CS194 Team Reflection

Cur8 | *the W in CS194W*

Introduction

From the moment we enrolled in CS194W on Axess, we were excited to assemble the squad and build something fun together. As we hit the ground running, it was clear that we had strong team dynamics — kind of like the Avengers, but better. At the beginning of the quarter, we were rapidly bouncing ideas off of each other in the cozy lounge of The WareHaus during brainstorming sessions. Memories like these will last a lifetime. Everyone demonstrated respect for one another, thoughtfulness and creativity in their contributions, and a unique enthusiasm. Team9 will always cherish these moments, and more importantly, one another. <3

Enough reminiscing. Time for real talk.

With a concrete idea in hand, we set onward into the depths of the quarter focusing on 3 main goals. Our first team goal, as a team of 2 student researchers and 2 former product managers, was to focus on translating the idea into an interactive, user-focused product. We wanted to heavily focus on the user experience and building out a robust and technical backend that had an intuitive and easy-to-use interface. This goal was directly oriented with our team's skill set and can be further decomposed into 2 specific metrics: intuitiveness and minimalism. We wanted to ensure that the complexity of the technology and models in the backend were not evident in the front-end and did not leak into creating an over-complicated interface. With regards to this goal, we were dictated by a counter-intuitive guiding principle that as our backend and feature-list got more complex, our front-end should become more minimalist and straightforward.

Our second team goal was to become implicit and effective communicators through writing well-structured and clear code, utilizing github's features, and remaining active on our messaging channels. Starting out, we realized how intertwined our tasks would be and this allowed us to initially set a goal of decomposing functions and naming variables and functions in very intuitive ways that enable another programmer to pick up the work at any point. In addition to being able to build off of each other's work, we wanted to maintain efficiency by parallelizing as much work as possible. This parallelization was made possible through using

the Project management feature on GitHub where people were allocated tasks individually and worked on them concurrently. We also made effective use of our messaging channel, consistently updating the team with feature deployments and completed tasks. This kept everyone in the loop and progressed the rate of feature deployment.

Our final goal was to be open to the evolution of our features, adapting to new feature sets as we got a better understanding of the technologies we would be using. This came naturally as our team's communication developed as we would consistently pitch new features and interfaces to interact with them. Eventually, we pivoted our user interface several times to include only certain filters and exclude and implicitly fill other ones that we thought users wouldn't use as much. We adhered to our team's goal of having a fluid and adaptive definition of the product, finally resulting in something we are all proud of.

Evolution, Process & Results

As our project progressed and evolved, so did our team dynamics. We started out our project with initial brainstorming sessions discussing products we thought would be useful to people. After an unfruitful brainstorming session, we decided to pivot our approach, focusing on finding a problem or use case that several of us would relate to. This initial shift in perspective, evolved our team's approach to the project, as we were now building a product for ourselves instead of an imaginary user. We now became great at critiquing what we wanted to see in the product and it catalyzed the progression of our iterations in the product vision. In addition to now being able to provide critical feedback on the current state of the product, our team also evolved to provide stronger and more accurate feedback to each other. This opened a dimension of communication that was great for improving our working styles to be more supplementary to each other. The evolution to being able to provide and receive constructive criticism was a great final transition.

From the start, our team created clear goals and expectations and this strong communication continued as the project progressed. We were relatively ambitious with our project from the start because we had multiple ideas that we wanted to fit into our final project. Execution of these ideas required communication and organization — as most of the time, each of us was responsible for a certain part of the project. In this sense, the team exceeded expectations because we were able to not only produce an end-to-end product but also worked together very well as a team. For example, we would ask each other questions about code that others had written, take ownership of different parts of the project, and hype each other up whenever someone accomplished something. We did a good job distributing work amongst ourselves — this work often played to our own individual strengths, as well as presented us with unique learning opportunities where we got exposed to parts of the tech stack that we didn't have as much experience with. From a concrete technical perspective, our team was able to get both

language summarization *and* video highlight creation working, as well as maintain strong product management flows.

