Projektbericht – „Video Summary Creator“

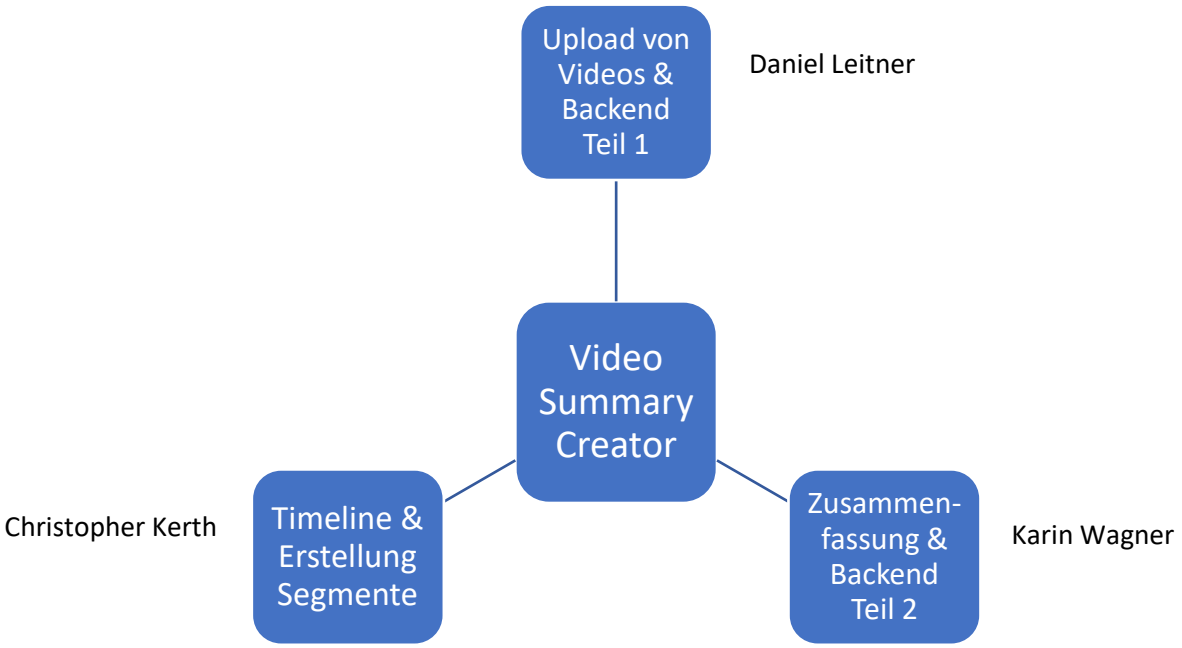
Allgemeines

Der „Video Summary Creator“ ist eine webbasierte Lösung zur Erstellung einer Zusammenfassung aus Segmenten bzw. Teilen eines Videos. Konkret sind dazu folgende Schritte erforderlich:

1. Upload des gewünschten Videos
2. Erstellung von Segmenten aus dem Videos mittels Timeline
3. Anpassung von Eigenschaften der Segmente (Reihenfolge, Beschreibungstexte, …)
4. Erstellung einer Zusammenfassung aus den Segmenten

Arbeitsteilung

Der Aufwand wurde wie in der folgenden Abbildung ersichtlich aufgeteilt:



Es folgen Beschreibungen zu den einzelnen Teilen u. a. auch aufgetretene Probleme und Lösungsansätze.

Upload von Videos & Backend Teil 1

Um das Video überhaupt erst anzeigen zu können, muss es auf den Webserver hochgeladen werden. Dafür verwenden wir Node js. Als erstes wählt man ein Video aus dem lokalen Dateisystem aus und drückt anschließend auf den Button „Upload“. Über den Socket, der sich bereits zu Beginn zum Server verbindet, wird das Video aufgeteilt in Megabyte-Blöcke hochgeladen. Der Fortschritt des Uploads wird textuell dargestellt. Die am Server ankommenden Blöcke werden zu einem File zusammengefügt und in den Unterordner „Temp“ zwischengespeichert. Nachdem das Video vollständig hochgeladen wurde, wird das Video aus dem Ordner „Temp“ gelöscht und in den Ordner „Video“ verschoben. Von da aus wird es als Source für das versteckte Video-Element auf der Website verwendet. Ab diesem Zeitpunkt kann man mit der Erstellung der Segmente beginnen.

Die Probleme meinerseits beschränkten sich auf die korrekte Verwendung von Node js. Es dauerte eine Weile bis ich den Server zum Laufen gebracht habe, da ich Node js davor noch nie benutzte. Der Zugriff auf Unterordner bereiteten mir unter Anderem Probleme. Das temporäre Video wurde aus dem „Temp“ Ordner nicht sofort gelöscht, sondern erst nachdem der Server heruntergefahren

wurde. Durch das Schließen des noch offenen `FileStreams` konnte ich das Problem beheben. Des Weiteren gab es Probleme, wenn das gleiche Video nochmal hochgeladen wurde, weil es bereits im Ordner existierte. Dies wurde von mir gelöst, indem ich vor dem Upload überprüfe, ob in den Ordnern „Temp“ bzw. „Video“ bereits so ein File existiert und es ggf. lösche. Auch die Segmente, welche im Ordner „Video“ gespeichert werden, werden vor jedem Upload gelöscht.

