Google Summer of Code 2018
Lesson translation dashboard
(Author: Sandeep Dubey)

Project Details:

Name of the project: Lesson translation dashboard

Why are you interested in working with Oppia?

Being a tutor and a web developer, who has learnt most of his skills from the internet world, I always wanted to do something for Online learners/creators. When I was planning to start contributing to OpenSource I searched “tutor” on github (Python as preference) and I came across Oppia/oppia (on top). I feel lucky that I started my opensource contribution journey with Oppia (also, I help new contributors to let them feel the same), while working on developing the core project I took part in *oppia-basic-mathematics* (2016-17) community and helped the community in creating lessons and while doing this I realized the potential of Oppia for creators, gradually, as I spent my time in the platform as a creator and a developer. I never really felt how it actually feels to be an Oppia learner in real, till one day I started play testing those explorations with my students and I saw the excitement & joy on their faces while they were playing with those explorations. (Most of these credits goes to the creator and his creativity for writing stories).

I have made a lot of stories and friends as well while going through this journey, and now I really feel Oppia as my own project. It will be great if I’ll get an opportunity to work for Oppia, which will let me contribute more to the core project and feel more connected to Oppia Foundation team and their mission.

.

What interests you about this project? Why is it worth doing?

I have worked with Oppia’s Delhi RCT program as an audio translator for the Fraction Collection. One of the important reason behind the success of that RCT program was the translated audio, and we found students were more interested with audio assets incorporated in the program. So being a member of Audio-Translation team for RCT, I have faced a lot of issues while recording, uploading & reviewing audio.

As this translated audio assets were something new to Oppia, it wasn’t easy for translators to work on audio related issues on editors tab. There were a lot of issues and difficulties which we had to face.

Few of them are listed below:

- Recording:
    - Recording audio in other device/platform.
    - Making those audio files compatible with Oppia's Audio Player.
- Uploading:
    - Uploading single audio at a time, this was tedious, time consuming & repetitive process.
    - Selecting the respective language for each upload. (This is resolved now)
- Overall:
    - It was tough to find a place where I have missed uploading audio, one had to take a look at each and every state and their components.
    - A translator would have to go through the entire exploration to update audio if the editor changed a single component of a card.

Keeping all these things in my mind, I found that a translator would need a separate page to do his/her role in a easy and efficient way. Instead of wasting most of their time in searching for missing places, converting audio files, uploading a single audio at a time etc.

Prior experience:

I have been contributing to Oppia for about a year. From my contributions to Oppia, I have gained a fair idea about the back-end and front-end aspects of Oppia. Being a user and a creator myself, I also have a good idea about what goes into making a delightful and cohesive user experience for both learners and creators.

My contribution to oppia can be seen here, also I have contributed to other open-source projects like google/blockly (I haven't done a lot of contribution there but I have a great affection toward this project and I'll be contributing to this project in future). I also contribute to my college website and other projects which runs on Django framework (A private repo on github).

Links(few) to issue I have solved for oppia:

- https://github.com/oppia/oppia/pull/4360/
- https://github.com/oppia/oppia/pull/4708/
- https://github.com/oppia/oppia/pull/4687/
- https://github.com/oppia/oppia/pull/4264/
- https://github.com/oppia/oppia/pull/4078/
- https://github.com/oppia/oppia/pull/4039/

Survey and Interview:

To make this translation page more useful for end users, I have done a survey and an interview with the audio translators of Oppia. This survey and the interview helped me to know the problems they used to face during audio translation work and it also helped me to discuss the preferred solution for the problems. In the below table I have enlisted few of the major problems and their response/solution which come out through the survey:

Problem	Solution/ideas	Response
Recording and converting audio on other platform/devices.	Recording audio on the oppia platform.	Yes (80%) Somewhat (20%) No (0%)
Uploading single audio at a time.	Multiple audio functionality for translators.	Yes (100%) No (0%)

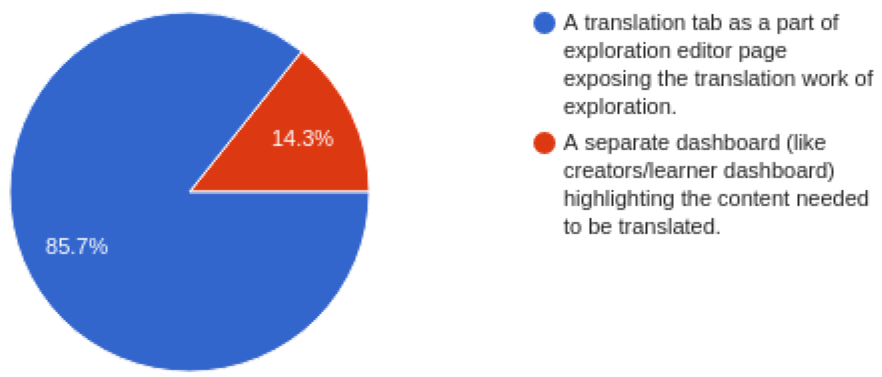
Finding places where audio needs to be updated or missing.	Status for each card and their components.	Yes(100%)
		No(0%)

Positioning translation page.	1. As a part of editor page. 2. Separated dashboard.	1 (85.7%) 2 (14.3%)
---	Accessibility of stats tab for a translator.	Yes (66.3%) No (33.3%)

Link to all the response: PDF, SPREADSHEET.
Link to the survey form: Google form.

Outcome of survey:

Positioning translation page:



Reasons:

- Translating an exploration is like story telling, so the translator use to connect themself with the flow of the exploration. It's pretty important for them to understand the reason behind any response while translating so that they can express them through the voice. So just enlisting the content in dashboard won't be that powerful way for story telling.
- Need of most of the functionality of editors page (like stats, history, feedback etc.) to the translators.
    - Few comments via translators in support of this:

      Access to stats tab: (1) Translator can analyze by the most frequent answer, which part of the concept is being misunderstood by most of the students and shape his/her translation accordingly. (2) In future we can also have stats regarding usage of audio.

      Access to feedback tab: This will help translators to read and reply on feedback related to audio.

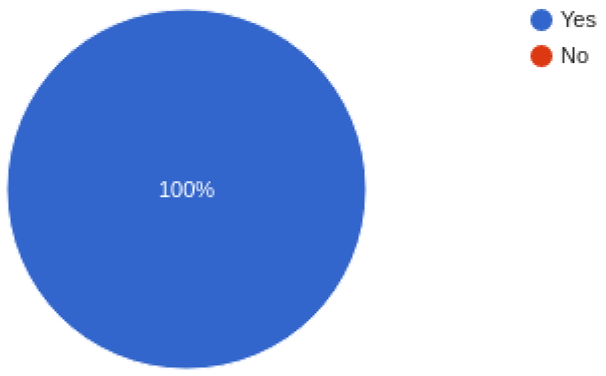
Comments in support:

- Having an Audio Translation Tab makes "Exploration" as our main focus (which actually should be). That makes exploration editors & translators, work as a team to prepare a great story. We are connected to the whole process, can suggest changes anywhere (editor things, which we did when we used to ask Sean to add an image somewhere or rephrase something. And in translation things similarly) *--by Shubha Gupta (translator)*

Conclusion::

The New translation panel should be a separate tab (translation tab) in editor page.

