfer und sind sowie vom Typ Male-Male als auch vom Typ Male-Female. Die Kabeln vom Typ Male-Female werden verwendet, um die Verbindungen zwischen den Elementen in der Steckplatine und den Raspberry Pi zu ermöglichen. Auf der anderen Seite werden die Male-Male Kabeln die Verbindungen innerhalb der Steckplatine machen.

### 3. *LEDs<sup>2</sup>*

LEDs sind elektronische Halbleiter Elemente, die Licht produzieren können, wenn sie Spannung kriegen. Ein LED besteht aus zwei Beine. Die längere Beine ist die Anode, die den Pluspol symbolisiert. Die andere Beine ist die Kathode, und symbolisiert den Minuspol. Durch die Beinen wird der Kontakt mit der Steckplatine erstellt.

### 4. *Widerstand*

Ein Widerstand ist ein elektrisches Bauteil, dass um die Reduzierung von Strom verwendet wird, damit es ein Gleichgewicht zwischen Strom und Spannung gesichert werden kann. Die Einheit ist Ohm.

### 5. *Tastern*

Ein Taster ist wird wie ein Button gedrückt, mit dem Zweck Impulse oder Signale zu schicken. Im Gegenteil zu einem Schalter wird der Taster nach der Betätigung wieder in der Basiszustand zurückgestellt. Ein Plusleiter, Minusleiter und ein Datenleiter sind bei einem Taster gefunden.

### 6. *Raspberry Pi*

Raspberry Pi ist ein Minicomputer, dass in diesem Projekt den normalen Computer ersetzt. Der verwendete Raspberry ist Version 3 und hat 4 USB-Anschlüsse, einem Netzteil, eine SD-Karte, 16 GPIO<sup>3</sup> Pins und einem VGA Schnittstelle. Die 32-Bit SD-Karte ist das wichtigste Element, weil dort alle Daten und Informationen gespeichert sind.

---

<sup>2</sup>Light Emmiting Diode

<sup>3</sup>Generated Input Output

### 7. Bildschirm

Ein Bildschirm ist nur eine Anzeige, dass um die visuelle Darstellung von verschiedenen Informationen oder Daten(wie Videos, Fotos, Statistiken usw) verwendet wird. Ein Bildschirm wird heutige Zeiten sehr häufig verwendet, aufgrund der hohen Benutzerfreundlichkeit dass angeboten wird.

### 8. Tastatur

Eine Tastatur ist ein Input Gerät, dass durch das Eindrücken von Buttonen den Benutzer die Eingabe von Daten oder Befehle ermöglicht.

### 9. Kamera

Letztens werden 2 Kameras benötigt, die die Fotos der Personengesichten machen werden. Sie werden auch im Raspberry integriert bzw. mit dem Raspberry verbunden. Die Kameras sind auf Typ A

## Schaltplan und Erklärung des Aufbaus

Die Hardware Komponenten werden in einer Steckplatine platziert. Dort werden die Verbindungen mit den anderen Komponenten sowie mit dem Raspberry Pi erstellt. Die elektrische Schaltung wird durch einen Schaltplan beschrieben. Dieser Schaltplan wurde mithilfe eines Programms, dass Fritzing heißt, gemacht und spielt eine sehr wichtige Rolle bei der Organisation und Planung des Schatzkreises. Der Schaltplan ist auf Abb.7.2 zu sehen.

