Projektdokumentation

im Studiengang

AIN

PIPCO

Private IP Camera Observation

Referent : Prof. Dr. Elmar Cochlovius

Vorgelegt am : 13.01.2019

Abstract

This is the project documentation of a group from the course of studies Computer Science at the Hochschule Furtwangen University located in Germany. The project is taking place in the sixth semester and the group is consisting of four members. The project is about the implementation of a software for remote camera observation, while a special focus is placed on the interchangeability of the hardware in use. Furthermore, registered users of the product shall be informed automatically when a motion is detected by the software. Many providers of similar solutions are using cloud based services to tackle such tasks. By making use of more direct connections between the IP camera and the end consumer, this project aims to achieve lower latency.

Dies ist die Dokumentation zum Semesterprojekt einer vierköpfigen Gruppe aus dem sechsten Semester des Studienganges Allgemeine Informatik der Hochschule in Furtwangen. Bei dem Projekt geht es um die Implementierung einer Software zur Kameraüberwachung, wobei ein besonderer Fokus auf die Austauschbarkeit der Hardware gelegt wird. Zudem sollen durch eine Bewegungserkennung ausgelöste Benachrichtigungen automatisch an registrierte Nutzer versendet werden können. Viele Anbieter ähnlicher Softwarelösungen greifen bei der Umsetzung auf Cloud-Dienste zurück. Durch eine direktere Verbindung zwischen IP-Kamera und Endanwender sollen zudem geringere Latenzzeiten als bei zuvor genannten, kommerziellen Produkten erzielt werden.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

IP Internet Protocol

CLI Command Line Interface

URL Uniform Resource Locator

1 Einleitung

1.1 Rahmenbedinungen

Dieses Projekt stellt das Semesterprojekt von vier Studenten des Studienganges Allgemeine Informatik der Hochschule in Furtwangen dar. Es handelt sich dabei um das zweite Semesterprojekt, welches im sechsten Semester stattfindet.
Ziel des Projektes ist es, eine Software zur Überwachtung mittels IP-Kamera zu implementieren, wobei die genutzte Hardware austauschbar bleiben soll. Die Anwendung soll die Fähigkeit besitzen, Bewegungen im Kameralivestream zu detektieren und zuvor hinterlegte Nutzer per E-Mail über die erkannten Bewegungen in Kenntnis zu setzen. Außerdem sollen Aufnahmen dieser Bewegungen erstellt und für den Endanwender einsehbar hinterlegt werden. Neben diversen Einstellungsmöglichkeiten für den Nutzer, wie zum Beispiel für die Sensitivität der Bewegungserkennung oder einer maximalen Anzahl an gespeicherten Aufnahmen soll die Anwendung über eine nutzerfreundliche Weboberfläche mit Login-Maske verfügen.
Unter der Betreuung von Prof. Dr. Elmar Cochlovius und Judith Jakob wurde das Projekt weitestgehend selbstorganisiert durchgeführt. Ein für Testzwecke erforderlicher Hardware-Aufbau konnte im Smart-Home-Labor am Campus in Furtwangen genutzt werden. Dort waren auch ähnliche Lösungen von kommerziellen Anbietern vorhanden, welche während dem Projekt als Referenzen gedient haben.

2 Installation

2.1 System

Die Installationsanweisungen wurden auf einem Ubuntu Server der Version 18.10 durchgeführt.
Hierzu wurde ein bereits installiertes Image für VirtualBox von
`https://www.osboxes.org/ubuntu-server/` verwendet.