Do you think "Multiple audio upload" functionality will be your option to upload audio?



Reasons:

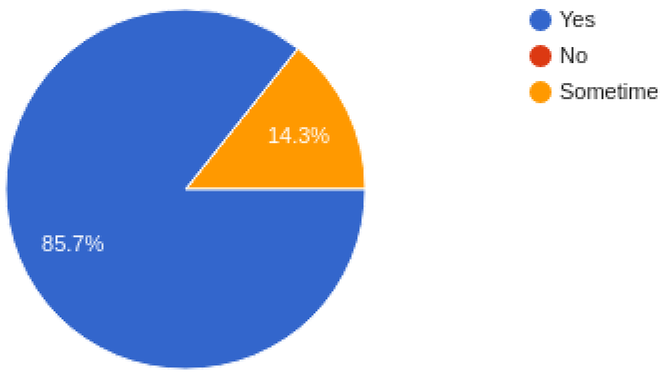
- Currently a translator can upload a single audio at a time and for that they have to do the same work repeatedly on each card for it's each component.

Comments in support:

- Bulk upload all the audios and let Oppia figure out how to map audios to their corresponding content. --*by Anmol Shukla (audio translator)*
- Bulk upload is a must! --*by Shubha Gupta (translator)*

Conclusion:

Do you transcript the content before recording?



Reasons:

- Transcripting the translated content helps translators to record audio smoothly i.e, they will just have to read the transcript.

- This also helps translators to keep records of the content and if in future editor do some changes then they'll just have to translate the newly updated part.

Conclusion:

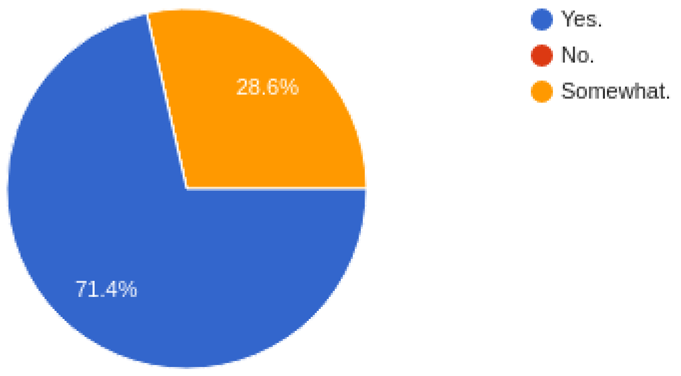
Translators need a place to keep their transcript for future use or reviewing audio.

NOTE: This idea was a part of an another Oppia project "Audio Bar Improvements", after the survey report we (me and mentors) were deciding to include this part too in this project, but on discussions we found this part needs a lot of back-end and design work, so we kept this as a future work for this project.

"I think it's worth considering the scope of what is manageable to do within GSoC.
My feeling is that adding transcripts on top of everything may seem a bit too much.
This is because there is not yet any backend architecture for handling transcripts either,
so you would have to devise the backend and storage models for that first.**"**

-- by Tony Jiang.

Will it be okay for you to directly record on oppia platform?



Reasons:

- Currently a translator would have to record audio on other platform and then they have to make those audios compatible with Oppia's demands using audio converter, this make this process tedious.

Conclusion:

Let translators record audio on the oppia platform.

Structural Layout and Integration of the translation_tab:

[R = Read and W = Write]

Back-end:

Introducing a new role:

- ROLE:
    - Add new role "translator" for an exploration.
    - "editor/collaborator", "owner" and "translator" will have read/write access to the translation page.
    - "translator" will only have read access to the editor page (like a guest user).

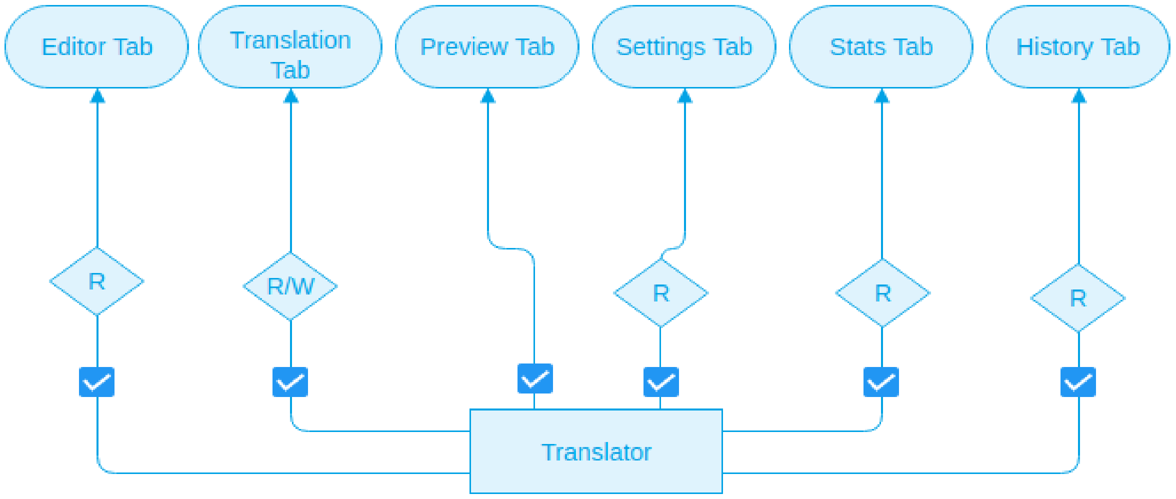


Figure showing translator accessibility on exploration editor page.

Front-end:

A new translation tab will be included in the current exploration editor page which will be accessible to the translator of the exploration.

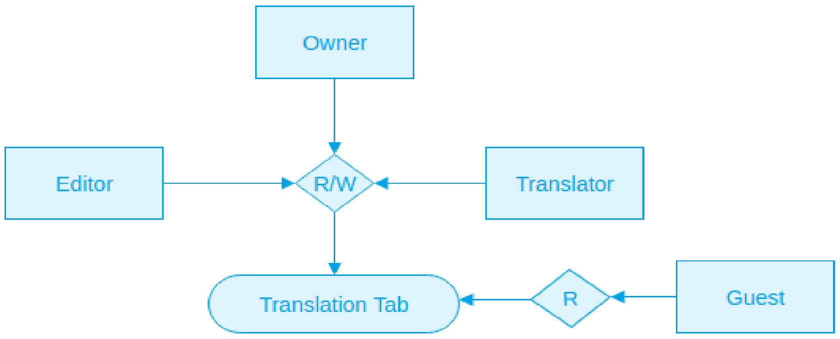


Figure showing accessibility of different roles on translation page.

