Sugarizer Story Activity

Introduction

Name - Adityakumar Sinha

Emails - adityakumar113141@gmail.com (Primary)
adityakumar.sinha@students.iiit.ac.in (Professional)

Github - aditya113141

IRC (freenode) : aditya113141

LinkedIn - https://www.linkedin.com/in/adityakumar-sinha-485a40193/

Website - https://aditya113141.github.io/

Languages - Hindi (Native) and English (Native)

Location - I am currently living in Vadodara, India. My timezone is UTC+05:30. I would like to work from 9:30 AM to 10:00 PM. In case of emergencies, it can be changed to 8:00 AM to 11:30 PM. I am ready to work even more if the necessity arises.

Degree : Bachelor in Technology

Major : Electronics and Communication Engineering

Institute : International Institute of Information Technology, Hyderabad

Resume : Click here

My Motivation

What is your motivation to take part in Google Summer of Code ?

I want to be a part of Google Summer of Code because of the learning experience that I would get and a chance to network and make friends with cool programmers. This will give me real exposure to the professional coding world and will help me develop as a programmer.

Why did you choose Sugar Labs ?

I came to know about Sugar Labs in December 2020. I researched about the organisation and then came to know about the OLPC initiative and the motive behind the Sugar Project. I have always wanted to contribute in the field of education and working with Sugar Labs will give me an opportunity to work for society.

Why do you want to work on this particular project ?

I chose the Story Activity because of my interests and skills in web development. I have previously used the Sugar Story activity and I am familiar with the program. Before the release of the 2021 idea list, I had a bunch of ideas in mind that I wanted to take up as a part of this program and it was coincidental that Sugarizer Story activity was one of them.

I would love to take it for Google Summer of Code 2021 and apply my web development skills in it.

What are your expectations from us during and after completion of the program ?

During the program, I expect the organisation and the mentors to be helpful and friendly like they have always been on the sugar-mail-list . After the program ends, my plan is to continue working with Sugar Labs and be a mentor to other participants in the upcoming years.

About My Project

My project is about creating a Sugarizer Story activity equivalent to Sugar Story activity.

Features from the original activity :-

- Both Grid display of Images and Slideshow display of images are to be implemented.
- Image Reloading Feature - Reload a new set of images.
- Toolbar Buttons - We will keep the Reload, Array, Linear and Playback Buttons from the original Activity

- We will keep the Sound Recording, text to speech and play the recording feature.
- We will keep image reloading animation from the original activity.

New Features :-

- New text formatting features in editor - Change color, font and alignment of the text. We will add new toolbar buttons for them.
- Image integration from Abecedarium activity. We will provide a button to switch to images from Abecedarium activity.
- Will allow the user to change the number of images - The original activity has 9 images being displayed at a time. We will allow the user to change this number in range 4-9 .
- Export the story as pdf and doc.

General Sugarizer Features :-

- Cross Platform : The activity will have the same functionality and behaviour on anybrowser (Chrome, Firefox, Safari) and any platform (Android, iOS, Windows, Linux, MacOS) supported by Sugarizer.

- Sugarizer Storage - Load / Save content into the Journal.

- Multi User Environment - Multiple users would be able to edit the same document in real time using the sugar presence framework.

- Autosave - All the content present inside the text editor will be autosaved in the journal after every change in the content . This will help the user to save his data if he forgets to properly exit the activity or in case of any other circumstances .

- Localisation - The activity will be fully localised like other sugarizer activities using webl10n library.

- Full Screen mode and Responsiveness - User will be able to enter full screen mode on clicking the full screen button and exit by pressing it again . Content should be responsive and adapt to any screen size, the fullscreen button should allow to mask the toolbar for smaller screens.

- The User interface of the activity will have Sugarizer look & feel . We will use Sugartoolbar and palette to show the features of the activity .

- Offline availability : All the main features would work smoothly without the requirement of any internet connection which will help to run this activity even in remote areas where internet connection is not available easily.

- Tutorial - An integrated documentation would be integrated to explain each feature of the activity

How will it impact sugar labs ?

- The new Sugarizer Story activity will be a great learning activity for students.
- The activity will provide the user with random images and they should try to tell a story that ties the images together into a comprehensive narrative.
- The new features will also allow the students to play with different fonts and colors.
- They can share their story with friends and also invite them to complete it.
- Integration of Abecedarium images locally will provide them with colorful pictures and enhance their creative thinking skills.

What technologies (programming languages, etc.) will you be using ?

