

# Modulare Text- Analyseplattform mit Segmentierer

*15.03.2021-21.05.2021 | P1 Projektmanagement | ON20 | 2. Semester*



## **Gruppenmitglieder**

Jonas Bloching, Nina Krauß, Linda Schäfer,  
Max Louis Scheufler, Laura Schramm & Dominik Thureau

# Inhaltsverzeichnis

<b>1. Projektorganisation</b>	2
1.1 Regeln der Zusammenarbeit	2
1.2 Kommunikationsregeln und -kanäle	2
1.3 Organisation der Dokumente und Informationen	2
1.4 Rollenverteilung	3
<b>2. Sprint 0</b>	4
2.1 Scrum-Methode und erste Überlegungen	4
2.2 Trello-Boards anlegen	4
2.3 Erstes Meeting mit Stakeholder und Scrum-Coach	4
2.4 Voraussetzungen und Zielsetzung	5
2.5 Ideen in Moodboard festhalten	5
2.6 Wettbewerbsanalyse	6
<b>3. Sprint 1</b>	7
3.1 Der Prototyp	7
3.1.1 Verwendetes Programm	7
3.1.2 Erstellen des Prototypen	7
3.2 Die HTML- und CSS-Struktur	7
3.2.1 Verwendete Programme und Services	8
3.2.2 Erstellen der HTML- und CSS-Struktur	8
3.3 Erstellen der JavaScript-Struktur	8
3.4 Einbinden der Daten mit JavaScript	9
<b>4. Sprint 2</b>	10
4.1 Anforderungen nach Sprint 1 Review	10
4.2 Segmentier-Funktion einbauen	10
4.3 HTML, CSS und JavaScript	11
4.3.1 Optimierung und Vereinheitlichung	11
4.3.2 CSS Effekte einbauen	11
<b>5. Sprint 3</b>	12
5.1 Anforderungen nach Sprint 2 Review	12
5.2 Dokumentation für neue Features	12
5.3 Tooltips	13
5.4 Hilfe Videos	13
<b>6. Fazit</b>	14

# 1. Projektorganisation

## 1.1 Regeln der Zusammenarbeit

- Wir sind pünktlich
- Wir geben den anderen Bescheid, wenn etwas dazwischenkommt
- Wir kommunizieren klar und deutlich, sodass jeder weiß, was zu tun ist
- Wir erledigen unsere zugeteilten Aufgaben zuverlässig und termingerecht
- Wir erwarten nach 22 Uhr keine Antworten/Ergebnisse mehr von den anderen
- Wir arbeiten mit Meilensteinen und kontrollieren, ob unser Anforderungsprofil erfüllt wurde
- Wir geben uns gegenseitig konstruktive Kritik
- Wir sind ehrlich zueinander
- Wir verhalten uns zuverlässig und arbeiten zuverlässig
- Wir verteilen die Aufgaben untereinander fair
- Wir lassen die anderen ausreden und hören einander zu
- Wir sagen rechtzeitig ab, sodass ein Ersatztermin gefunden werden kann
- Wir treffen Entscheidungen im Team und nicht als Einzelgänger
- Wir einigen uns bei Differenzen in der Gruppe diplomatisch
- Wir haben alle das gleiche Stimmrecht

## 1.2 Kommunikationsregeln und -kanäle

Für Textnachrichten beziehen wir uns auf den Messenger "WhatsApp" und bei Meetings treffen wir uns über Discord oder den angelegten Moodle-Raum. Hier können komplexere Themen "persönlich" besprochen werden und Bildschirme geteilt werden.

Wir halten uns an die allgemein bekannten Kommunikationsregeln.

## 1.3 Organisation der Dokumente und Informationen

Wir speichern alle unsere Dateien/Dokumente in einem freigegebenen Google Drive Ordner. Hier haben alle unabhängig voneinander Zugriff auf die Dateien und diese können in Echtzeit und parallel bearbeitet werden. Zur besseren Projektorganisation und Projektkoordination wird die Plattform Trello verwendet.

