THM | CAMPUS GIESSEN | MNI
TECHNISCHE HOCHSCHULE MITTELHESSEN
Mathematik, Naturwissenschaften und Informatik

Bericht

Softwaretechnik- Praktikum (SWT-P)
(CS1023)

SS 2020

Thema:
Data Analytics

Teammitglieder:

Timon Pellekoorne (Projektleiter) – timon.pellekoorne@mni.thm.de

Tanja Gutsche - tanja.gutsche@mni.thm.de

Jannik Lapp - jannik.lapp@mni.thm.de

David Martschenko - david.martschenko@mni.thm.de

Lisa Soboth - lisa.soboth@mni.thm.de

Max Stephan - max.stephan@mni.thm.de

Inhaltsverzeichnis

Zeitmanagement

Das Projekt findet vom 23.04. – 03.08.202 statt.

Inspiriert von der Methode „Scrum“ wird dieser Zeitraum in einwöchige Sprints unterteilt, wobei die ersten eineinhalb Wochen die Planungsphase darstellen.

Ab Montag dem 04.05.2020 beginnen die wöchentlichen Sprints. Dazu findet jeden Montag ein *Sprint-Planning* statt, in welchem alle Teammitglieder über ihre Arbeit in dem vergangenen Sprint berichten, neue *Issues* erstellt, und eventuelle neue Herausforderungen besprochen werden.

Hinzu kommen die wöchentlichen Meetings mit dem Auftraggeber. Da dieses Meeting mitten in einem Sprint liegt, kann der Projektleiter dem Auftraggeber eine gute Übersicht über den zuletzt abgeschlossenen Sprint, sowie eine Zwischenbilanz des aktuellen Sprints, geben.

In der „Abbildung 1“ wird die zeitliche Aufteilung des Projektzeitraums grafisch dargestellt. Die rot markierten Linien stellen besondere Termine dar:

1. Lastenheft
2. Pflichtenheft
3. Zwischenpräsentation
4. Abschlusspräsentation

Wochentag	Datum		
Donnerstag	23.04.2020	Woche 1	Planung
Montag	27.04.2020		
Donnerstag	30.05.2020	Woche 2	
Montag	04.05.2020		Sprint 1
Donnerstag	07.05.2020	Woche 3	
Montag	11.05.2020		Sprint 2
Donnerstag	14.05.2020	Woche 4	
Montag	18.05.2020		Sprint 3
Donnerstag	21.05.2020	Woche 5	
Montag	25.05.2020		Sprint 4
Donnerstag	28.05.2020	Woche 6	
Montag	01.06.2020		Sprint 5
Donnerstag	04.06.2020	Woche 7	
Montag	08.06.2020		Sprint 6
Donnerstag	11.06.2020	Woche 8	
Montag	15.06.2020		Sprint 7
Donnerstag	18.06.2020	Woche 9	
Montag	22.06.2020		Sprint 8
Donnerstag	25.06.2020	Woche 10	
Montag	29.06.2020		Sprint 9
Donnerstag	02.07.2020	Woche 11	
Montag	06.07.2020		Sprint 10
Donnerstag	09.07.2020	Woche 12	
Montag	13.07.2020		Sprint 11
Donnerstag	16.07.2020	Woche 13	
Montag	20.07.2020		Sprint 12
Donnerstag	23.07.2020	Woche 14	
Montag	27.07.2020		Sprint 13
Donnerstag	30.07.2020	Woche 15	
Montag	03.08.2020		

Abbildung 1

Versionsverwaltung: Git und Github

Um das Projekt ordentlich zu strukturieren, benutzen wir Github. Dies dient uns zum einen als Versionsverwaltung, da Git dort integriert ist, sowie auch zum managen unseres Projektes.

Um unser Zeitmanagement in Github umzusetzen, wird in jedem *Sprint-Planning* ein neuer *milestone* mit dem Namen des Sprints (z.B. „Sprint 2“) und dem Ende des Sprints (darauffolgender Montag) angelegt. Danach werden ebenfalls im *Sprint-Planning Issues* gemeinsam mit allen Teammitgliedern erstellt und dem passenden *milestone* zugewiesen. Diese *Issues* werden den Teammitgliedern zugeordnet, welche sie bearbeiten sollen.

