Sprint Retrospective, Iteration #2

Task Overview

Task #	Task Assigned To	Estimated Effort per Task *(in hours)*	Actual Effort per Task *(in hours)*	Done *(yes / no)*	Notes
Display video in browser with OpenCV.js	Dean	8	2	No	This has been deemed infeasible as the OpenCV.js library is not well documented and lacks essential functionalities in the core OpenCV library written in C++.
Add the ability to fetch BLOB from server	Dean	2	3	Yes	Additional time was used for testing and investigating why the client video was not properly displayed in the browser (encoding issue).
Add roles (admin/user) to the front-end	Dean	3	4	Yes	All admin restricted resources, in addition to the authorization in the back-end, are protected in the front-end by only displaying the UI elements to admins.
Send user identification to the server	Dean	1	1	Yes	Add a header with the unique oid of a user. This is encrypted when the application switches to using HTTPS for communication.
Preprocess the video with OpenCV Java and FFmpeg	Dean	10	26	Yes	This task took much longer than expected due to the lack of documentation for OpenCV Java which led to memory leakage and other issues. Additionally, learning about FFmpeg for video compression and getting the CLI tool to work in Java proved to be quite a hurdle.
Add page to view project	Ege	5	15	Yes	This was my first time working with the front-end. I had to learn javascript and react so the first task took much longer than expected since I experimented.

Add page to edit project	Ege	5	7	Yes	This time since I knew the basics. However to make the web app look more intuitive and user friendly, I had to look up material ui documentation.
Add page to view single project in detail	Ege	5	7	Yes	I had to learn more about react switches, routes and how to use routes with parameters.
Styling pages with css and inline styling	Ege	10	10	Yes	One of the most important parts of the product is the user friendliness and ease of use.
Fix database constraints	Akash	8	10	Yes	Due to miscommunication amongst team members, we have changed the database schemas for a few times
Build 3-tier architecture at the server side	Akash	15	15	Yes	I used 3-tier architecture for the backend because it separates the system into different layers. In other words, it provides ease for maintaining the code base. The endpoints should only invoke service layers. And the service layer handles all the logic and invoke methods from the data access layer. Data access layer is responsible for fetching data from the database. I also used the strategy design pattern to reduce code clones.
Fix some entities, add some services for the service layer, add database test for some services	Thang	10	15	Yes	It is the first time I've worked with 3-tier architecture, it took some time for me to understand how it works. The cascade errors with hibernate made it difficult to test the update methods in the service layer
Add roles-based authorization using Azure	Tan	3	6	Yes	Azure did not have enough documentation on its authorization library how it handled roles. It took some time for me to figure out how @PreAuthorize works as well.
Add video player on the frontend	Tan	5	15	Yes	Since the prerequisite is that we need to precisely extract frames using time + fps, I did some experimentation on what video player I should use and that takes a

					long time. I chose VideoJS but I tried to use hooks instead of class components, that took time to research as well because it’s my first time working with React.
Integrate blob fetch from the web server with the video player	Tan + Dean	0.5	4	Yes	We tried to fetch the blob from a web server then serve it using VideoJS. There was an unexpected problem where the error showing up wasn’t very descriptive. We finally figured out that the blob fetch method was async and VideoJS can’t display it due to it’s ‘lazy initialization’.

Main Problems Encountered

Problem 1: Time underestimation

Description: Some tasks were underestimated and took much longer than expected

Response: Adjust the expectation appropriately from our experiences

Problem 2: Inexperience with React and the VideoJS library

Description: Inexperience with React led to the integration of fetching and displaying videos taking much longer than expected

Response: Continue to work with and look for more resources on React

Problem 3: Database schema design and miscommunication

Description: Due to miscommunication amongst team members, we have changed the database schemas for a few times

Response: The issue has been resolved and there is working implementation of the database