Milestone 2 Submission 1 by Group 1

Proposed Level of Achievement: Gemini

Panopto Plus

A Chrome extension to improve the user experience of webcasting

**Purpose of this document:**

1. Allow evaluators to evaluate implemented feature and perform acceptance testing

2. Allow evaluators to evaluate planned features

3. Allow evaluators to evaluate UI, UX and usability of product

4. Allow evaluators to evaluate testing procedures used during the project

Table of Contents

[**Video URL**](#_xpo2zbos30h1) **1**

[**Recap on the Scope of Project**](#_brqln99omcld) **2**

[**Features/Tasks Completed**](#_h3mnv2m1spjr) **2**

[In-depth description of Completed Features](#_rri1kd1i29mu) 4

[Limitations of Completed Features](#_yedj426qdton) 9

[**Features/Tasks In Progress**](#_typcrsrktleu) **10**

[In-depth description of Features in progress](#_g4lxctaksafz) 10

[**Features/Tasks Pending**](#_ykyomq1w37n3) **11**

[In-depth description of Pending Features](#_vwzxavxtm88k) 13

[**Features Kept In View**](#_yckw3ado3ih3) **15**

[**Usability / User Interface / User Experience**](#_90qx67ruup04) **15**

[**Types of methods used to evaluate the suitability of solution(s):**](#_dhvvw2rqiotw) **16**

[**Survey Results**](#_z53l5t9n41hc) **18**

# Video URL

<https://www.youtube.com/watch?v=IwWY9Yh84co&feature=youtu.be>

# Recap on the Scope of Project

The Chrome Extension will serve 3 main purposes:

· The Chrome extension will inject HTML, CSS and JS into the page to improve user-friendliness. There will be a side-bar for users to configure the extension settings.

o Wider range of video playback speeds, restructuring of page to allow for more space for the video webcast

o Persistent settings across webcasts e.g. always play all webcasts at 2.0x speed.

· Machine Text Transcription (Machine-generated subtitles for webcasts) will by generated by Panopto. This Chrome extension will leverage on that and process the transcript into subtitles and a user interactable transcript.

o Users can click sections of the transcript to seek to that video timeframe

· The Chrome extension will also process files of the webcast for voice detection. Sections in the webcast where there is no speech will be automatically skipped.

These features are toggleable (except user-interface modifications to the Panopto webpage).

# Features/Tasks Completed

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **User / Main Beneficiary** | **Details** | **Status (Pending / In Progress / Testing / Complete)** | **Deadline** |
| UI | Student | Smaller carousel\* | Complete | Mid June |
| UI | Student | Removal of Panopto header if loaded on LumiNUS | Complete | Mid June |
| UI | Student | Replace playback Speed buttons with slider | Complete | Mid June |
| Subtitles | Student | Retrieval and caching of transcript from Panopto via REST API | Complete\* | Mid June |
| Subtitles | Student | Conversion of transcript from Panopto to suit app’s purposes | Complete | Mid June |
| Subtitles | Student | Display of subtitles on video(s) | Complete | Mid June |
| Subtitles | Student | Properly synced subtitles on both videos (two video webcasts) | Complete | Mid June |
| Silence Removal | Student | Basic implementation (demuxing, decoding & processing using AudioWorklet) | Complete | Mid June |
| Silence Removal | Student | Skipping mechanism based on silent sections detected | Complete | Mid June |
| Silence Removal | Student | Extension of basic implementation: caching of results, async, multiple webworklets and audioworklets | Complete | Mid July |
| Misc | Contributor(s) | Generation of JSDocs | Complete | Mid July |
| Misc | Student | Fix Panopto’s “buffering” issue | Complete | Mid July |
| Silence Removal | Student | Re-implement with web worker because Chrome’s OfflineAudioContext current implementation has memory leaks | Complete | Mid July |

*\*Carousel refers to the carousel web element below the main webcast on the Panopto page, which allows users to see a “snapshot” of the webcast at that time.*

Acceptance testing should be performed via the video or directly via the chrome extension that can be found at<https://github.com/crazoter/panplus> (however you’ll need to enable developer mode in order to load an unpacked chrome extension). The following features discussed below simply help flesh out their purpose and implementation detail.