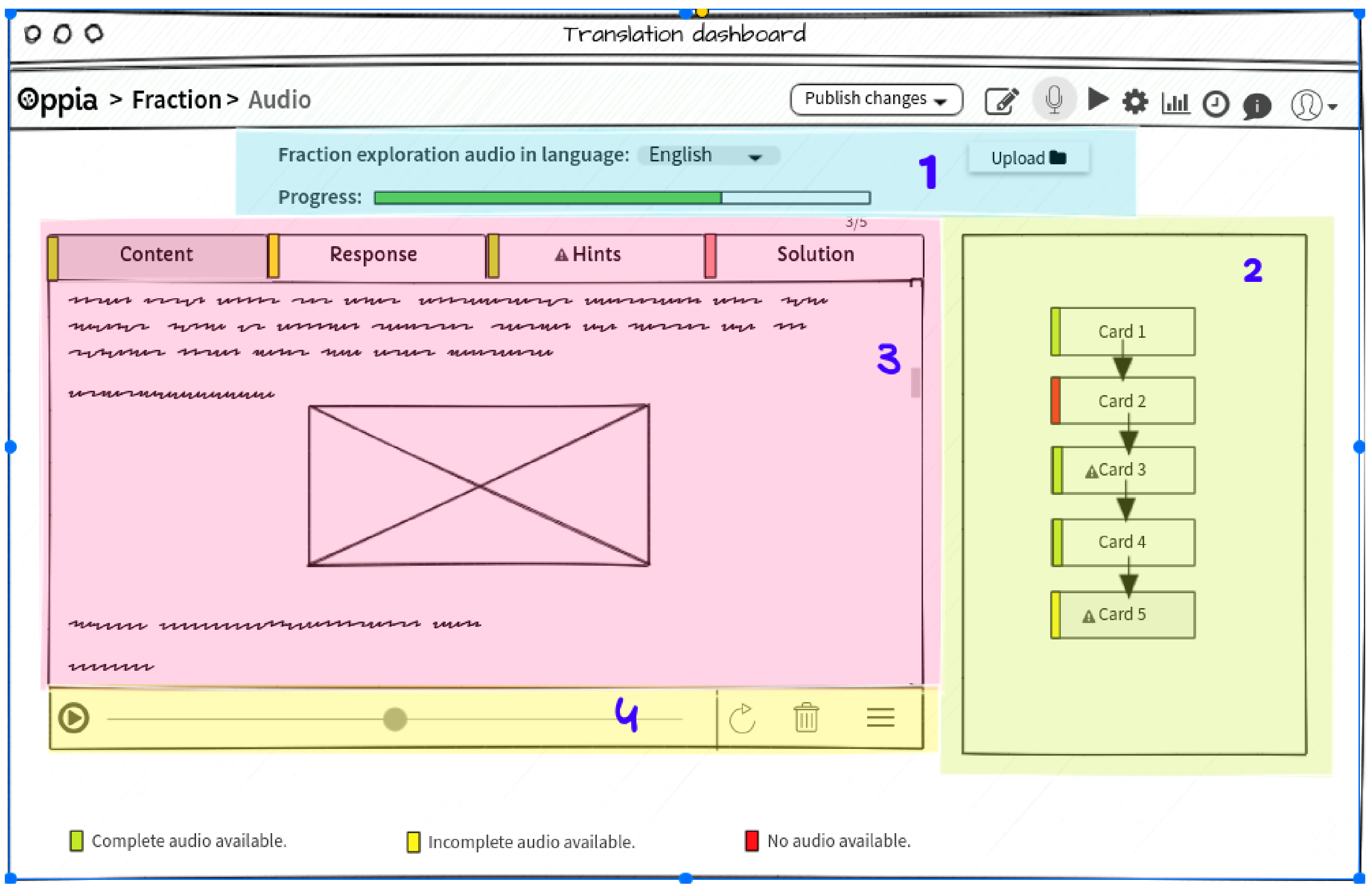


Figure showing the new translation tab (different sections are highlighted and annotated below with the num.)

Features:

- ★ Status of the entire exploration on selected language. (1)
- ★ Status of each state in preferred language. (2)
- ★ Status of each component of the state (content/responses/hints/feedback) (3)
- ★ Audio recording. (4)
- ★ Multiple audio upload. (1)
- ★ Progress bar for the translation work. (1)

User flow:

Assigning a translator for an exploration:

This will be similar to the current flow of assign roles to an exploration.

Owner/Manager → Settings → Roles → Select translator → Provide username → Save.

Translating an exploration:

When a translator will open the translation tab entire tab will show the status of the cached language if cache language will be undefined default language will be selected(default will be English). Once a user will change the language, this info will be cached.

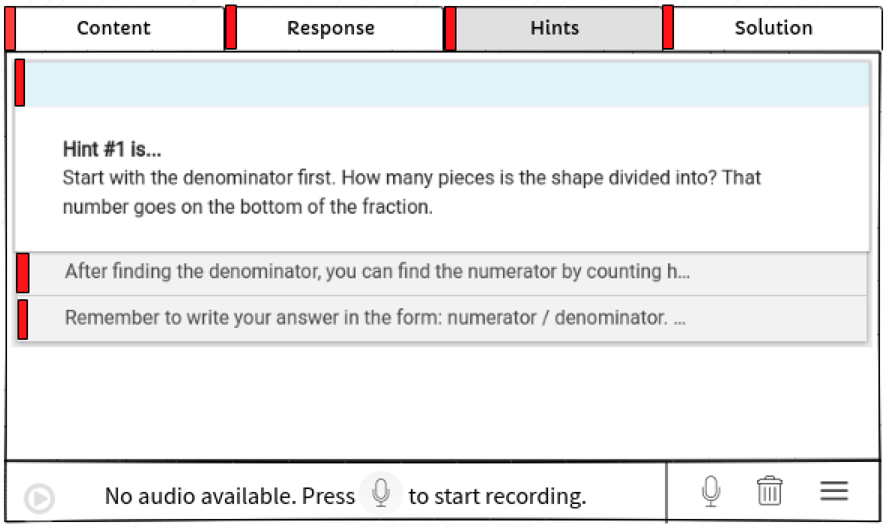
1. Translation tab's header will have 3 features:
   - Select Language: This will be the core of the entire page as the status is dependent on preferred language.
   - Progress bar: This will show the progress of translation in the selected language i.e, total no. of audio uploaded / total numbers of audio requires. (Total no. of audio = $\sum(\text{content} + \text{hints} + \text{response} + \text{solution})$ for all states.)
   - Multiple audio upload button: A button in top right corner which will help translator to upload bulk of audios in one chance. Entire audio upload process is given below.