Bei Erstellung von neuen Dokumenten erfolgt dies direkt in unserem Google Drive Ordner, sodass jederzeit von überall darauf zugegriffen werden kann.

Zum Abgleichen von Gruppen-Terminen oder Aufgaben, die zu erledigen sind, haben wir einen gemeinsamen Google-Kalender erstellt. Auf diesen hat jedes Teammitglied Zugriff und hat damit jederzeit einen Überblick über folgende Punkte:

- ✓ Was muss bis wann erledigt werden? Und von wem?
- ✓ Wann ist das nächste Team Meeting?
- ✓ Wann ist der nächste Termin mit dem Kunden?

## 1.4 Rollenverteilung

Rolle	Name	Persönliche Skills
Product Owner	Laura Schramm	Organisation, Projektmanagement, Design, Webentwicklungs- Grundkenntnisse
Scrum Master	Jonas Bloching	Kontrolle, Organisation, Design, Qualitätsbewusstsein, Web-Entwicklungs-Grundkenntnisse
Entwicklerteam	Nina Krauß	Konzeption, Design, Webentwicklungs-Grundkenntnisse
	Linda Schäfer	Organisation, Projektmanagement, Design, Qualitätssicherung, Webentwicklungs-Grundkenntnisse
	Dominik Thureau	Softwareentwicklung, Datenbanken, fortgeschrittene Webentwicklungs-Kenntnisse
	Max Scheufler	Kreativität, Konzeption, Entwicklung, Webentwicklungs-Grundkenntnisse

## 2. Sprint 0

### 2.1 Scrum-Methode und erste Überlegungen

Dieses Projekt wurde mit der Scrum-Methode umgesetzt, welche in unserem Fall drei Sprints mit unterschiedlicher Länge beinhaltet. Daily Scrums des Entwicklerteams wurden ab dem ersten Sprint täglich für ca. 15 Minuten abgehalten. Diese dienten dazu kurz und knapp auf alle abgearbeiteten Aufgaben zurückzublicken und neue Aufgaben zu verteilen und zu fokussieren.

### 2.2 Trello-Boards anlegen

Um die Backlogs der Scrum-Methode verwalten und einsehen zu können, entschieden wir uns für die Gratis-Webanwendung "Trello". Hierbei werden die beiden Boards "Sprint Backlog" und "Product Backlog" unterschieden.

Das Product Backlog beinhaltet hierbei die Ideen und Überlegungen des Product Owner die mit dem Stake Holder besprochen wurden.

Die genaue Planung für die einzelnen Sprints wurde dann in den Sprint Reviews entwickelt. Dabei wurden die einzelnen User Stories auf deren Schwierigkeit geschätzt um den Workload besser verteilen und der Sprint besser geplant werden konnte.

### 2.3 Erstes Meeting mit Stakeholder und Scrum-Coach

In unserem ersten Meeting mit unserem Stakeholder, Herrn Ostermann, haben wir die grundlegenden Formalitäten des Projektes geklärt, sowie das Thema etwas näher kennengelernt.

Zur Abgabe gehört eine Dokumentation, mit ca. 25 Seiten. In dieser soll unsere Vorgehensweise, unsere Gedankengänge und unsere Umsetzung dokumentiert werden, sodass im Nachhinein verfolgt werden kann, was wir uns dachten und was wir gemacht hatten.

Ebenfalls gehört zur Abgabe ein Projektarchiv. Dies beinhaltet den Quellcode in einem Repository, Hilfevideos und mehrere Dokumentationen zum Projekt. Zum besseren Verständnis soll ebenfalls eine Dokumentation abgegeben werden, welche dem Betrachter einen Überblick verschafft, was er wo findet und warum.