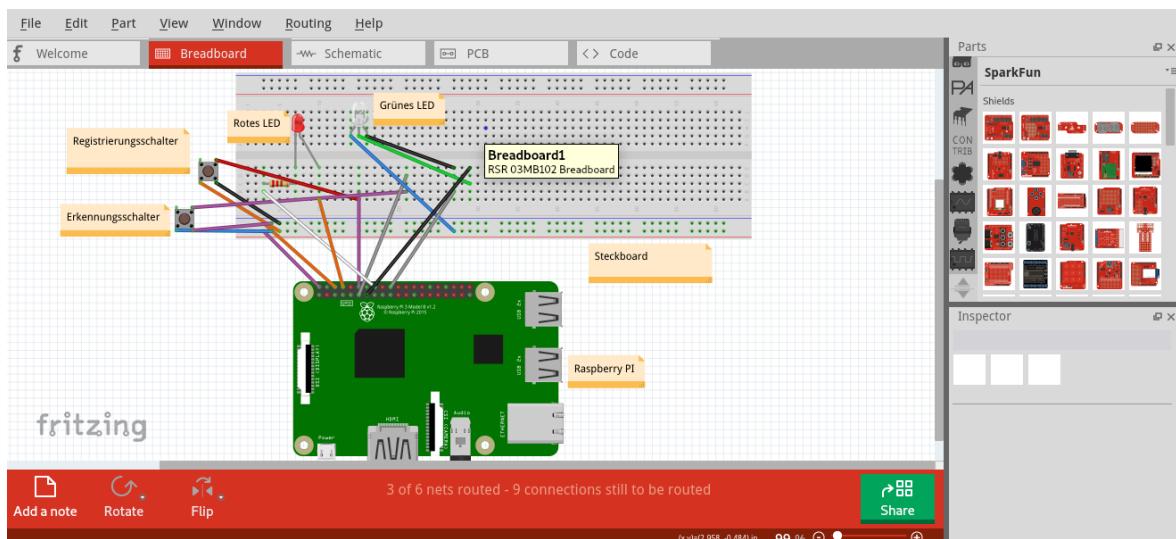


Abbildung 7.2: Schaltplan des Systems

Wie der Schaltplan zeigt, besteht das System aus zwei LEDs, zwei Tastern, einem Widerstand, einer Steckplatine und aus einem Raspberry Pi. Sehr wichtig für den Aufbau der Schaltung sind die GPIO Pins. Diese Pins sind in dem Raspberry Pi platziert und können als Input, als Output oder Spannung Pins verwendet werden. Der erste Taster dient für die Gesichtsregistrierung und besteht aus einem Pluspol, einem Datenleiter und aus dem Minuspol. Der Pluspol Anschluss wird mittels Steckplatine mit

dem 5-Volt Pin des Raspberry Pi verbunden, während der Minuspol Anschluss verbindet sich mit dem Minuspol der Steckplatine. Der Datenanschluss ist mit dem GPIO18 Pin verbunden. Wie bei der Registrierung-Taster wird auch bei der Erkennungstaster<sup>4</sup> der Minuspol Anschluss mit einem Pin des Minusbereichs der Steckplatine verbunden. Der Pluspol Anschluss gehört zu dem 5-Volt Pin des Raspberry, während der Datenanschluss mit dem GPIO17 Pin sich verbindet. Die rote LED wird verwendet wenn die Registrierung oder Erkennung der Benutzer im System nicht erfolgreich war. Die längere Beine oder die Anode wird mit dem GPIO23 Pin des Raspberry verbunden, während die Kathode wird in dem Minuspol Bereich der Steckplatine eingeschlossen. Die grüne LED ist eine RGB(Red Green Blue) LED. Diese LED kann die Farbe ändern. Wird im Fall einer erfolgreichen Registrierung oder Erkennung des Benutzers im System verwendet. Hat im Gegensatz zu der normalen LED 3 Anschlüsse. Der Minus Anschluss wird in dem Minuspol Bereich der Steckplatine eingeschlossen und der Pluspol Anschluss verbindet sich mit dem Pluspol Bereich der Steckplatine. Mit dem GPIO27 Pin des Raspberry wird der Datenanschluss verbunden. Die GPIO Pins sind extrem wichtig für die Integration der Tastern, LEDs und der anderen Bauteile in der technischen und logischen Teil bzw. in der Software und in der verwendete Skripte.

## 7.2.2 Software

Nachdem der Aufbau und die verwendete Hardware des Systems beschrieben wurde, ist die Software dran. In diesem Unterkapitel wird alles was mit der logische Teil der Umsetzung hat: die verwendete Betriebssystemen, Programmiersprachen, Frameworks, Technologien und Planungsmethoden. Es wird jede Aufgabe zusammen mit der zugehörigen Lösung im Details beschrieben und jeder programmierte Skript erklärt.

### Verwendete Betriebssystemen, Programmiersprachen, Frameworks, Technologien und Planungsmethoden

In diesem Teil des Projekts dass Gesichtserkennung heißt, werden für die Umsetzung verschiedene Technologien, Frameworks und Planungsmethoden verwendet.

#### 1. Betriebssystem und Programmiersprache

- Linux(Debian)

Linux ist ein von die wichtigsten Betriebssystemen die zur Verfügung stehen. Linux wird meistens für Programmierungs- und Konfigurationszwecke. Dieses Betriebssystems wichtigste Bestandteile sind der Bootloader, der Kernel, und das Init System. Linux würde für die Umsetzung dieses Projekts gewählt, obwohl es auch andere Möglichkeiten wie zum Beispiel Microsoft Windows gab. Die Gründe dieser Wahl sind die folgende:

- Open Source und Kostenlosigkeit

Linux wurde genutzt, weil es komplett kostenlos benutzt werden kann und weil es ein Open Source Betriebssystem ist.

---

<sup>4</sup>Auch als Login Taster genannt

- Hohe Sicherheit

Für dieses Projekt ist Sicherheit sehr wichtig, und Linux bietet eine hohe Sicherheit an. Es gibt keine Viren und keine Abstürze. Backup ist auch sehr leicht machbar.

- Vielfältigkeit an Möglichkeiten und Benutzerfreiheit

Linux wurde auch aufgrund der Vielfältigkeit der Möglichkeiten dass angeboten wird und aufgrund der Freiheit, die die Benutzern haben, verwendet. Beliebige Systemkonfigurationen, verschiedene Gesichtserkennungspaketen(OpenCV<sup>5</sup>) und andere Programmen oder Technologien(Fritzing) sind vom Linux besser und günstiger als bei anderen Betriebssystemen gefunden und installiert. Endlich ist der Raspberry auch mehr kompatibel mit Linux. Als Linuxderivat wurde aufgrund der großen Menge angebotenen der Paketen, der hohen Geschwindigkeit und der schnellen Korrigieren des Fehlers Debian gewählt.[[Linux](#)]



Abbildung 7.3: Debian

[[DebianBild](#)]

- Python

Die gewählt Programmiersprache ist Python. Der wichtigste Grund dieser Wahl ist hohe Leichtigkeit der Verwendung der OPEN-CV Pakete für die Gesichtserkennung Algorithmen. Im Vergleich zum C zum Beispiel werden die OpenCV Bibliotheken und Programmen viel praktischer und schneller gemacht. Python hat auch Vorteile im Bezug der Skripten Einbindung oder der Skripten Aufruf, als ein oder zwei Output Parametern genug sind um zwei Skripten miteinander zu verbinden. Version 3 wurde verwendet.



