

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

In diesem Kapitel wird feiner beschrieben, wie der Gesichtsregistrierungsteil funktioniert.

1.1 Umsetzung

Hier wird erklärt, wie es gedacht ist, den Gesichtsregistrierungsteil zu implementieren.

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 Gesichtern von Personen zu tun. Alle wissen, dass das Gesicht für jede Person anders ist. Jede Person wird mit ihrem Gesicht identifiziert, weil es einzigartig 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 gemacht.

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 Implementierungsphase verwendet, um die Veränderungen der Source-Code, wann geändert hat, wer geändert hat, deutlich zu sehen. Cmake ist ein Paket, das gebraucht wird, wenn das System mit OpenCV-Framework arbeitet wird. Es muss das System so konfiguriert sein, damit das OpenCV-Framework in einem C++ Programm verwendet werden kann. Deshalb brauchen wir spezielle cmake Befehle, die es ermöglichen.

”CMake wird verwendet, um den Softwarekompilierungsprozess mithilfe einfacher plattform- und compilerunabhängiger Konfigurationsdateien zu steuern und native Makefiles und Arbeitsbereiche zu generieren, die in einer Compilerumgebung Ihrer Wahl verwendet werden können.”[1]

Das gleiche passiert auch, wenn z.B. Python statt C++ verwendet wird. Die anderen 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 weit verwendete 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 ganzen 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 es 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.”[2]
Warum Python gewählt wurde, hat verschiedene Gründe. Python hat weniger Schlüsselwörter, reduziert die Syntax auf das Wesentliche und optimiert die Sprache. Ein Programm, das in Python geschrieben ist, ist vom Betriebssystem unabhängig. Das bedeutet, sie können Plattform unabhängig interpretiert werden. Python hat auch eine gute Lesbarkeit.

Das System besteht aus verschiedenen Terminatoren. Ein Terminator befindet sich außerhalb des zu definierenden Systems. Es kann eine andere Person, System oder eine Organisation sein. Terminatoren können von unserem System Informationen, Nachrichten, Materialien oder Energie erhalten oder das System empfängt diese.”[7]

Der wichtigste Terminator ist der ”Register-Schalter”. Er initialisiert das ganze Programm. Schalter in Technik ist nichts anders, nur ein Gerät zum Ein- und Ausschalten des Stroms oder zum Leiten des Stromflusses. 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 nicht stimmt, z.B. nicht richtige Inputdaten, dann wird mit einer bestimmten Farbe geleuchtet, nämlich mit rot. Wenn etwas passt, dann wird mit grün geleuchtet. 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 zu verwenden.

Eine spezielle Eigenschaft des Systems ist die Verwendung einer Tastatur. Sie wird verwendet, weil die einzelnen Personen ihren Namen, bzw. Email schreiben müssen. 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 es z.B. ein Problem mit dem System gibt, damit er nicht vor der Kamera warten muss. Eigentlich die Hauptfunktion des LCD-Screens ist, alles was der Benutzer mit der Tastatur schreibt, da zeigen zu lassen. Warum es so geplant ist? Das Problem steht daran, wenn der Benutzer seine Email schreibt, kann er Fehler machen, weil er nicht sieht, was er schreibt. Um das zu vermeiden, wird das LCD-Screen verwendet, damit der Benutzer sehen kann, was er schreibt.

Um registrierte Personen mit ihren Gesichtsdaten zu speichern, braucht das System einen Server. 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.

Das System hat auch einen Backup-Server. Die Daten werden parallel bei dem Hauptserver sowie bei dem Backup-Server gespeichert, damit die Daten noch gesichert sind, wenn der Hauptserver 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.