## In-depth description of Completed Features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Smaller carousel | Complete | Mid June |

Why smaller carousel: Simplicity. Panopto’s elements and their positioning were more coupled than anticipated. A smaller carousel instantly translates to a larger webcast screen while still retaining the same functionality that the carousel had intended to offer (allow users to preview & seek to point in video).

Implementation Details: This was already implemented using CSS and was evident in the first video.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Removal of Panopto header if loaded on LumiNUS | Complete | Mid June |

Why: This is a small UI improvement that removes an unnecessary second header that can be used for other elements on the page.

Implementation Details: I can’t show this in action because I currently do not have access to any modules on Panopto through LumiNUS. This was implemented prior to NUS dropping access to our previous modules (hence the necessity to begin work quickly).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Replace playback Speed buttons with slider | Complete | Mid June |

Why: Amongst most UI/UX improvements, the one that catches the attention of students is the one that allows them to increase the playback speed beyond 2x (which is understandable given that would directly translate to shorter weblectures). On top of that, the previous UI with buttons was imprecise; users couldn’t adjust as per their liking.

Implementation Details**:** With this new slider, users can adjust (in real-time) the playback speed by clicking or sliding on the notched slider up to a precision of x0.1 up to a ludicrous playback speed of x7.0. The playback speed is also adjusted as users slide, allowing them to better adjust their playback speed and immediately get feedback without having to release their mouse. If necessary, they can also reduce it down to x0.5 (although I believe that the use cases for *slowing* down a webcast are few and far between).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subtitles | Student | Retrieval and caching of transcript from Panopto via REST API | Complete\* | Mid June |

Why: Our subtitles and transcripts come directly from Panopto. No retrieval, no subtitles. The reason why caching is used is to reduce the load on Panopto’s servers, if this is not an intended use case for their APIs. In my opinion, the load is nominal and shouldn’t cause any issues.

Implementation Details**:**  The work has already been done; the only thing that needs to be finished is an expiry date for the cached transcripts and an expiry system to remove expired transcript data. This is done behind the scenes, so from the user’s perspective they don’t even know this is happening.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subtitles | Student | Conversion of transcript from Panopto to suit app’s purposes | Complete | Mid June |

Why: Self-explanatory

Implementation Details: Use of OOP to make things easier for future contributors if transcripts don’t come from Panopto’s API.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subtitles | Student | Display of subtitles on video(s) | Complete | Mid June |

Why: Subtitles means glanceable data, which means that if you know what the Professor is talking about, you can quickly just skip over that section of the webcast. Even with poorly translated transcripts, if you can derive a few pieces of information about what the Professor is saying, or if you find yourself amused by how bad the subtitles is (and as a result injected humour into an otherwise boring weblecture) then the subtitles have helped improve your webcast experience. If all else fails, the subtitles can be disabled in the final product.

Implementation Details: You can check the GitHub repo, code and JSDocs for the nitty gritty details on the implementation details from a developer’s perspective. From the user’s perspective, the subtitles are slightly translucent (but not to the point where they make it hard to read) to reduce the amount of stuff they block on the webcast. In the final product, users will be able to toggle the subtitles.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Subtitles | Student | Properly synced subtitles on both videos (two video webcasts) | Complete | Mid June |

Why: Most webcasts have 2 videos. Have you noticed that sometimes one video plays at a higher FPS than the other? That’s right; both videos are playing at a different timestamp, and the difference between their timestamps can vary. This makes syncing subtitles, transcripts and even silent sections a challenge that needed to be resolved to ensure both videos play the subtitles and whatnot at the same time.

Implementation Details: From the user’s perspective there should be no changes in their user experience. The implementation itself is simple; because the offset is only known at runtime, all one needs to do is inject the subtitles from the first video to the second when the subtitles are displayed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Silence Removal | Student | Basic implementation (demuxing, decoding & processing using AudioWorklet) | Complete | Mid June |
| Silence Removal | Student | Extension of basic implementation: caching of results, async, multiple webworklets and audioworklets | Complete | Mid July |

Why: First, before I talk about why we even need to demux, decode and perform voice activity detection, I will need to discuss why we even need Silence Removal as a function. There will be users who may not see the point of this function, because it’s meant to be an “invisible” function; users will enjoy the benefits of the function without really realizing that it’s saving them a lot of time. I myself, am sometimes unsure about the amount of time I’m saving when I watch a webcast using this feature until I check the console logs. This is because there are many moments when the Professor will subconsciously pause to think or breathe and even these moments are shortened with this feature. It is exceptionally helpful if the Professor likes to speak in short bursts with pauses in-between, an issue that cannot be rectified simply by pumping up the playback speed.