Abbildung 7.4: Python

[[PythonBild](#)]

---

<sup>5</sup>[Open Computer Vision](#)

## 2. Frameworks

In diesem Projekt werden einige Frameworks verwendet, die nach unten beschrieben werden.

- Open CV

Die wichtigste Framework ist OpenCV. OpenCV ist eine Softwarebibliothek, dass für Computer-Vision und maschinelles Lernen verwendet wurde. Die Bibliothek verfügt über mehr als 2500 optimierte Algorithmen, die sowohl klassische als auch moderne Computer Vision- und maschinelle Lernalgorithmen umfassen. Diese Algorithmen können verwendet werden, um Gesichter zu registrieren und erkennen, um Objekte zu identifizieren, menschliche Handlungen in Videos zu klassifizieren und Kamerabewegungen zu verfolgen. Dieser Framework wurde deswegen gewählt, weil die Vielfältigkeit der angebotenen Optionen und Paketen einfacher größer als bei anderen Frameworks ist. Ein anderer Vorteil ist dass OpenCV Open-Source ist. Die größte Herausforderung ist die lange und komplizierte Installation auf Linux.[**OpenCV**]



Abbildung 7.5: Open CV

### [OpenCVBild]

- Git

Git ist ein Versionsverwaltungssystem, dass für den Back-Up des Systems verwendet wurde. Dieser Framework ist ein verteiltes Versionsverwaltungssystem, das heißtt, die unterschiedlichen Versionen der Dateien werden funktional gespeichert, in dem die ganze Repository vom Server kopiert wird. Dieses erlaubt dem Benutzer lokal zu arbeiten, als alle Änderungen nachträglich im Server aktualisiert werden. Nicht nur das, sondern auch die hohe Geschwindigkeit der Fehlerfindung und die perfekte Unterstützung vom großen Projekte, in dem alle Teammitgliedern auf alle Daten Zugriff haben, waren die Gründe, warum Git für den Back-Up des Systems gewählt wurde.[**Git**]



Abbildung 7.6: Git

## [GitBild]

### 3. Technologien

Das System braucht auch ein paar Technologien für die Umsetzung. Unter Technologien sind die Programme, die Systeme und die spezielle Komponenten, dass benutzt worden sind.

- Raspberry Pi

Wie es vorher erwähnt wurde in dem Gesichtserkennungsteil des Projekts statt ein Computer ein Raspberry Pi verwendet. Was ein Raspberry Pi ist wurde in dem Kapitel Hardware erklärt. Die richtige Gründe der Wahl sind aber erst jetzt genant zu werden. Das kleinere Platzbedarf und die hohe Flexibilität dass der Raspberry anbietet sind wesentliche Vorteile. Die Bestandteile bewegen sich nicht, und es gibt gar keinen Lärm. Was sehr beliebt ist ist dass der Raspberry kann gleichzeitig für die Hardware(GPIO PINS) und die Software(wie ein normaler Computer) verwendet werden. Der unterstützt Raspberry Linux sehr gut. Ein der wichtigsten Gründe dieser Wahl ist auch dass und dass im Fall eines Fehlers oder irgendwann wenn es notwendig ist, wird ein Raspberry sehr schnell und problemlos geändert oder repariert(einfach SD-Karte kopieren und dann ändern, alles wird gespeichert und die Informationen werden zuverlässig in einem anderen Raspberry transportiert).



Abbildung 7.7: Raspberry Pi

## [RaspberryBild]

- Fritzing

Fritzing ist ein Programm das für die graphische Darstellung der Schaltkreis des Systems verwendet wurde. Durch dieses Programm wurde auch der Schaltplan des Systems erstellt(Abb.7.2). Fritzing bietet eine sehr große Menge von elektronischen Komponenten und ein PCB, Steckplatine und Schematic View. In diesem Fall wurde die Breadboard View gewählt, weil dort alles übersichtlicher ist. Was noch gut ist, ist dass Fritzing kostenlos im Linux angeboten wird, was für Windows nicht der Fall ist.

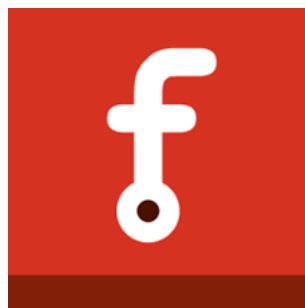


Abbildung 7.8: Fritzing

#### [FritzingBild]

- MySQL und MariaDB

Um die Testdatenbanken zu erstellen damit die Benutzerfotos und Benutzerdaten gespeichert werden können, wurde die berühmteste Open-Source relationales Datenbank System, MySQL. Zusammen mit MySQL wurde auch die Pakete MariaDB auf dem System installiert, damit Python mit dem MySQL-Python Pakete besser umsetzen kann. Mithilfe der MariaDB werden die ganzen Tabellen mit ihren Daten gesehen.