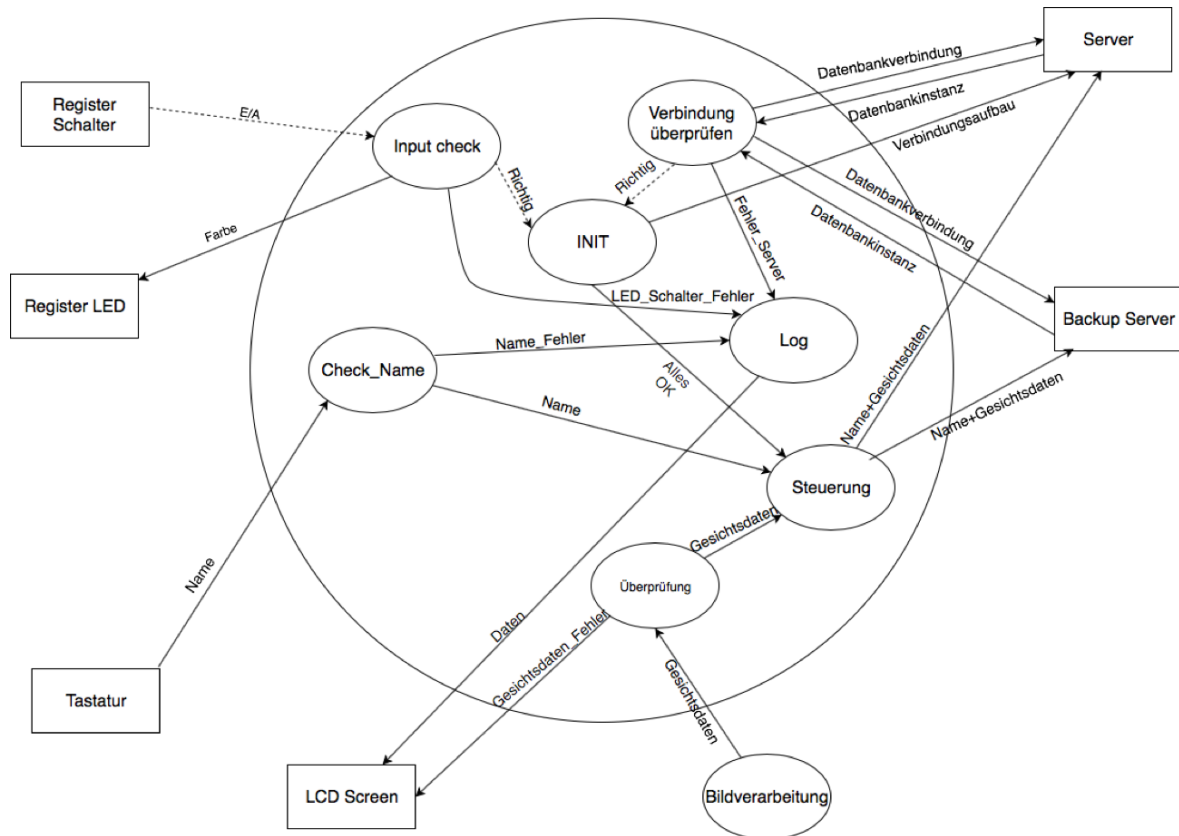


Abbildung 1.1: Structed Software Design bzw. erste Ebene

Um Schalter und LEDs im System verwenden zu können, brauchen wir ein spezielles Paket namens „RPi.GPIO“. Dieses Paket macht es möglich, den Raspberry PI mit HW (LED und Schalter) verbinden zu können. Dafür werden GPIOs verwendet. Der Schalter hat 3 Beine. Eines wird mit 5V verbunden, das andere mit Ground und das andere ist für Daten. Dies wird dann mit einem GPIO-Port in Raspberry PI verbunden. Das gleiche ist auch für die LED, damit es von Raspberry PI kontrollieren verwenden kann, wird mit einem GPIO-Port verbunden. 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.

