Networked Multimedia Systems

Project report

VideoCloud

Edgar Passos, Gustav Janér

August 19, 2019

Contents

1 Introduction

VideoCloud [1] is a video streaming platform where video comments are associated with a timestamp and are shown as the video plays. This allows a user to see other's remarks on particular moments while watching the video. The user also has a view of all the comments of the video and can click on a particular comment to get to that timestamp of the video.

2 Problem Statement

When watching a video on any popular video streaming platform, there might be some interesting details that we miss, a detail in the background of a film scene or foreshadowing of other action at some point in the video.

The standard solution to the use case of commenting on a specific time of a video, is just to add the video timestamp to a normal comment that is simply displayed under the video in a comment feed. This has the limitation that a user cannot watch the video and read comments simultaneously. The user has to stop watching the video in order to scroll down to read the comments.

3 Proposed Solution

A video streaming web application, where the comments are not only shown in the comment feed, but are also displayed below the video, in a video strip.

The application's interface is similar to popular video streaming websites such as YouTube [2], with a video feed acting as the main page, and a video playback page where the video is played and comments on the video are displayed.

In our solution, the comments are also shown overlaid on a set of screenshots of the video track, which doubles as a seeking bar, allowing the user to have an idea of the section of the video where the comment will appear, and also to visually seek to a certain time in the video.

This solution is inspired by Soundcloud's [3] user interface, where the comments made on an audio track are shown overlaid on the visual representation of the track.

3.1 Features

The web application provides the following features to its users:

3.1.1 Registration and Authentication

Users register to the website using an email and a password which are later used to log in. An authenticated user can upload and comment on videos, and like other users'

comments.

Unauthenticated users can watch and export videos, but not upload new videos, comment on them or like comments.

3.1.2 Video Upload

Authenticated users can upload videos to the website that are publicly viewable for other users.

3.1.3 Video Playback

All users can watch the videos that have been uploaded to the platform. Users should also be able to select the video playback quality, if there is more than one available.

3.1.4 Video Export

All users can download the video in the desired quality together with its comments as a subtitle file, allowing users to watch the video with the comments offline.

3.1.5 Video Commenting

Authenticated users can comment on videos, and like or unlike comments made by other users.

All comments are associated with a timestamp which is the time in the video at which the comment was made. This timestamp is displayed alongside the comment, and clicking on it will make the video seek to that timestamp.

All comments are shown in a comment feed under the video. A video strip under the video displays a set of screenshots from the video in chronological order, over which are overlaid icons indicating that a comment was made at that point in the video. Hovering over the comment icon displays the comment.

Some comments are also shown below the video strip when the video playback reaches the timestamp at which the comment was made.

4 Technologies

Figure 1 displays a high-level representation of the architecture of the platform and some of the technologies used.

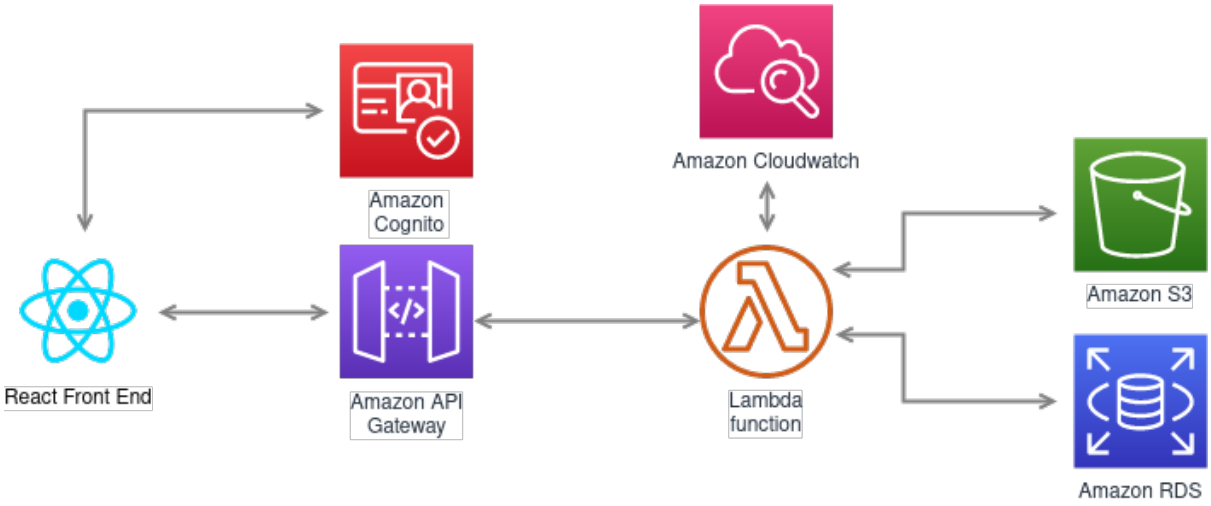

Figure 1: High-Level Architecture of the Platform

4.1 Front End

The platform's front end is implemented in React [4], a Javascript framework. Using this framework sped up the front end development work. The React Bootstrap [5] library was used in order to not have to develop certain React components from scratch.