Abbildung 7.9: Mysql

#### [MysqIBild]



Abbildung 7.10: Mariadb

#### [MariadbBild]

##### 4. Planungsmethode

- Structed Design Structed Software Design ist die Methode, die um das System zu planen verwendet wurde. Diese Software-Architektur ermöglicht

den Entwurf, das Implementierungskonzept und das Integrationskonzept des Systems zu planen. Obwohl Structed Design großer Aufwand benötigt, hat diese Methode viele Vorteile, die bei der Wahl der Planungsmethode eine große Rolle gespielt haben: Structed Design verwendet Logik, ist von allem verstehtbar, beschreibt alles ohne Abgrenzungen und ganz detailliert. Die wichtigste Elemente die in der Structed Design beschrieben werden sind:

- Bubbeln  
Die wichtigste Arbeitsbereiche des Systems(Gesichtsregistrierung, Gesichtserkennung, Bildverarbeitung)
- Schnittstellen  
Sind die Interfaces, die die Verbindung zwischen die Bestandteile machen /
- Nachrichten  
Sind die geschickte Signale innerhalb des Systems.
- Flüsse  
Zeigt die Daten die innerhalb des Systems fließen.
- Modulen  
Bestandteil des Systems, dass innerhalb liegt.
- Terminatoren  
Ähnlich mit der Modulen, aber befinden sich außerhalb des Systems und bewirken es
- definierte Abgrenzungen  
Mithilfe der Abgrenzungen wird vom Systemhersteller genau definiert, was er nicht machen wird, sei es für Zeit-,Schwierigkeits-, oder Komplexitätsverbundende Gründe. Diese Abgrenzungen dienen damit der Auftraggeber genau weiß was er erwarten muss, damit es keine Missverständnisse gibt.[**StructuredDesign**]

Mithilfe der Structed Design wurde die BigPicture des Systems und die Erste Ebene der Gesichtserkennung erstellt.

### Lösungsweg- Beschreibung und Erklärung

Der Lösungsweg der Umsetzung der Aufgaben der Gesichtserkennung ist streng mit einer guten vorherigen Planung verbunden. Deshalb wurde nicht nur die Big Picture(Großes Sicht des Systems nach Außen), die in die vorherigen Kapiteln beschrieben wurde, sondern auch die Erste Ebene der Gesichtserkennungsbubbel. Die Big Picture steht unter Abb.7.11