Ist das *Sprint-Planning* abgeschlossen stehen alle erstellten *Issues* im *Project Board* in der Spalte „To-Do“. (siehe Abbildung 2)

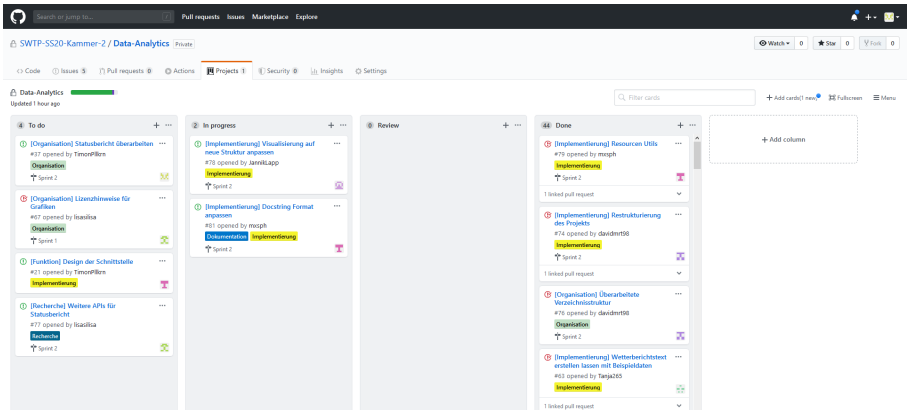


Abbildung 2

Wird nun ein *Issue* in die Spalte „In Progress“ gezogen, so wird automatisch ein „Branch“ erstellt, welcher den Namen des zugehörigen *Issues* trägt.

Hat ein Teammitglied den *Issue* erledigt, so zieht er diesen in die Spalte „Review“. Nun wird automatisch ein *Pull Request* erstellt. Ebenfalls werden alle Unit-Tests, die Github finden kann automatisch ausgeführt. Verlaufen diese fehlerfrei, so wird dieses dem Entwickler angezeigt.

Die *Pull Request* wird dann einem anderen Teammitglied zum reviewen zugeteilt. Diese überprüft noch einmal manuell den Code. Wenn alles in Ordnung ist, wird der Code *gemerged*, also dem Hauptcode hinzugefügt.

Statusberichte

Statusbericht 05.05.2020 (Planung)

Zusammenfassung

Das Ziel der Planungsphase war es, das Projekt zu strukturieren. Dies bedeutete konkret:

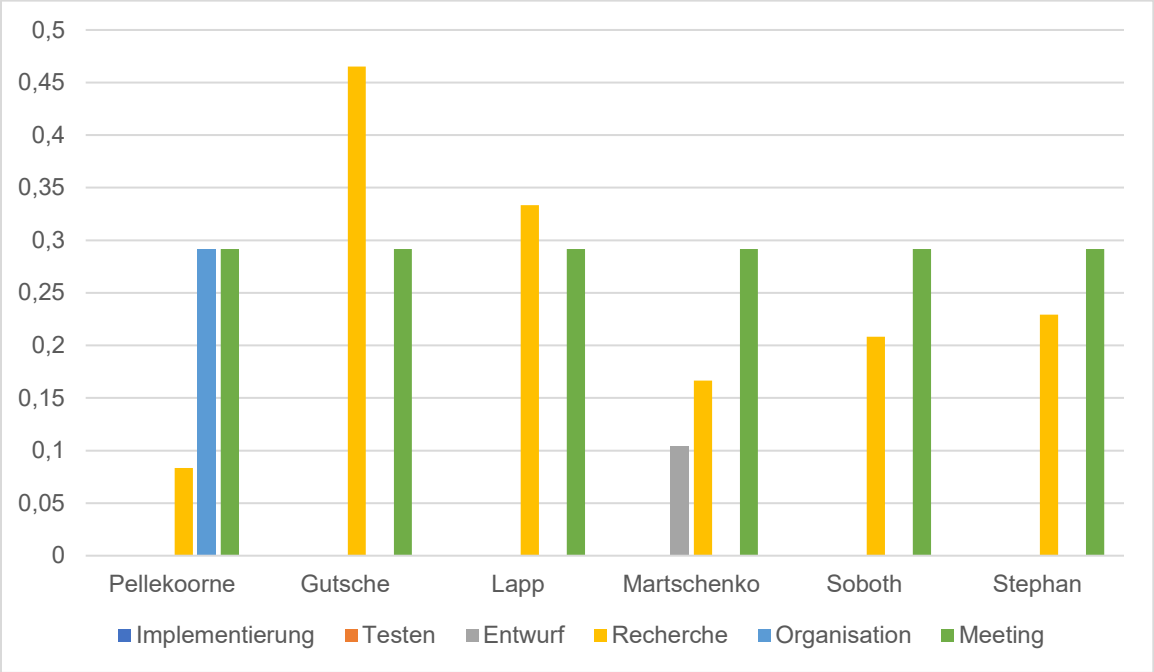
- **Zeitmanagement ausarbeiten**
  Dazu wurde der vorgegebene Zeitraum so aufgeteilt, dass ein möglichst effektives Arbeiten ermöglicht wird. Die genauere Erläuterung dieser Struktur finden sie hier.
- **Festlegen der einzusetzenden Technologien**
  Jedes Teammitglied hat recherchiert, welche Technologien für dieses Projekt zum Einsatz kommen könnten und wie diese zu Verwenden sind. Bei einem Meeting wurden diese dann dem restlichen Team vorgestellt und sich auf einige festgelegt. Welche Technologien in diesem Projekt verwendet werden, finden sie in dem Pflichtenheft.
- **Lastenheft erstellen**
  Das Lastenheft fasst die Anforderungen an das Programm zusammen. Dazu haben wir ein grobes Konzept erstellt, welches einen Überblick über die Funktionen des Programmes darstellen soll. (zum Lastenheft)

Ziel für Sprint 1

Das Ziel für den Sprint 1 ist es, die Ergebnisse aus der Planungsphase so aufzubereiten, dass die Implementierung beginnen kann.

- Verzeichnisstruktur des Programmcodes erstellen
- Einen Style Code erstellen um einheitlich zu arbeiten
- Vorgaben für Tests herausarbeiten

Arbeitszeiten



Statusbericht 12.05.2020 (Sprint 1)

Zusammenfassung

Die Ziele, die für diesen Sprint gesetzt wurden, wurden vom Team schon nach paar Tagen erreicht. So konnte schon in diesem Sprint mit der Implementierung begonnen werden.

Um mit der Implementierung zu beginnen, wird zuerst die Wetter API benutzt, um anhand dieser, einen kompletten durchlauf des Programmes umzusetzen.

Außerdem wurde sich mit den Lizenzen beschäftigt, um Wetter Icons oder andere Grafiken legal nutzen zu können.