4.2 Back end

The platform leverages Amazon Web Services' (AWS) [6] technologies. Since this is a heavily web based project, many of AWS's technologies allowed us to focus on developing the features without having to worry about server and network configuration.

AWS services utilized:

- **API Gateway** - To set up the connection between the front end and the back end.
- **Cognito** - To handle user registration and authentication
- **Lambda** - All of the API calls are handled by Lambda functions
- **Cloudwatch** - To monitor the platform and log API calls and Lambda function invocations
- **Simple Storage Service (S3)** - To store assets such as thumbnails, video strips and the videos themselves
- **RDS** - To host a PostgreSQL database which holds video, user and comment information

All of the handler functions were implemented in Javascript, using NodeJS, version 10. FFMPEG [7] and ImageMagick [8] were used for video and image manipulation, respectively.

5 Implementation

For the full implementation, see the GitHub repositories of the front end and the back end.

5.1 Front End

Five containers constitute the different website pages in section 5.1.1. In the different containers, elements of each page are subdivided into different React components to create a modular architecture with high cohesion.

The front end uses the API from AWS Amplify service to issue requests to the Lambda functions through the API Gateway.

5.1.1 Pages

- Home
- Registration
- Login
- Profile
- Video Upload
- Video Playback

5.2 Back End

5.2.1 API

Name	Method	Path
Create User	POST	/users
List Users	GET	/users
Get User	GET	/users/:userId
Create Video	POST	/videos
List Videos	GET	/videos
Get Video	GET	/videos/:videoId
Create Comment	POST	/comments
List Video Comments	GET	/comments/:videoId
List User's Comments	GET	/comments/:videoId/:userId
Create Like	POST	/likes/:commentId/:userId
Delete Like	DELETE	/likes/:comment/:userId
Export Video	POST	/exportVideo

Table 1: VideoCloud's API

5.2.2 Video Upload Workflow

The process of uploading a video to the platform involves all of the technologies in the platform, as seen in Figure 2. The details of the video upload process are the following:

1. The user fills in the upload form: inserting the video title, description, and the video file itself. Then clicks on the submit button.
2. A call is made through Amplify to upload the video to the application's S3 bucket, in the videos/ folder. This is the first step in a process that runs in the back end:
   (a) The video is placed in the inputs/ folder, and an s3:ObjectCreated* event is emitted by S3.
   (b) The multiRes lambda function is triggered by the event, after verifying that it was triggered by an input in the videos/ folder. It downloads the video from S3 and uses FFMPEG to create MP4 versions of the input video in different resolutions, up to 1920x1080, depending on the initial resolution of the input video. E.g. if the input video is in 1280x720 resolution, the lambda creates 256x144, 640x360, 960x540, and 1280x720 resolution versions of the video. These versions are uploaded to the outputs/ folder of the S3 bucket, emmiting another s3:ObjectCreated* event.
   (c) The createStrip function is triggered by the event. It downloads the 640x360 resolution version of the video and, using FFMPEG, takes 20 equidistant screenshots of the video. The 10th screenshot is used as the video thumbnail and is stored in the thumbnails/ folder in S3. Using ImageMagick, the 20 screenshots are concatenated side-by-side to create the video strip, which is stored in S3.
3. In parallel, the createVideo API is called to insert the video's information in the database.

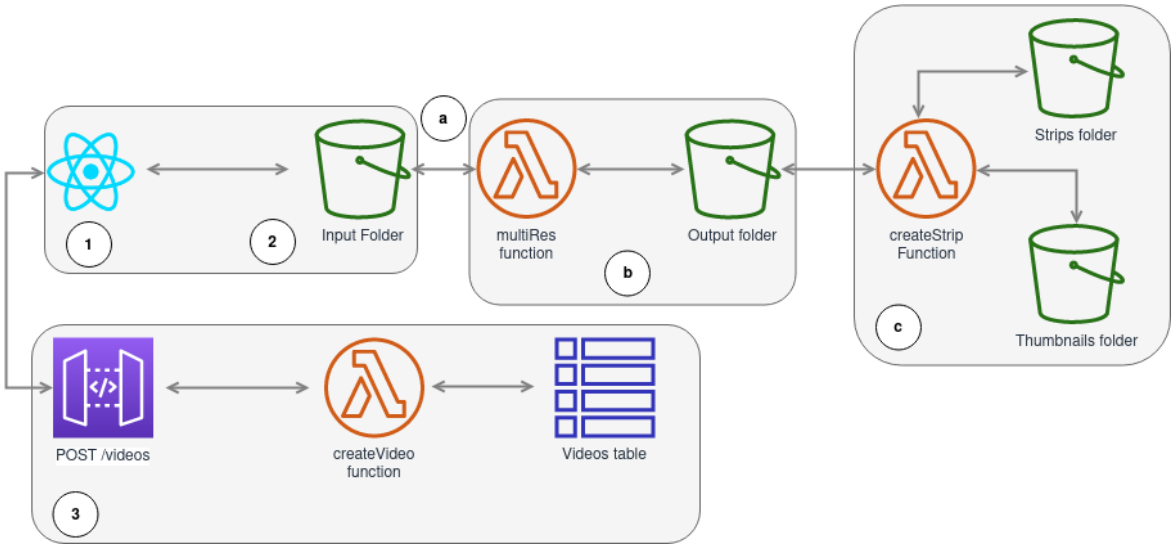

Figure 2: Video Creation Workflow

5.2.3 Video Export

The export feature allow the user to download the video with the comments as a subtitle file, so that they can be played offline. Figure 3 provides a visual representation of the project, which works as follows:

1. The client makes a request to the exportVideo lambda, containing the video ID, comments as JSON and the desired resolution
2. The Lambda function downloads the video from S3, creates a SubRip subtitle file containing the comments, and creates a ZIP archive containing both the video and the subtitle file. The archive is uploaded to the exports/ folder on the platfrom's S3 bucket.
3. A download link is returned by the API
4. The client can now download the archive through the link

The archive has a 24 hour time-to-live in S3, i.e. it is automatically deleted after one day.

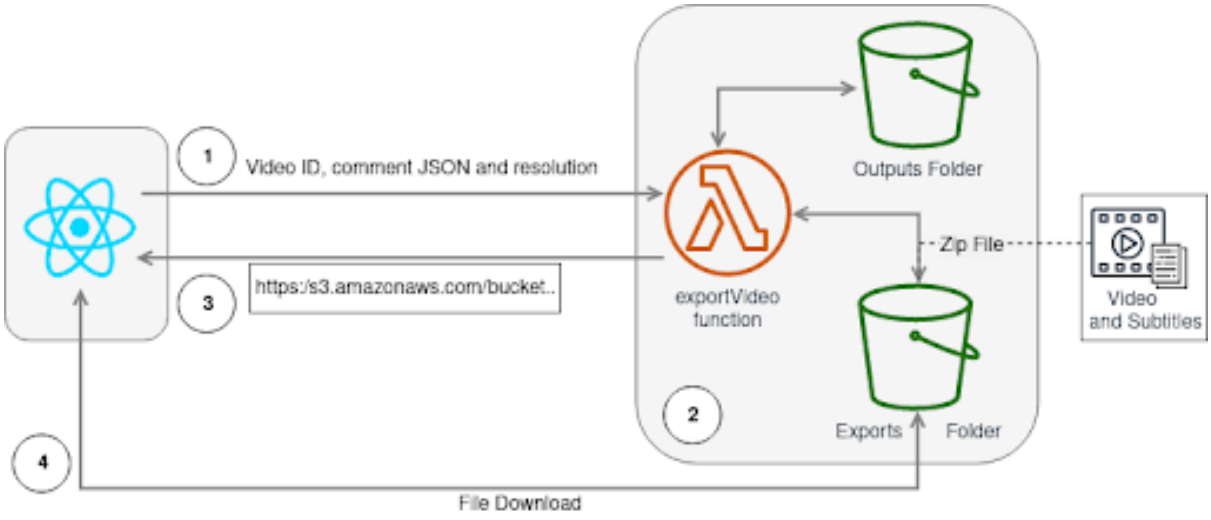

Figure 3: Video Export Workflow

6 Result

The finished application is a web based video streaming platform where users can upload and watch videos. The application has a comment feature that allows users to comment on specific video timestamps. These comments are displayed in a video strip below the video on the specific timestamps as the video is played. It is possible to export videos together with the attached timestamp comments. The platform offers video playback in different qualities.

7 Evaluation

The developed application satisfies the required core features stated in section 3.1. The current state of the application includes the basic functionalities necessary to be used as a video streaming platform.

8 Future work

Due to time constraints, the development was focused on implementing core features regarding the multimedia aspects of the project. This includes; displaying timestamp comments during video playback, video export with the attached timestamp comments and video playback in different qualities.

While prioritising the core features of the project, some usability features and and styling was left out. In order to make the application into a more complete video streaming platform, the features listed below could be implemented:

- Change password and other account settings
- Edit details of a uploaded video
- Implement adaptive streaming
- Optimization of the resizing function in order to prevent timeouts and support larger files

References

[1] (2019). Videocloud, [Online]. Available: https://priceless-carson-aa7a3a.netlify.com/ (visited on 08/14/2019).

[2] (2019). Youtube, [Online]. Available: http://www.youtube.com (visited on 08/11/2019).

[3] (2019). Soundcloud, [Online]. Available: http://www.soundcloud.com (visited on 08/11/2019).

[4] (2019). Reactjs web page, [Online]. Available: https://reactjs.org/ (visited on 08/11/2019).

[5] (2019). React bootstrap web page, [Online]. Available: https://react-bootstrap.github.io/ (visited on 08/11/2019).

[6] (2019). Amazon web service's web page, [Online]. Available: https://aws.amazon.com/ (visited on 08/11/2019).

[7] (2019). Ffmpeg, [Online]. Available: https://ffmpeg.org/ (visited on 08/11/2019).

[8] (2019). Imagemagick, [Online]. Available: https://imagemagick.org/index.php (visited on 08/11/2019).