Implementation Details: The only reason why this is defined as “basic implementation” and the complete status is starred is because the demuxing and decoding is not done on a webworklet (which would reduce the strain on the main thread of the browser; understand that all javascript runs on only one thread). Even without the use of web worklets, this feature can be considered implemented, considering that the main processing of voice activity detection is already done in a web worklet.

The actual implementation is far more complicated, considering that TS files are not readily supported by Chrome and must be demuxed and decoded into a format that can be used.

This data then undergoes even further processing using audio worklets which is still a somewhat new feature (its use is necessitated by the fact that it replaces a function that will be deprecated in the future); its new-ness and rawness causes complications especially when sometimes the documentation doesn’t describe the implementation.

The logic behind this processing was initially straightforward: under the basis that human speech has a frequency range, one could perform a Fast Fourier Transform to get the magnitude of the frequencies of the human speech. If the magnitude crossed a certain threshold, one could assume that someone was speaking. This didn’t work very well, and I figured that it had to work well, or there was no point in having the feature in the first place.

I thus adopted a second approach based on this research article:<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4142156/>. The core idea is to distil a signal down to 5 variables. They then obtained a set of variables as a reference set by assuming a certain section of the audio comprises only of noise and comparing the difference of the sum of each variable of that reference set to any other set of variables derived from other signals. If the difference was large enough, then the signal wasn’t noise (and thus likely to be voice). For the noise reference, I used the last second (or two) of the webcast because that is where the Professor would’ve stopped speaking (and it is much more reliable as a source of noise than the first second, as some webcasts immediately begin with the Professor speaking). However, the paper does not discuss very much about what is the threshold for “noise” and “voice” which made it difficult to implement it. As a result, I instead gathered 30 samples for the noise reference. Assuming a normal distribution of each variable due to the central limit theorem and using a confidence level, I then assigned lower and upper bounds for each variable. If most of these variables fell outside of these bounds, then they would be considered as not noise. This approach worked exceptionally well, even for webcasts with a lot of noise (with the exception of certain scenarios like the Professor playing a video during the webcast). The confidence level could be further tweaked of course, and I am thinking of adding it as a setting for users to play with to improve their experience if possible. However, it works well as is, and the setting would be considered an additional feature.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Silence Removal | Student | Skipping mechanism based on silent sections detected | Complete | Mid June |

Why: Save time plus you don’t have to struggle to understand what the Professor is saying if you’re playing at too fast a playback speed. Also very effective for webcasts that have exceptionally long and frequent pauses.

Implementation Details: Once all the heavy lifting has been done, these silent sections are then added as cues to the video. The videos then jump when they reach these cues and need to be synced.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Misc | Contributor(s) | Generation of JSDocs | Complete | Mid July |

Why: There will come a time when the summer ends, Orbital ends and everyone goes back to studying. In order to make sure a project is well maintained, others also need to understand how the project is implemented and that is where documentation comes in to help others understand and maintain the code. JSDocs is a means to convert all those comments that is in the codebase into a set of pages that make it easy for developers to read the code (much like Java API Documentation).

Implementation Details**:** The package.json is in place to help others build and update the JSDocs if required. The documents are already in the GitHub repository.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Misc | Student | Fix Panopto’s “buffering” issue | Complete | Mid July |

Why: This is not so much a feature as it is a quick fix. From the testing that I’ve been doing using the speed playback increase and silent section removal, Panopto has a bug that causes the webcast to stutter and pause before continuing its playback even though the video should’ve been buffered. This restricts the effective playback speed (even without the chrome extension enabled, this bug can still occur).

