Google Summer of Code, 2020 Proposal
Organisation : sugarlabs

Applicant Details:

❖ **Name** : Hemansh Khaneja

❖ **Country of Residence** : India

❖ **Timezone** : IST (GMT +0530 hrs)

❖ **University** : Indian Institute of Technology, Delhi

❖ **Email** : hemansh3172@gmail.com

❖ **Github Profile** : https://github.com/Hemansh31

❖ **Personal Background** : I am a Sophomore at the prestigious Electrical Engineering Department, Indian Institute of Technology, Delhi. I was among the top 0.003% of the 1.3 million candidates who appeared for admission into the highly competitive and the prestigious Indian Institute of Technology. Although I am majoring in the field of Electrical Engineering I am a bit inclined towards Software Development which has also encouraged me to take up certain campus wide projects in Software Development. I love problem solving and I try to work on interesting projects in my free time.

❖ **Estimated last day of University Examinations** : My University Examinations were scheduled to be over by the first week of May, 2020 (i.e by 9th May, 2020) but due to the coronavirus outbreak my university had to suspend all campus activities until 31st March,2020 (beginning from 15th March, 2020) which may result in my semester getting extended a few days although I haven't got any confirmation regarding this from my university yet.

❖ **Planned GSoC work hours** : If I am selected, I shall be able to work 45 to 55 hrs a week, though I am open to putting in more effort if the work requires. Also, if some weeks of my semester clash with GSoC, then during those weeks I can devote upto 30 hrs a week.

❖ **Planned absence/vacation days or other commitments during GSoC 2020 period** : As of now, I don't have any other commitments or planned vacations during GSoC 2020 period and I intend to keep it that way.

- **Skill Set relevant to complete this project** : Although I am new to Open Source, I have done several projects on web development and I also did a technical internship in web development in the summers of my Freshmen year and thus I am quite fluent and experienced in web development (both client side and server side). Also, I am quite confident that I can deliver a better project in terms of the UX and UI than anyone else. Here are some relevant details regarding the proposal :
  - **Familiarity with Languages** : HTML, CSS, Javascript, Node.js, NoSQL
  - **Familiarity with tools** : Bootstrap, Vue.js, EJS, Mongo, Git

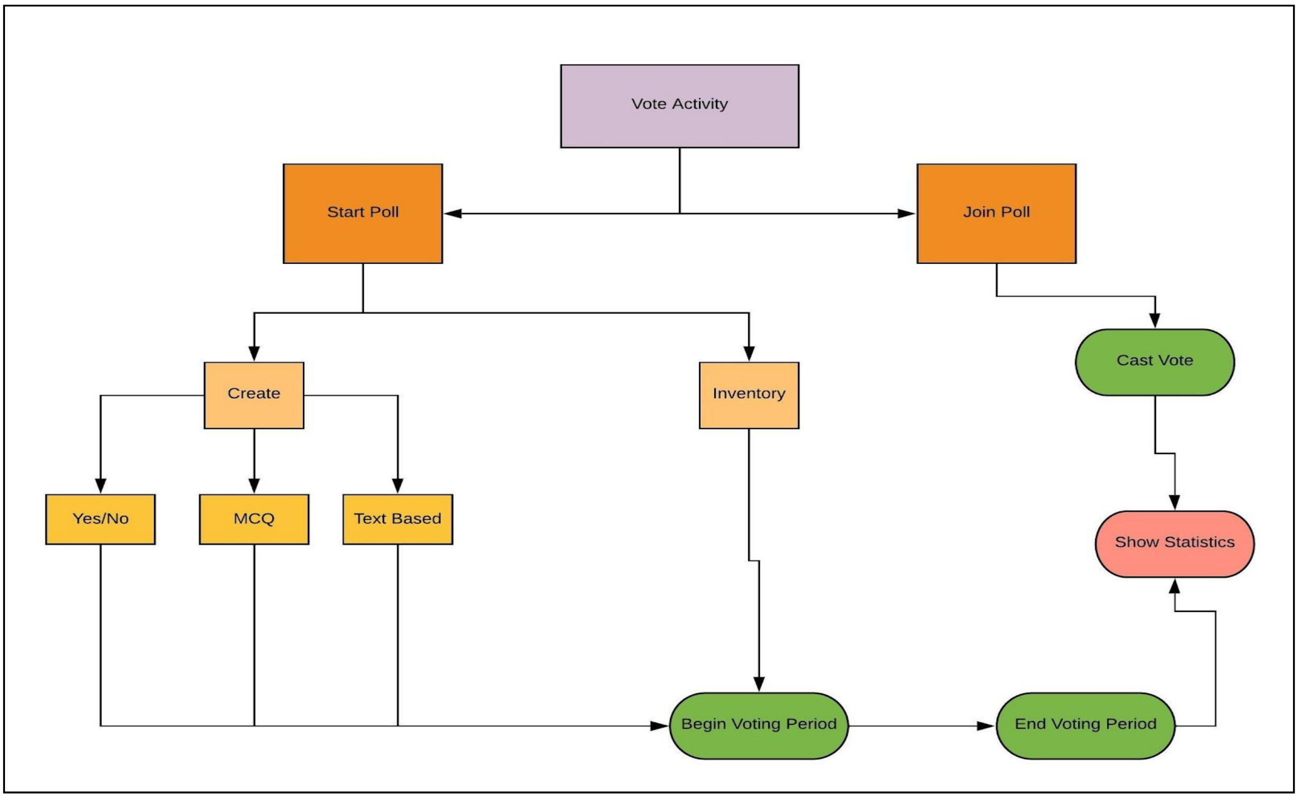
  I have a basic understanding of the sugarizer code base and would keep learning more during the pre-GSoC period. Here are details of my few contributions to the sugarizer code base :

Github Link	Type	Title	Status
#648	Pull Request	Issue #360 Resolved	Merged
#645	Pull Request	Chat Activity Tutorial	Merged
#644	Challenge	ChessBoard Task	Under Review
#707	Pull Request	Nell Colours	Under Review
#679	Issue	TamTam MIcro Activity Not working in file:\\\	Fixed
#677	Issue	Export in Write activity	Closed