Unsere Webanwendung sollte am Ende des Projektes folgende Features enthalten:

- Start Empfehlung (in Boxen)
- Benutzerfreundliche Bedienungsfläche
- integrierte Hilfe (Tooltips & Erklärvideos)
- Datenbestände einlesen (mindestens ein gängiges Importformat → XML)
- Funktion zum Filtern der Artikel
- Zerlegung des Textes in Absätze/Sätze → Segmentieren
- Clustern/Kategorisieren
- Dokumentationen (allgemeine Dokumentation TEXTALYSE, Wie erstell ich neue Features? How to start - TEXTALYSE)

Zur Projektbearbeitung haben wir einen Datenbank-Corona-Korpus von Herrn Ostermann zur Verfügung gestellt bekommen.

## **2.4 Voraussetzungen und Zielsetzung**

Ziel ist es im Laufe des Projektes eine erweiterbare Grundstruktur zu entwickeln, welche als erstes Modul es ermöglicht eine Textsammlung nach Begriffen zu filtern. Außerdem soll sie einen Segmentierer enthalten, der die Texte in kleine Einheiten unterteilt. Unser Projekt möchte die bisher bestehenden Text Analyse Plattformen kombinieren und mit einer guten Usability ausstatten, um zukünftig ein besseres Interface zur Verfügung zu stellen und somit die Nutzergruppe zu erhöhen.

## **2.5 Ideen in Moodboard festhalten**

Zur Vorbereitung unseres Prototypen erstellten wir zunächst ein Moodboard. Hier platzierten wir Screenshots von ähnlichen Webanwendungen oder gar Konkurrenten. In einem gemeinsamen Meeting legten wir, basierend auf dem Moodboard, die Farbgebung fest. Ebenfalls befassten wir uns näher mit den Konkurrenten und konnten so entscheiden, welche Komponenten wir für wichtig erachten und dann in die Webanwendung eingearbeitet werden sollen. Gleichzeitig erkannten wir hier auch einige Komponenten, die sich für unsere Webanwendung als nicht praktikabel erwiesen hätten. Diese Komponenten wurden bereits im Prototypen außer acht gelassen.

Nach der Besprechung des Moodboards erstellten wir das Logo und konnten mit der Erstellung des Design-Prototypen (Kap. 3.1) starten. Neben der Logogestaltung entwickelten wir in dieser Phase auch den Namen unserer Webanwendung.

## **2.6 Wettbewerbsanalyse**

Thema eines der ersten Meetings war die genauere Betrachtung und Analyse des Wettbewerbs. Hierfür wurde besonders die Plattform “ICE-Analysis”, eine Textanalyse Plattform die bereits in der Beta Version der Entwicklung steht, von uns angesehen und deren Funktionen analysiert.

Der Prozess stellte sich als sehr umständlich heraus, da nur ein Mitglied unseres Teams erste generelle Erfahrungen in der Arbeit mit einer Textanalyse Plattform hatte. Schwierigkeiten gab es in dem generellen Verständnis wofür und in welchem Zusammenhang solche Textanalyse Plattformen Verwendung finden. Auch war nicht klar, welches Ziel oder Ergebnis man sich aus der Arbeit mit solch einer Plattform erhofft. Die verschiedenen Funktionen, die wichtig sind um Texte erfolgreich und systematisch zu analysieren und zu sortieren, lagen nicht direkt auf der Hand, wurden allerdings durch den Blick auf ICE Analysis ein wenig klarer. Die Wettbewerbsanalyse war für dieses erste Verständnis sehr hilfreich und einzelne Funktionen wurden bei der Erstellung des Design-Prototypen übernommen.

Die Recherche nach weiteren Wettbewerbern blieb erfolglos. Plattformen mit vergleichbaren Funktionen, wiesen diese nur nebensächlich auf, befassten sich jedoch schwerpunktmäßig mit anderen Thematiken. Für den direkten Wettbewerb waren diese daher nicht relevant, weshalb wir uns dafür entschieden uns hauptsächlich auf “ICE-Analysis” zu fokussieren.