2. State translation graph will show the status of each state:
   - **Red**: If none of the component (content/hints/response etc.) will have audio in the selected language.
   - **Yellow**: If any of the component of the state will not have audio in the selected language.
   - **Green**: If all the components are having audio in the selected language.
   - **Warning(⚠)**: If any of the component will require a change (marked by the editor).

3. State components section will show the status and content of each component:
   - **Red**: If the component (content/hints/response etc.) will not have audio in the selected language.
   - **Yellow**: If any part of the component will not have audio in the selected language.
   - **Green**: If all the part of components are having audio in the selected language.
   - **Warning(⚠)**: If any part of the component will require a change (marked by the editor).

4. Audio-bar is the most dynamic component of this tab. Entire audio upload, recording and other related interaction (record (like record, re-record, pause, stop), audio (play, delete) etc.) will be done via this section.

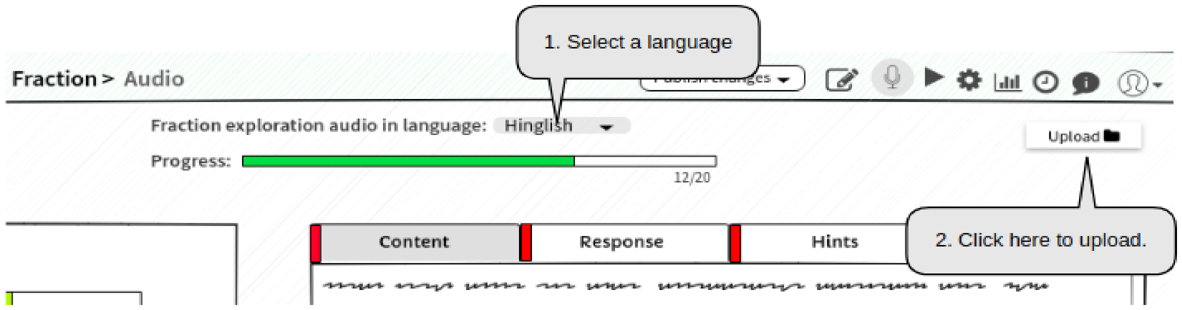
Note: Hint and Response sections will look different from the Content and Solution sections as because hint and response will be in multiple subsection. See Image below


Content
Response
Hints
Solution
Hint #1 is...
Start with the denominator first. How many pieces is the shape divided into? That number goes on the bottom of the fraction.
After finding the denominator, you can find the numerator by counting h...
Remember to write your answer in the form: numerator / denominator. ...
No audio available. Press to start recording.

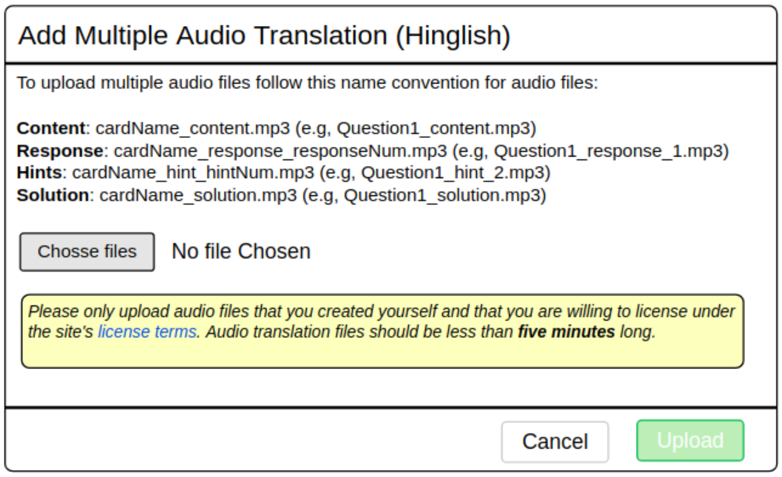
Multiple audio files upload:

Multiple audio files can be uploaded via the translation tab. To upload multiple audio in Hinglish language can be done via these steps:

1. Select a language.
2. Click the top right "Upload dir." button.
3. And follow the instruction provided in a modal. (Click here to see this in more interactive way.)



- After a user clicks on "Upload" button a modal will appear (image below):



Add Multiple Audio Translation (Hinglish)

To upload multiple audio files follow this name convention for audio files:

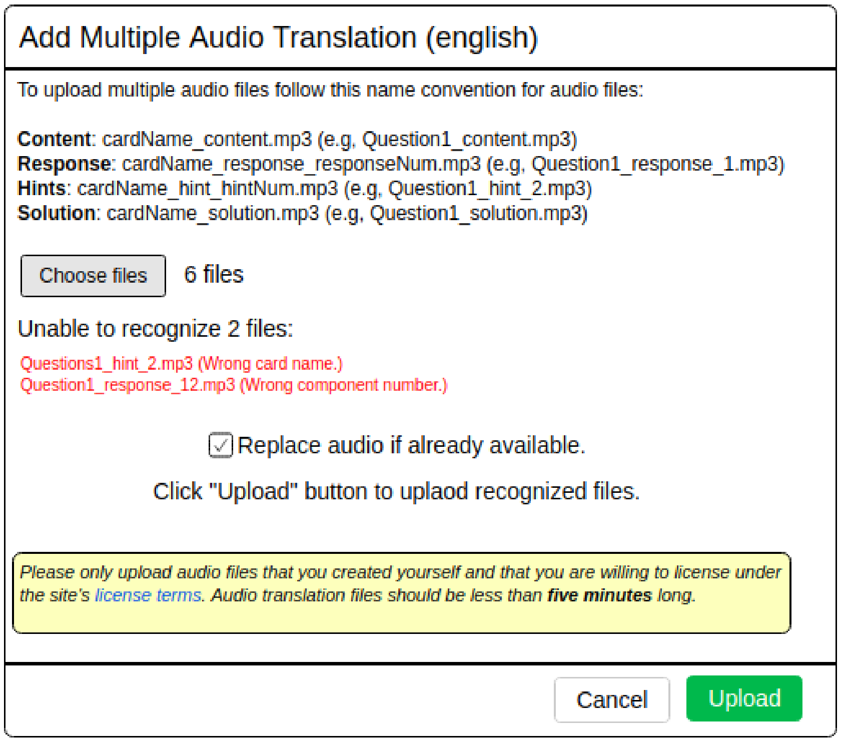
Content: cardName_content.mp3 (e.g, Question1_content.mp3)
Response: cardName_response_responseNum.mp3 (e.g, Question1_response_1.mp3)
Hints: cardName_hint_hintNum.mp3 (e.g, Question1_hint_2.mp3)
Solution: cardName_solution.mp3 (e.g, Question1_solution.mp3)