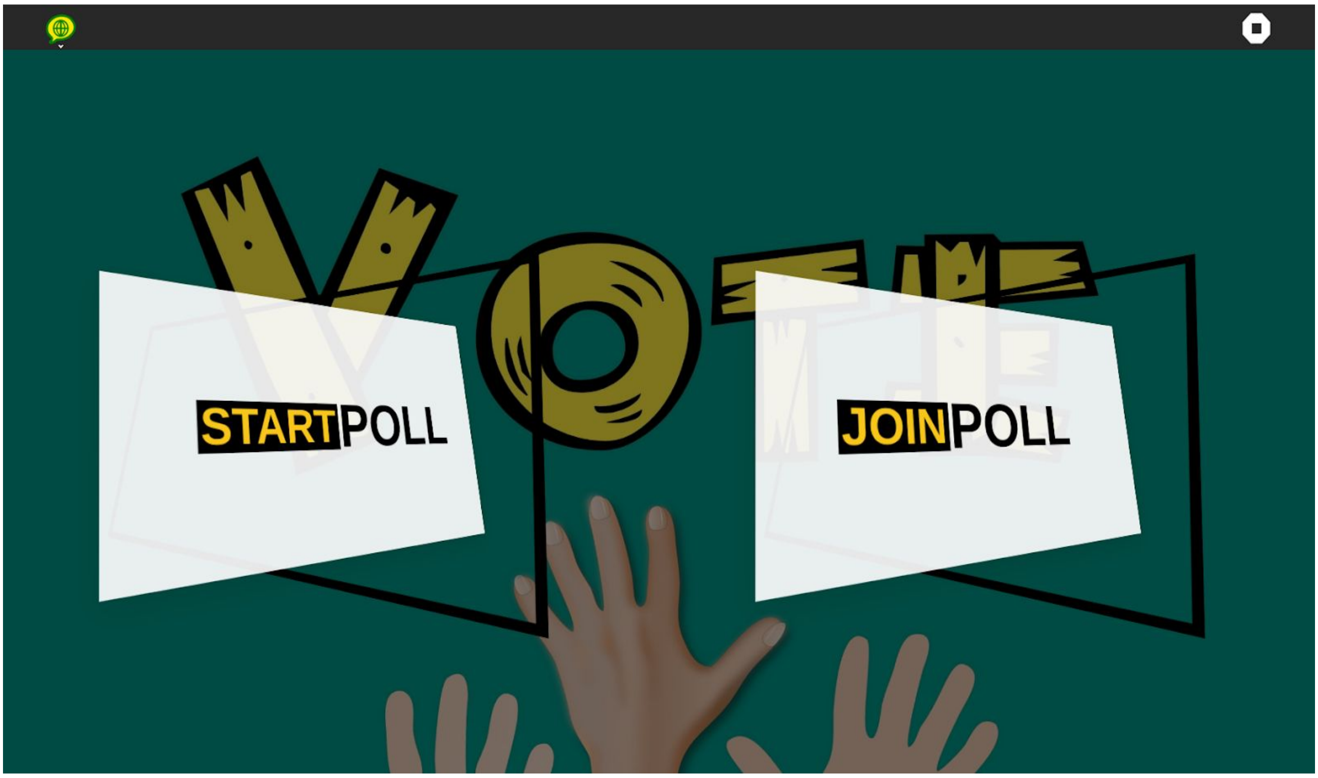
Project Proposal Details:

- **Project Title** : Sugarizer Knowledge Activity Pack
- **Potential Mentors** : Lionel Laské and Michaël Ohayon

- **Project Summary : Specifically, the objective of this project is to develop two new activities:**
  - A Curriculum activity : The Curriculum activity will be a way for a student to self check their skill in a set of knowledge and provide multimedia elements to demonstrate these skills.The Curriculum activity will display a hierarchical set of skills grouped by categories then let the user explore the tree. On each skill the user should be able to validate (i.e. skill acquired) and provide multimedia elements (pictures or sounds coming from Journal) to demonstrate the skill.The Curriculum activity will provide a settings mode to edit the set of skills: Create/Update/Delete/Sort skills or categories. A category should have a title and a color, a skill should have a title and an image. It should be possible to generate a Word/ODT document with all skills and dated multimedia elements. It should be possible to share its skills on the network.
  - A Vote activity : The Vote activity will allow easy building of a poll system. The user creates a poll (yes/no, chooses a value in a list, enters a value) then shares it on the network so any user could vote in real time. At the end of the vote, a screen will sum up results of the vote: stats, who voted for what, who voted first, etc.A poll is a label (question) and could integrate a multimedia element (image, audio, video, speech to text).
- **Vote Activity :**
  - Disclaimer : The User Interface shown below is just a basic template implementation, the final User Interface would contain all the fine details. This is just to give an idea what the final User Interface would look like. The basic implementation of the User Interface I coded can be found here: https://github.com/Hemansh31/sugarizer/tree/VoteActivity
  - Workflow Diagram :