Generally, turning points for the team were whenever someone accomplished something big! We were all very excited about one another's successes throughout the project and that created a wonderful team culture. These wins occurred throughout the project lifecycle, so we were continually motivated. For example, toward the beginning of the project when Ethan made a beautiful pitch deck and logo (*fig. 4*); in the middle of the project lifecycle when Alex got video clipping with audio working (*fig. 3*) and Kaylee showed a demo of Flask and Next.js communication with frontend (*fig. 2*); and at the end of the project when Amol got summarization working (*fig. 1*). All of these moments brought our team together, as shown:

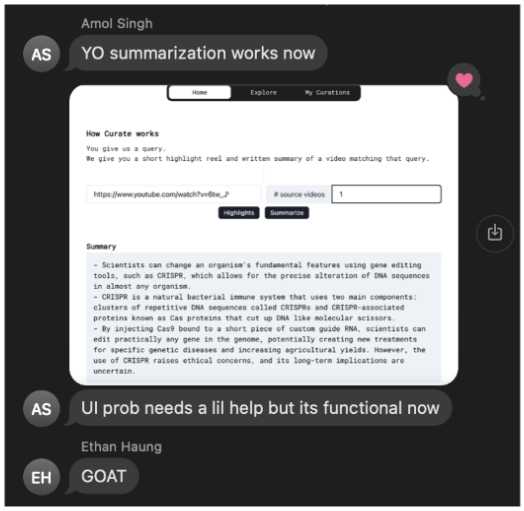

Fig. 1: Ethan calling Amol a "GOAT" for summarization

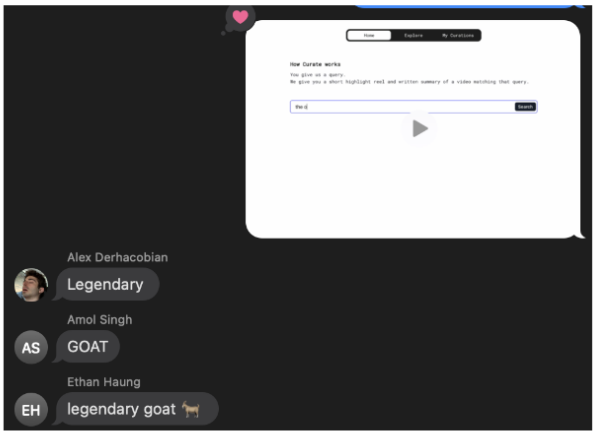

Fig. 2: Kaylee sends demo video, receives praise.

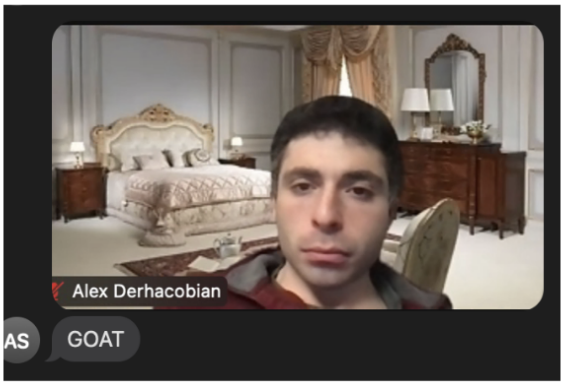

Fig. 3: Amol calls Alex a "GOAT"

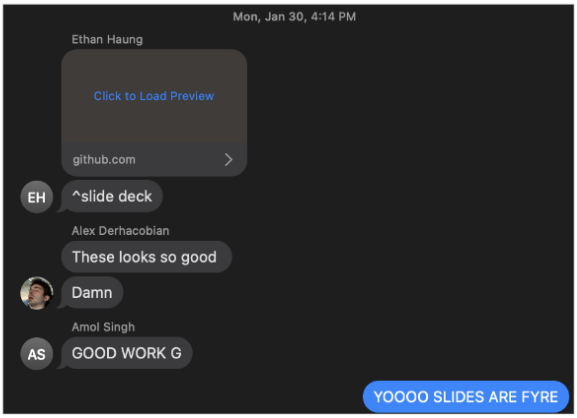

Fig. 4: Ethan's slides "looks so good" and "are fyre"

Throughout the course of our project, we discovered multiple insights about the product space: (1) large amounts of data are required to produce accurate highlight reels and summaries, (2) users want to optimize their time by maximizing the amount of video content they can consume, (3) adjacent to time optimization, users want a simple, easy-to-use interface that gets them from no content enjoyed to maximized content enjoyed, and (4) the content we

generate needs to actually be valuable to users by providing *accurate* highlights and summaries of videos.