Chosse files No file Chosen

*Please only upload audio files that you created yourself and that you are willing to license under the site's license terms. Audio translation files should be less than **five minutes** long.*

Cancel Upload

- After a user selects (assume 6 files):
    - In this stage name of the audio file names will be parsed and if the filename will not be recognized then the name of those files will be shown to the users in "**Unable to recognize X files:**" where X is the number of unrecognized files.
    - Possible reasons behind errors:
        - Wrong card name.
        - Wrong component name.
        - Components like hint and response are without component number.
        - Wrong Component number.
            - If an user wants to assign an audio to 10th response although there's just 5 response in that card.
            - Non-numeric character in place of component number.
    - After verification of filename if any of the files will be found correct then translator would have chance to upload "only" the correct file(s) through "**Upload**" button otherwise they can terminate this process through "**Cancel**" button.
    - Before uploading any files translators also have an option to replace audio files through a check box.
        - If "Replace audio if already available." is unchecked then audio will be uploaded to the components if and only if there will be no audio available for that component (component and card name will be decided through the audio filename).
        - If "Replace audio if already available." is unchecked then the audio which are already available in the components will be replaced by the new audio.


Add Multiple Audio Translation (english)

To upload multiple audio files follow this name convention for audio files:

Content: cardName_content.mp3 (e.g, Question1_content.mp3)
Response: cardName_response_responseNum.mp3 (e.g, Question1_response_1.mp3)
Hints: cardName_hint_hintNum.mp3 (e.g, Question1_hint_2.mp3)
Solution: cardName_solution.mp3 (e.g, Question1_solution.mp3)

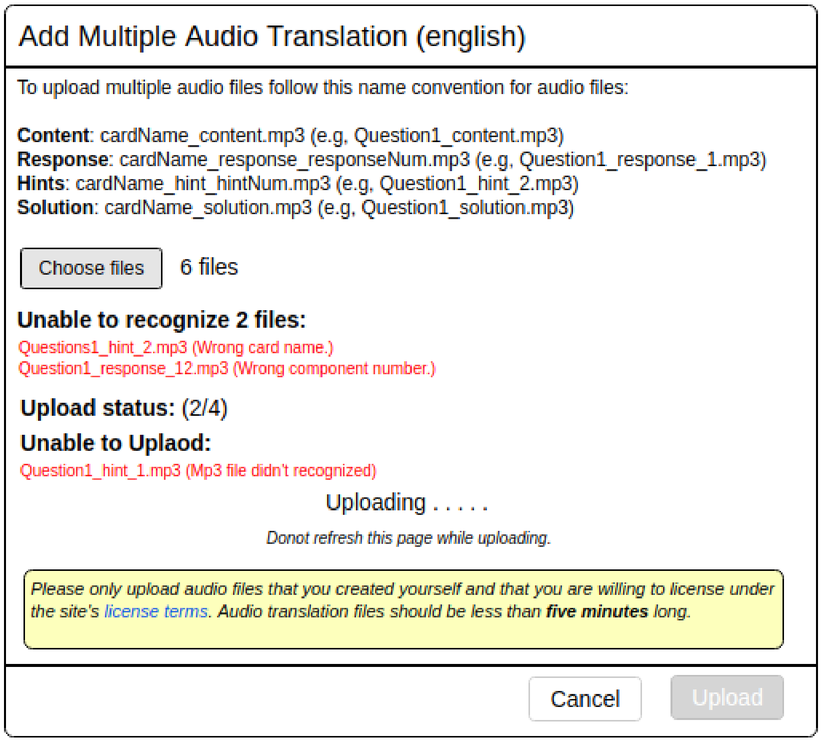
Unable to recognize 2 files:
Questions1_hint_2.mp3 (Wrong card name.)
Question1_response_12.mp3 (Wrong component number.)

[✓] Replace audio if already available.

Click "Upload" button to uplaod recognized files.

*Please only upload audio files that you created yourself and that you are willing to license under the site's license terms. Audio translation files should be less than **five minutes** long.*

- Once a user clicks “Upload” button Upload process starts:
    - In this stage audio files with correct name will start getting uploaded and the upload status will get updated accordingly. If there will be any issue regarding uploading any audio then this will be shown to the translator in “**Unable to Upload:**” section of the modal, but the upload process will be continued for other files.



Add Multiple Audio Translation (english)

To upload multiple audio files follow this name convention for audio files:

Content: cardName_content.mp3 (e.g, Question1_content.mp3)
Response: cardName_response_responseNum.mp3 (e.g, Question1_response_1.mp3)
Hints: cardName_hint_hintNum.mp3 (e.g, Question1_hint_2.mp3)
Solution: cardName_solution.mp3 (e.g, Question1_solution.mp3)

Choose files 6 files

Unable to recognize 2 files:
Questions1_hint_2.mp3 (Wrong card name.)
Question1_response_12.mp3 (Wrong component number.)

Upload status: (2/4)
Unable to Uplaod:
Question1_hint_1.mp3 (Mp3 file didn't recognized)

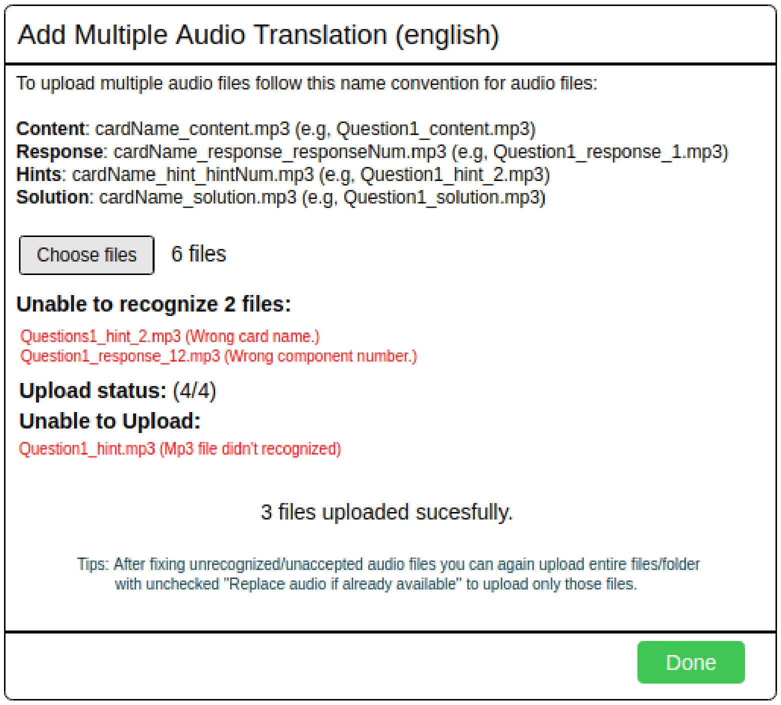
Uploading . . . . .