## 3. Sprint 1

### 3.1 Der Prototyp

#### 3.1.1 Verwendetes Programm

Den Prototypen erstellten wir in Adobe XD. Adobe XD ist in der Adobe Creative Cloud enthalten und ein vielseitiges UI/UX-Design-Programm zum Erstellen und Freigeben von Designs und Prototypen. Der größte Vorteil dieses Tools ist, dass Dateien für andere Creative Cloud Nutzer freigegeben werden können und eine gleichzeitige Bearbeitung stattfinden kann. So war die Erstellung unseres Prototypen gut auf mehrere Teammitglieder aufteilbar und weitere gestalterische Ideen konnten direkt ausgetauscht werden.

#### 3.1.2 Erstellen des Prototypen

Die Aufgabenstellung beinhaltet eine Webanwendung zu entwickeln, weswegen wir den Prototypen in der entsprechenden Standardgröße (1920 x 1080 px) anlegten. Die Farbgebung wurde bereits bei dem Besprechen des Moodboards festgelegt, ebenso wie die Komponenten, die in der Webanwendung vorhanden sein sollen. Uns erschien es wichtig zuerst mit der Startseite anzufangen, damit wir sehen, wie die Farben harmonisieren und was als Titelbild in Frage kommt.

Der Prototyp wurde in fünf Unterseiten, nach der Erstellung, unterteilt:

- Die Startseite ("Start") mit allgemeinen Informationen über die Webanwendung
- Die Übersichtsseite der Projekte ("Projekte"), auf der alle hochgeladenen Projekte angezeigt und verwaltet werden
- Die Übersichtsseite des Benutzers ("Nutzerprofil"), auf dieser können die persönlichen Daten des Nutzers geändert werden, die einzelnen Projekte werden hier ebenfalls nochmals aufgelistet
- Die Hilfeseite ("Hilfe"), hier werden hilfreiche Tipps zum Einstieg in die Webanwendung geliefert, sowie How-To-Tutorials zur optimalen Nutzung
- Die Anmelde-/Registrierungs-Seite ("Anmelden"), hier loggt sich der Benutzer ein bzw. ein neuer Benutzer registriert sich auf dieser Seite



## 3.2 Die HTML- und CSS-Struktur

### 3.2.1 Verwendete Programme und Services

Die Entwicklung unserer Plattform führten wir mit Hilfe der Applikation “Visual Studio Code” durch. Als Backup und Arbeitsumgebung legten wir ein Repository bei GitHub an, sodass alle gleichzeitig und gemeinsam entwickeln konnten.

### 3.2.2 Erstellen der HTML- und CSS-Struktur

Anhand des erstellten Design-Prototyps wurde die HTML- und CSS-Struktur in Visual Studio Code entwickelt. Hierfür wurde jede Seite in einem eigenen Ordner, der jeweils eine entsprechende HTML, CSS und JavaScript Datei beinhaltet, angelegt.

Im ersten Schritt entwickelte jedes Teammitglied ein Teilelement, welches für alle Seiten von Relevanz ist. Diese Entwicklung sollte mit Hilfe von StencilJS geschehen. StencilJS ist ein Compiler, welcher es ermöglicht einzelne Webkomponenten zu entwickeln die dann einfach abgerufen und eingebunden werden können. Allerdings stellte sich dieses Vorgehen für den weiteren Verlauf der Entwicklung unserer Plattform als eher unvorteilhaft dar, da StencilJS ein für uns noch neues Framework ist, welches wir erst dieses Semester kennengelernt hatten und so die Entwicklung viel langsamer vorangeschritten wäre. Nach dieser Realisation wurden die einzelnen Komponenten im Header und Footer zusammengefasst und mussten im Anschluss dann in jede HTML-Datei kopiert werden. Diese Integration des Headers und Footers sowie die weitere Gestaltung der einzelnen Seiten war unser zweiter Schritt in der Entwicklung, nachdem wir die Entwicklung, weg von StencilJS, umstellten.