Aktueller Stand der Implementierung

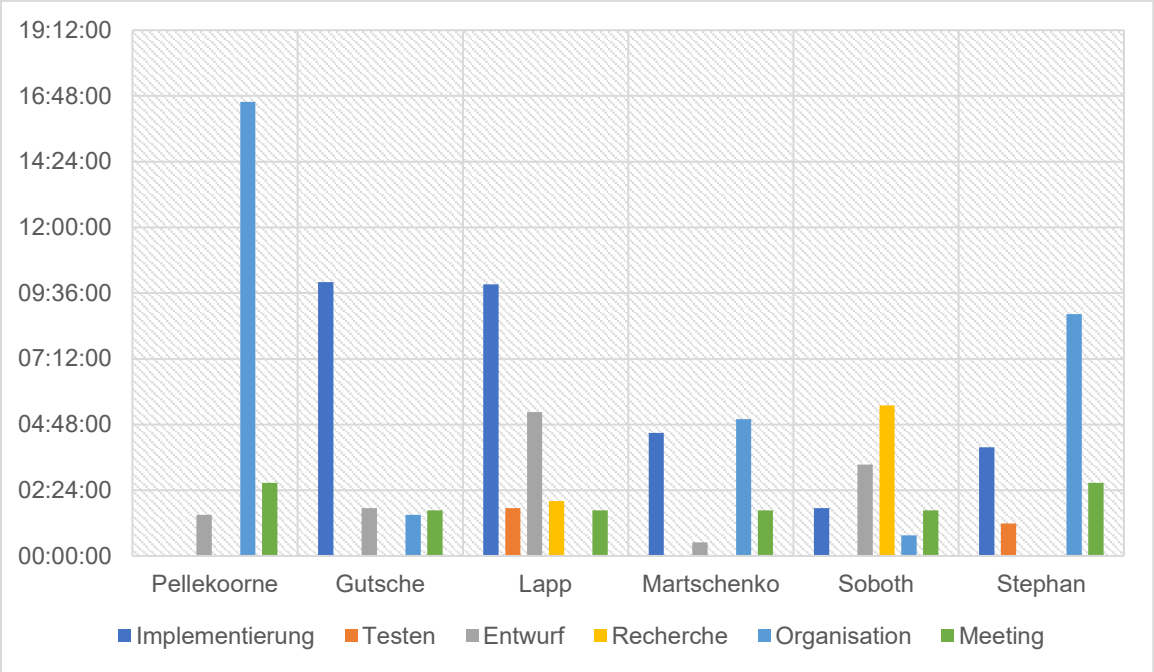
Die Grundstruktur des Servers ist aufgesetzt.

Die Wetterdaten der API werden ausgelesen und in einem „Dictionary“ abgelegt.

„Lückentexte“ erstellt, um die Daten aus der API in diese einzufügen und einen Fließtext daraus zu bekommen. Mithilfe eines „Dictionary“ Mehrere Sätze pro Stadt oder ähnlichem angelegt, um variable Texte zu generieren.

Wetterdaten sowie Icons der API über die erstellten Grafiken legen, um daraus ein Video zusammen zu schneiden.

Arbeitszeiten



Lastenheft

Zielbestimmung

Das Programm soll verschiedene Informationen aus dem Internet erfassen und automatisiert verarbeiten. Dazu soll am Ende ein Video, eine Grafik etc. erstellt werden, um die erfassten Informationen übersichtlich und verständlich zu präsentieren.

Dazu sollen dem Benutzer mehrere Themenbereiche zur Verfügung stehen:

Beispiele:

- Wetter
  - Wettervorhersage für den nächsten Tag und die drei darauffolgenden
  - Aktuelles Wetter
  - Aktuelle Luftqualität
- Corona-Daten
  - Zahl der Infizierten, Genesenen, Toten weltweit und Deutschlandweit
- Historische Ereignisse
  - Nach ausgewähltem Datum wird eine Wordcloud erstellt, mit den Begriffen, welche am wichtigsten zu dem Zeitpunkt waren.

Must-Have/Nice-to-Have/If-Time

Must-Have

- Das Programm soll in der Lage sein, aus mindestens 4 Themengebieten Informationen dem Benutzer visualisiert darzustellen.
- Diese Ausgaben soll in Videoform, mit Hilfe einer Wordcloud, Diagrammen und Grafiken wiedergegeben werden. Die ausgegebenen Videos sollen eine Audiospur enthalten, welche die visualisierten Daten verständlich erläutert.
- Das Programm soll in die Website https://biebertal.mach-mit.tv/ integriert werden. Alle Besucher dieser Seite sollen Zugriff auf die Videos haben.
- Der Administrator der Website soll einen Zyklus einstellen können, in dem das Programm automatisch die Ausgabe in diesem Zyklus erstellt.

Nice-to-Have

- Die Einbindung des Programmes in die Website per Wordpress-Plugin realisieren.
- Es kann die Möglichkeit bestehen, Videos zu generieren, welche die Daten aus mehreren Themenbereichen zusammenführt.
- Dem Benutzer können Eingaben zur Verfügung stehen, welche die Ausgabe präzisieren.
- mindestens 7 Themenbereiche für den Benutzer zur Auswahl
- Der Benutzer kann das Video downloaden oder Teile davon. (Wordcloud, Diagramme, Grafiken)

If-Time-Allows

- Audiospur und Visualisierungen auch auf Englisch bereitstellen.
- 10 T Themenbereiche für den Benutzer zur Auswahl

Produkteinsatz

Das Produkt soll auf der Internetseite https://biebertal.mach-mit.tv/ öffentlich zur Verfügung stehen. Dort soll jeder auf die generierten Videos zugreifen können.