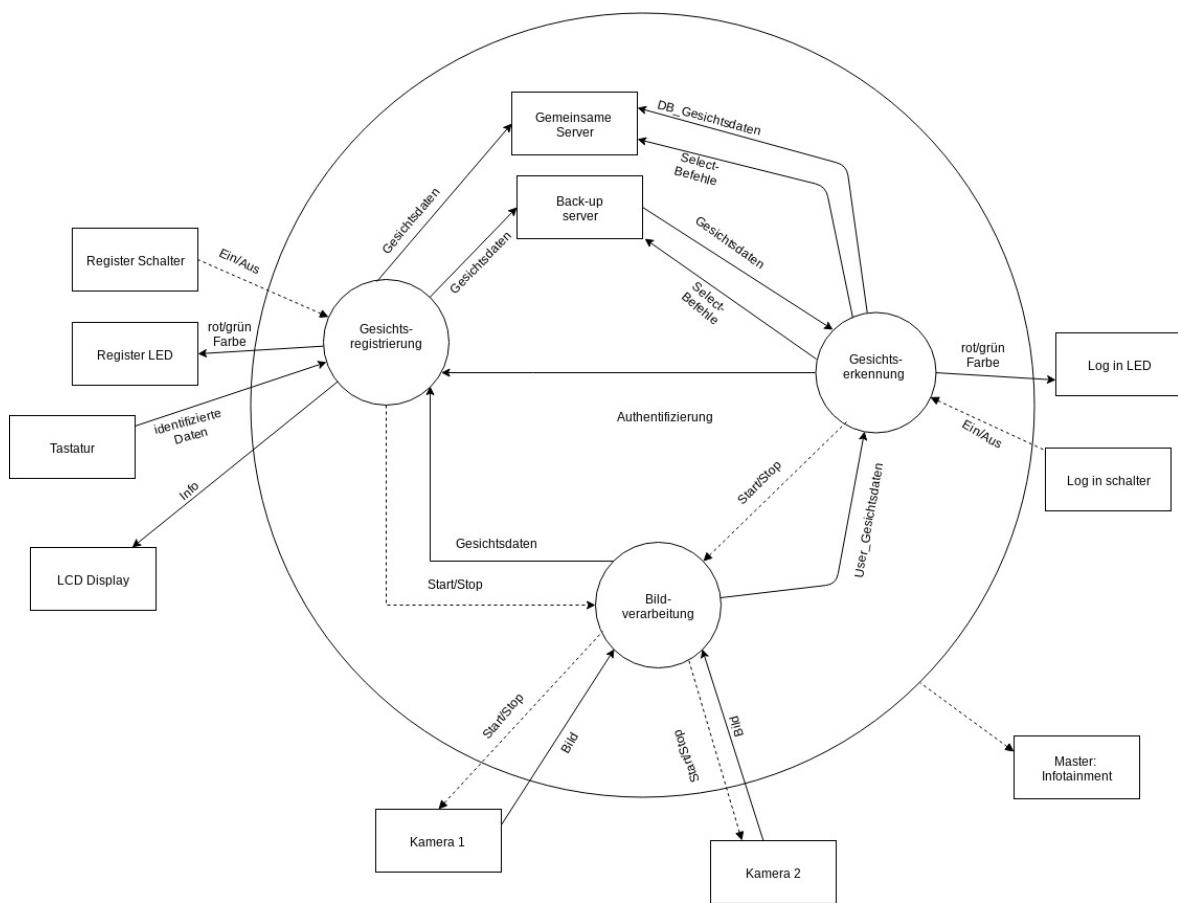


Abbildung 7.11: Big Picture

Die erste Ebene, die steht unter Abb.7.12. Sie ist ein mehr detailliertes Version der Big Picture, dass sich nur auf dem Erkennungsteil konzentriert. Man spricht von einer Iteration, dass hier passiert. Folgend wird es die Vorgehensweise und der Lösungsweg dieser Aufgabe mithilfe der Ersten Ebene erklärt und beschrieben.

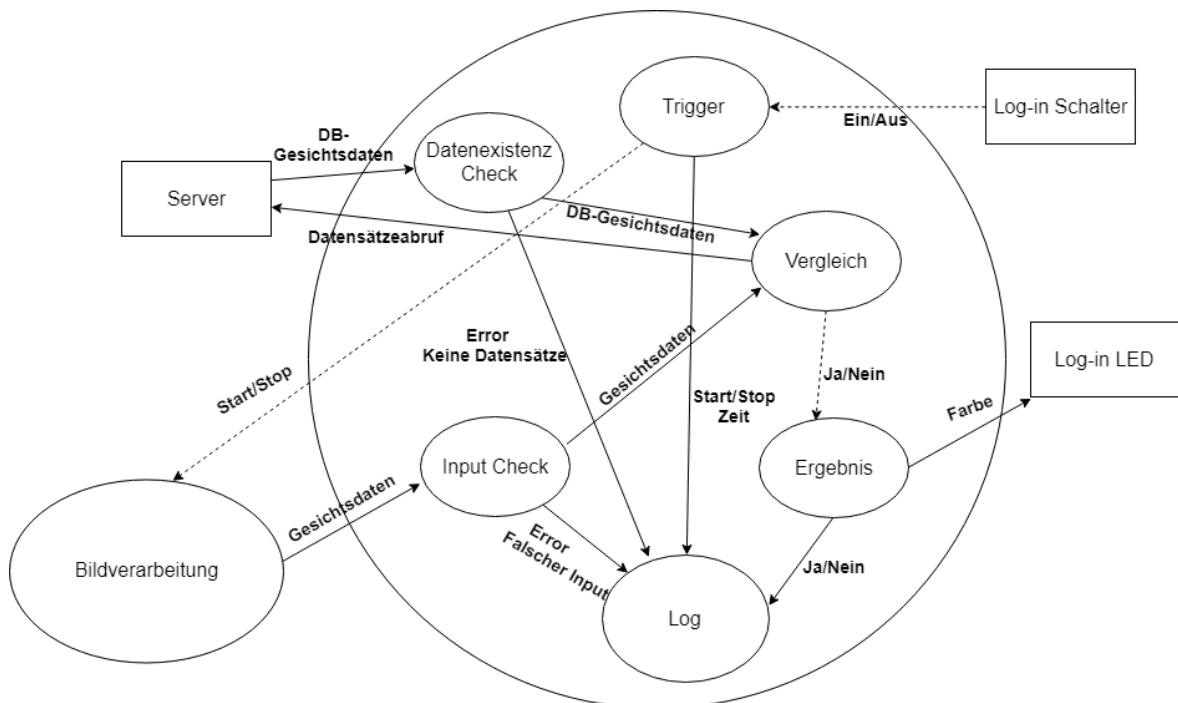


Abbildung 7.12: Erste Ebene

### Einzelne Arbeitsschritte:

#### 1. Erkennungstaster wird eindrückt

Will der Benutzer im System einloggen um reinzukommen, muss er zuerst den Erkennungstaster eindrücken. Wird dieser Taster gedrückt, dann kriegt der Benutzer eine Anmeldung, dass er seine Emailadresse eingeben muss. Um diese Aufgabe zu erledigen, muss die Pakete RPi.GPIO importiert werden. In dieser Weise ist der Zugriff auf den GPIO Pins ermöglicht, damit die Taster- Zustand, oder die LED Zustand erkannt werden. Die Pakette mysqldb wurde auch in der Zukunft benötigt, um Zugriff an die Datenbank zu haben.

```
if GPIO.input(17):
    exec(open('Existiert_nExistiert.py').read())
```

#### 2. Benutzer gibt seine Emailadresse ein

Es ist wichtig zu erwähnen, dass das Skripten Anruf innerhalb anderer Skripten extrem wichtig ist. Das wird durch Variablen gemacht. Deshalb ist das Importieren der Pakete sys nötig. Der Benutzer ergibt seine Emailadresse an, dass verwendet wird, um in der Datenbank schneller auf den Benutzerbilddaten zu suchen.