I would be using HTML5, CSS, Vue.JS and Sugar Web library. I have completed the Vue.JS tutorial for Sugarizer. You can find my work on Pawn Activity here.

Implementation

Editor

We need to use a WYSIWYG editor in our project. There are many such editors available online. One of the most popular ones made in Vue.Js is **Vue2-editor**.

Vue2-editor - It is a free open source editor based on **Quill.js**. It has a lot of formatting features and most of them are not required for this project. It is highly customizable and we can use custom quill modules in it.

I tried integrating Vue2-Editor in Sugarizer. But I was skeptical whether we are allowed to use third party editors or not. After a discussion with mentors, I came to know that Sugarizer already has an editor in **Write activity**. I realized that **Write activity** has an editor that is similar to **Vue2-editor**, and after some investigation, I found that both are based on **Quill.js .**

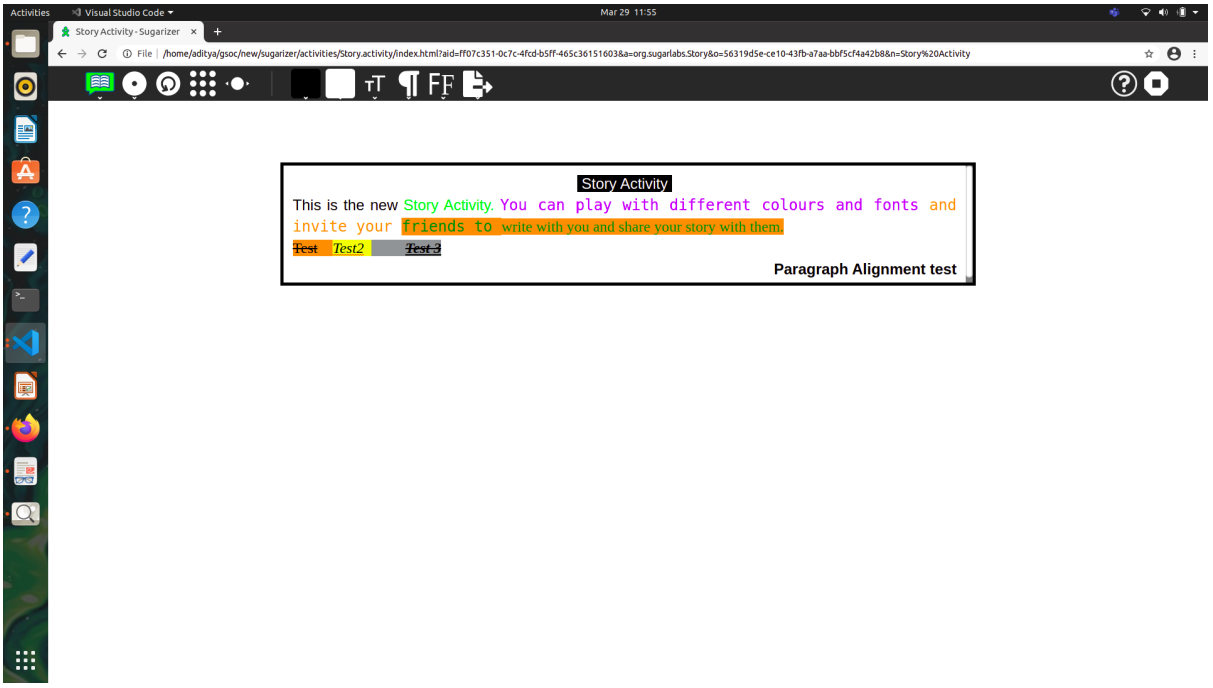
I decided to leverage the editor from Write activity to maximum extent.

- It already provides text formatting features that we need - Font, Paragraph alignment and foreground and background color. We can remove the unnecessary features from the original Write activity.

- It already has exporting features to odt, doc, pdf and image.

- It already has Sugarizer features - Multi User Environment, Localization, Autosave and Sugarizer Storage.