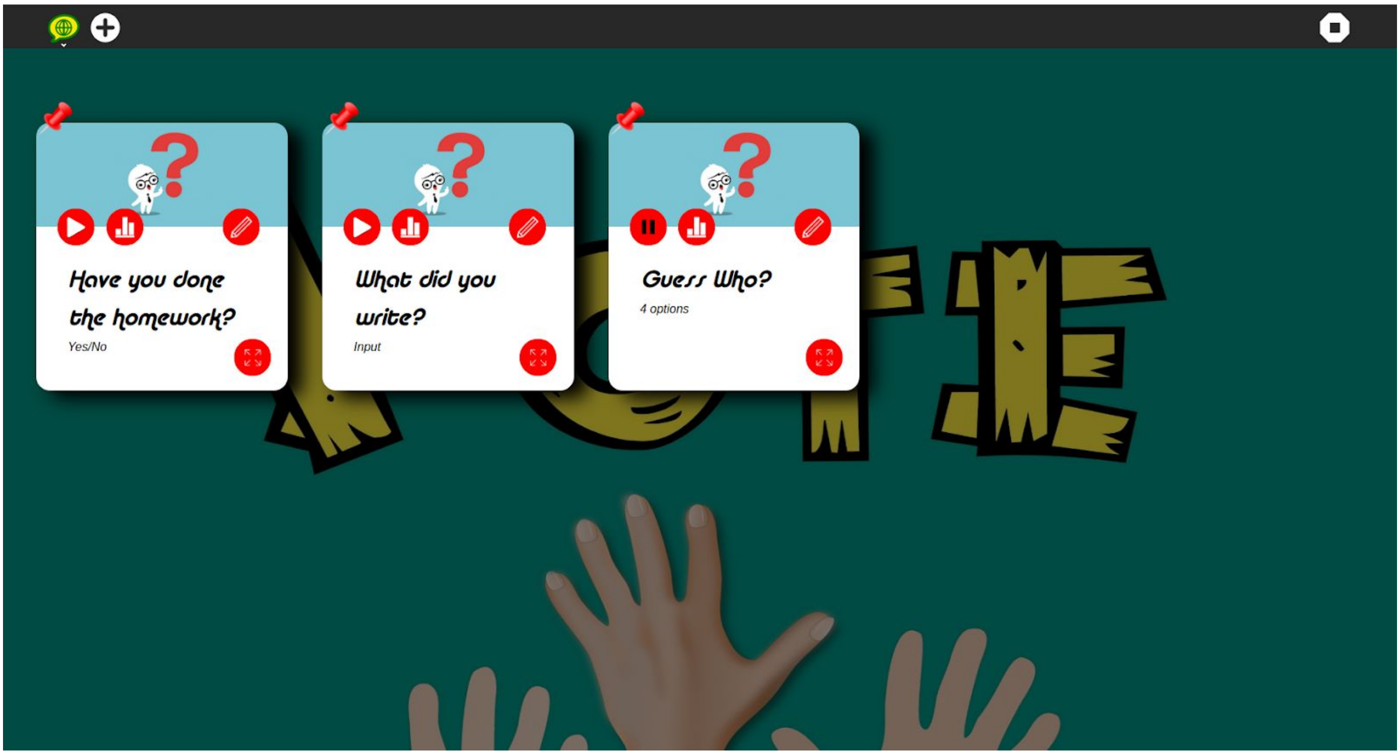


- **Workflow Details :** Upon launching the activity the home page would present two options for starting or joining a poll.

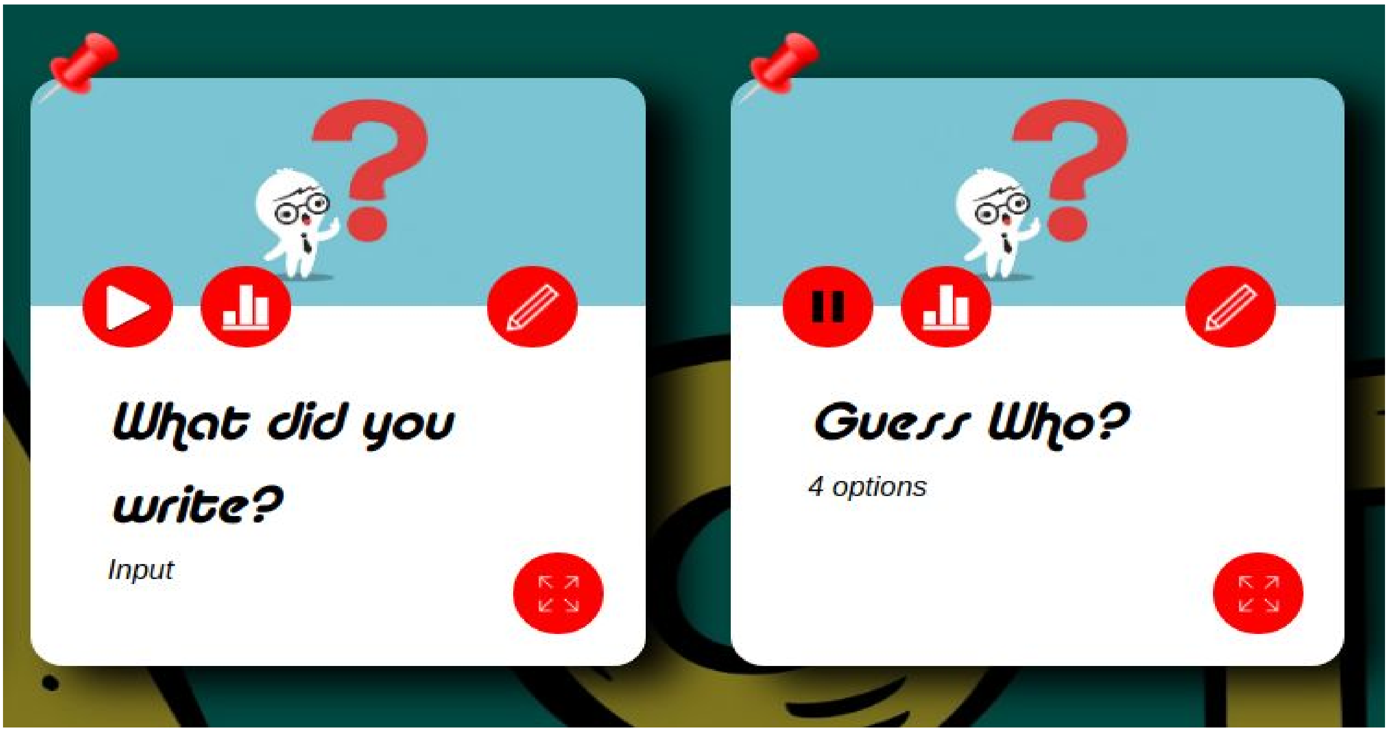


Home Page for Vote Activity

- **Start Poll** : Upon clicking on Start Poll a page would open showing past polls stored in the journal and running polls and a button for adding a poll would be present on the Toolbar.

