GSoC - 2018 Project Proposal

* Akshay Anand

## Why I am interested in working with Oppia.

I found about Oppia in December 2017, when I was searching for organizations to start for Open Source contributions, and from then on, have been a regular contributor to the Oppia platform. My initial reason to start contributing to Oppia was that I knew the languages used by the platform (AngularJS and python) well and so decided, this would be the best place to start with open source development. As I got involved in the project, I became more and more invested in the ideals of the Oppia Foundation, which is to provide a simple and easy to use learning platform in which anyone can share their knowledge about a subject to the world.

The thing that stood out most to me about the entire structure of the Oppia website was its simplicity. I have used some learning platforms, but most of those would not be ideal for anybody to use. With Oppia, as seen from the recently conducted RCT, even primary school children were able to use Oppia with ease, which is something to be really proud of.

Coming to the RCT, I was also involved in developing certain new features which was tested in the recently conducted RCT. From the review that we had after it, everyone shared a lot of insights on how to make Oppia better and even more easy to use and intuitive for people to use.

Also, interacting with the community as a whole, it is evident that everyone, from different parts of the world, are working toward a common goal for providing free education to anyone in a simple and easy to access platform.

## My interests about this project and why I think it’s worth doing.

Going back to the RCT, as I mentioned, I was involved in developing a new feature for it, which was refresher explorations. This feature, which although served it purpose to some extent, was not a very streamlined approach to the problem, and hence, when I found a project which replaced that with a much better alternative, I thought, this would be the perfect project for me to do.

This would also greatly increase the learner experience as currently, with refresher exploration, a lot of redirection to and from pages are present, which could confuse learners, while with Skills, it is going to be played as a part of the exploration itself, and therefore would make the transition to and from a “refresher skill” seamless.

Coming to the introduction of topics and stories to Oppia, to replace collections, these would also greatly increase the site’s functionality. One problem, that was seen in the RCT, that could be fixed is that the learner was jumping to later explorations in a collection, without completing the previous ones. With the skills construct added to stories, this could be prevented and as such, make sure that the learner knows all the prerequisites before starting a lesson.

## Prior Experience

I was part of the web operation teams at the technical festival (Shaastra) of our institute (IIT Madras) in my 3rd semester in college ( I am currently in my 4th Semester). I had started getting involved with web development as a preparation for this in my first year itself. In this team, we had used MEAN stack for creating the web portals and hence I got hands-on experience with general backend development (node js in particular) as well as AngularJS. I had already learnt python as a part of a course in our college, and thus combining this with my knowledge on backend development as well as AngularJS, I was able to grasp the overall flow of control in the Oppia codebase, as well as knowing which files to change to implement specific features.

Other than this, I have also participated in hackathons related to web development and also done competitive coding, in which I participated in major competitions like Codechef Snackdown (passed pre-elimination stage), ACM-ICPC (qualified for regionals), Google Code Jam (passed 1st stage) etc. I have also done contests in coding platforms like Codechef and Codeforces. All of this also helped me in tackling bugs or adding features to the codebase as it made me think about effective ways to do the same.

## Link to Pull Requests with Oppia

The first major PR I had of fixing a bug in non-inline interactions: [#4232](https://github.com/oppia/oppia/pull/4232)

My PRs related to refresher explorations: [#4307](https://github.com/oppia/oppia/pull/4307), [#4344](https://github.com/oppia/oppia/pull/4344), [#4373](https://github.com/oppia/oppia/pull/4373), [#4611](https://github.com/oppia/oppia/pull/4611), [#4644](https://github.com/oppia/oppia/pull/4644)

The complete list of pull requests that were merged with Oppia is [here](https://github.com/oppia/oppia/pulls?q=is%3Apr+author%3Aaks681+is%3Aclosed).

# Project Plan

## Current skills implementation and additions done in this project

Currently, the exploration title for each node in a collection is taken as the skill acquired when that node is completed and that is stored in the skill ids section for a collection. For explorations, for certain answer groups, redirection is currently being done to other independent explorations called refresher explorations.

Some shortcomings of the above method are that, the refreshers need not be full-fledged explorations, but just a concept card explaining the lacking skill, and a series of questions that test that skill. Also, the multiple redirections that happen during an exploration play, might confuse the learner, and coming to collections, the skills should not be something restricted to a collection, but it should be a global construct, that is uniform across a general topic.

Hence, the existing collections are to be replaced with topics and story constructs. To achieve this, this project lays the groundwork by adding the topics and stories constructs in the backend and hence, once these are fully implemented in the future, all the collections can be migrated to stories and topics, and collections can be removed.

Also, in this project, the usage of refresher exploration is being replaced by the general skills construct which consists of a concept card and a series of questions testing that concept and the redirections to these skills during exploration play would not require a page reload, as the skill states can be loaded beside the current exploration itself, and played in the same window.

# Milestone 1: Complete the backend required for skills, topics and stories and the frontend for a story and topic editor

In this milestone, the backend models, domain objects and controllers related to both the creator and learner experience for skills, and the creator experience for topics and stories is implemented. In addition to this, a simple topic and story editor is also done.

The way the UI is implemented is as follows:

* In the dropdown in the main header, a topics and skills editor link would be there, if the logged in user is an admin.
* Clicking this would go to the topics and skills editor library page, where a fixed number of topics and skills currently in the database would be displayed.
* Here, there would be a Create Topic and Create Skill button which opens the topic and skill editor respectively.
* Then, inside either the Topic editor, the Create Story button would be present to add a story to a topic.

After this milestone, stories and topics editor would be done, and hence new topics and stories can be added to the database. Though, prerequisite skills won’t be there, as the skills editor is done as part of the second milestone.

As far as testing its working, a new topic can be created for Fractions, and the current series of explorations in the Fractions collection can be split up into discrete storylines and can be added as stories (that may or may not be linear) to this topic.

## Topics

A topic is high-level concept that consists of one or more stories and skills and ties them all in to completely explain a concept. Eg: Fractions.

### Schema for Topic

* **Name**: The name of the topic
* **Description**: This is the content to be displayed on the landing page that will explain what the learner will learn through the topic.
* **Canonical story IDs**: These will be a list of ordered story ids that are linked to each other i.e there will be an overall story arc across the list and the learner would play them in a specific order.
* **Optional story IDs:** These will be a list of independent story ids that are optional and are not directly related to the main storyline.
* **Skill IDs:** These consist of the set of skills that the learner will acquire as he/she progresses through the topic.
* **Version:** The current version of a particular topic ( will increment after each update ).

### Schema for TopicSummary

This model would be responsible for storing the summary details of topics, to be displayed as tiled in the topics and skills editor page

* **Name**: The name of the topic
* **topic\_id**: the id of the topic that this summary corresponds to.

### Schema for TopicCommitLogEntry

This model would be responsible for storing the log of all commits to topics. Every time, an edit is made to a topic, an entry is made in this model.

* **User\_id**: The id of the user making the commit.
* **Username**: The username of the user at the time of making the commit.
* **Topic\_id**: the id of the topic that was updated during this particular commit.
* **Commit\_type**: The type of the commit. Currently, this would contain one of ‘create’, ‘edit’ or ‘delete’. Once, the topic editor becomes more full fledged, ‘revert’ can also be added.
* **Commit\_message:** The message given by the user regarding the changes in this commit.
* **Commit\_cmds:** The commit command dicts for this commit.
* **Version:** The version number of the topic after the commit.