Drawing from these insights, the ideal team would consist of diverse backgrounds and skill sets. To identify key data points, gather the data, and process it in a useful way, having someone with data science or analytics experience would be highly useful. To fully understand what is considered “optimal time and content management,” the team must also stay close to users through product managers who are consistently in touch with how users are reacting to the product. Having seasoned UX/UI designers who understand common design heuristics, especially for web applications will minimize friction in the flow of a primarily web-based product. Lastly, the team should consist of members who are passionate about solving the issue that is a “firehose of content” the everyday person is tasked with consuming. Every team member should want to deliver a valuable tool that makes people’s lives more enjoyable when it comes to consuming media.

There were many aspects of our team dynamic that made this past ten weeks an extremely enriching experience. We all came into this project with a diverse array of skills. For example, Amol and Alex primarily had experience in backend engineering, whereas Kaylee and Ethan had the design prowess and front-end capabilities to realize a deliverable product. This combination of skills made for a senior project experience that allowed us to work off of our own strengths but also learn and grow from others. Another enjoyable aspect of our team dynamic was the fact that we conveniently live together. This allowed us to leverage our connections *outside* of the classroom when working on our project. This demonstrated the importance of meaningful, personal connections in collective productivity and group dynamics. It might not always be enough to have the “perfect storm” of skills. In actuality, having personal compatibility with your close collaborators can make a world of difference.

One of the most challenging aspects of the project was the initial ideation phase. Everyone came into this project with their own visions and aspirations for what to do. It was after a couple of meetings that we settled on building some sort of video summarization tool. While Ethan was thinking of applications in the educational space of such a tool and Amol was thinking about monetization, it quickly became obvious to us that there were many directions that we could take this project. The most challenging experience for our team was coming to a consensus on the finalized *purpose* and *vision* for our project. This required very effective communication from everyone on our team. Poor communication would not only result in frustration but also missed opportunities. Over the course of the past couple of weeks, we developed better communication skills. We learned that communicating in loose terms can often become frustrating and lead to misunderstandings among teammates. We learned how to effectively and succinctly communicate complicated ideas to a team. This required not

assuming prior knowledge, thinking about who your audience really is and speaking to that, and effectively building a train of thought before explaining complicated visions or concepts.

Individual Reflections

Kaylee

Links: Code commits for initial Flask/Next.js setup, frontend, and API communication. Created Figma mockup for frontend.

I think my biggest contribution to this project was maintaining a high-level understanding of how the web application worked end-to-end. I created the initial project scaffold of the web app using Flask (backend) and Next.js (frontend) and set up the communication between the backend and the frontend. I also mocked up the frontend design and implemented the initial groundwork for the frontend. When it came time to deploy, I worked on implementing the Heroku flow and the S3 bucket handling, so that we could move our interface off localhost and into production. It was cool to learn about what goes into production and this helped me learn more about the build process.

I think I could have improved by better communicating with my team regarding the functionality of my code. I think I would write big chunks of code, push it, and not really explain it well to my teammates, or document it well. Thus, it probably made it harder for my teammates to parse through what I had written and understand the functionality. This project has demonstrated to me the importance of good documentation, and I will continue working on improving that skill. Additionally, I would work on having better commits because sometimes I would have commit messages that were non-descriptive and unhelpful — further contributing to the non-helpfulness of my documentation.