Donot refresh this page while uploading.

*Please only upload audio files that you created yourself and that you are willing to license under the site's license terms. Audio translation files should be less than **five minutes** long.*

Cancel Upload

- Once the upload process completed:



Add Multiple Audio Translation (english)

To upload multiple audio files follow this name convention for audio files:

Content: cardName_content.mp3 (e.g, Question1_content.mp3)
Response: cardName_response_responseNum.mp3 (e.g, Question1_response_1.mp3)
Hints: cardName_hint_hintNum.mp3 (e.g, Question1_hint_2.mp3)
Solution: cardName_solution.mp3 (e.g, Question1_solution.mp3)

Choose files 6 files

Unable to recognize 2 files:
Questions1_hint_2.mp3 (Wrong card name.)
Question1_response_12.mp3 (Wrong component number.)

Upload status: (4/4)
Unable to Upload:
Question1_hint.mp3 (Mp3 file didn't recognized)

3 files uploaded sucesfully.

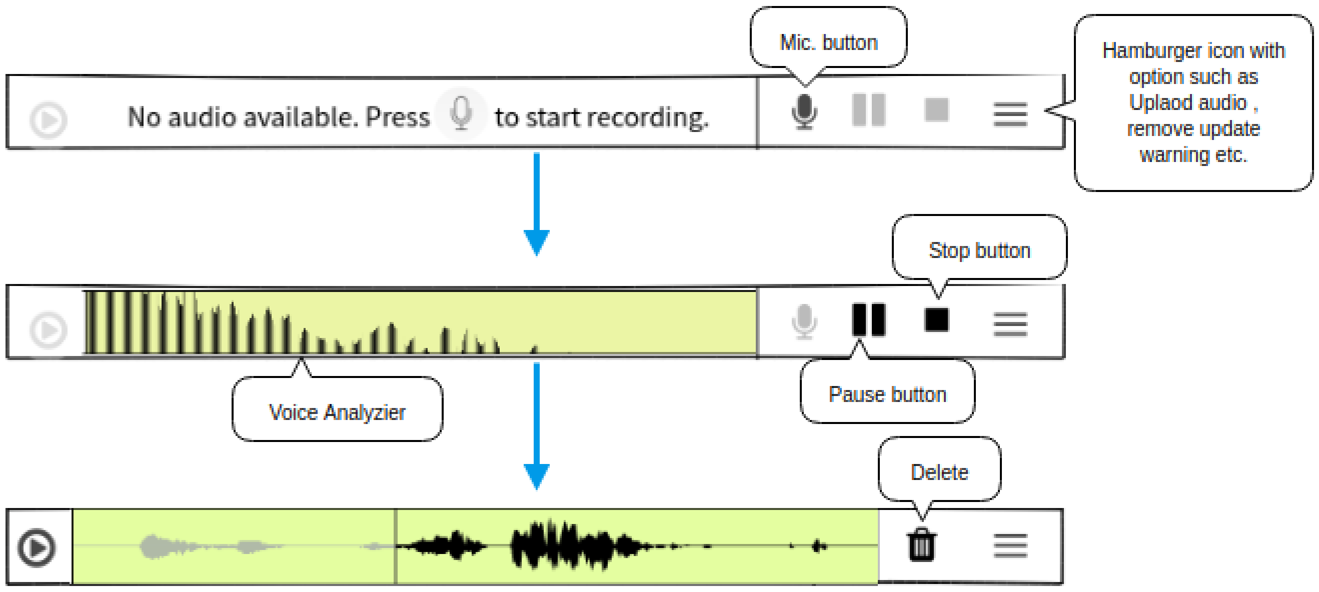
Tips: After fixing unrecognized/unaccepted audio files you can again upload entire files/folder with unchecked "Replace audio if already available" to upload only those files.

Done

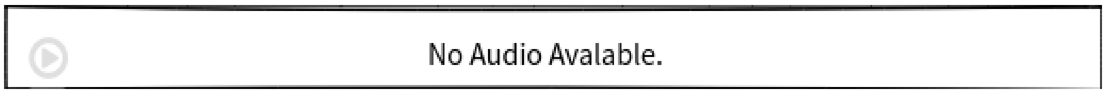
Note: After completion of the above process an user can do rectification to those audio files which failed to complete the audio upload process (because of incorrect naming, file extension issue etc.) and then re-upload those files through same process.
(They can upload entire audio files and can uncheck "Replace audio if already available." and this will take care of uploading only those files which are required or have failed earlier .)

Audio bar without audio available:

- For user having "R/W" access to translation page:

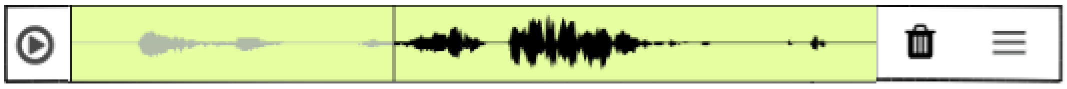

Mic. button
Hamburger icon with option such as Uplaod audio , remove update warning etc.
No audio available. Press to start recording.
Stop button
Voice Analyzier
Pause button
Delete

- For user having "R" access only:

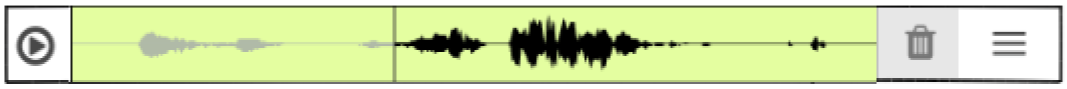


Audio bar with audio available:

- For user having "R/W" access to translation page:



- For user having "R" access only:



    - Disabled "Delete" button.

Project plan and implementation strategy:

Implementation of Multiple audio upload functionality:

Multiple audio upload will be handled via the help of audio file name and as underscore (_) is a part of INVALID_NAME_CHARS (i.e, it cannot be a part of state name) this will help us to determine the position of audio file in the exploration (i.e, where we have to place it.). Name convention for the each components are defined below:

- **Content**: cardName_content.mp3 (e.g, Question1_content.mp3)
- **Response**: cardName_response_responseNum.mp3 (e.g, Question1_response_1.mp3)
- **Hints**: cardName_hint_hintNum.mp3 (e.g, Question1_hint_2.mp3)
- **Solution**: cardName_solution.mp3 (e.g, Question1_solution.mp3)

Multiple audio upload will follow these steps (in brief):

1. Validate file names:
    a. Use *isValidAudioFileName(fileName)* from AudioUploadService to validate each file name.
    b. Show the invalid files to users and remove those audio files from the audio list.
2. Upload audio:
    a. Use *AudioUploadService.uploadAudio* to save audio files (one at a time).
    b. Show the name and error message for files which are rejected or having problem in uploading.
    c. Save the language, filename and size of the audio for that component of exploration in it's subtitledHtml.