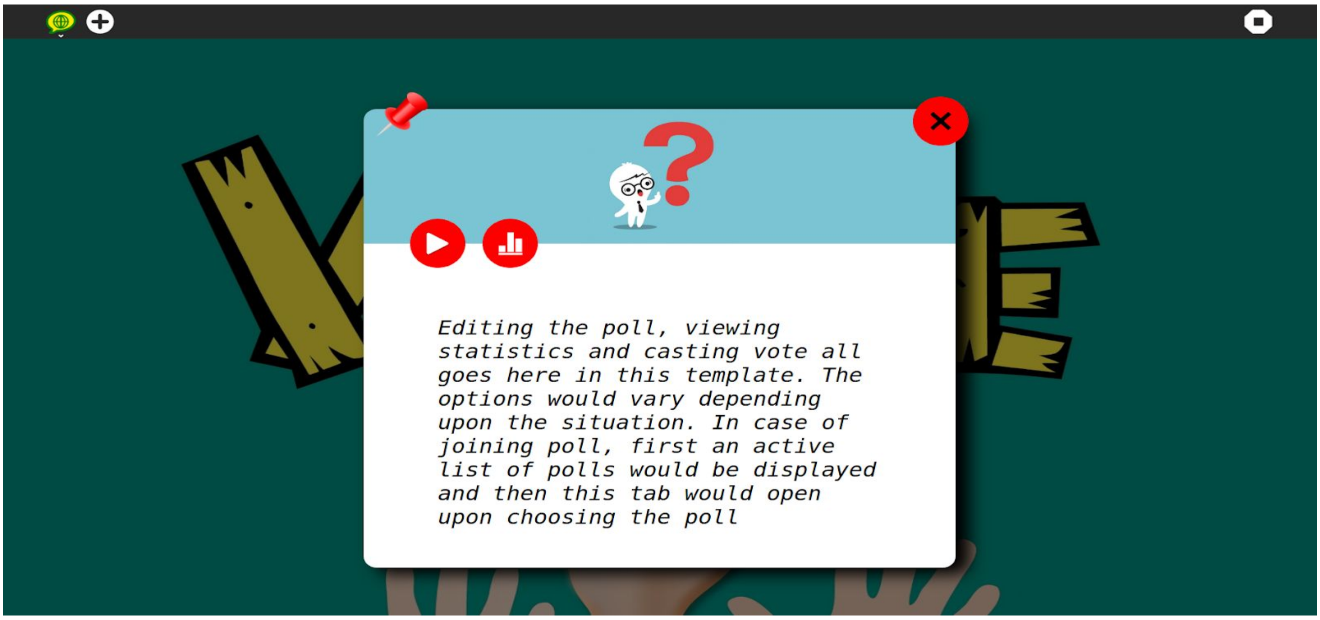


Page showing saved and running Polls with Add button in Toolbar



Poll Cards

- Poll Card : The poll cards shown in above figure represent a saved poll and a running poll respectively. The poll card has several options :
    - Play Button : Upon clicking this poll would begin, i.e the user using Presence would be able to share the poll over the network and anyone on the network can vote.
    - Pause Button : This button would appear in place of the play button when a poll is started. Upon clicking this the poll would end and the play button would be visible again.
    - FullScreen Button : This button would enlarge the poll card and would display the complete details of the poll.
    - Edit Button (Pencil Icon) : Upon clicking the user would be able to change details in the poll in FullScreen mode.
    - Statistics Button : Upon clicking this the user would be able to view the statistics (in a graphical way using Chart.js) for that poll along with other details such as the date and time of voting beginning and ending, who voted first, number of votes etc. in FullScreen mode. There would be Pagination if multiple statistics for a single poll are present.



Editing the poll, viewing statistics and casting vote all goes here in this template. The options would vary depending upon the situation. In case of joining poll, first an active list of polls would be displayed and then this tab would open upon choosing the poll

Template for displaying content in FullScreen Mode

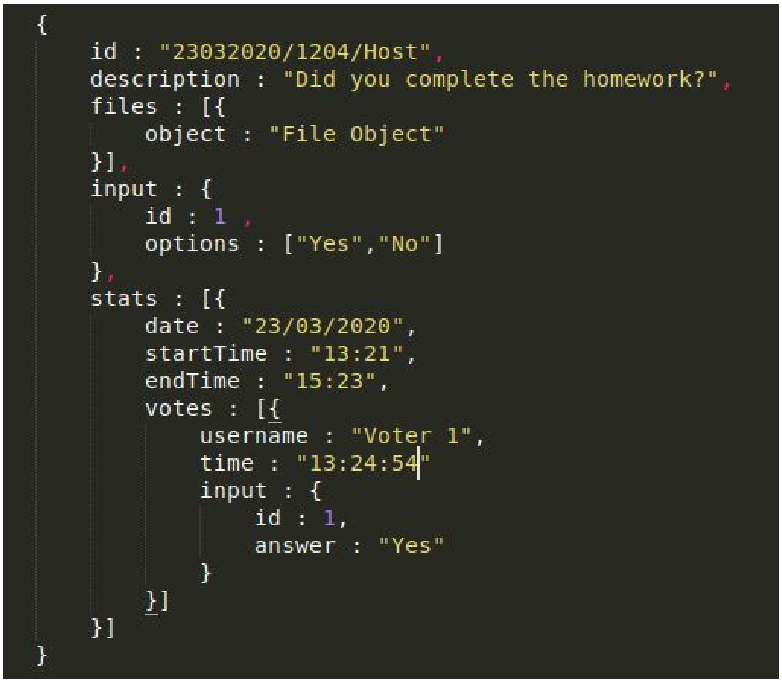
- **Add Poll** : Upon clicking this button a form for adding a poll with fields for selecting type of input (Yes/No, MCQ, Text based), the question for poll, link to upload media related to the poll etc. would be displayed in **FullScreen mode** in the template shown above.
- **Join Poll** : Upon clicking the Join Poll button from the homepage first a list of active polls (possibly the neighbourhood view) would be displayed and the user would be able to join any one of them using **Presence**. After joining a poll the Poll would be visible in **FullScreen Mode** and the user can cast a vote using a submit button until the poll ends. Upon ending the **Statistics** would be displayed in **FullScreen Mode**.