2.2 Backend

Für das Backend muss OpenCV, sowie der Flask-Webserver mit allen notwendigen Modulen installiert werden. Die Lightweight Installation von OpenCV, welche einfach mit pip installiert werden kann, enthält nicht den passenden Encoder für mp4, weshalb der aktuelle Stand selbst geladen und compiliert werden muss. Hierzu kann folgende Anleitung verwendet werden:
`https://www.pyimagesearch.com/2018/05/28/ubuntu-18-04-how-to-install-opencv/`
Zusätzlich zur Installation von OpenCV muss Flask mit pip installiert werden. Hierzu muss wie in der Anleitung beschrieben .bashrc mit

```
source ~/.bashrc
```

geladen werden. Anschließend kann mit

```
workon cv
```

in der Virtuellen Python-Umgebung gearbeitet bzw. flask wie folgt installiert werden:

```
pip install flask flask-cors
```

Nun muss nur noch das Repository ausgecheckt und ausgeführt werden.

```
git clone https://github.com/PIPCO-1819/PIPCO-Backend
cd PIPCO-Backend
python Main.py
```

2.3 Frontend

Für das Frontend wird Node.js, npm, sowie Angular verwendet.

```
apt-get install npm nodejs
sudo npm install -g npm@latest
sudo npm install -g @angular/cli
```

Repository auschecken und restliche dependencies installieren:

```
git clone https://github.com/PIPCO-1819/PIPCO-Frontend.git
cd PIPCO-Frontend
npm install
```

Anschließend den Server wie folgt starten:

```
ng serve --host 0.0.0.0
```

2.4 Run on Startup

start_backend.sh in PIPCO-Backend

```
#!/bin/sh
printf "<LOGIN>\n<PASSWORD>\n" | \
/home/osboxes/.virtualenvs/cv/bin/python Main.py
```

start_frontend.sh in PIPCO-Frontend

```
#!/bin/sh
printf "<LOGIN>\n<PASSWORD>\n" | \
/home/osboxes/.virtualenvs/cv/bin/python Main.py
```

start_frontend.sh in PIPCO-Frontend

```
#!/bin/sh
ng serve --host 0.0.0.0
```

start_pipco.sh

```
#!/bin/bash
screen -dmS frontend bash -c \
'cd␣/home/osboxes/PIPCO-Frontend;␣./start_frontend.sh'
screen -dmS backend bash -c \
'cd␣/home/osboxes/PIPCO-Backend;.␣start_backend.sh'
```

rc.local bei Start des Systems ausführen

```
printf '%s\n' '#!/bin/bash' 'exit␣0' | sudo tee -a /etc/rc.local
sudo chmod +x /etc/rc.local
```

Skript zu rc.local hinzufügen

```
...
/home/osboxes/start_pipco.sh
exit 0
```

3 Tests

Diese Testdokumentation wurde erstellt, um die Herangehensweise, Durchführung sowie die Ergebnisse unseres Testprozesses festzuhalten.

3.1 Testplan

3.1.1 Ziele

Ziel unseres Testprozesses ist es garantieren zu können, dass die in diesem Projekt geschaffene Software unter den von uns festgelegten Vorraussetzungen annähernd bis vollständig fehlerfrei und mit möglichst guter Performance betrieben werden kann. Auffälligkeiten sowie nach dem Testprozess bekannte und unbehobene Fehler sollen am Ende des Testprozesses dokumentiert sein.

3.1.2 Rahmenbedingungen

Grundsätzlich wurde während der Entwicklung der Anwendung stets darauf geachtet, dass die jeweils neu implementierten Features einwandfrei funktionieren und auch, dass durch die Implementierung jener Features keine der zuvor vorhandenen Teile beschädigt werden. Dennoch haben wir in unserer Projektplanung eine gesonderte Testphase geplant, bei der wir im Zeitraum von zwei Wochen alle nötigen Schritte abschließen möchten, um die von uns erstellte Anwendung ausgiebig zu testen.

3.1.3 Teststrategie

Um unser Testziel zu erreichen greifen wir auf verschiedene Testmethoden zurück. Da die Testphase sowohl durch einen kurzen Zeitraum, als auch die Anzahl der Tester eingeschränkt ist, müssen wir diese Ressourcen bestmöglich nutzen. Nach längeren Diskussionen innerhalb des Entwicklungsteams haben wir uns dazu entschlossen, auf eine Kombination von automatisierten Unit-Tests, manuellen System- und UI-Tests, sowie Last-Test zu setzen. Auf diese Weise decken wir beim Testen nicht nur funktionale sondern auch nichtfunktionale Anforderungen der Software ab.
Welche Methodik bei den einzelnen Teilen der Anwendung verwendet wurde wird in der folgenden Tabelle dargestellt.
Trotz dessen, dass wir eine eigene Testphase geplant haben, ist es uns wichtig über den gesamten Entwicklungsprozess der Software für eine stets einwandfrei lauffähige