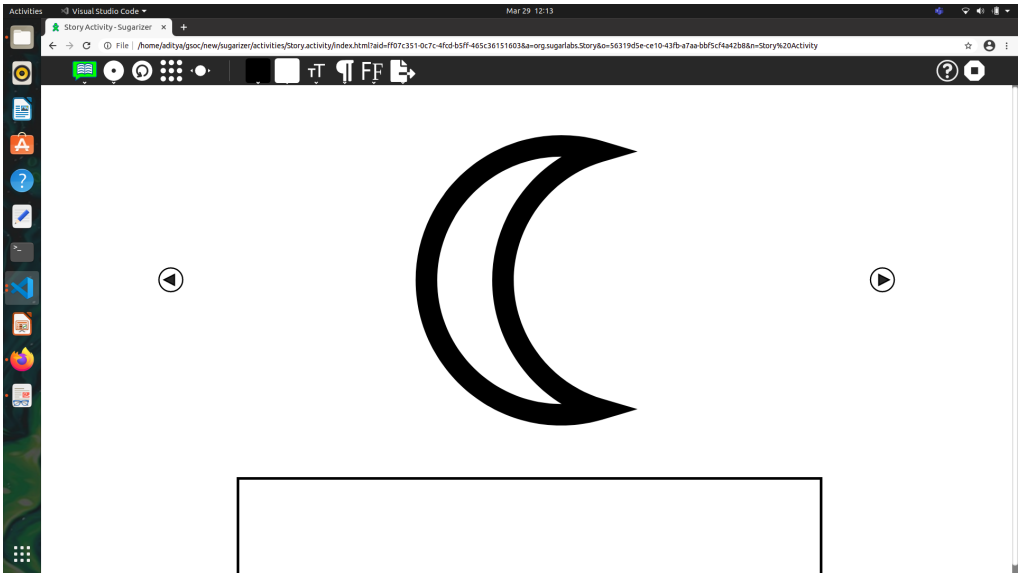
I tried making changes in the original Write Activity to make it look like Story activity.



It saves the data in the Journal.

Image Slideshow

Taking inspiration from the original activity, I tried implementing an Image slideshow using pure JS.



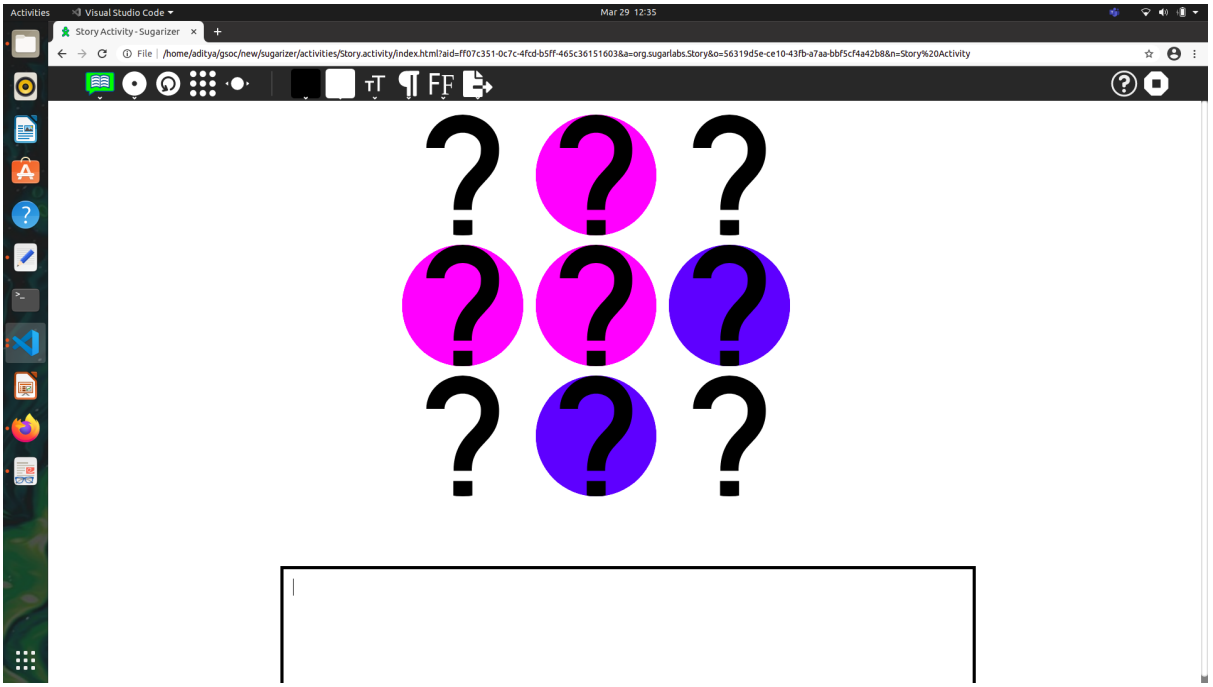
In the final project, the pure JS code will be written in Vue Component (As the project requires to use Vue.JS)

In the slideshow, for each picture, we need a separate editor and the content of each editor is to be saved separately. To implement this, we would use a reusable Vue Component. Each instance of the component will have a unique picture and the editor. We will initiate a new instance by passing an editor and the picture to the component.