➢ External Libraries to be used :

- Chart.js : This library would be used to draw graphs and other representations of the votes data.
- Vue.js

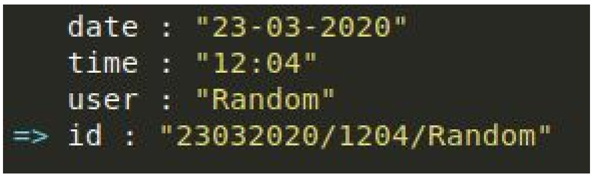
➢ Technical Implementation :

- **Journal** : The data of a poll would be stored in the Journal. The structure of the JSON data would be :



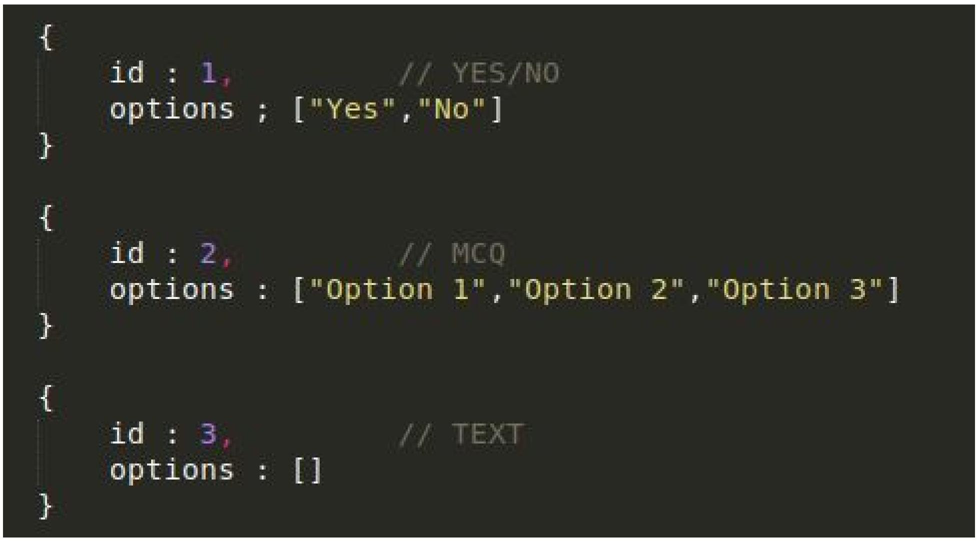
```
{
    id : "23032020/1204/Host",
    description : "Did you complete the homework?",
    files : [{
        object : "File Object"
    }],
    input : {
        id : 1 ,
        options : ["Yes","No"]
    },
    stats : [{
        date : "23/03/2020",
        startTime : "13:21",
        endTime : "15:23",
        votes : [{
            username : "Voter 1",
            time : "13:24:54"
            input : {
                id : 1,
                answer : "Yes"
            }
        }]
    }]
}
```

- **id** : ID for a poll is of the following form:



```
   date : "23-03-2020"
   time : "12:04"
   user : "Random"
=> id : "23032020/1204/Random"
```

- **description** : Description contains the title or question of the poll.
- **files** : Since all multimedia can be converted to Javascript objects, this field stores all such multimedia objects in an array. Example : images can be converted to base64 data URL using *readAsDataURL()* function of the File Reader and stored as strings.
- **input** : It stores the form of input for the poll

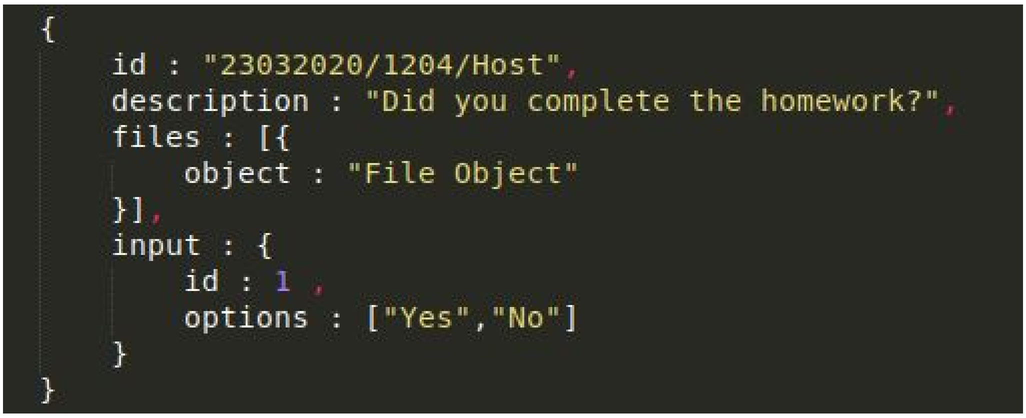


```
{
    id : 1,          // YES/NO
    options ; ["Yes","No"]
}

{
    id : 2,          // MCQ
    options : ["Option 1","Option 2","Option 3"]
}

{
    id : 3,          // TEXT
    options : []
}
```

- **stats** : It stores statistics of the voting on a particular poll in form of an array (as the poll could be open more than once). A single object in this array contains the date, start time, end time and an array of votes of users (which includes the username, time of vote and vote casted by the voters).