Testobjekt	Art des Testens
Front-End	Unit-Tests, Manuelle Tests
Back-End	Unit-Tests. Manuelle Tests
Gesamtsystem	Manuelle Tests

Anwendung zu sorgen. Dies entspricht nicht nur unserem agilen Softwareentwicklungsprozess nach Scrum, sondern erleichtert auch die gemeinsame Arbeit durch mehrere Entwickler jeweils an Front-End sowie Back-End. Um dies gewährleisten zu können, haben wir abseits der Testphase jedes neu implementierte Feature sowie die Auswirkungen der Implementierung auf den Rest der Anwendung manuell getestet.

3.2 Testen des Front-Ends

3.2.1 Unit-Tests

Bei unserem Front-End sind wir zu dem Schluss gekommen, dass ein automatisiertes Testen nur bedingt sinnvoll ist. Ein großer Teil der Implementierungen dort bezieht sich rein auf die Darstellung der vom Back-End erhaltenen Daten im Webbrowser, oder um das Beschaffen und Versenden eben dieser Daten. Ein automatiertes Testen der Weboberfläche ist dabei überproportional aufwändig und in unserem Falle in den meisten Fällen nicht sinnvoll, da es sich vor allem um statische Inhalte oder um Video- beziehungsweise Bildinhalte handelt. Außerdem muss beim Testen einer Weboberfläche auf Faktoren wie Browserkompatibilität geachtet werden, was durch manuelles Testen besser umsetzbar ist. Nichts desto trotz wurde für jede Komponente des Front-Ends ein eigener Unit-Test erstellt, der die vollständige Erzeugung eben dieser Komponente simuliert und Testet. Dabei werden für die Komponente erforderliche Abhängigkeiten durch Mock-Objekte ersetzt, um ein unabhängiges Testen zu ermöglichen.
Standardgemäß verwenden wir beim automatisierten Testen unseres Angular-Front-Ends das Testframework Karma. Dieses ist bereits beim Erzeugen eines neuen Angular-Projektes per Angular-CLI integriert und vorkonfiguriert.

3.2.1.1 Ausführen der Unit-Tests

Nachdem das Projekt korrekt auf die in 23123 Dargestellte Art und Weise installiert wurde und lauffähig ist, können die automatisierten Tests durch das Aufrufen eines Konsolenbefehls gestartet werden. Dazu muss im Projektordner ein Terminal geöffnet werden und der Befehl nng testäusgeführt werden.

Einstellung	Wert
Sourrce.Livestrean-URL	https://webcam1.lpl.org/axis-cgi/mjpg/video.cgi
Maximum Clip Length	10
Maximum Clip Count	20
Maximum Clip Storage	1024

3.2.1.2 Ergebnisse der Unit-Tests

HIER SCREENSHOT VON UNITTEST ERGEBNISSEN EINFÜGEN

3.2.2 Manuelle Tests

Beim manuellen testen handelt es sich um einen Testprozess, bei dem der Tester ohne die Verwendung von Automatisierungstools vorgeht. Dabei können durch die systematische Verwendung der Software und das Nutzen von Diagnosetools oft Fehler aufgedeckt werden, die etwa bei Unit-Tests häufig nicht gefunden werden. Insbesondere Benutzeroberflächen können auf diese Weise unkompliziert getestet werden.
Im folgenden wird tabellarisch festgehalten, welche Aktionen getestet wurden und von welcher Ausgangssituation aus getestet wurde. Alle Tests wurden in den beiden Browsern Google Chrome (64-Bit Version 71.0.3578.98 Offizieller Build) und Mozilla Firefox (64-Bit Version 63.0.1 Offizieller Build) auf einem mit Windows 10 betriebenem Laptop mit einer Auflösung von 1920x1080 durchgeführt.