On clicking the Image Reload Button, new images will be loaded in the slideshow. During the transition from old images to new images, there would be an image-loading animation same as in the original Sugar Story activity. I have implemented the animation in grid display (next section). Using similar logic, we can implement it in the slideshow.

Grid display of Image

In the grid display, there would be a grid having 4-9 images (depending upon the user's choice). Here, we only need a single editor for all images.
I tried making the original image - loading animation. The code can be found here. The logic of this code will be used in the Grid Vue component. I integrated my pureJS code in the Sugarizer environment.



The final grid display and the slideshow will be coded as separate Vue components.

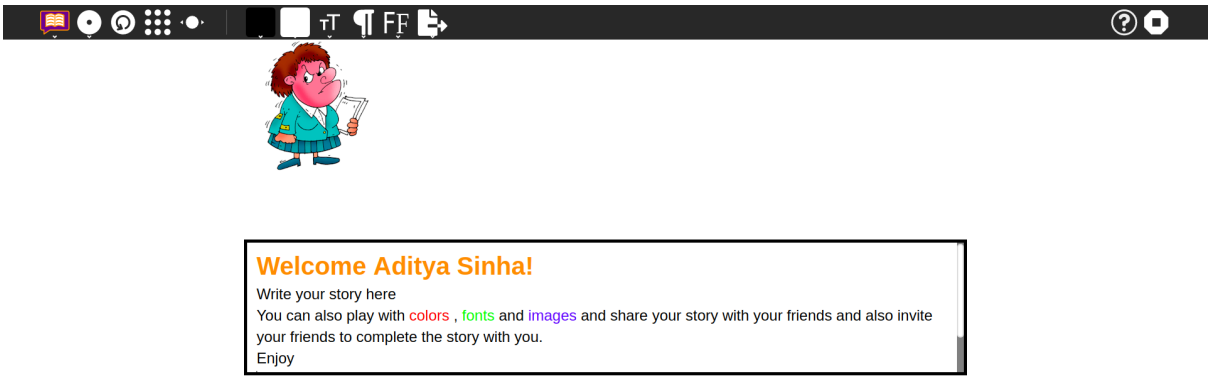
Toolbar

The toolbar displayed in above pictures has the following Buttons :-

- Activity Button
- Network Button
- Buttons from original Story activity
  - Image Reload Button - Displays new images
  - Array Button - Displays Images in Grid
  - Linear Button - Displays Images in Slideshow
- Buttons from Write Activity
  - Foreground Color - Change foreground color of text
  - Background Color - Change background color of text
  - Format Text - Bold, Italic, Underline, Strike
  - Paragraph Alignment
  - Choose Font
  - Export
- Tutorial
- Stop

New buttons and features to add :-

- Abecedarium - A button to switch to images from Abecedarium activity.
  Images from abecedarium activity can be fetched using relative path locally as shown below.



- Number of images - A button to change the number of images in the range 4 to 9.

- Full screen Button

Export

The Write activity already has implementation of export features
Below are the snippets from Write Activity

```
// save as PDF
document.getElementById("20").addEventListener('click',function(){

        var title = document.getElementById("title").value;
        editor.scrollingContainer.style.overflowY = 'visible';
        console.log(editor.scrollingContainer.style.height);
        var h = editor.scrollingContainer.offsetHeight;
        editor.scrollingContainer.style.height="auto";
        editor.scrollingContainer.scrollTop=0;
        html2canvas(editor.scrollingContainer).then(function(canvas){
                editor.scrollingContainer.style.height=h;
                editor.scrollingContainer.style.overflowY = 'auto';
                var imgData = canvas.toDataURL('image/png');
                var imgData = canvas.toDataURL('image/png');
                var imgWidth = 210;
                var pageHeight = 295;
                var imgHeight = canvas.height * imgWidth / canvas.width;
                var heightLeft = imgHeight;

                var doc = new jsPDF('p', 'mm' , '',true);
                var position = 0;

                doc.addImage(imgData, 'PNG', 0, position, imgWidth, imgHeight , '' , 'FAST');
                heightLeft -= pageHeight;
                while (heightLeft >= 0) {
                position = heightLeft - imgHeight;
                doc.addPage();
                doc.addImage(imgData, 'PNG', 0, position, imgWidth, imgHeight , '' , 'FAST');
                heightLeft -= pageHeight;
                }
                var inputData = doc.output('dataurlstring');
                var mimetype = 'application/pdf';
                var metadata = {
                        mimetype: mimetype,
                        title: title+".pdf",
                        activity: "",
                        timestamp: new Date().getTime(),
                        creation_time: new Date().getTime(),
                        file_size: 0
                };
                datastore.create(metadata, function() {
                        console.log("export done.");
                        humane.log(webL10n.get("Pdf"));
                }, inputData);
        });
});
```

