

Diplomarbeit

Gesichtsregistrierung



AUTOR: ARON TERZETA

Inhaltsverzeichnis

1	Gesichtsregistrierung	1
1.1	Umsetzung	1
1.1.1	Allgemein	1
1.1.2	Technische Lösung	2
1.1.3	Herausforderungen	7
1.1.4	Qualitätssicherung und Controlling	9
1.2	Ergebnisse	10
1.2.1	Implementierung	10

Kapitel 1

Gesichtsregistrierung

1.1 Umsetzung

1.1.1 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 Gesichte 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-MySQLdb“ ist eine Schnittstelle, das nur für Python gültig ist, und die Verbindung zwischen Python (Scripts, Programme) und dem Datenbank Management System ermöglicht. Dient als Verbinder 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.

1.1.2 Technische Lösung

Technologien, die ich für die Implementierung verwendet habe sind:

- Linux als Betriebssystem [6] Linux ist das verwendeste Betriebssystem der Welt. Es ist eine open-source Software. Linux ist flexibel, man kann die einzelnen Modulen wegnehmen, 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-Foundationen 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." [1] Warum ich python gewählt habe, gibt es verschieden Gründe. Python hat weniger Schlüsselwörter, reduziert den Sytax 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 weiß, 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. 1.1 können Sie in einem technischen Weg besser sehen, wie der Gesichtsregistrierung-Teil arbeitet.

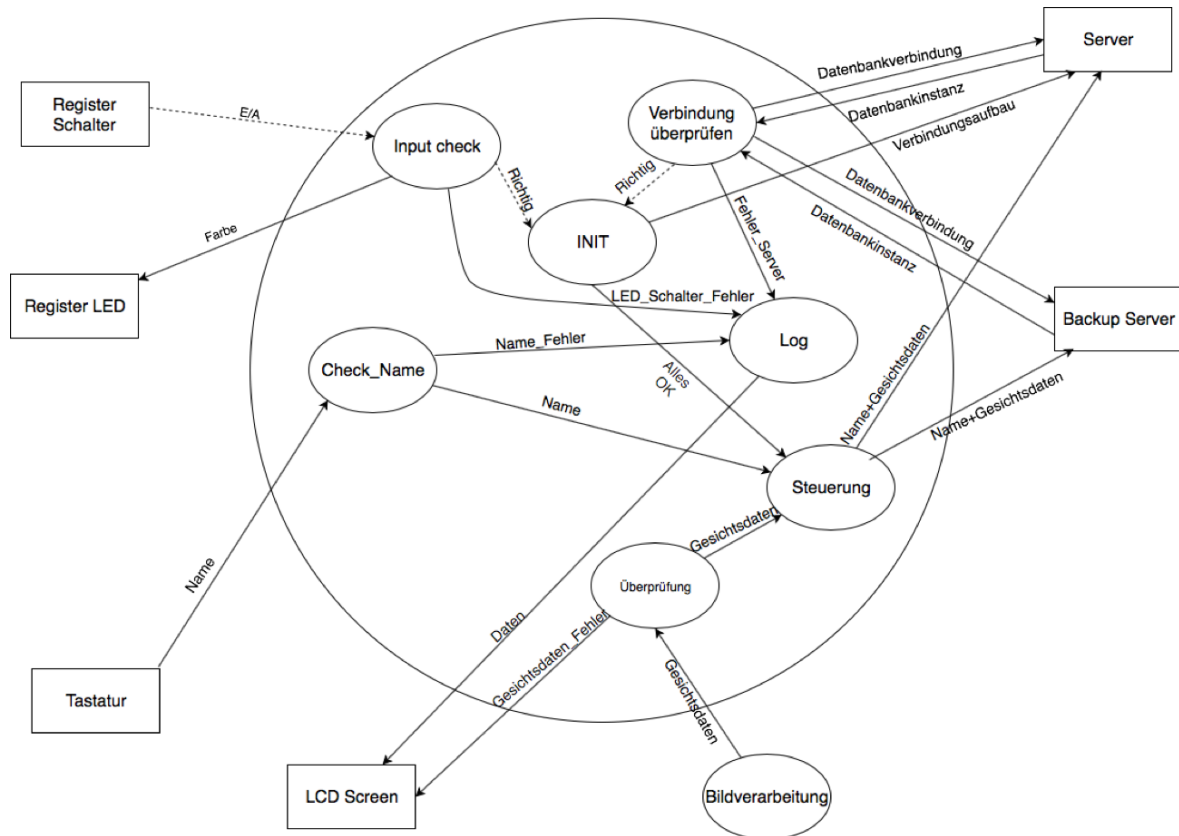


Abbildung 1.1: Structed Software Design bzw. erste Ebene

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 geschrieben wird. Der erste Grund ist, dass dieses Programm mit einem Python-Interpreter ausgeführt wird, der zweite ist, Verwendung des Programmsuchpfads, um es zu finden.