#### 3. Bild wird gemacht und wird temporär gespeichert

Gleich nachdem der Benutzer sein Email gegeben hat, wird die Kamera das Bild machen. Dieses Bild wird deswegen temporär gespeichert, weil nachdem der Vergleich von diesem Bild mit den anderen Bilderdaten, dass schon in der Datenbank

liegen, ist die Verwendung dieses Bildes nicht sinnvoll. Sonst würde es extrem viele Bilderdaten in der Datenbank, dass nur einmal verwendbar sind. Stattdessen wird nur das Path des Bildes zusammen mit den extrahierten Punkten in der Datenbank gespeichert.

4. *Die Existenz der selben Emailadresse des Benutzers in der Datenbank wird geprüft*

Dieser Schritt ist deswegen wichtig, weil falls das gegebene Email nicht in der Datenbank schon existierte, braucht das System gar nicht besorgen, um die Bildvergleichung zu machen.

5. *Wird die selbe Emailadresse in der Datenbank gefunden, werden die zugehörige Bildinformationen gekriegt*

Falls aber die vom Benutzer gegebene Emailadresse die gleiche mit einer Adresse in der Datenbank ist, weißt das System Bescheid, dass eine Bilderdatenvergleichung stattfinden muss. Um das zu erreichen, werden alle Bildinformationen gekriegt. Unter Bildinformationen sind die extrahierte Gesichtspunkte, die mindestens 16 sein werden, zu verstehen. Die kurze Codeabschnitt unten zeigt genau wie dieser Information Aufnahme erreicht wurde.

```
if b=="existiert":  
    mycursor.execute(  
        """ select * from info i \  
        join person p \  
        on i.idP=p.idP \  
        where p.email=%s """ %var1)  
    myresult=mycursor.fetchall()  
    for x in myresult:  
        print(x)
```

6. *Die Bildinformationen der beiden Bildern werden vergleicht*

Die wichtigste Aufgabe ist die Vergleichung der Bildinformationen. Es werden die extrahierte Punkten der bereit gemachten Bildern mit den Punkten der schon in der Datenbank existierte Bildern vergleicht. Dazu wird die wichtigste Pakete, dass Open CV heißt, benötigt. Nachdem diese Pakete importiert wurde, kann die Vergleichung beginnen. Es wurde so gedacht, dass jeder Punkt von einem Bild mit dem zugehörigen Punkt des anderen Bildes vergleicht wurde. In dieser Wiese wird eine Toleranz kalkuliert. Danach wird der Skript dass die Bilder vergleicht angerufen, dass die Toleranz als Output gibt. Passt sie zu der kalkulierten Toleranz, stimmt die Vergleichung, die Bilder passen miteinander.

7. *Passen die beiden Bildern zusammen, darf der Benutzer reinkommen*

Wenn die Vergleichsergebnisse positiv sind, wird die grüne LED leuchten und der Benutzer darf weitermachen. Sonst wird die rote LED eingeschaltet.

## 7.3 Herausforderungen, Probleme, und wie wurden sie gelöst

Während dieser Arbeit wurden einige Problemen und Herausforderungen eingetreten.

- *OpenCV Installation*

Die größte Probleme hat die Installation der OpenCV gegeben. Sie dauerte sehr lang und es war sehr schwer zu bestimmen, genau was ausgelassen werden sollte und was nicht. Dieses Problem wurde nach vielen Proben gelöst, durch das Kopieren von einer anderen SD-Karte, dass OpenCV schon installiert hatte. CMake und Make waren sehr wichtige Paketen um diese Aufgabe zu erledigen.

- *Fritzing*

Das Problem beim Fritzing war folgendes, dass jedes Mal dass der Programm beendet wurde, konnte er nicht mehr geöffnet werden. Es fehlten die Paths oder die Bauteile, die für die Schaltung zu verwenden waren. Die Lösung war eigentlich sehr leicht. Der Programm wurde komplett gelöscht und dann wieder im System installiert, und danach wurde die ganze Schaltung gemacht, ohne den Programm zu beenden. Sonst wurden die selben Probleme wieder auftreten.

- *Kurzschluss beim Raspberry Pi*

Was noch passiert ist, ist dass beim Anfassen vom Raspberry ein Kurzschluss gab. Der Raspberry war kaputt und musste ersetzt werden.

- *Pakete FaceRecognition Installation*

Die Installation der Pakete FaceRecognition kann nicht erfolgreich gemacht werden, weil die dlib Pakete auch gebraucht wird. Wahrscheinlich aufgrund des nicht genügenden RAMs kann diese Pakete nicht installiert werden. Dieses Problem wurde noch nicht gelöst.

- *Probleme bei dem Skripten Aufruf*

Am Beginn war das Umgehen mit der Skriptenaufrufe sehr schwer. Es war schwer zu verstehen wie das genau funktionierte, als der große Variablenanzahl die Arbeit kompliziert machte. Dieser Anzahl wurde reduziert und es wurden viele Recherche an der korrekten Verwendung von der sys Pakete gemacht um das Problem zu lösen.

## 7.4 Projektmanagement und Controlling

Im Bezug auf Projektmanagement und Controlling wurde die Methode des Fehlerbaums verwendet. Diese ist eine berühmte Methode um die Aufwandschätzung zu kalkulieren und um eine Fehlerursache zu finden. Es wird der Problem in kleinen Teilen geschnitten damit es alles klarer wird. Es wurde noch eine detaillierte Soll-Ist Analyse gemacht, die bei der neuen Aufgabeteilung sehr geholfen hat, und einen Balkendiagramm, dass für die Organisation und Festlegung der Arbeitstermine dient.