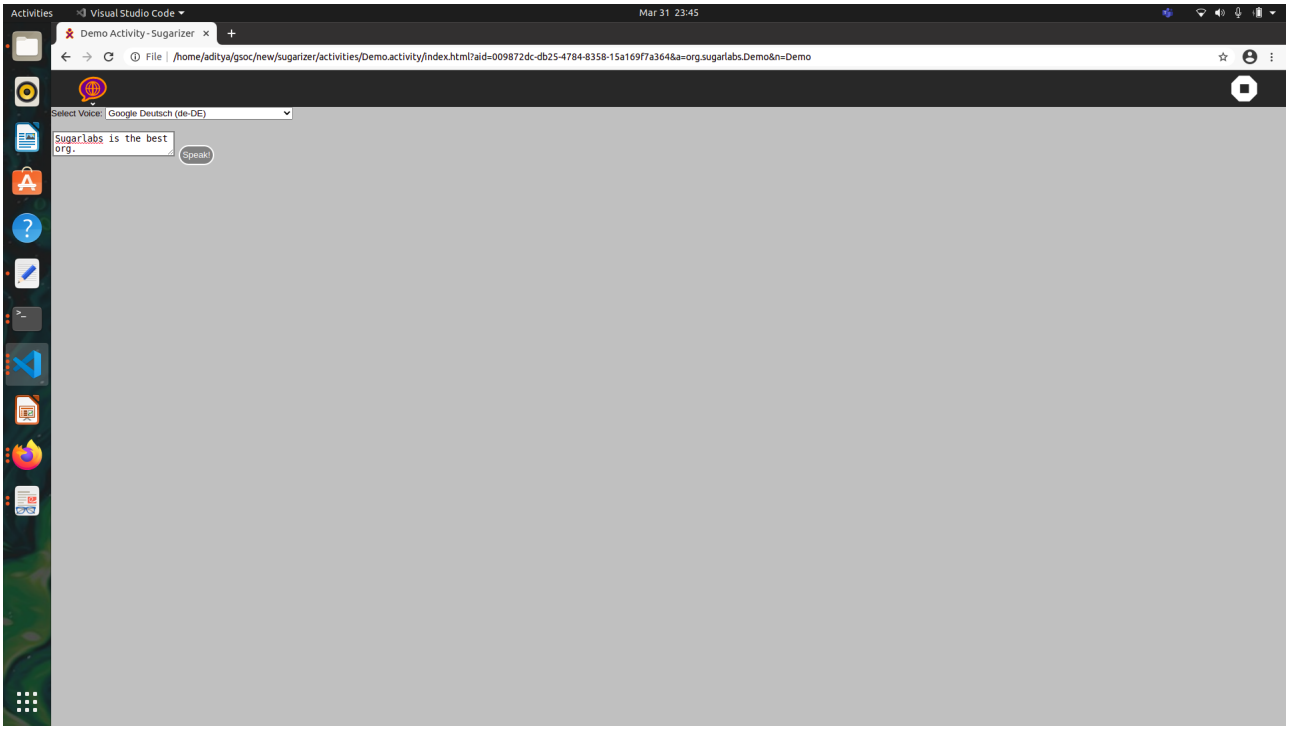
```
// Save as doc
document.getElementById(21).addEventListener("click",function(){
        var title = document.getElementById("title").value;
        var content =document.getElementById("editor").innerHTML;
        var header = "<html xmlns:o='urn:schemas-microsoft-com:office:office' "+
        "xmlns:w='urn:schemas-microsoft-com:office:word' "+
        "xmlns='http://www.w3.org/TR/REC-html40'>"+
        "<head><meta charset='utf-8'></head><body>";
        var footer = "</body></html>";
        var sourceHTML = header+content+footer;
        var inputData = 'data:application/vnd.ms-word;charset=utf-8;base64,' + btoa(unescape(encodeURIComponent( sourceHTML )));
        var mimetype = 'application/msword';
        var metadata = {
                mimetype: mimetype,
                title: title+".doc",
                activity: "",
                timestamp: new Date().getTime(),
                creation_time: new Date().getTime(),
                file_size: 0
        };
        datastore.create(metadata, function() {
                console.log("export done.");
                humane.log(webL10n.get("Doc"));
        }, inputData);
});
```

After understanding the export process, I implemented basic pdf-exporter(here) and doc-exporter(here) in Vue.JS.