3.2.2.1 Ergebnisse der Manuellen Tests

Vorraussetzung für alle Tests ist selbsterklärend, dass Front-End sowie Back-End korrekt installiert und gestartet sind. Zudem sind alle Einstellungen sinnvoll gewählt. Das bedeutet beispielsweise, dass ein funktionierender MJPEG-Stream hinterlegt ist.
Folgende Einstellungen waren bei den folgenden manuellen Tests vorhanden:
Bedeutung der Spalte C*: Der Test wurde in der zuvor genannten Version von Google Chrome erfolgreich durchgeführt.
Bedeutung der Spalte F*: Der Test wurde in der zuvor genannten Version von Mozilla Firefox erfolgreich durchgeführt.

#	Komponente	Vorraussetz-ungen	Aktion	Erwartetes Ergebnis	C*	F*

1	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen den richtigen Usernamen (user) und das richtige Passwort (geheim) ein.	Der Anwender wird auf die Hauptseite der Anwendung weitergeleitet und ist korrekt eingeloggt.	X	X
2	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen einen falschen Usernamen (Verwendet: test) und das richtige Passwort (geheim) ein.	Rechts neben dem Login-Button erscheint eine Nachricht (Login failed) in roter Schrift.	X	X
3	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt. Das Back-End ist nicht erreichbar.	Der Anwender versucht sich einzuloggen.	Rechts neben dem Login-Button erscheint eine Nachricht (Login failed) in roter Schrift.	X	X
4	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen den richtigen Usernamen (user) und ein falsches Passwort ein. (Verwendet: test)	Rechts neben dem Login-Button erscheint eine Nachricht (Login failed) in roter Schrift.	X	X
5	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen sowohl einen falschen Usernamen (Verwendet: test1) als auch ein falsches Passwort (Verwendet: test2) ein.	Rechts neben dem Login-Button erscheint eine Nachricht (Login failed) in roter Schrift.	X	X

6	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen einen falschen Usernamen (Verwendet: test) und das richtige Passwort (geheim) ein.	Zwischen dem Absenden der Logindaten und dem Empfangen einer Antwort durch das Back-End wird rechts neben dem Login-Button eine Ladeanimation angezeigt.	X	X
7	Login	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender gibt beim Einloggen den richtigen Usernamen (user) und das richtige Passwort (geheim) ein.	Zwischen dem Absenden der Logindaten und dem Empfangen einer Antwort durch das Back-End wird rechts neben dem Login- Button eine Ladeanimation angezeigt.	X	X
8	Header	Der Anwender befindet sich auf der Login-Seite und ist demnach nicht eingeloggt.	Der Anwender klickt auf das PIPCO-Logo auf der linken Seite des Headers.	Die Webseite wird neu geladen.	X	X
9	Header	Der Anwender befindet sich auf der Settings-Seite und ist demnach bereits eingeloggt.	Der Anwender klickt auf das PIPCO-Logo auf der linken Seite des Headers.	Die Webseite wird neu geladen. Der Anwender ist nicht länger eingeloggt und wird daher auf die Login-Seite weitergeleitet.	X	X
10	Header		Der Anwender hovert mit dem Cursor über das PIPCO-Logo auf der linken Siete des Headers.	Ein Tooltip (Refresh Page) wird neben dem Cursor angezeigt. Der Cursor ändert sein Styling zu Pointer.	X	X