Implementation Details: When the app detects that the videos are not running when they are supposed to be running, then it will make it run. This detection is currently done by overriding the Panopto’s logging system to conditionally execute a quick fix.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Silence Removal | Student | Re-implement with web worker because Chrome’s OfflineAudioContext current implementation has memory leaks | In Progress | Mid July |

Why: On further testing it is discovered that there is a memory leak that will cause the webpage to run out of memory and subsequently crash after around an hour (or when it runs out of memory). This is not due to our own code, but due to Chrome’s implementation OfflineAudioContext which currently has garbage collection issues… Although it is not a critical issue (because one can just reload and continue where they left off), I’d much rather the program *didn’t* crash.

Implementation Details: Fixed by directly retrieving the PCM data from the AudioBuffer and processing it using a web worker. Demuxing & Decoding still remains on the main thread because AudioContexts cannot be initialized on the worker thread (and I’d rather not bounce that data to and fro worker and main thread) so it is not as efficient as I’d like it to be, but the problem is resolved and there are no longer any memory leaks.

## Limitations of Completed Features

Transcript & Subtitles do not exist for some webcasts

Because the transcript and subtitles are retrieved directly from Panopto, if the webcast doesn’t have machine transcribed transcript attached to it (because the webcast is too old / the feature isn’t enabled on Panopto’s side), then there will not be transcripts or subtitles. This applies to a lot of the old open webcasts. However, in earlier testing sessions on webcasts for the modules I was taking, the webcasts have machine transcribed transcripts. I doubt this will be a big issue.

The alternative is to use a paid service like Google Cloud Speech To Text. However, these alternatives are paid services and they can only transcribe 1 hour of audio per month for free per user. This is infeasible as the goal is to get it into the hands of NUS students as a free service. On top of that, these solutions are not that much more accurate than what Panopto offers based on some preliminary testing using some audio files.

Transcript & Subtitles are sometimes inaccurate

As you might have guessed, the transcripts are machine transcribed. This means that although some webcasts will have some decent transcripts, some webcasts will have transcripts that aren’t so good (and some will have transcripts that are laughable). I thought of a crowd-sourcing feature that students can use to contribute to fixing the machine transcripts, but it was counter-intuitive; the goal of transcripts is to *shorten* the time taken to watch a webcast, not increase it. Alternatives such as Google Cloud Speech To Text are also unable to provide a better transcription than what Panopto offers based on preliminary testing of some audio files, thus the team decided that the additional effort and cost of these alternatives is simply not worth it.

Not all silent sections may be removed

This is still undergoing refinement, but ultimately, the result may not be a perfect removal of all silent sections because there is a need to balance the likelihood of false positives (voice recognized as noise/silent sections) and the false negatives (noise/silent sections recognized as voice), especially when noise is involved. Although it won’t be perfect, the existing implementation has already removed a substantial amount of noise/silent sections. By adding the confidence level as a setting, the feature should hold its own quite well.

# Features/Tasks In Progress

A feature in progress is a feature that is currently in development

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **User / Main Beneficiary** | **Details** | **Status** | **Deadline** |
| UI | Student | Sidebar with tabs | In Progress (High Priority) | Mid June (Overdue) |

## In-depth description of Features in progress

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Sidebar with tabs | In Progress (High Priority) | Mid June |

Why sidebar (instead of navbar at the top or bottom): A sidebar was chosen for two reasons. First, simplicity. It would be tedious to revamp Panopto's UI for minute improvements of the user interface. At this point, users would also have expectations of how Panopto's UI will be like. Changing it drastically will force users to relearn, which I believe is not not worth the change. Second, the aspect ratio of the webcast. The webcast is limited by the height of the screen; one can observe that by reducing the height of the carousel, one is able to increase the size of the main webcast video stream. Placing other information on the side of the webcast thus allows for a better user interface.

Why tabs: The reason why we include the settings tab (instead of placing the settings options available only on right clicking the chrome extension icon) is to make it easier for users who may not be familiar with how chrome extensions work. There is also enough space on the tab bar to accommodate this settings tab. Users can choose to have a second screen + transcript, transcript alone, settings, or just hiding the bar entirely.

Implementation issues encountered: Panopto's implementation of their two video stream system is quite quaint, in the sense that if you want to switch webcasts (when you click the swap button), Panopto literally swaps their positions around. This caused some difficulty in effectively implementing a proper sidebar.

Survey: Ultimately the end-users will be the students of NUS. Although I myself am a current user of the system, I figured it would be nice to get more feedback from other students. I am currently conducting a google survey to see which design they like best using screenshots.