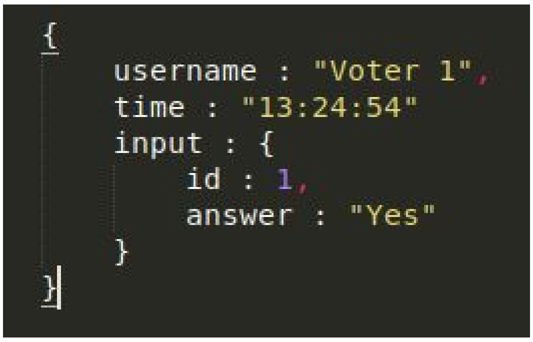
- **Presence** : Joining or Starting a poll over a network would be achieved using Presence and the JSON data to be shared depends on the case :

- User just after joining the poll would receive the JSON data:



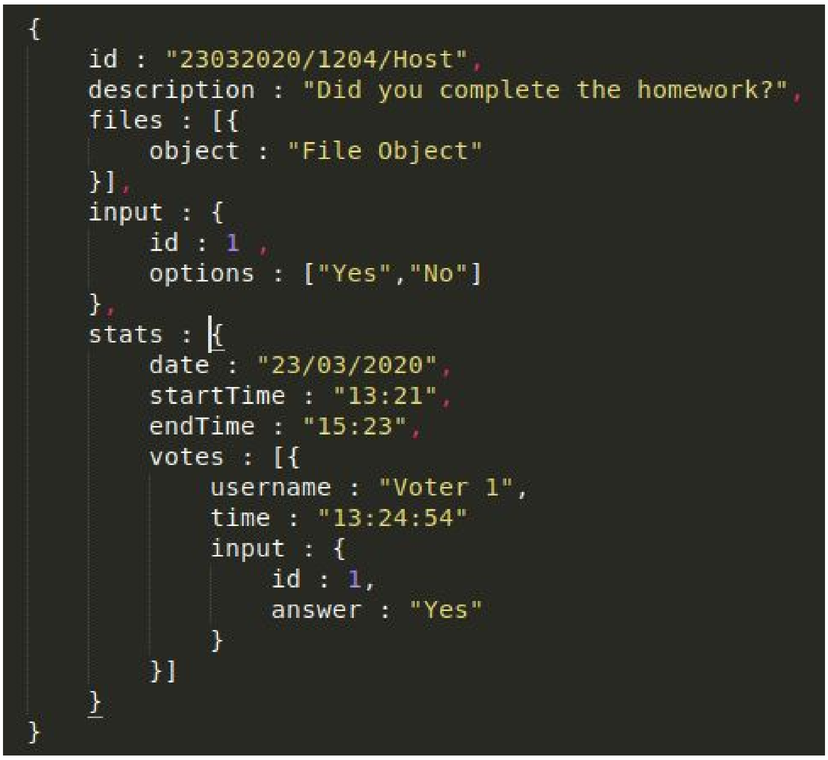
```
{
    id : "23032020/1204/Host",
    description : "Did you complete the homework?",
    files : [{
        object : "File Object"
    }],
    input : {
        id : 1 ,
        options : ["Yes","No"]
    }
}
```

- Users after casting the vote would send this data to the host:



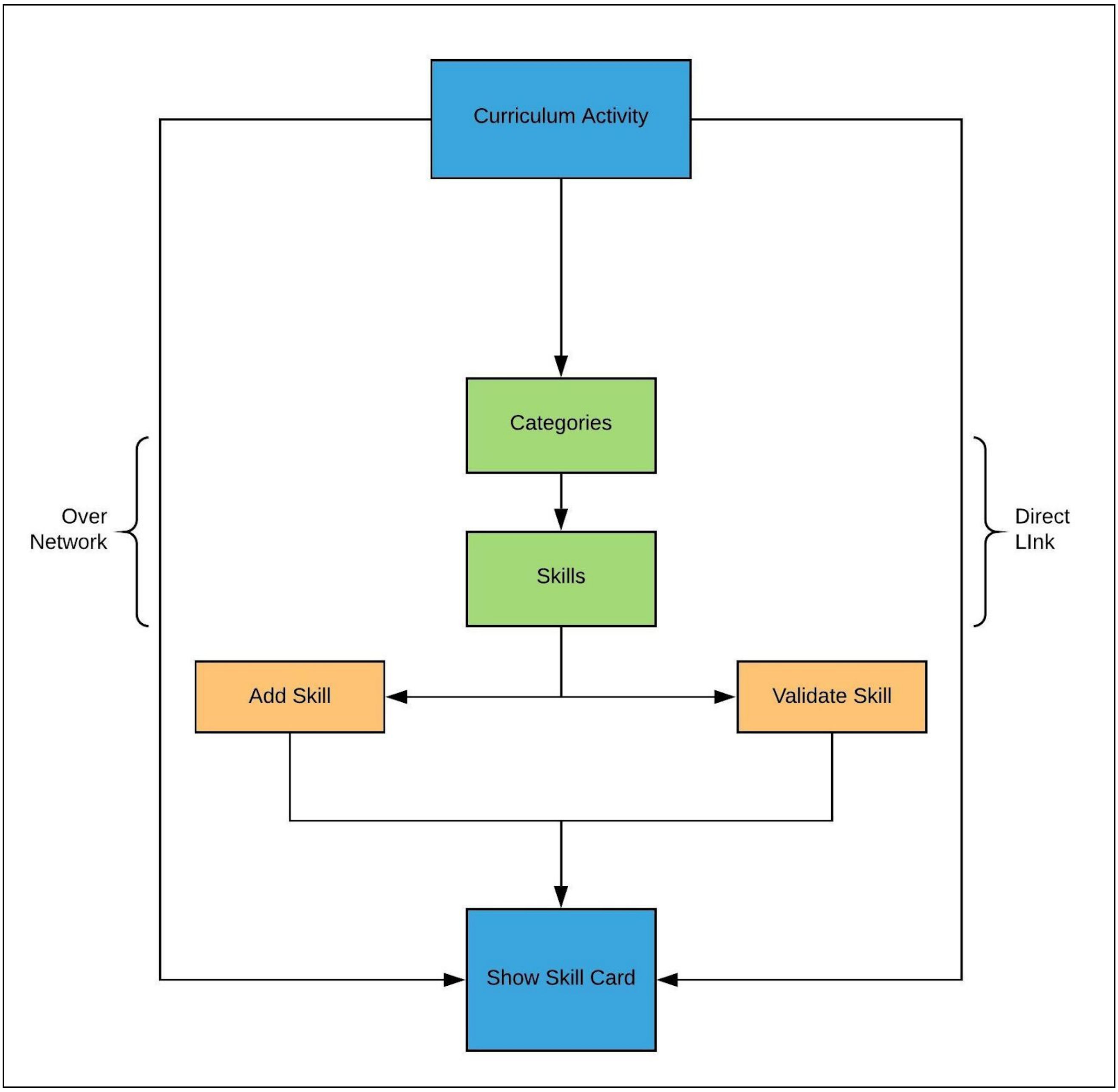
```
{
    username : "Voter 1",
    time : "13:24:54"
    input : {
        id : 1,
        answer : "Yes"
    }
}
```

- After the voting is closed the users would receive the stats as a JSON data from the host.