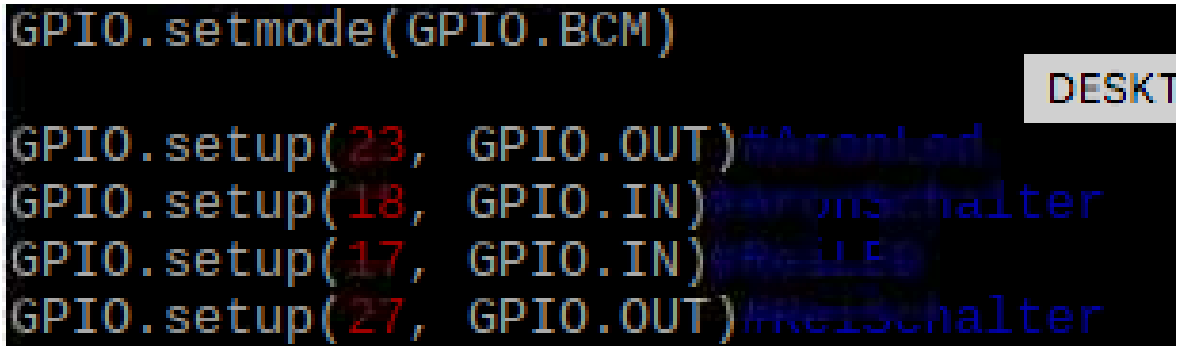
2. Alle Paketen importieren. Sehen Sie auf Abb. 1.2

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

Abbildung 1.2: Packages zu importieren im Main-Skript

- RPi.GPIO ist ein Paket, das verwendet wird, um Zugriff auf die sogenannten 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 sogenannte 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. [7]
- time ist ein Paket in Python. Von diesem Paket habe ich nur die Funktion 'sleep' verwendet. Diese Funktion pausiert das python-Programm. [4]
- 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. [3]
- 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. 1.3 ist auch der Code in python.



```
GPIO.setmode(GPIO.BCM)

GPIO.setup(23, GPIO.OUT) #Aron LED
GPIO.setup(18, GPIO.IN) #Aron Schalter
GPIO.setup(17, GPIO.IN) #Lara LED
GPIO.setup(27, GPIO.OUT) #Lara Schalter
```

Abbildung 1.3: 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 Betriebsart 'BCM'. [5] 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 Bilder 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 beide 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 Bilder 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 beide 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';"%(variable3))
myresult=mycursor.fetchall()
for x in myresult:
var1=x[0]
mycursor.execute("insert into info(imagePath,idP)
    values('%s',%s);"%( './'+variable3+'.jpg',var1))
```

1.1.3 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 Managment 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 Verzeichnisse 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 Missverständnisse 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. Strukturiertes 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 Teammitglied 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.

1.1.4 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.[2] Auf dem Abb. 1.4 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.	Wahr- sch.	Aus- wirk.	Ampel	Manager	Beschreibung	Behandlung und Kontrolle	Termin / Nächster Schritt
Standardrisiken								
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 Hasmegaj	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	Alle	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	Alle	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	Alle	Mangelnde Einführung	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	Alle	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	Alle	Ausfall einzelner Projektglieder	Ein anderer Projektglieder diese Arbeit machen	Einem neuen Team das Projekt einrichten

Abbildung 1.4: Risikoanalyse in Excel

1.2 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.

1.2.1 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 sogenannte 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 der Gesichter erfolgt.
- Die Person mit ihrem Vorname, Nachname, 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 geleuchtet.

Abbildungsverzeichnis

1.1	Structed Software Design bzw. erste Ebene	4
1.2	Packages zu importieren im Main-Skript	5
1.3	GPIO-Ports Konfiguration	6
1.4	Risikoanalyse in Excel	10

Tabellenverzeichnis

Literatur

Der ganze Rest

- [1] Stephan Augsten. *Was ist Python?* <https://www.dev-insider.de/was-ist-python-a-843060/>. [Online; accessed 2019]. 12/7/2019.
- [2] MA BSc Bekim Alibali. “Projektmanagement Teil5”.
- [3] Jackson Cooper. *Miscellaneous operating system interfaces*. <https://docs.python.org/3/library/os.html>. [Online; accessed 2001]. 2001.
- [4] Jackson Cooper. *Python’s time.sleep() – Pause, Stop, Wait or Sleep your Python Code*. <https://www.pythoncentral.io/pythons-time-sleep-pause-wait-sleep-stop-your-code/>. [Online; accessed Tuesday 23rd July 2013]. 2013.
- [5] Ben Croston. *RPi.GPIO Python Module*. <https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/>. [Online; accessed 2014]. 2014.
- [6] CHIP Digital GmbH Niels Held. *Linux-Umstieg: So einfach gelingt der Windows-Wechsel*. https://www.chip.de/artikel/Linux-Umstieg-So-einfach-gelingt-der-Windows-Wechsel-2_140047889.html. [Online; accessed 2007]. 2007.
- [7] Jeff Tranter. *Control Raspberry Pi GPIO Pins from Python*. <https://www.ics.com/blog/control-raspberry-pi-gpio-pins-python>. [Online; accessed Wednesday, July 31, 2019]. 2019.