Intended Design: Based on the responses from the survey, the current design for the sidebar is as follows: It will have 3 tabs and 1 hide (sidebar) button.

Tab 1: “Default”

1. Retain the same UI that Panopto is using.
2. Make the “Search this webcast” field shorter.
3. Add Transcript as an additional tab under “search this webcast”

Tab 2: “Transcript”

1. Just the transcript

Tab 3: “Settings”

1. Settings checkboxes and input fields for users to input their settings.
2. Settings are split into “Live Settings” and “Settings that require reloading”. They will be placed on top of each other.
3. There will be 3 Save buttons, only upon clicking any of these buttons are settings committed:
   1. “Save for this webcast only”: Settings are only saved for this particular webcast.
   2. “Save for all webcasts under this module”: Settings saved for all webcasts under this module.
   3. “Save for all webcasts”: Settings saved for all webcasts. A confirmation alert will be prompted.

Tab 4 (or button, implementation to be fleshed out): Hide sidebar button

# Features/Tasks Pending

A pending feature is a feature that is planned to be implemented

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **User / Main Beneficiary** | **Details** | **Status** | **Deadline** |
| Misc | Contributor(s) | Error logging module | Pending (Medium Priority) | Mid July |
| UI | Student | Consistent UI & UX for single video and double video webcasts | Pending (High Priority) | Mid July |
| UI | Student | Customizable carousel via settings | Pending (High Priority) | Mid July |
| UI | Student | Customizable speed modifier options | Pending (Medium Priority) | Mid July |
| UI | Student | Volume booster | Pending (High Priority | Mid July |
| UI | Student | Option to make the webpage fullscreen | Pending (Medium Priority) | Mid July |
| UI | Student | Allow webcast to be shown even on smaller screens (responsiveness) | Pending (Medium Priority) | Mid July |
| UI | Student | Other miscellaneous improvements\*\* | Pending | Mid July |
| UX | Student | Persistent settings across webcasts | Pending (High Priority) | Mid July |
| Transcripts | Student | Visible Transcripts on the sidebar | Pending (High Priority) | Mid July |
| Transcripts | Student | Highlighted line in transcript in sync with video | Pending (High Priority) | Mid July |
| Transcripts | Student | Clicking of line in transcript will seek to section of the video | Pending (High Priority) | Mid July |

*\*\*More features may be introduced to meet user requirements.*

## In-depth description of Pending Features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Misc | Contributor(s) | Error logging module | Pending (Medium Priority) | Mid July |

Why: The rationale behind this module is to use an error logging system (e.g. google analytics) for chrome extensions to submit error logs when something happens, so that developers can identify if something goes wrong without the user having to report it. Although this will be invaluable once the product is distributed and users start using it, its priority is currently lower than the rest of the core features.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Consistent UI & UX and functionality for single video and double video webcasts | Pending (High Priority) | Mid July |

Why: Panopto’s implementation for the sidebar (and to a certain extent the carousel as well) is different if there is only 1 video stream. All functions must work similarly on streams with only 1 video stream.

For this version, the sidebar will be the same design as the sidebar described above (under “Sidebar with tabs”).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UX | Student | Persistent settings across webcasts | Pending (High Priority) | Mid July |

Why: This is a UX improvement that allows user to set their settings and retain their settings across multiple sessions. For more details, see sidebar described above (under “Sidebar with tabs”).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Customizable carousel via settings | Pending (High Priority) | Mid July |

Why: Based on the survey, some prefer to leave the carousel unchanged, some prefer to reduce its side and some prefer to hide it entirely. A setting should be in place to accommodate their preferences.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Customizable speed modifier options | Pending (Medium Priority) | Mid July |

Why: Some students have indicated in the survey that they prefer buttons. A setting should be in place to accommodate their preferences.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Volume booster | Pending (High Priority | Mid July |

Why: Some webcasts are soft and hard to listen to based on the survey and personal experience. This feature will help amplify the gain of the webcast so that viewers can better hear the lesson. The UI design for this is still unconfirmed, but most likely put in the volume settings of the main video.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Option to make the webpage fullscreen | Pending (Medium Priority) | Mid July |

