Erkenntnisse



|  |  |
| --- | --- |
| Autor: | Alexander Schmid |
| Datum: | 19.11.2021 |
| Klasse: | API20 |
| Abgabetermin: |  |
| Dozent: | Fabian Dennler |

Inhaltsverzeichnis

[1 Einleitung 3](#_Toc74247191)

# Einleitung

Wir müssen mittels einer agilen Methode eine Software entwickeln. Das Ziel dabei ist eine funktionsfähige Software zu haben die mittels einer API funktioniert. Wir haben uns entschieden, eine Software zu machen, wo alle Followers/Freunde etc. zusammenzählt und aufzeigt.

# Tag 1

Der erste Tag haben wir mehr oder weniger drauf los Programmiert nachdem alles besprochen wurde. Ich habe das Gefühl, dass wir eigentlich relativ wenig Aufwand haben werden, da wir Python verwenden. Ich habe ¾ vom tag an der API von Facebook/Instagram verwendet. Diese ist nur über einen Authentifizierungstoken der Software verfügbar, leider funktioniert dies nicht so, wie es Dokumentiert ist.

Nach langer suche und ausprobieren mit Herr Dennler selber habe ich einen weg bei Instagram gefunden, durch eine Hintertür diese Informationen raus zu lesen. Leider ist es so gesagt nicht wirklich eine API die dabei verwendet wird, jedoch kommen die Daten trotzdem in JSON.

Gleichzeitig haben wir uns entschieden, dass wir eine eigene API erstellen, mit den Informationen die wir fürs anzeigen der Software brauchen. Dafür hat uns Herr Dennler <https://replit.com/> gezeigt, wo wir so gesagt wie als Server verwenden kann.

## Erkenntnisse

Ich habe gesehen, dass mittels Python sich sehr vieles auch sehr einfach machen lässt… Jedoch eignet sich nicht alles um mit Python zu machen, da im Frontend und Serverseitig nicht so geeignet ist wie PHP zum Beispiel.

# Tag 2

Wir mussten uns heute eingestehen, dass wir mit Python an sich nicht genug weit kommen würden und ebenfalls sehr viele verschiedene Programmiersprachen verwenden müssten. Dazu überredet Node zu verwenden und schlussendlich eine Electron-Application zu erstellen hat uns Herr Dennler. Ich war zuerst ein bisschen skeptisch, ob die Zeit überhaupt dafür ausreichen würde, aber am Ende vom Tag sah es eigentlich gut aus.

Am Anfang des Tages hatten wir keinen Strom an den Arbeitstischen der Schule… Ich habe die Zeit ausgenutzt um Gedanken über die Applikation zu machen. Nebenbei habe ich auf einen Block Papier schriftlich festgehalten, was wir noch genau tun müssen:

* PHP-Form für das Python Script
* Python Übermittlung der Daten an den Node-Server
* Readme schreiben
* Logo erstellen
* Frontend Programmieren
* Functions auf Parameter umschreiben und einen Return-Wert geben

Anschliessend haben wir eine kleine Besprechung durchgeführt und die Arbeit unter uns verteilt und waren am arbeiten. Schlussendlich hat uns Herr Dennler wie erwähnt aufgezeigt, dass sich für so ein Projekt Node viel besser eignet.

Ich habe somit ein Electron Projekt aufgesetzt und ein neues Git-Repository erstellt, während Claudio mit Herr Dennler die Funktionen von Python auf Node umgeschrieben haben. Yassine war zwischenzeitlich mit mir dran und hat noch das Logo mit der Userstory fertig gestellt. Anschliessend fügten wir alles zusammen und ich habe die Applikation so erweitert, dass wir die Daten die wir brauchen Lokal auf den PC abspeichern können und ein Formular dafür gemacht (Herr Dennler musste mir dabei helfen).

## Erkenntnisse

Wie am ersten Tag schon gemerkt gab es viele Probleme mit dem Python-Script die wir am einfachsten mit der Programmiersprache Node umgehen konnten.

Was ich sicherlich gelernt habe ist, dass wirklich viele Sachen einfach mittels Packages gelöst werden können. Persönlich habe ich noch ein Problem mit den Async Funktionen und deren «Promises». Momentan habe ich den Fehler in der Applikation, dass ich die gespeicherten Daten zwar auslese, aber diese zu schnell wieder brauche und somit nur eine undefinierte Variable habe. Aus diesem Grund habe ich das Auslesen der Daten als eine Asynchrone Funktion in unser «function.js» geschrieben und diese mit einem Rückgabewert verziert. Nun bekomme ich ein «promise { <pending> }» zurück… Anschliessend habe ich an jene möglichen Stellen versucht, ein «await» einzubauen. Leider hat Node nie dieses Await akzeptiert und gab entweder die Meldung «'await' has no effect on the type of this expression.» oder «SyntaxError: await is only valid in async functions and the top level bodies of modules»… Ich habe leider wirklich keine Ahnung, wieso ich diese Fehler bekomme, aber ich bin mir sicher, dass ich diese Fehler in Zukunft sobald ich das gelernt habe nicht mehr haben werde da ich eine grosse Verzweiflung daran hatte und es Schmerzvoll war.