[isValidAudioFileName and uploadAudio functions are explained below in detail.]

Implementation of Audio recording functionality:

I am going to use angularAudioRecorder, which is basically an AngularJS plugin for recording audio input.

Reason behind the choice of plugin:

- Well suited API for development.
- Well documented.
- Ability to convert the resulting Wave audio file into MP3.
- Uses FlashWaveRecorder in web browsers that do not support *getUserMedia().

Recorder Options:

- **Record**: To record an audio.
- **Delete**: To delete recorded audio.

- **Stop**: To stop recording audio.
- **Play**: To play the recorded audio.
- **Pause**: To pause the audio player.
- **Show audio waveform**: Show the audio waveform of the recorded/uploaded audio.
- **Show Recording analyzer**: Show the waveform which is getting recorded.

Show Recording analyzer & Show audio waveform:

Both of these can be easily implemented using *angularAudioRecorder* plugin, for audio waveform this plugin uses wavesurfer.js.

Directives for analyzer:

<ng-audio-recorder-analyzer></ng-audio-recorder-analyzer>

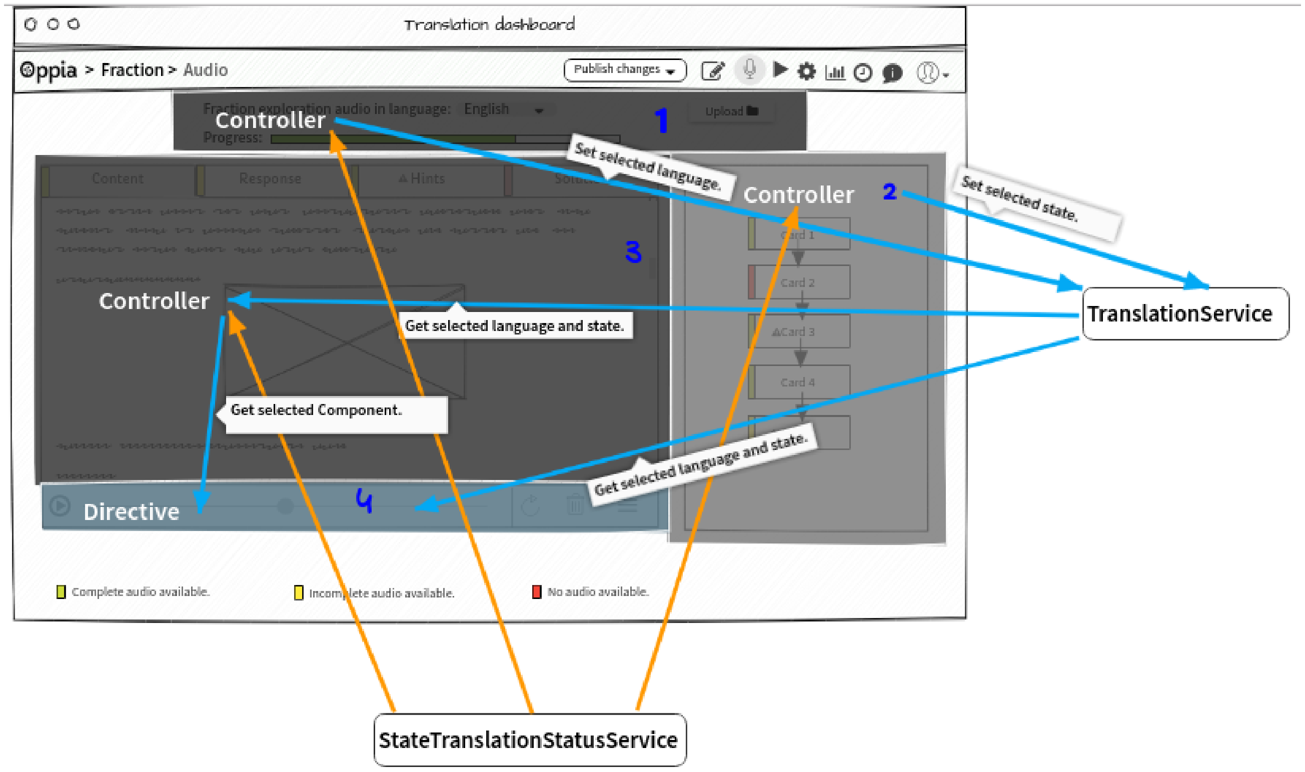
Directive for recorded audio wave visualization:

<ng-audio-recorder-wave-view></ng-audio-recorder-wave-view >

I have also created a demo version to check the recording quality, mic. intensity analyzer & audio waveform of the audio produced through this plugin: https://codepen.io/S4ND33P/pen/aYyOEv

Frontend implementation:

The entire front end part is broken down into 3 part (image below) each part is connected via "TraranslationService".



Files overview:

- exploration_editor/translation_tab/
    - translation_tab.html
        - translator_header.html (1)
            - TranslatorHeader.js (Controller)
            - Multiple_upload_modal_directive.html
        - state_translator.html (3)
            - StateTranslator.js (Controller)
        - audio_translation_bar_directive.html (4)
            - AudioTranslationBarDirective.js
        - state_translation_status_graph.html (2)
            - StateTranslationStatusGraph.js (Controller)
        - StateTranslationStatusService.js
        - AudioServices.js
        - TranslationService.js

translation_tab.html: This file will contain the overall structure of the translation tab.
translator_header.html: This will contain the structure of header and this will be controlled via *TranslatorHeader* controller.
TranslatorHeader.js: Controller for translator_header.html
Multiple_upload_modal_directive.html: Structure for multiple_upload_modal.
state_translator.html: This will contain the navbar to switch between the components of the selected card. This will be controlled via *StateTranslator* controller.
StateTranslator.js: Controller for state_translator.
Audio_translation_bar_directive.html: Template for AudioTranslationBarDirective.
AudioTranslationBarDirective.js: Audio translation bar Directive.
state_translation_status_graph.html: Template for translation tab's state graph.
StateTranslationStatusGraph.js: Controller for state graph.

StateTranslationStatusService.js: A service to provide the status of card and their components.

```
// Constants for color of status.
var COLOR_ALL_AVAILABLE = '#4EA24E';
var COLOR_NONE_AVAILABLE = '#DC143C';
var COLOR_FEW_AVAILABLE = '#FFD700';
```

```
Each status will be a dict/object factory:
{
    'color': string(any color constant),
    'warning': boolean
}
```

- *StateTranslationStatusService.getStateStatus(stateName)*
    - Returns a status dict.
- *StateTranslationStatusService.getContentStatus()*
    - *Returns a status dict.*
- *StateTranslationStatusService.getResponseStatus()*
    - Returns list(status dict, list(status dict))
        - 1st element of returned list will represent the overall response status for nav bar.
        - 2nd element of returned list will be a list of status dict representing each of the response.
- *StateTranslationStatusService.getHintsStatus()*
    - Returns list(status dict, list(status dict))
        - 1st element of returned list will represent the overall hint status for nav bar.
        - 2nd element of returned list will be a list of status dict representing each of the hints.
- *StateTranslationStatusService.getSolutionStatus()*
    - *Returns a status dict.*

AudioUploadService.js: A service to Upload audio.