Eine genaue Zielgruppe wird bei der Herstellung nicht berücksichtigt, die Themengebiete werden nach Interesse der breiten Masse ausgesucht.

Produktübersicht

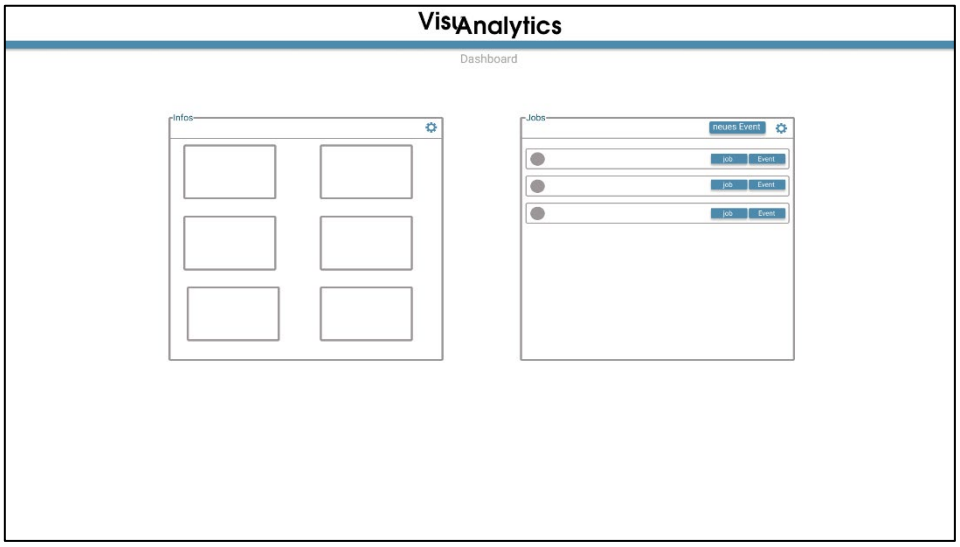
Benutzeroberfläche

Begriffserklärung:

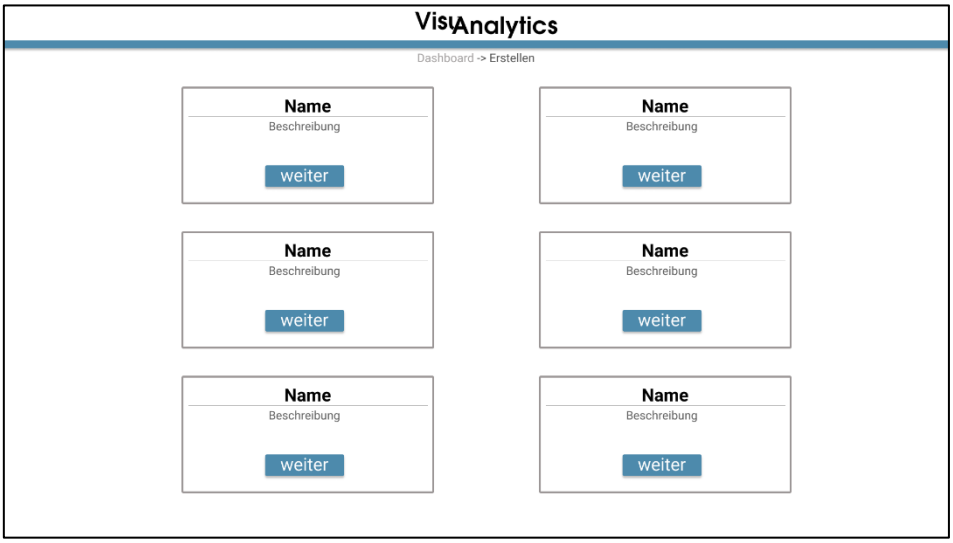
Job: Ein Job besteht aus einem Themengebiet und eventuellen präziseren Auswahlmöglichkeiten, welche das Output-Video einschränken.