# Tag 3

Heute war mehr oder weniger war das Problem heute, das gleiche wie letztes Mal. Yassine hat heute unser Front-End gemacht und ich habe weiter am Back-End gearbeitet. Mir persönlich gefällt Back-End viel mehr, als Front-End. Leider ist das Backend bei uns momentan so komplex und noch ein durcheinander mit Issues, dass wir nicht viele und gleichzeitig daran arbeiten können.

Am Anfang des Tages habe ich das Backend, respektive unser Index.js, so umgesetzt, wie es vorgesehen war von der Letzten Woche aus. Hierbei hatte ich wieder das Problem mit den Asynchronen Funktionen, dass die Rückgabewerte zu spät übergeben wurden. Darauf hin habe ich unser ganzes Backend von Asynchronen Funktionen auf Promises umgeschrieben. Ich hatte jedoch das gleiche Problem mit den Promises… Darauf hin habe ich jeden noch so kleinen Task in einem Promise geschrieben und jeden Promise aufgrund eines anderen Promise durchlaufen lassen, so dass das Letzte Promise das fertige JSON bekommt und diese zurück resolved. Ich dachte, dass dies so funktionieren könnte, auch wenn das ganze ein sehr grosses durcheinander war. Leider gab es nur ein <Promised> zurück, welches ich nicht brauchen konnte. Als ich dies mit Herr Dennler angeschaut habe (ebenfalls mit ein paar Klassenkameraden, da ich mir mit der Zeit dumm vor kam immer fast alles mit Herr Dennler anzuschauen), sagte er, dass ich alles über die IPC-Kommunikation zwischen dem Renderer und dem Main machen sollte.

Ich tat dies anschliessend, so dass jede Sache die unser Index machen muss aufgrund eines Aufrufs des IPC-Renderer erledigt wird. Anschliessend gab es vom IPC-Main einen respons. Der Vorteil hierbei ist, dass wir im Frontend alles dynamisch anzeigen kann. Allerdings gibt es auch hier noch ein Problem mit der Asynchrone Funktion um die Anzahl Follower raus zu lesen, da die respons vor der Ausführung der Funktion kommt.

Ich habe anschliessend zuhause noch das Kanban-Board von unserem Python Projekt übernommen und noch ein paar weitere Tasks hinzugefügt. Ebenfalls ist mir aufgefallen, dass ich einen unathorized login in meinem Instagramm vorgefunden habe (von einem Linux System). Ich denke dieser Login kam von unserem Python-Script, ich habe allerdings diese Sache trotzdem als einen issue eingetragen.

Alle offene Punkte:

* Frontend dynamisch aufzeigen
* Readme fertig schreiben
* Logo erstellen
* Asynchrone Funktion Problem mit dem Rückgabewert lösen

## Erkenntnisse

Ich stosse auf sehr viele Probleme, welche ich so bis jetzt noch nie gesehen habe… Ehrlich gesagt verzweifle ich fast daran, was eigentlich noch ok wäre, allerdings muss ich meinen Teammitglieder über all das ebenfalls bescheid geben. Dies ist sehr schwer, da ich beschäftigt bin mit dem Backend und nicht wirklich vom Fleck komme aber gleichzeitig den Teammitgliedern bescheid geben soll. Wenn ich ihnen alles erkläre und bescheid gebe, bringen sie noch ihre Gedanken und Meinungen dazu, jedoch habe ich all diese schon ausprobiert und verliere somit nur Zeit.

Ich habe von dieser Woche nicht so viele neue Erkenntnisse gewonnen. Was sicher neu für mich ist, ist dieses im Team arbeiten, welches mir noch ein wenig schwerfällt, da mir das Abgeben eines Codes welches ich nur mässig verstehe nicht an einer nicht eingearbeiteten Person weitergeben will, da es für mich wie mein Baby ist.

Die Asynchrone Funktionen kann ich leider immer noch nicht als eine Erkenntnis einschreiben… Ich weiss zwar mittlerweile wie sie Funktionieren, aber verstehe nicht, wieso es keine Möglichkeit gibt diese abzuwarten…

Die Promises waren sicher eine Bereicherung für mein Wissen. Ich denke in Node.js sind diese bestimmt noch wichtig und denke auch, dass ich mich noch mehr mit diesen auseinander setzen muss für dieses Projekt. Während dem schreiben, kam mit wieder eine neue Idee, wie ich dieses Problem lösen könnte.