As part of this project, I think I improved my understanding of how the frontend and backend are separated by functionality, as well as how they fit together to create a cohesive web application. I enjoyed increasing my understanding of how Flask works and what it takes to have a backend written in a different language than the frontend. I think it was a good exercise to work through the tradeoffs of creating this architecture (as I haven't personally done that before) — and ultimately, it paid off in the end. I also feel that my understanding of how to maintain a good repo when working with a team improved. I learned a lot about task management in software development, which I hope to continue to improve in the future.

Amol

Links: Code commits for summarization feature, frontend, and api-hack. Helped create and maintain OKRs and KPIs.

My biggest contribution to this project was building out the summarization feature. I did this initially by using a hack to directly access the API for ChatGPT for free. Once that was patched

up by the OpenAI team, I had to re-implement an approach using the newly released GPT-3.5 turbo API. This approach worked great and allowed us to summarize youtube videos in a fairly fast manner. Along with building this backend, I also implemented this feature on the frontend, giving consumers a way to interface with this new feature. In addition, I had the initial idea of generating highlights for youtube sports videos and defined the first version of our product that our team collectively iterated on.

As a team member, I think I did a good job in communicating my status on milestones and when I would wrap them up. I think that I could work on documenting my code, enabling other people to pick up from where I left off and making the functionality of my code more straightforward. I think in the future I will also work on setting a more concrete timeline with my teammates, ensuring that we have a better shared understanding of when certain features are expected to be done.

In this project, I furthered my understanding of how a team larger than 2 people would collectively work together on a shared codebase to build out a product. I learned about how important it is to modularize the implementation as much as possible and parallelize tasks when possible to ensure work is completed in a timely manner. I also enjoyed learning about how a react frontend and a python backend would interface with flask.

Ethan

Links: Code commits for multi-video sourcing and updated estimates. Created slide deck and rapid prototyping deck. Helped create and maintain OKRs and KPIs. Created user testing guide and issues management board.

My major contributions came in the form of leading non-technical, qualitative work and project management. After our initial ideation discussions, I synthesized our most salient points and ideas into a concise slide deck. We were then able to use this deck to pitch the idea to multiple people to gather initial feedback and reactions. As we delved deeper into a solidified product idea, I led the creation of our OKRs and KPIs, drawing inspiration from metrics and goals I've seen in past product internships. We used these throughout the quarter as key milestones to measure success. In terms of project management, I led the generation of issues on our GitHub repository so we had a clear understanding of which bugs and features to prioritize. Using a kanban board, we were able to actively prioritize new issues that arose through internal testing and our mid-point check-in. In addition to non-technical aspects of the project, I was able to modify our product to work with multiple video sources, avoid unnecessarily downloading videos that we did not end up using, and include better estimates of how long the video generation would take. We initially downloaded all videos, even those without "most replayed" data, so I implemented the logic to avoid this issue. The video estimates also provided a much better experience for users instead of just receiving a static "estimated <1 min remaining."

As a team member, I felt that I communicated well with each individual team member and was able to facilitate productive conversations during team meetings. I was able to help get the team excited about what we were working on and acted as a sounding board for different potential solutions and general ideas. In terms of improvement, I could have tried staying more on top of our codebase and the content being pushed by other team members. When I was contributing code, it took a while to fully understand everyone's code before I could contribute meaningfully and cleanly. I could have also been more proactive in identifying potential solutions others were facing (e.g., Heroku issues, S3, etc.).

Throughout this project, I learned how to not just work with provided starter code, but how to set up an application from the ground up. I also learned how to work effectively in a group that has to collectively code and communicate to make meaningful contributions. It was also great to get exposure to exploring non-traditional ways of solving technical problems such as getting the "most replayed" data and how we actually implemented summarization.

Alex

Links: Code commits for sound engineering, video highlight generation algorithm. Helped with Figma mockup for frontend.

Most of my contributions as a team member were on the core backend code for generating the video highlights. There were multiple parts to this code. The first part was code that was required to interface with the official YouTube API. Given a user query for a youtube video or a youtube search query, the YouTube API was required in order to access unique identifying information about the video, such as the videoID of the given video. Previous versions of our project also required the search results for a given YouTube search query. Once we accessed the public metadata for the video we wanted to generate highlight clips for (either from a specific link or the top hit from a user-provided search query), we generated the highlights.