Für die Einbindung jeglicher Bilder und Icons wurde ein entsprechender Assets Ordner angelegt, indem alle Bild- und Icon-Dateien gespeichert werden konnten. Im <head> der HTML-Dateien konnte dieser Ordner nun verknüpft werden und die einzelnen Bilddateien im <body> dann abgerufen und an den entsprechenden Stellen im Code eingebaut werden.

Nach dem Abschluss dieses zweiten Schrittes waren nun also sowohl die HTML- als auch die CSS-Grundstruktur für alle Seiten gegeben.

### 3.3 Erstellen der JavaScript-Struktur

Nach der Erstellung der HTML- und CSS-Grundstruktur sah die Plattform nun optisch bereits fertig aus, allerdings war die Seite so noch nicht interaktiv. Hier fing der weitaus anspruchsvollere Teil unseres Projekts an.

Als erstes musste ein Node/Express-Server erstellt werden, der die Plattform hosted, damit Dokumente vom Endgerät auf die Plattform hochgeladen werden können. Grund hierfür ist, dass Webapplikationen nicht direkt auf Dokumente, die auf der Festplatte des Endgeräts liegen, zugreifen dürfen, um die Sicherheit des Gerätes zu erhöhen. Somit gibt es nun einen Server, der zwischen der Webapplikation und den Dokumenten auf dem Endgerät des Nutzers steht.

Außerdem wurden die Header- und Footer-Leiste statisch gesetzt, sodass diese immer am oberen und unteren Bildschirmrand angezeigt werden, wenn der Content die Seite nicht ganz füllt. Ansonsten befinden sich Header und Footer, wie gewohnt, ober- und unterhalb des Contents der jeweiligen Seite. Die Unterseiten werden in die Index dynamisch per AJAX (Asynchronous JavaScript and XML) geladen.

Da jede Seite ihre eigene Dynamik verlangt, gibt es für jede Seite eine eigene JS-Datei. So behält man auch leichter den Überblick, sollte etwas geändert werden. Eine Seitenübergreifende JS-Datei kümmert sich hierbei um den Austausch des Contents.

Die Daten, die mit TEXTALYSE selektiert werden sollen, werden per API-Aufruf zur Verfügung gestellt und können so in die Seite hinein geladen werden.

### 3.4 Einbinden der Daten mit JavaScript

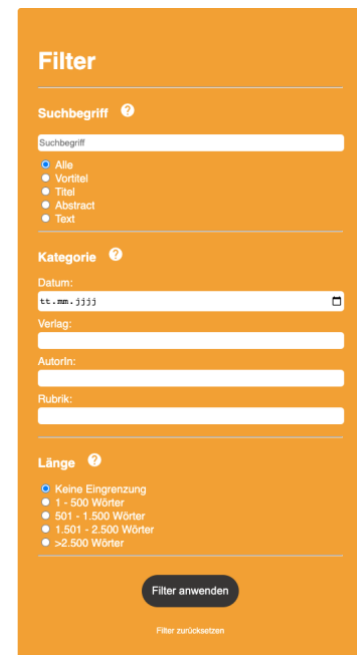
Die ausführliche Erklärung zum Code und dem Einbinden neuer Daten wurde in dem Dokument "Wie erstelle ich neue Features" und dem [Erklärvideo "Technische Einführung in TEXTALYSE"](#) dargestellt.

## 4. Sprint 2

### 4.1 Anforderungen nach Sprint 1 Review

Um die Unterteilung der Datenmenge zu erleichtern und lange Ladezeiten zu vermeiden, sollten Filter-Buttons erstellt werden. Die Filterfunktion beinhaltet folgende Elemente:

- Einen Suchbegriff: Hier kann das gesuchte Wort eingegeben werden. Ebenfalls kann der Nutzer sich entscheiden, in welchem Textabschnitt sich das Wort befinden soll.
- Die Kategorie: Hier kann die Suche noch spezifischer gestaltet werden und nach dem Datum des Textes, dem Verlag, dem Autor / der Autorin und der Rubrik (z.B. Wirtschaft) gesucht werden.
- Die Länge des Textes: Sucht der Nutzer nach einem Text mit einer gewissen Länge kann er hier seine Auswahl treffen.