Timeline & Erstellung Segmente

Es wurde ein GUI zur Anzeige und Auswahl von Frames des hochgeladenen Videos erstellt. Diese als Timeline bezeichnete GUI ermöglicht die sekundenweise Auswahl von Frames des Videos. Alternativ könnte jeder Frame des Videos (z. B. 25 je Sekunde) einzeln in der Timeline dargestellt und auswählbar gemacht werden. Der Grund für die sekundenweise Anzeige der Frames ist jener, dass man so zwar weniger Frames auswählen kann, dafür jedoch schneller Unterschiede zwischen benachbarten Frames erkennbar sind. Somit ist eine schnellere Suche gewünschter Szenen mit geringerem Aufwand möglich. Besagte Szenen werden als Segmente bezeichnet. Sie enthalten eine ID (Identifikation von Segmenten), eine textuelle Beschreibung und die korrespondierenden Frames inkl. Anfangs- und Endzeit. Bei jedem Klick auf den aktuell dargestellten Frame wird dieser zwischengespeichert. Bei zwei zwischengespeicherten Frames bzw. nach jedem zweiten Klick wird aus dem zuvor und dem aktuell gespeicherten Frame ein Segment gebildet und der im unteren Bereich der GUI dargestellten Liste von Segmenten hinzugefügt. Bei der Darstellung von Segmenten, genauer dessen Frames, gab es ein Problem. Die Frames werden als `Image`-Objekte innerhalb der Segmente gezeichnet. Bei „schneller“ Auswahl zweier Frames (z. B. via Doppelklick) wurden diese Frames nur als weißes (leeres) Quadrat gezeichnet. Der Grund hierfür war, dass die zu den Frames erstellten `Image`-Objekte vor dem Zeichnen nicht fertig geladen worden waren. Abhilfe wurde hier durch Auslagerung der Zeichenlogik in einen `onload`-Callback der `Image`-Objekte erzielt. Innerhalb der Liste von Segmenten kann die Reihenfolge zwischen Segmenten vertauscht und textuelle Beschreibungen ergänzt werden. Ferner wurde eine maximale Anzahl von Segmenten definiert um die Einhaltung des beschränkten Wertebereiches für die ID der Segmente sicherstellen zu können. Durch die maximale Anzahl ist sichergestellt, dass es nie zwei Segmente mit derselben ID geben kann.

Zusammenfassung & Backend Teil 2

Nachdem alle Segmente ausgewählt wurden, wird die Start- und Endzeit der einzelnen Segmente und die Beschreibung des Users an den Server übergeben. Dieser parsed den String und erhält somit alle wichtigen Informationen, die notwendig sind, um im ersten Schritt mithilfe von fluent-ffmpeg die Segmente zu erstellen. Dabei sieht der String der übergeben wird beispielsweise so aus:

-ss 00:00:33 -to 00:00:47 -txt Waking up...;-ss 00:03:10 -to 00:03:22 -txt Butterfly....;-ss 00:06:32 -to 00:08:56 -txt Forest;

Fluent-ffmpeg ist ein node.js Modul, dass die Verwendung von ffmpeg erleichtert. Wichtig für die Verwendung ist, dass ffmpeg bereits am System installiert ist. Zu Beginn wird ein ffmpeg command erstellt, zu dem eine Vielzahl an Inputs und Optionen hinzugefügt werden können. Die Verwendung von fluent-ffmpeg stellte sich als äußerst hilfreich heraus, und auch das Einbetten der Beschreibung konnte mittels Hinzufügen eines `complexFilter` durchgeführt werden. Im zweiten Schritt werden alle zuvor erstellten Segmente (inkl. Beschreibung) zu einem Video zusammengefügt. Dies erfolgt ebenfalls mittels fluent-ffmpeg. Danach wird das Video in einem neuen Tab angezeigt und als Download zur Verfügung gestellt.

Probleme gab es, wenn der User Segmente mit einer Dauer von < 1 Sekunde ausgewählt hat. Die Start- und Endzeit muss variieren. Dies wurde behoben, indem nur Segmente mit einer Länge >= 1 Sekunde erlaubt sind. Da bei großen Videos das Erstellen des Summary-Videos einige Minuten dauern kann und der User kein Feedback zum aktuellen Fortschritt erhalten hat, wurde am Ende eine

Progress Bar hinzugefügt, die den Fortschritt nach jedem erstellten Segment anzeigt. Im Falles eines Fehlers beim Erstellen der Zusammenfassung wird der User ebenfalls informiert.

Fazit

Die Erstellung des „Video Summary Creator“ stellte sich als ein interessantes Projekt heraus. Hierbei konnten wir unser Wissen bzgl. Node.js bzw. fluent-ffmpeg etc. erweitern. Auch die Arbeit im Team funktionierte sehr gut. Zu Beginn wurden die einzelnen Arbeitspakete aufgeteilt und jeder fokussierte sich hauptsächlich auf seinen Teil, jedoch gab es bei Problemen und notwendigen Tests jederzeit Unterstützung von den anderen Teammitgliedern. Die Dokumentation/User Guide wurden ebenfalls im Team erstellt.