- *uploadAudio()*
    - This will call *AssetsBackendApiService.saveAudio* to save audio and in case audio already available for the component and "replace audio" (default true) is false then no audio upload will happen for that file (in case of multiple audio upload). Also, this will handle single audio file upload through hamburger icon of AudioTranslationBar and recording.
- *markReplaceAudio()*
- *unmarkReplaceAudio()*
- *isValidAudioFileName(fileName)*
    - This function will use similar code structure:

```
var audioPositionList =  fileName.split('_');
```

```
// validate 2 <= audioPositionList.length() <= 3
var cardName = audioPositionList[0];
//validate cardName
var componentName = audioPositionList[1]
If (audioPositionList.length() == 2 && (componentName == 'content' ||
    componentName == 'solution')) {
            return true;
}
If (audioPositionList.length() == 3 && (componentName == 'response' ||
   componentName == 'hint')) {
            var componentNum = audioPositionList[2];
            // validate componentNum.
}
else {
        return false;
}
```

TranslationService: A service to connected the info flow between different controllers.

TranslationService.setSelectedLanguage(LanguageId)
TranslationService.getSelectedLanguage()
TranslationService.setSelectedState(stateName)
TranslationService.getSelectedState()

Milestones:

- ### Milestone 1: (7th May - 31st May)
  Summary:

  In this milestone, in first two week I'll be working on adding "translator" role for the exploration in the back-end and making it compatible with current oppia's situation i.e, allowing translators to access audio translation bar of the editor page. In the remaining two weeks I'll be writing necessary service for the new audio translation tab. So after this month release (6th may release cut) an exploration creator will be able to assign a translator to their exploration who(translator) can access the audio translation bar of the editor tab.

  - Milestone 1.1: (1 week)
    - Adding "translator" role for the exploration.
      - Adding new entity transltor_ids in ExplorationSummary and ActivityRights object.
      - Adding `transltor_ids` in ExplorationSummaryModel and ExplorationRightsModel.
      - Writing @acl_decorators for can_translate_exploration.
    - Writing back-end test for the same.
  - Milestone 1.2: (1 week)
    - Making old audio translation bar of editors tab accessible to the new "translator" role.
    - Making other tabs (stats tab, preview tab etc.) accessible to the new role.
    - Testing and making this new role compatible with the oppia in every place.
  - Milestone 1.3: (1 week)
    - Writing StateTranslationStatusService.js (with unit test).
    - Writing TranslationService.js (with unit test).

- ### Milestone 2: (1st June - 6th July)
  Summary:

  In this milestone, I'll be working on creating the frontend of the new translation tab. After this release (6th july release cut) translator will be able to access their new translation tab with all of the functionality.

  - Milestone 2.1: (1 week)
    - Making editor navbar functional for the translator.
    - Writing translator_header.html and it's controller.
    - Writing multiple_upload_modal_directive.html and it's controller.
  - Milestone 2.2: (1 week)
    - Writing state_translation_status_graph.html & StateTranslationStatusGraph.js.

        - This would be done using state-graph-viz directive.

    - Milestone 2.3: (1 week)
        - Designing state_translator.html and writing controller.
        - Integrating this with the graph and header.

    - Milestone 2.4: (1 week)
        - Writing AudioServices.js (with unit test).
        - Writing AudioTranslationBarDirective.js.
        - Removing audio-translation directive from the editor page.

- ### Milestone 3: (8th July - 1st August)

  Summary:

  In this milestone, I'll take feedback from the users for any further changes and most of the time will be spent in writing e2e tests and documentation.

    - Milestone 3.1: (1 week)
        - Ensure the translation tab is working smoothly, take community feedback for any further changes.
        - Analyzing feedbacks and doing required changes.
    - Milestone 3.2: (1 week)
        - Writing e2e tests.
    - Milestone 3.3: (1 week)
        - Documentation.

Timeline:

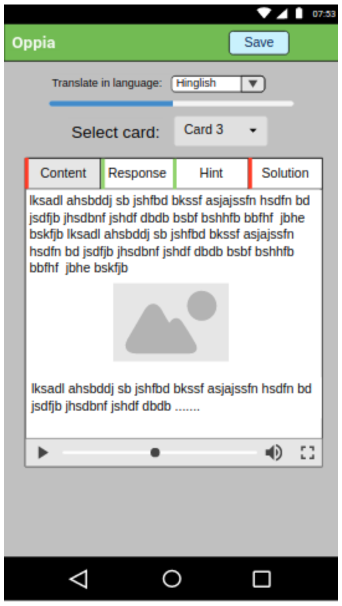
The deadlines outlined above is the worst case timeline - by these dates, the given goals will definitely be completed.

Future work:

There has been many ways to make this translation page more useful to translators which can be done after the GSoC period:

- Adding option to simple edits of audio, after recording it on translation page.
- Providing a way for translator to save their transcript on translation page.
- Providing a place for audio translator to write and save the audio transcript. (If this will not be covered in other project.)

- Make this page mobile friendly, which can be done if and only if the editor page will be mobile friendly.



Sample design for mobile-friendly translation page.

Summer Plans:

- **Which time zone(s) will you primarily be in during the summer?**

  I will be in Indian Standard Time (GMT +5:30).

- **How much time will you be able to commit to this project?**

  I'll be able to give around 6-7 hrs. a day and 6 days a week i.e, around 36-42 hrs. a week. (Last week of may will be bit busy for me so I will give up to 4 hrs. a day)

- **Please list jobs, summer classes, and other obligations you may need to work around.**

  Currently I'm in final year of B.Tech and I have my exam in May(last week) and we just have 3 subjects/papers in final year so I'll be busy for 5-6 days in that, after that I'll be totally free as I don't have to attend any classes or do any assignments/projects except GSoC project in this summer (June - August).

Communication :

- **What is your contact information, and preferred method of communication?**

  E-mail : dubeysandeep.in@gmail.com

  Mobile no. : +91-8981656526

  Github handle : DubeySandeep

  Oppia is active on gitter, so preferred method for most of the communication and meetings will be gitter (or Google Hangout as preferred by the mentor). I'll keep daily devlogs as recommended by mentors to document my daily work.

- **How often do you plan on communicating with your mentor?**

  We'll remain in continuous touch with gitter/Google Hangout whenever I need advice. There will be bi-weekly (or weekly based on mentor's consent) meetings to discuss the workflow to be followed.

Location: India