The image shows a mobile app filter interface with an orange background. It is titled 'Filter' at the top. Below the title, there are three main sections: 'Suchbegriff', 'Kategorie', and 'Länge'. Each section has a search input field and a list of options. The 'Suchbegriff' section has a search bar and a list of options: 'Alle', 'Voritel', 'Titel', 'Abstract', and 'Text'. The 'Kategorie' section has a search bar and a list of options: 'Datum', 'Verlag', 'Autorin', and 'Rubrik'. The 'Länge' section has a search bar and a list of options: 'Keine Eingrenzung', '1 - 500 Wörter', '501 - 1.500 Wörter', '1.501 - 2.500 Wörter', and '>2.500 Wörter'. At the bottom of the filter interface, there are two buttons: 'Filter anwenden' and 'Filter zurücksetzen'.

Nach dem ersten Sprint, sah unsere Anwendung dem Prototypen schon sehr ähnlich, jedoch war sie noch nicht interaktiv. Dies wurde durch den entsprechenden JavaScript-Code im zweiten Sprint erledigt, sodass man nicht mehr manuell auf andere Seiten/Funktionen gelangt.

Eine Webanwendung, die Texte segmentieren und annotieren kann, macht natürlich nur Sinn, wenn die Ergebnisse anschließend entsprechend gespeichert werden können. So hat der Nutzer auch im Nachhinein noch die Möglichkeit, sich seine Projekte anzusehen. Da die Webanwendung nicht im Internet gehostet werden kann, entschlossen wir uns dazu, dass die Projekte ausgedruckt werden können. Über das Dialogfeld "Drucken" kann das Projekt ebenfalls als PDF auf dem Computer des Nutzers gespeichert werden.

### 4.2 Segmentier-Funktion einbauen

Mit der Segmentier-Funktion können Texte aus den JS-Objekten in Sätze geteilt werden und anschließend mit farbigen Tags versehen werden.

Die entsprechende Javascript-Funktion erzeugt für jeden Satz einen eigenen p-Tag. Durch das Anklicken des jeweiligen p-Tags, also des Satzes, wird abhängig vom aktuellen Zustand entweder eine Klasse mit der ausgewählten Farbe hinzugefügt oder die bestehende Farbe entfernt.

## **4.3 HTML, CSS und JavaScript**

### 4.3.1 Optimierung und Vereinheitlichung

Für ein einheitliches Gesamtbild der Website und eine ordentliche Struktur des Codes, wurden HTML und CSS optimiert und vereinheitlicht. Dazu wurde in einer globalen CSS-Datei beispielsweise die Schriftart "Helvetica" festgelegt, die für alle Seiten übernommen wird.

### 4.3.2 CSS-Effekte einbauen

Für eine bessere Führung der Aufmerksamkeit des Nutzers auf der Website, wurden zusätzlich CSS-Effekte eingebaut. Buttons wurden mit Hover-Effekten ausgestattet, die die Färbung des Buttons von einem dunklen Grau auf ein Orange wechseln, sobald mit dem Mauszeiger darüber gegangen wird. Der Call-to-Action Button auf der Startseite "Starte jetzt", der zum sofortigen Beginn mit dem Programm aufruft, wurde direkt blau gefärbt, um die Aufmerksamkeit auf ihn zu ziehen. Ein Farbwechsel ist daher nicht nötig. Geht man mit dem Mauszeiger auf ihn, wird dieser jedoch vergrößert, um dem Nutzer eine direkte Rückmeldung zu geben, dass dieser funktioniert und etwas passiert. Das gleiche geschieht mit den Informationskästen auf der Start- und Hilfeseite, damit dem Nutzer vermittelt wird, dass eine Aktion ausgeführt werden kann.