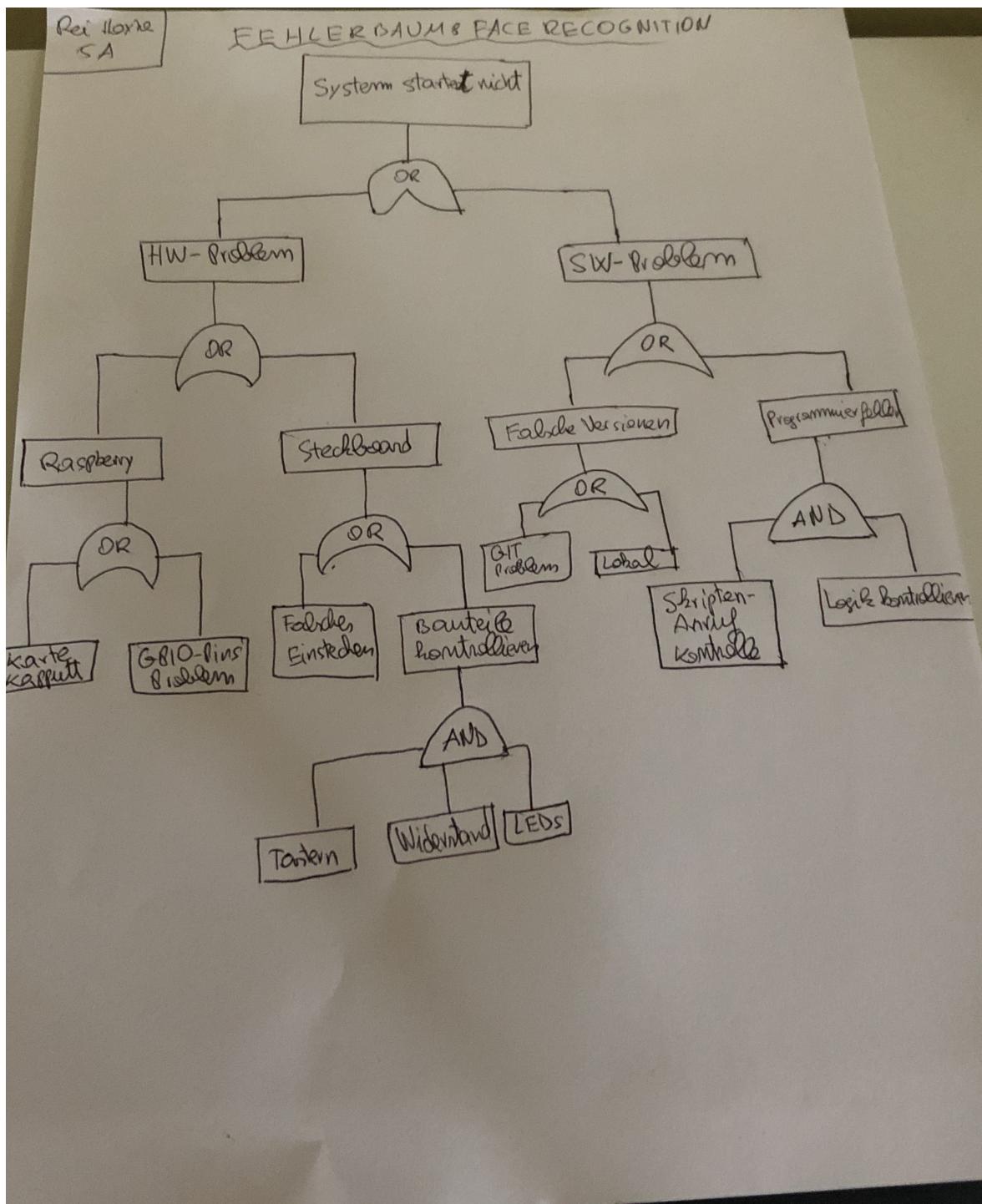


Abbildung 7.13: Fehlerbaum

## 7.5 Ergebnisse

Die bisherige Ergebnisse sind die folgende:

1. Systemaufbau  
Das ganze System wurde aufgebaut zusammen mit der ganzen Hardware.
2. Digitale Darstellung des Systemaufbaus  
Der Schaltplan des Systems wurde durch Fritzing digital dargestellt.
3. Erkennung der Admin  
Der Admin wurde sofort vom System erkannt, mithilfe einer Skript dass OpenCV verwendet um das Adminbild mit den anderen Bildern zu vergleichen.
4. Aufnahme der Benutzer Gesichtsdaten mithilfe der Emailadresse  
Gibt der Benutzer seine Emailadresse ein, werden die zugehörige Gesichtsdaten von der Datenbank selektiert und gekriegt, um den Vergleich zu beginnen.