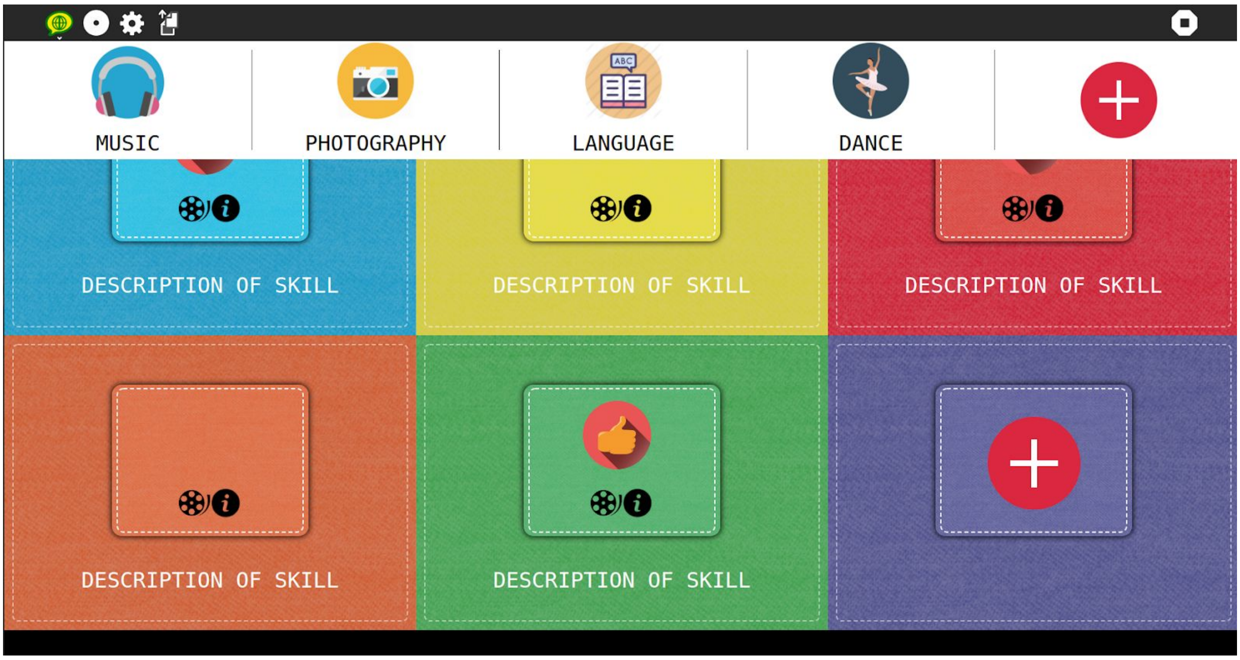


```
{
    id : "23032020/1204/Host",
    description : "Did you complete the homework?",
    files : [{
        object : "File Object"
    }],
    input : {
        id : 1 ,
        options : ["Yes","No"]
    },
    stats : {
        date : "23/03/2020",
        startTime : "13:21",
        endTime : "15:23",
        votes : [{
            username : "Voter 1",
            time : "13:24:54"
            input : {
                id : 1,
                answer : "Yes"
            }
        }]
    }
}
```

- **Curriculum Activity :**
  - **Disclaimer** : The User Interface shown below is just a basic template implementation, the final User Interface would contain all the fine details. This is just to give an idea what the final User Interface would look like. The basic implementation of the User Interface I coded can be found here: https://github.com/Hemansh31/sugarizer/tree/CurriculumActivity
  - WorkFlow Diagram :

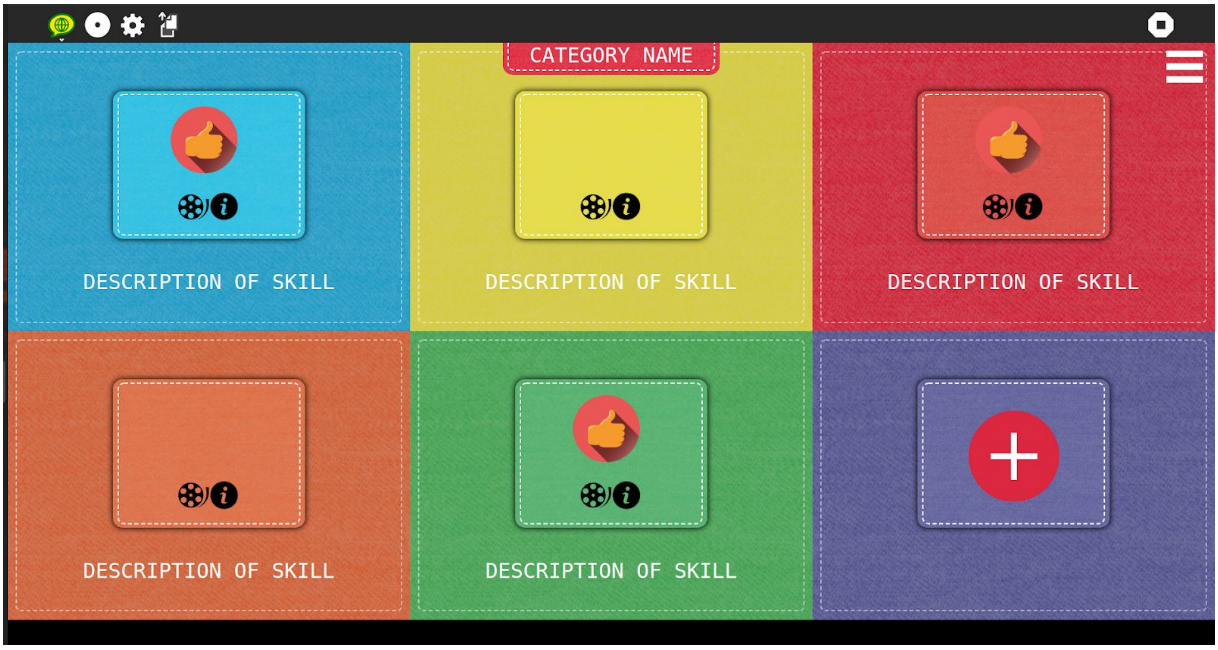


- **WorkFlow Details** : The home page of the Curriculum Activity will show skills of a particular category with a navbar to switch categories.