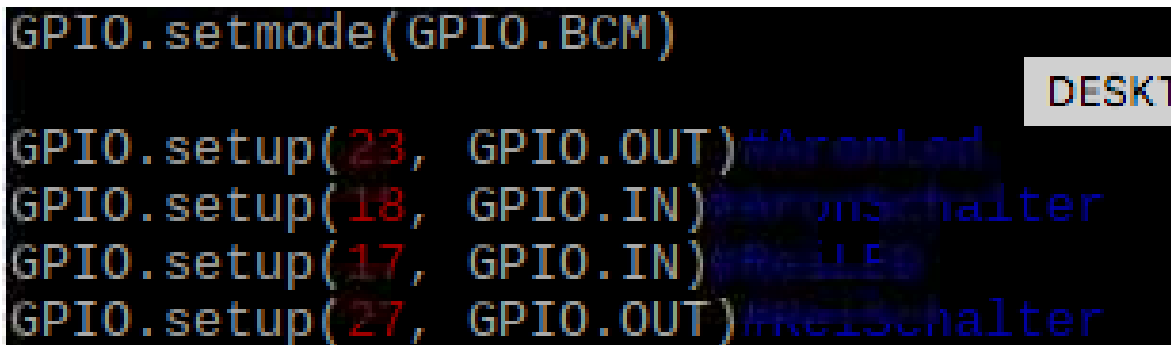
2. Alle Paketen importieren. Sehen Sie auf Abb. 1.2

```
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. 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. Es wird das rpi.gpio Paket verwendet, weil es leicht verständlich, programmierfreundlich und einfach zu verwenden ist. [8]
- time ist ein Paket in Python. Von diesem Paket wird 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. Es 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.
- subprocess Paket dient zur Verbindung zwischen verschiedenen Prozessen, in meinem Fall, ein Prozess heißt, ein Aufruf eines Skriptes, aber wird nicht im Skript verwendet.
- Das sys Paket wird verwendet, um Console Parameter zu geben. Das bedeutet, wenn der Skript aufgerufen wird, 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 Python-Code.



```
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.OUT) #LED
GPIO.setup(18, GPIO.IN) #Schalter
GPIO.setup(17, GPIO.IN) #Schalter
GPIO.setup(27, GPIO.OUT) #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 nur die Betriebsart 'BCM' verwendet werden. [5] Für die Registrierung der Schüler und Schülerinnen bzw. Lehrer und Lehrerinnen ist es nötig, dass der Admin physisch da ist. 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, wird der Wert '1' herausgekommen und 1 repräsentiert '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 physisch da ist. Ist der Admin da, können die Personen mit der Registrierung beginnen. Diese Person wird nach ihrem Vornamen, Nachnamen, Email und Rolle gefragt. 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 für das Einfügen der Daten in der Datenbank und bei der Speicherung des Paths des Bildes in der Datenbank mit dem gleichen Namen wie E-Mail einfacher ist.

Abbildung 1.4: Beispiel in Python, wie die Personen registriert werden

```
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))
```

Es wird die E-Mail verwendet, weil es eine performantere Suche in der Datenbank ermöglicht. Die E-Mail ist einzig, nur einmal für jede Person und die SQL-Anweisung (select-Abfrage) eine Selektion durchführt. Von 1000 Datensätzen wird nur der Datensatz ausgegeben, in dem die E-Mail mit der eingetippten E-Mail vom Benutzer übereinstimmt.

Anschließend, wenn die Person in der Tabelle 'person' gespeichert ist, hole ich ID dieser Person und füge dann diese ID mit der E-Mail und Path des Bildes in der Tabelle 'info' ein. Sehen Sie auf Abb. ??, wo einen Code in Python dargestellt ist.

1.1.3 Herausforderungen

Eigentlich hatte das Projekt für mich viele Herausforderungen. Die Gründe dafür sind, weil es ein ziemlich großes Projekt ist, haben wir neue Technologien verwendet, die wir vorher nie verwendet haben. Ich habe 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 wütend, weil das Datum des Ende des Projekts verzögert ist. Die Herausforderungen waren:

- opencv zu installieren. Das war eigentlich die größte Herausforderung. Es hat mir 3 Wochen 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 es ihnen erklären. Manchmal 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 ist nicht gegangen, deshalb sind wir zu Python gewechselt. 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 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 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 ein bisschen Zeit gekostet.
- Abhängigkeiten zwischen einzelnen Arbeitsteilen. Die Aufgaben sind so geteilt, dass sie Abhängigkeiten zwischen einander liegen. Das hat dann zu einer Verspätung der Projektabgabe geführt, weil jeder Teammitglieder aufeinander warten mussten.
- Was ich geplant habe, hat nicht gut funktioniert. Die großen Teile meiner Planung haben gepasst, nur wenige Kleinigkeiten musste ich ändern. Sie sind erst in der Implementierungsphase angezeigt.