11	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite		Auf der rechten Seite des headers befinden sich ein Settings-Button sowie ein Logout-Button (in dieser Reihenfolge)	X	X
12	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.	Der Anwender hovert mit dem Cursor über den Settings-Button auf der rechten Seite des Headers.	Ein Tooltip (Settings) wird neben dem Cursor angezeigt. Der Cursor ändert sein Styling zu Pointer.	X	X
13	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.	Der Anwender hovert mit dem Cursor über den Logout-Button auf der rechten Seite des Headers.	Ein Tooltip (Logout) wird neben dem Cursor angezeigt. Der Cursor ändert sein Styling zu Pointer.	X	X
14	Header	Der Anwender ist korrekt eingeloggt und befinet sich auf der Hauptseite.	Der Anwender klickt auf den Settings-Button auf der rechten Seite des Headers.	Der Anwender wird auf die Settings-Seite weitergeleitet.	X	X
15	Header	Der Anwender ist korrekt eingeloggt und befinet sich auf der Hauptseite.	Der Anwender klickt auf den Logout-Button auf der rechten Seite des Headers.	Der Anwender wird korrekt ausgeloggt und auf die Login-Seite weitergeleitet.	X	X
16	Header	Der Anwender ist korrekt eingeloggt und befinet sich auf der Settings-Seite.		Auf der rechten Seite des Headers befinden sich ein Home-Button sowie ein Logout-Button (in dieser Reihenfolge)	X	X

17	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Settings-Seite.	Der Anwender klickt auf den Home-Button auf der rechten Seite des Headers.	Der Anwender wird auf die Hauptseite weitergeleitet	X	X
18	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Settings-Seite.	Der Anwender klickt auf den Logout-Button auf der rechten Seite des Headers.	Settings-Button sowie Logout-Button im Header sind nicht mehr da.	X	X
19	Header	Der Anwender ist korrekt eingeloggt und befindet sich auf der Settings-Seite.	Der Anwender hovert mit dem Cursor über den Home-Button auf der rechten Seite des Headers.	Ein Tooltip (Home) wird neben dem Cursor angezeigt. Der Cursor ändert sein Styling zu Pointer.	X	X
20	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.		Der in den Einstellungen hinterlegte MJPEG-Stream wird angezeigt.	X	X
21	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.		Über dem MJPEG-Stream wird eine Überschrift dargestellt (Currentyl Watching: IP Camera Live Stream)	X	X
22	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.	Aus der Event-Log-Komponente wird das Thumbnail einer Aufnahme angeklickt.	Der MJPEG-Stream wird durch eine Video-Wiedergabe des ausgewählten Clips ersetzt.	X	X
23	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.	Aus der Event-Log-Komponente wird das Thumbnail einer Aufnahme angeklickt.	Der Titel über der Clip-Wiedergabe ändert sich (Currently Watching: Motion Detection Clip)	X	X

24	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite.	Aus der Event-Log-Komponente wird das Thumbnail einer Aufnahme angeklickt.	Neben dem Titel über der Clip-Wiedergabe erscheint rechts ein Button (RETURN TO LIVESTREAM)	X	X
25	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite. Über die Event-Log-Komponente wurde die Wiedergabe eines Clips gestartet.	Es wird auf den Return-Button (RETURN TO LIVESTREAM) rechts oben in der Komponente geklickt.	Die Clip-Wiedergabe wird durch den MJPEG-Stream ersetzt.	X	X
26	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite. Über die Event-Log-Komponente wurde die Wiedergabe eines Clips gestartet.	Es wird auf den Return-Button (RETURN TO LIVESTREAM) rechts oben in der Komponente geklickt.	Die Überschrift über der Wiedergabe wird zurückgesetzt (Currentyl Watching: IP Camera Live Stream)	X	X
27	Video	Der Anwender ist korrekt eingeloggt und befindet sich auf der Hauptseite. Über die Event-Log-Komponente wurde die Wiedergabe eines Clips gestartet.	Es wird auf den Return-Button (RETURN TO LIVESTREAM) rechts oben in der Komponente geklickt.	Der eben betätigte Return-Button verschwindet.	X	X

Tabelle 1: Manuelle Front-End-Tests

4 [Eigene Kapitel]

5 Ausblick

6 Fazit

Literaturverzeichnis

Eidesstattliche Erklärung

Ich versichere, dass ich die vorstehende Arbeit selbständig verfasst und hierzu keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Stellen der Arbeit die wörtlich oder sinngemäß aus fremden Quellen entnommen wurden, sind als solche kenntlich gemacht.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt oder an anderer Stelle veröffentlicht.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben kann.

Furtwangen, den 13.01.2019