The first step in generating the "most replayed" clips from a video, or the "highlight" clips from the video, included finding the "most replayed" data for a given video. This is a new feature implemented in highly trafficked videos on YouTube which allows users to buffer over a video and see a distribution indicating the popularity of a given timestamp. This creates a distribution over the entire video. We used a python library in order to access this data. This data was served to us in a set of tuples which consisted of [start_time, end_time, average_most_replayed]. These start and end times were discrete and of fixed length. Therefore, we wrote a simple algorithm by which to aggregate these tuples into intelligible clips that can be served to the user.

A hyperparameter was set which indicated the number of clips to aggregate. We varied this number from 5 to 50 but decided to settle on a maximum of 20 clips per video. For a given clip i in a video, the clip was represented by a start and end time such as [start_i, end_i]. Each of the clips for the video were required to be non-overlapping so they were not repetitive for the user. We did this by combining two tuples A = [start_time_a, end_time_a, average_most_replayed_a], B = [start_time_b, end_time_b, average_most_replayed_b] if they overlapped in any way or were within a certain number of milliseconds of one another.

In this project, I believe that I was able to create robust code that lasted for the entirety of the project. Although the algorithm that I wrote for clip concatenation was a rather naive approach, it ended up serving our purposes rather well. One thing that I could improve on was that I could have communicated with my team members doing front-end work more during the beginning of the project. This could have alleviated a lot of headaches of misunderstandings from both sides about each other's code. Pair programming could have also been a great way to have everyone on the same page and also let people learn about other skill sets being used on the project.

As part of this project, I furthered my understanding of Flask and how it interfaces with backend code. I also learned a great deal about how to deploy products using blob storage such as Amazon S3 buckets.

Conclusion

Having achieved the core milestones of our project, the Cur8 team is happy with our progress so far! We have created a product that generates custom highlight reels based on 1 to 50 YouTube videos that the user specifies through links or search queries. Our product is also able to summarize any YouTube video into a concise paragraph for users to read while on the go. We are in the process of optimizing our video delivery speed, continuing to debug our Heroku server usage, and bug-testing our product for robustness.

Throughout this process, we gained multiple insights. We experienced first-hand the importance of effective communication and collaboration within the team. The more we communicated and the more intentional we were when communicating, the faster we were able to make progress and the better each iteration of the product was. We also realized the importance of working with existing libraries and tools to streamline the development process. Without the usage of online resources and open-source libraries, we would have been unable to generate nearly the same amount of progress. This quarter also brought about a revelation of the power of large language models. With the release of GPT-4, the opportunities seem endless, and we are just scratching the surface.

Some challenges we faced this quarter include gathering and processing the proper data to generate highlights, issues with API access, and deploying the product on an actual server. We had to use open-source libraries and APIs to find a workaround to YouTube's lack of disclosure of "most replayed" data through their native API. We had to try going through personal Open AI accounts to access ChatGPT without their API. We are also currently working with Heroku and Amazon S3 buckets to fully deploy the app, but are still experiencing some CORS issues when running two simultaneous servers. We were disappointed that we were unable to improve upon our current highlight reel quality (e.g., start and end at appropriate breaks, etc.).

Midway through the quarter, we were pleasantly surprised that prominent figures such as Garry Tan, a former partner at Y Combinator and founder of Initialized Capital, actually wanted a product like ours to exist. In fact, he tweeted: "Is there a summarization service for YouTube links and/or podcast links? What I want is to submit a URL, and then get LLM summarization in bullet points of specific stories or segments with timestamps…Willing to pay per link!" This further fueled our work for the last 5 weeks of the quarter.

Overall, our team is proud of the work we've accomplished for Cur8 this quarter and we hope to continue developing the product so it becomes a useful tool for the public to actually use. We would also like to thank our team's TA, Federico Zalcberg, for his guidance throughout the quarter. He was instrumental in our success and we would not be where we are without him!