User test format

Procedure: The goal of user testing is to make sure users will successfully use our software. It is unclear how literate or clever these users will be. Our procedure is therefore very simple. Given a computer with a browser open to the homepage of our website, we task our user to find a video they want to watch, comment on it, edit their comment, and finally delete it.

Reasoning: Many of the important features of the API must be used to complete our procedure. While we agree using the API is very different than using our front end, we spent a good chunk of time making sure our API functions interface well with the front end. Most of the API calls boil down to specifically formatted database queries interfaced with an abstract front end. If the user has trouble using or understanding a certain task, at worst we found a failure in effective documentation. At best, we get insight into a new bug in the API and immediately resolve the issue.

User Test Demographic: While the main product we deliver is a comment API, we target the secondary users - people accessing a front end website using the API. We target secondary users for two main reasons. First, within the scope of this class, we do not expect to have a developer build a front end and implement our comment API within our short sprint time-frame. Second, we believe valuable information about the comment API remains even from secondary user feedback.

User test 1:

Tanner Smith (my roommate, No CS background, but computer literate)

Immediately left documentation page
Resolution: Make it shorter/more appealing??

Clicked around for a while (Super mislead by Comment API tab)
Resolution: Delete Comment API tab

After he found the videos tab, he tried to click on a video he wanted to watch. He had to try a couple times before he figured out you have to click on the video title.
Resolution: Make entire div the a=href

There is no comment button unless you are logged in, which I had to tell him about.
Resolution: Make alt display saying you have to be logged in to comment

Liking comments, disliking, and funny were semi intuitive. He asked if the word ‘likes’ corresponded with the thumbs up and down icons.
Resolution: Replace those words ‘likes + funny’ with the icons

FEEDBACK: commenter icons (instead of blank boxes), actual video playback (instead of text)
Resolution: Add those in (like we have been talking about)

User Test 2:

Carson Hansen (No CS Background, my brother, computer literate)

Looked at getting started page

Scrolled through pictures super fast (did not read anything)
Resolution: Less text to read, better pictures

Saw picture that showed videos tab on navigation, so left

Found a video, dead clicked on the box a few times
Resolution: Make entire div the href

Could not comment (because not logged in)

Watched the video a bit

Realized he should make an account to be able to comment/react
-- Make this part more clear

Commented on the video

Started messing around with likes and dislikes

Reactions + Commenting reloads the page
Resolution: Make a static function such that no routes need to be called to load in the page (with components)

Noticed Infinite reactions, and abused it.
-- We already decided this was out of the scope of our project

Feedback: Fix infinite reactions