Event: Ein Event besteht aus einem oder mehreren Jobs. Events können angelegt werden, um einen Job einmalig oder in einem bestimmten Zyklus auszuführen.

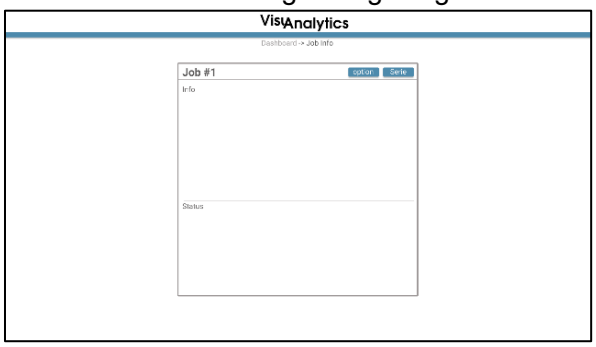
1. Zum Beginn des Programms befindet sich der Benutzer auf seinem Dashboard. Dort findet er zum einen allgemeine Informationen und zum anderen eine Übersicht über seine Jobs.

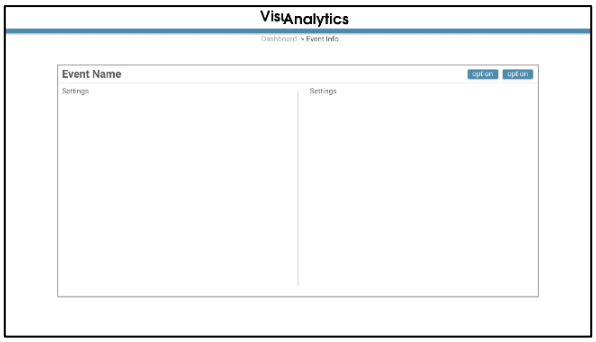


2. Dort kann der Benutzer auch einen neues „Event“ anlegen. Wählt er diese Option aus, so wird ihm eine Auswahl aller Themen gegeben. Zuerst erstellt der Benutzer einen Job zu einem der angezeigten Themengebiete. Danach legt der Benutzer ein Event an, in welchem er festlegt, wie häufig der vorher angelegte Job ausgeführt werden soll.



Will der Benutzer genauere Informationen zu einem einzelnem „Job“ eines Events, so kann er dieses sich durch klicken auf den gewünschten Job ansehen. Klickt er bei einem „Job“ auf „Event“, so werden ihm alle „Jobs“ die mit dem ausgewählten Event zusammenhängen angezeigt.

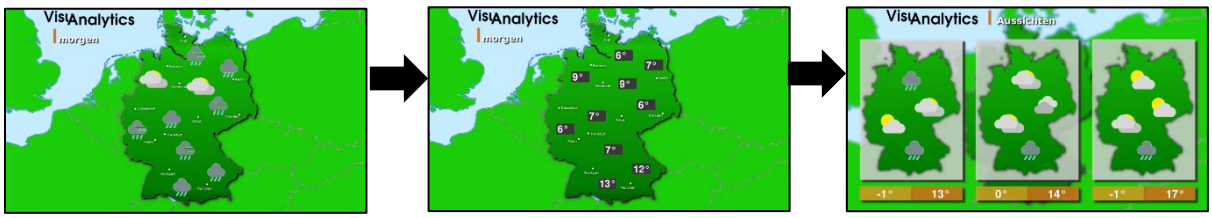




Ausgabe am Beispiel Wetterbericht:

Aus dem Internet sollen die aktuellen Wetterdaten erfasst werden, diese zu Text und Grafiken verarbeitet werden. Aus diesen Ergebnissen soll ein Video erstellt werden, welches einen Wetterbericht für den nächsten Tag und die darauffolgenden drei Tage darstellt.

Beispiel:



Qualitätsanforderungen

Das Produkt soll ohne großen Aufwand um weitere Funktion, vor allem aber um weitere Schnittstellen erweiterbar sein.