# Kapitel 8

## Planung vs Realisierung

Planung vs Realisierung Viele Sachen sind anders umgesetzt worden als sie geplant waren. Die Aufgabenteilung wurde anders geplant aber bei der Implementierung wurde es als vernünftiger angesehen, dass die verändert werden. Es war die Wichtigkeit und die Größe der Arbeitspakete beim Beginn nicht richtig abgeschätzt weil wir uns nicht gut ausgekannt haben. Konkretes Beispiel ist das Datenbankdesign. Zuerst war diese Aufgabe an Aron zugeteilt, und erst später wurde festgestellt, dass Jordi besser für das geeignet war und so hat er sie übernommen. Gewisse Ziele wurden auch entsprechend angepasst und sogar weggenommen, weil sie als unnötig galten. Zum Beispiel: „Licht neben der Kamera“ wird nicht mehr berücksichtigt. Wir wollten auch eine minimale Verarbeitungszeit erreichen, aber dadurch wir fertig trainierte Modelle und Formvorhersager verwenden, auf denen wir nicht zugreifen können, ist es also schwierig so was zu machen. Es war früher auch geplant nur Gesichtsdaten in der Datenbank zu speichern aber erst wenn es in der Implementierungsphase gekommen ist, haben wir gemerkt, dass wir auch Fotos von den Personen(ins Besondere die von den Administratoren) in der Datenbank speichern sollen. Das wurde aber noch nicht fest entschieden. Auf Abb. 8.2 sind die erreichten und nicht erreichten Ziele.

Ziel	Status
Datenbankdesign wurde erstellt.	✓
Datenbank wurde eingerichtet.	✓
Zugriffsberechtigungen wurden implementiert.	✓
Gesichts-Schlüsselpunkte wurden extrahiert.	✓
Position/Verhältnisse der Hauptmerkmale wurden relativ zueinander herausgeholt.	✓
Daten werden für Gesichtserkennung aufbereitet.	✓
Gesichtserkennung wurde implementiert. Data streams von den extrahierten Bildern werden mit erheblichen Daten die auf der Datenbank gespeichert sind.	✓
Gesichter von zwei Administratoren wurden auf der Datenbank gespeichert für den Zweck der Admin Account.	✓
Admin Account (Register-Rechte nur für Schüler und Lehrer eingeben)	✓
Ein Maximum von 500 Personen ist gedacht, in der Datenbank gespeichert zu werden.	
10 Tests wurden gemacht, jeder Test in einer anderen Raumkondition. Alle Betriebskonditionen wurden getestet.	

Abbildung 8.1: Planung vs Realisierung

"Gefälschte" Gesichter wurden gegenüber "echte / legitimate" Gesichtern erkannt. Unterschied zwischen einer reellen Person und einem Bild bzw. einer Visuellen Darstellung von ihnen gefunden.	
Error checking ausgeführt. Es wird signalisiert wenn ein Gesicht nicht 100% identifiziert werden kann.	

Safe Mode erstellt. Backup Server wurde erstellt und eingerichtet.	
Min. Verarbeitungszeit (min. Zeit für Gesichtsdetektion) eingestellt. Es dauert min x Sekunden um ein Gesicht erfolgreich zu erkennen.	✓
Aufbau des Systems wurde erledigt.	✓
Es ist ein Benutzer-freundliches System mittels einer minimalen Anzahl von Schalter und einem Display erstellt.	✓

Abbildung 8.2: Planung vs Realisierung

# Kapitel 9

## Evaluierung und Resümee

### 9.1 Wertschöpfung und Lessons Learned

Also, nach einige Wochen sind wir zu Idee gekommen, dass die Planung eine große Rolle in diesem Projekt spielt, weil das Team must genau spezifizieren was jeder macht, und auch die wichtigsten Punkte, wie z.B der Lösungsweg. Bei den nächsten Treffungen soll mehr über den Problemen diskutiert werden, damit klar für jeden was genau zu tun ist. Eine chaotische Arbeit wird keine guten Ergebnisse bringen, weil auch eine Software bzw. Programm, dass ohne Struktur geschrieben wird, wird schwierig zum Lesen und Verstehen.

# Abbildungsverzeichnis

2.1	Big picture	4
2.2	Scrum	5
3.1	Structed Software Design bzw. erste Ebene	9
3.2	Packages zu importieren im Main-Skript	10
3.3	GPIO-Ports Konfiguration	11
3.4	Risikoanalyse in Excel	15
4.1	MySQL Logo	17
4.2	MariaDB Logo	18
4.3	Python Logo	19
4.4	OpenCV Logo	19
4.5	Plan-Do-Check-Act-Zyklus	25
6.1	OpenCV logo	28
6.2	Numpy logo	29
6.3	dlib logo	29
6.4	Bildverarbeitung Structed Design	31
6.5	Haar Features	33
6.6	Haar Einsetzung	33
6.7	Box on face: Aron	35
6.8	Aron Gesicht	36
6.9	Facial Landmarks	37
7.1	Grobe Skizze des Systems	42
7.2	Schaltplan des Systems	44
7.3	Debian	46
7.4	Python	46
7.5	Open CV	47
7.6	Git	47
7.7	Raspberry Pi	48

7.8 Fritzing . . . . .	49
7.9 Mysql . . . . .	49
7.10 Mariadb . . . . .	49
7.11 Big Picture . . . . .	51
7.12 Erste Ebene . . . . .	52
7.13 Fehlerbaum . . . . .	55
8.1 Planung vs Realisierung . . . . .	58
8.2 Planung vs Realisierung . . . . .	59

# Tabellenverzeichnis

4.1 Technologien . . . . .	17
----------------------------	----

# Literatur