Why: This feature is very easy to implement and can be added into the settings page. By removing the navbar of the browser, it allows for more space for the webcast etc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Allow webcast to be shown even on smaller screens (responsiveness) | Pending (Medium Priority) | Mid July |

Why: A student commented that he/she normally watch webcasts on half screen so making the webpage and extension half screen friendly would be good.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UI | Student | Other miscellaneous improvements\*\* | Pending | Mid July |

Why: This feature is purposely left vague as there may be other improvements requested by the student population with regards to improvements to the UI.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Transcripts | Student | Transcripts on the sidebar | Pending (High Priority) | Mid July |
| Transcripts | Student | Highlighted line in transcript in sync with video | Pending (High Priority) | Mid July |
| Transcripts | Student | Clicking of line in transcript will seek to section of the video | Pending (High Priority) | Mid July |

Why: Transcript and clickable lines allow users to jump from one point of the webcast to another, which is very convenient if the transcripts are accurate enough for use.

# 

# Features Kept In View

Features that are put on hold indefinitely

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Misc | Student | Optimizations, use of webpack to reduce size of chrome extension and manifest links | KIV | - |

Why: A smaller file for distribution is good and will improve the scalability of the project. However, considering the file size growth (and file size reduction) is currently nominal, this is a low priority task, and may be scrapped if deemed unnecessary.

# **Usability** **/ User Interface / User Experience**

Disclaimer: Although some elements of the UI has been implemented, the focus of our developmental work in Milestone 2 was not on UI and UX but instead on the subtitles and silence detection.

Considering that the baseline of the UI and UX is what Panopto already offers (which is already quite usable), the chrome extension can actually be released now (after transcripts has been implemented) and it would still be considered an UI/UX improvement to the current webpage. Ultimately the reason why users visit the webpage is to watch a webcast, and it isn’t that hard for users to do so given the current UI.

In addition, a number of the completed changes (subtitles, transcript, silence removal) and proposed changes (volume booster etc) require no action on the user’s part and each feature / tab / setting will be appropriately labelled to help the user understand what each feature does. All customizable settings are found in 1 centralized settings tab or in the appropriate section (e.g. speed modifier or volume modifier), so it is easy for users to enable or disable the features they want. These settings will persist across sessions after they save the settings, so they don’t have to reconfigure and can immediately start watching the webcast(s).

With the improvements we can make based on the survey responses, the team is confident that we can further enhance the UI to suit each user and their wishes.

Note that the scope of this project is restricted only to pages that display the webcasts hosted on Panopto. Any other pages (e.g. webcast listings, webcast/bookmark/module search etc) is untouched. This is because these other features (e.g. bookmarks) are not used by the students, or webcast listings and module search are already improved upon by LumiNUS.

# Types of methods used to evaluate the suitability of solution(s):

Disclaimer: These options offered by the Milestone 2 criteria sheet focuses largely on the suitability of the UI / UX of the application, which wasn’t the focus of our developmental work in Milestone 2 itself.

☑ Expert / self evaluation

**UI / UX / Usability**: Self-evaluation has been performed. I have been using and testing the application on my own for the purposes of studying AI using the open weblectures.

**Silence Detection:** Technically, “expert evaluation” is involved, because the implementation is based on the research paper <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4142156/> . By definition, however, expert evaluation is usually with regards to the usability of the application.

☑ Cognitive walkthrough / ~~heuristic evaluation~~

**UI / UX / Usability**: The team and a few of our friends have been trying the application out. As of now, the cognitive walkthrough is as follows (it is very simple):

1. Visit any webcast (by the IVLE open weblectures list)
2. Watch the webcast
3. Adjust the speed modifier if necessary

There is currently no way that the user can get lost in the UI at this point.

☒ Simulated user focus group

☑ Actual user focus group / ~~interview~~

The team and a few of our friends have been trying the application out and giving suggestions. However, because most of the work done in Milestone 2 doesn’t really affect the UI or UX (and testing was focused on silence detection and subtitles), there were no major complaints on the UI or UX in particular.

☑ Usability testing with potential users on low­ fidelity artefacts (e.g., PowerPoint mockup)

A survey comprising of with low fidelity artefacts to help respondents envision the improvements to Panopto has been released. Feedback has been received and is being applied to future features (e.g. greater customizability due to the large variance of preferences by the users in general).

# Survey Results