We will modify and integrate them in the final project.

Sound Activities

I initially thought of using the Web Speech API for Text-to-Speech and Speech-To-Text features. It is compatible with Cordova. I experimented with it in Vue JS Template. It was working fine on Google Chrome.



But it did not work when I started Sugarizer from terminal using ‘npm start’.

I decided to use the **SugarSpeak component** from Vue template. This will serve the purpose of the Text-to-Speech feature but we won't be able to implement the Speech-to-Text feature. I decide to leave it for now. For the recording feature, I manipulated the **Record Activity** and removed the Image capturing and Video Capturing Feature. We can use this as a component in our project.

Timeline

During the coding period, I have vacation in my college and I am not doing any other internship and will devote my complete time towards my Google Summer of Code Project

Duration	_Task
May 17 - June 7	Bonding with mentors and colleagues Discuss potential implementation ideas with mentors. Test the various ideas based on discussion.
June 7 - June 14	Make the editor component with all the required formatting features. Make toolbar with all basic buttons
June 14 - June 21	Create Slideshow component
June 21 - June 28	Create Grid component with Animation
June 28 - July 5	Add images from Abecedarium Button to toggle between Noun-project images and Abecedarium images Button to choose number of images Full screen Button with proper working
July 5 - July 12	Use extra time to cover up unexpected delays
July 12 - July 16	Phase I evaluations (Separate Grid and Slideshow Components working)
July 16 - July 23	Integrate Grid and Slideshow Components in the same activity
July 23 - July 30	Work on sound features - Recording, Text-to-Speech
July 30 - Aug 2	Add Localisation and Export features
Aug 2 - Aug 9	Add load/save data in Journal. Add network Sharing for multiple users through SugarPresence Component from Vue Template and add UI for it and test the workflow.
Aug 9 - Aug 16	Review Documentation and prepare for final evaluation
Aug 16 - Aug 23	Final Evaluation (Activity working properly - Export feature, Sound feature, Journal Saving, Documentation)

My Working Environment and Tools

- **Ubuntu 20.04 LTS Operating System**
- **VS Code**
- **Git**
- **Chrome Dev Tools**

Open Source Contributions :

- I am currently part of another open source program called Girlscript Summer of Code.
- I am working on the project ML Projectyard. I am looking forward to taking it up as my major project during the later part of the program.
- **This program ends in May and will not be a hindrance to my Google Summer of Code schedule.**
- I decided to participate in this program to gain relevant open source experience before Google Summer of Code.
- My contribution so far - Ramen Ratings

Major Web-Development Projects

IIITH-Buy and Sell Website - Created a project as a part of Hackathon 2020 orgainised by IIIT Hyderabad in my first year of college. Being from the Electronics branch, I had no prior experience with Web Development. In spite of having competition from students of Computer Science branch, **my team won the competition**. This was my first project in Web Development.

Chat-App - Made a MERN Stack Chat application with Firebase Authentication. Currently used by me and a few of my batchmates. Work on group messaging is going on. We are learning to manage an app with a large number of Users.

XMeme - Worked on a Meme Streaming Web application as part of Crio Winter of Doing. Frontend work was done in React while Backend in Flask. Learned working on a CRUD Application.

Contribution to Sugarlabs

https://github.com/sugarlabs/flappy/pull/13#issue-569011605

https://github.com/sugarlabs/sugar/issues/937#issuecomment-774803849

https://github.com/sugarlabs/flappy-birds-activity/pull/24#issue-573635780

https://github.com/sugarlabs/GSoC/pull/131

Other Questions

How many hours will you spend each week on your project ?

I am planning to spend 45 hours per week on average.

How will you report progress between evaluations ?

I will be in contact with my mentors through mail and will update them about my progress.

Will you continue contributing to Sugar Labs after GSOC ’21 ends ?

I would like to keep contributing to the Sugar Labs community and work on other ongoing projects - Musicblocks 2.0 and musicblocks-v4-lib after GSOC ‘21 ends. (I am really interested in joining the community and will keep contributing irrespective of my selection in GSOC ‘21)