Spezielle Anforderungen

Dem Benutzer soll es möglich sein, Einstellungen vorzunehmen, sodass in einem bestimmten Zyklus z.B. Wetterberichte erstellt werden. So kann auf der Internetseite täglich der aktuelle Wetterbericht zu finden sein, ohne dass der Administrator, dieses manuell erstellen lassen muss.

Quellen/Lizenzen

- https://de.wikipedia.org/wiki/Vorlage:Positionskarte_Europa
  (GNU-Lizenz für freie Dokumentation)
- https://commons.wikimedia.org/wiki/File:Karte_Deutschland.svg
  (Attribution-Share Alike 2.0 Germany)
- Wetter-Icons: https://www.weatherbit.io/api
  (Wetter-Icons werden durch die kostenlose API mitgeliefert)

Pflichtenheft

Zielsetzung

Das Programm soll um weitere Schnittstellen erweiterbar sein. Außerdem sollen die Informationen aus verschiedenen APIs kombiniert werden können.

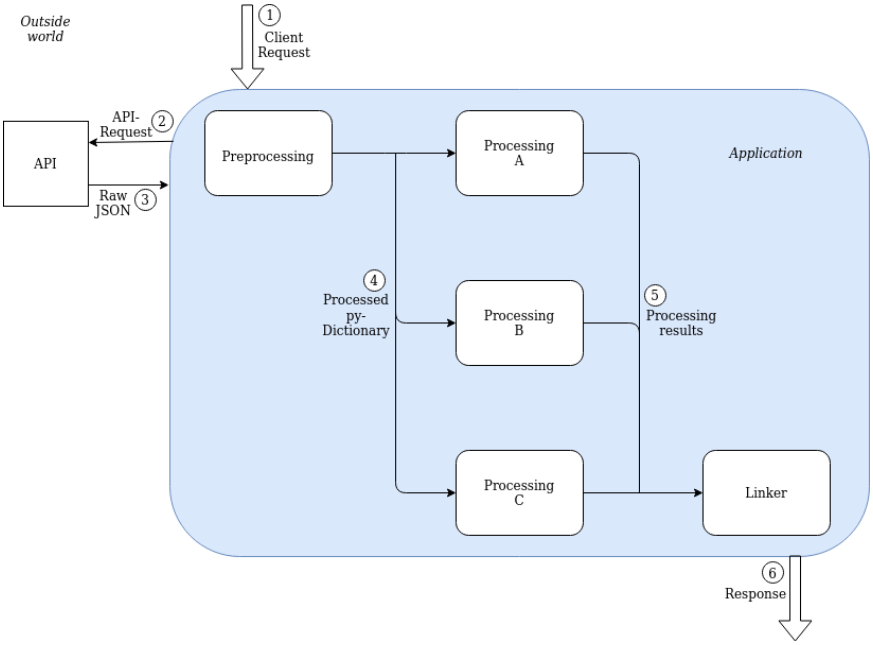
Der Administrator der Website soll einen Zyklus einstellen können, in dem das Programm automatisch die Ausgabe in diesem Zyklus erstellt. Diese Einstellungen werden in eine config-Datei geschrieben, aus der das Programm dann die Videos zum gewünschten Zeitpunkt erstellt.

Benutzerschnittstellen

Der Benutzer kann über die Website https://biebertal.mach-mit.tv/ auf das Programm zugreifen, deren Ausgaben anschauen oder auch eigene Eingaben tätigen, um die Ausgabe nach seinem Wunsch zu gestalten.

Funktionale Anforderungen

Um die Funktionalen Anforderungen des Programmes zu verdeutlichen, ist unser Programm folgedermaßen strukturiert:



1. Der Client schickt eine Anfrage an das Programm.
2. Das Programm schickt eine Anfrage über die Schnitstelle an die gewünschte API.
3. Die Antwort der API landet im „Preprocessing“. Dort werten die Daten aufbereitet, so dass jeder Prozess einheitlich darauf zugreifen kann. In unserem Fall ist dies ein Python Dictionary.
4. Die einzelnen „Processing“ Stationen entnehmen die Daten dem Dictionary und verarbeiten diese (z.B. Diagramme erstellen).
5. Die daraus resultierenden Daten werden im „Linker“ zu einem Endergebnis zusammengefügt (z.B. Video)

Benutzte APIs

Historische Ereignisse

Das Programm soll eine Wordcloud zu historischen Daten ausgeben, in der die wichtigsten Themen den größten Anteil haben.