Ich habe diese Lösungen für die Herausforderungen gefunden:

1. Ich habe viel Tutorials geschaut, Websites gesehen, wie opencv in Raspberry PI installiert werden kann. Ich habe viele verschiedene Methoden probiert,aber mit keinem guten Ergebnis. Nach vielen Proben ist es gegangen. Es ist installiert, und habe ich dann verschiedene Skripte in python gemacht, um es zu testen. Manche der Skripte sind gegangen, manche nicht. Jetzt war eine kleine Herausforderung für mich, dass ich die Skripte, die nicht ausgegangen sind, verbessere. 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. Ein Treffen mit meiner Gruppe vor dem Beginn des Projektes war notwendig. Ich hab es Ihnen gesagt und erklärt, in welchen Verzeichnisse sie arbeiten sollten, 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 dieses Treffen hat zu lange gedauert, bis alle verstanden hatten, wie, wo,was, wann machen sollen. Aber auch nach dem Treffen gab es zwischendurch Missverständnisse bzw. Probleme mit der Kommunikation, z.B. wurde 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. Um strukturierter zu werden, habe ich dann verschiedene Branches angelegt. 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 wurden, mit deiner Version am Computer synchronisiert werden. 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 mit dem Lernen von Python verbringen. Tutorials ansehen, Beispiele selbst probieren, die Quellen dafür selbst 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 minimieren, habe ich gedacht, dass jeder Teammitglied 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 machte. Aber ich habe diese Kleinigkeiten zur Kenntnis genommen, damit ich keinen solchen Fehler(Kleinigkeiten) mehr in der Planungsphase machen werde.

1.1.4 Qualitätssicherung und Controlling

Ein Risiko ist meistens nur eine Einschätzung, was kostet es einem Unternehmen, wenn die Projektziele nicht erreicht werden. Ich, als Projektleiter, muss das machen. Eine Risikoanalyse zu planen ist sehr schwierig, weil es mit der Zukunft zu tun hat. Zuerst muss ich an die Zukunft denken, bei welchen Bauteilen z.B. können Fehler auftreten, welche Programme können ausfallen. Das bedeutet, einen Überblick an der Zukunft und einschätzen, was für Fehler und Risiken es geben kann. Dann schätze ich die Wahrscheinlichkeit ihres Eintretens und am Ende die Maßnahmen. Dahinter versteckt sich eine große Arbeit.[3] Auf Abb. 1.5 können Sie die Risikoanalyse sehen. Wahrscheinlichkeit, Kosten usw. alles sollen beachtet werden.

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(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 1.5: Risikoanalyse in Excel

1.2 Ergebnisse

Es sind 3 Monaten vergangen, seit ich angefangen habe zu arbeiten. Weil meiner 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, befindet sich das Produkt in der Inbetriebnahme-Phase. 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 nichts geht. Die Funktionen, die in diesem Prototyp implementiert sind, sind folgende:

- Admin Account. Wenn sich eine Person registriert möchte, dann muss sich der Admin sich einloggen. Für diesen Prototyp gibt es nur ein Passwort, damit der Admin erkannt wird. Auf dem anderen Prototyp wird es kein Passwort geben, sondern die Überprüfung wird durch den Gesichter-Vergleich erfolgt.
- Eine Person wird mit ihrem Vorname, Nachname, Email und Role in der Datenbank gespeichert
- Ein Bild von einer Person wird gemacht, und in der Datenbank speichern mit der ID der Person speichern
- Wenn die Person erfolgreich in Datenbank gespeichert wird, leuchtet die LED.

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	Beispiel in Python, wie die Personen registriert werden	7
1.5	Risikoanalyse in Excel	10

Tabellenverzeichnis

Literatur

Aus dem Netz

- [1] Kitware inc. *CMake*. 2000. URL: `url`.

Der ganze Rest

- [2] Stephan Augsten. *Was ist Python?* <https://www.dev-insider.de/was-ist-python-a-843060/>. [Online; accessed 2019]. 12/7/2019.
- [3] MA BSc Bekim Alibali. “Projektmanagement Teil5”.
- [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] Dominik Stocklasser. “Architektur”.
- [8] 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.