## 5. Sprint 3

### 5.1 Anforderungen nach Sprint 2 Review

Nach der Präsentation unseres Zwischenstandes, wurde uns durch unseren Stakeholder weitere Funktionen in Auftrag gegeben, die am Ende in der Webanwendung vorhanden sein sollten. Folgende Funktionen wurden gewünscht:

- Die bearbeiteten Dateien sollen nicht nur als PDF speicherbar sein, sondern auch als XML-Datei.
- Die Bearbeitung der Daten soll direkt in der Datenbank erfolgen und auch dort gespeichert werden.
- Eine Dokumentation, in der beschrieben wird, wie neue Features optimal eingebunden werden können.
- In der Filterfunktion sollen mehrere Optionen ausgewählt werden können.
- Die Webanwendung sollte am Ende über einen Server gehostet werden, sodass jeder einen Zugriff darauf hat.

Nach einem ausführlichen Gruppenmeeting sind wir zu dem Entschluss gekommen, dass die ersten beiden Funktionen in der kurzen Zeit des Sprints (1,5 Wochen) und unserem bisherigen Können und Wissen leider nicht umsetzbar sind. Unsere Konzentration liegt in diesem letzten Sprint nun vielmehr auf der Feinarbeit und kleineren Änderungen. Besonders die Elemente, die die Bedienung vereinfachen bzw. erläutern wurden in diesem Schritt erstellt und eingebaut.

### 5.2 Dokumentation für neue Features

Eine weitere Anforderung des Kunden war es, eine Dokumentation zu erhalten, welche alle Schritte erklärt, wie neue Features in die bereits bestehende Applikation eingefügt werden können. Hierfür gibt es neben einer Dokumentation, die den Code beschreibt auch ein aufgenommenes Erklärvideo, welches Schritt für Schritt die Implementierung einer neuen Seite erklärt.