Als Schnittstelle dafür verwenden wir die http://developer.zeit.de/index/ API. Diese enthält gesamtes Archiv der ZEIT bzw. ZEIT ONLINE.

Corona

Das Programm soll die aktuellen Daten zur Corona Pandemie ausgeben. Dazu wird die https://covid19api.com/ API verwendet. Diese liefert die aktuelle Corona Statistiken weltweit und für Deutschland.

Wetter

Anhand der Werte welche wir aus der https://www.weatherbit.io/api API beziehen, soll ein Wetterbericht als Video erstellt werden. Die solch ein Bericht kann folgende Informationen beinhalten:

- Wetter heute
- Wettervorhersage für morgen und die darauffolgenden drei Tage
- aktuelle Luftqualität für vorausgewählte Städte in Deutschland

weitere APIs

Finanzen

https://rapidapi.com/apidojo/api/yahoo-financ

- Zusammenfassung zum angefragten Zeitpunkt
- Gewinner / Verlierer in einer bestimmten Region
  z.B. Day Gainers - US, Day Losers - US, Most Actives - US
- Daten, um Diagramme zu bestimmten Akteuren zu erstellen
- Gewinne in einer bestimmten Region in einem eingegrenzten Zeitraum

Klimawandel

https://datahelpdesk.worldbank.org/knowledgebase/articles/902061-climate-data-api

- Daten des World Bank’s Climate Change Knowledge Portal
- basieren auf 15 global ciculation models (GCMs), welche die Reaktion des globalen Klimasystems auf steigende Treibhauskonzentrationen simulieren
- Niederschlag und Temperatur
- Tages-, Monats- und Jahresdurchschnitt, Veränderungsrate
- Zeitspannen in der Vergangenheit
- Zeitspannen in der Zukunft (Prognose)

Erdbeben

https://earthquake.usgs.gov/fdsnws/event/1/

- Erdbeben in einem bestimmten Zeitraum
- Erdbeben in bestimmten Höhen und Breitengraden
- Intervall von Erdbebenstärken
- Filter für bestimmte Alarmlevel: grün, gelb, orange, rot
- in Kombination anwendbar
- keine Vorhersage

Eingesetzte Technologien

Datenvisualisierung

Um die Daten zu visualisieren Benutzen wir NumPy und Matplotlib. Desweiteren kann auch Basemap eingesetzt werden, um Verteilungen anhand einer Karte darzustellen.

Für die Erstellung der Wordcloud wird die Python-Bibliothek „wordcloud verwendet.

Audio

Um aus einem generierten Text, welcher z.B. den Wetterbericht erläutert benutzen wir die Python-Bibliothek „gtts“. Diese Bibliothek kann aus einem vorgegebenen Text eine mp3 Datei erzeugen.

Bildbearbeitung

Zur Bildbearbeitung benutzen wir die Python-Bibliothek „Pillow“. Anhand dieser Bibliothek können wir vorangefertigte Grafiken mit Inhalt füllen, wie z.B. die Temperaturen oder passende Wetter Icons bei einem Wetterbericht.

Videobearbeitung

Um am Ende alle Grafiken, Diagramme etc. zu einem Video zusammenzufügen, benutzen wir das Tool „FFMPEG“. Dieses Tool gibt uns die Möglichkeit Bilder aneinander zu schneiden und mit einer Audiodatei zu unterlegen. Es kann die Anzeigelänge eines Bildes an die Länge einer Audiodatei anpassen, was uns die Möglichkeit gibt das Bild passend zur Sprachdatei auszugeben.

Nichtfunktionale Anforderungen

Das Programm soll um weitere Schnittstellen einfach erweiterbar sein, dazu muss die Schnittstelle so designt sein, dass das Hinzufügen einer weiteren ohne großen Aufwand von statten geht.

Abnahmekriterien

Das Programm muss fehlerfreie Ausgaben zu den zur Auswahl stehenden Themen liefern.