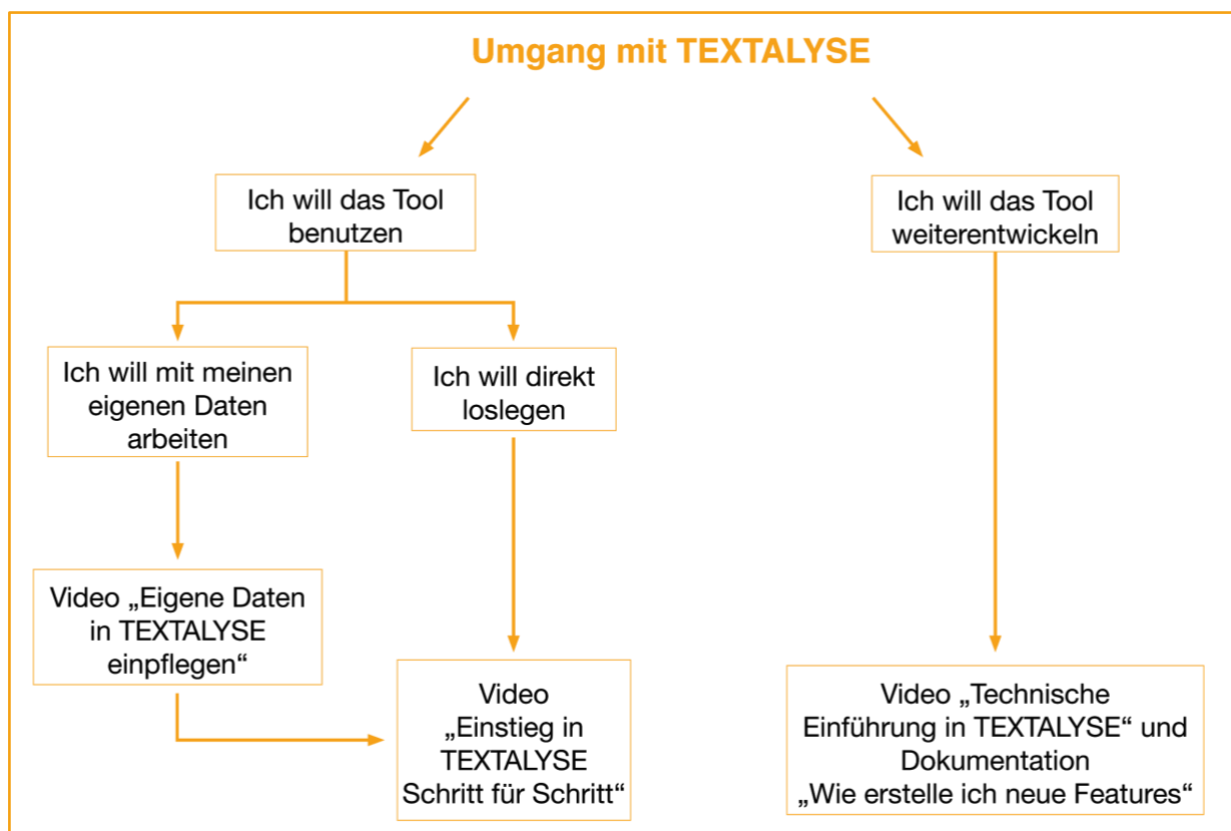
### 5.3 Tooltips

Zur direkten Hilfe entschieden wir uns dazu sogenannte Tooltips an verschiedenen Stellen der Seite, die potenziell schwer verstehbar sind, einzubauen. Hinter den weiß hinterlegten runden Fragezeichen-Symbolen sind die Tooltips hinterlegt. Hovort man über das Fragezeichen bekommt man nun eine kurze Erklärung was die Funktion macht und wie sie zu benutzen ist.



### 5.4 Hilfe Videos

Zur benutzerfreundlichen Erklärung der Plattform wählten wir zusätzlich zu den verschriftlichten Erklären auch noch das Video Format. Hierbei wurden unsere Inhalte in drei verschiedene Videos aufgeteilt. Zu welchem Wissensstand welches Video ideal ist, zeigt unser Flussdiagramm, welches zusätzlich auf unserer Hilfeseite eingebunden ist.



## 6. Fazit

Am Anfang unseres Projektes, waren wir uns sehr unsicher bezüglich des Themas. Nach ersten Absprachen mit unserem Stakeholder, war uns allerdings klar, was zu tun ist. Auch die Scrum-Methode war anfangs sehr ungewohnt für uns. Das Arbeiten mit dieser Methode fiel uns von Tag zu Tag leichter.

Dank des tollen Gruppenklimas hatten wir über die Zeit sehr viel Spaß an dem Projekt und hauptsächlich an unserer Zusammenarbeit. Was man leider sagen muss ist, dass der Projektumfang unserem Können überhaupt nicht angemessen war. Eine Webanwendung mit den gegebenen Funktionen zu programmieren ist sehr komplex und kann nicht von Studenten im 2. Semester bewältigt werden. Ohne Dominik und auch der Hilfe seines Vaters, wäre das Ergebnis lange nicht so zufriedenstellend, wie es nun im Endeffekt ist. Wir würden uns wünschen, dass die Projekte in Zukunft dem Können der Studierenden angepasst werden. Andernfalls erlebt man von Woche zu Woche nur Enttäuschungen, da nichts klappt und verliert schnell die Motivation an dem Projekt.

Ebenfalls ist uns die Kommunikation und die Beratung durch unsere Scrum-Expertin als negativer Punkt aufgefallen. Wir hätten uns hier mehr Beratung und Hilfestellung gewünscht und eventuell auch ein paar Tipps, inwiefern man den Stakeholder bei Scrum mit einbezieht oder nicht.

In der folgenden Grafik wird die Verteilung der Arbeitszeiten an dem Projekt in die einzelnen Kategorien eingeteilt. Auffällig ist hier, dass Gruppenbesprechungen und die Programmierung ca. 80% der Projektzeit in Anspruch genommen haben. Daraus resultiert, dass der größte Teil des Projektes in der Programmierung und Entwicklung lag, da auch in den Gruppenbesprechungen gemeinsam programmiert und entwickelt wurde.