Home Page Showing category navigation bar (white bar)

- **Category Navigation Bar** : This will be a horizontally scrollable navigation bar showing all the categories with an add category button at the end of the navigation bar.



Template for Showing Skills of a Category

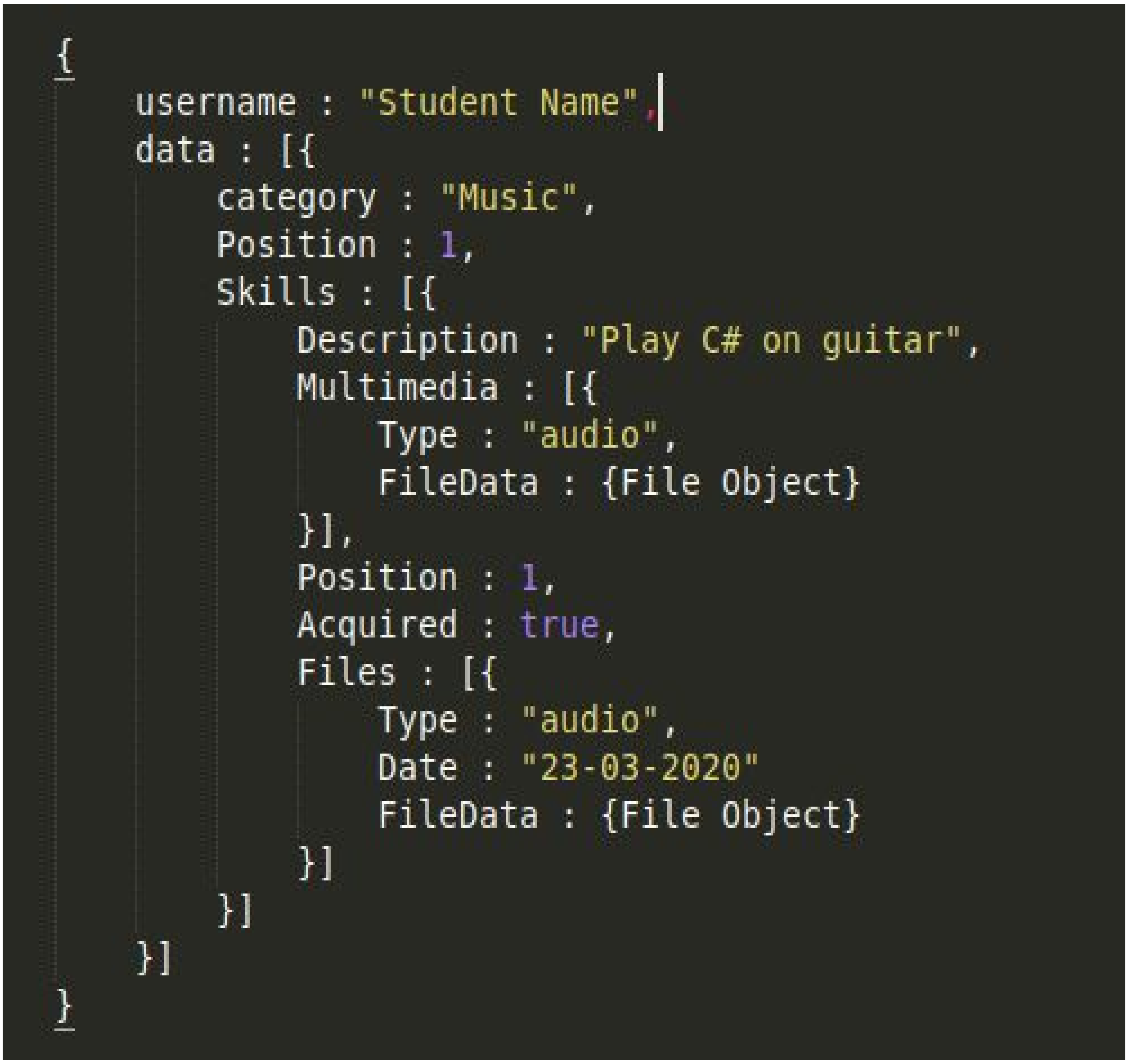
- **Skill Display** : The skills for a particular category would be displayed in the form of 6 tiles per page with pagination if more skills exist. One extra tile would be occupied by the add skill button. A particular tile consists of the description of the skill, a thumbs up to signify whether it is acquired or not, a info button to view complete details of the skills in **FullScreen Mode** and edit it if the user wants and a multimedia button to upload multimedia related to the skill. All files related to the skill would be stored in the **Journal.**
- **FullScreen Mode** : The FullScreen mode of any tile would be a magnified version of that tile (same background as the tile) and covering almost half the page and it would overlay over the current view in the center of the page.
- **Settings Mode** : Upon clicking the settings button the user would enter the settings mode where he would be able to reorder skills or categories using **drag and drop functionality** and could also delete or add a skill or a category if he wants.
- **Skill Card** : The Skill Card could be viewed and exported in form of pdf by the student by clicking on the Show Skill Card button and by any other user who joins the shared curriculum activity. Skill Card would be displayed in **Full Screen Mode** and that could also be exported in form of a Word/ODT document in a tabulated form and the same view can also be exported as a screenshot in form of pdf.

➢ **External Libraries to be used :**

- SortableJS : This library would be used to drag and drop functionality in the settings mode.
- Vue.js
- HTML2canvas : This library is used to convert a DOM section to canvas and take it's screenshot.
- jsPDF : Used along with HTML2canvas to generate pdfs.

➢ **Technical Implementation :**

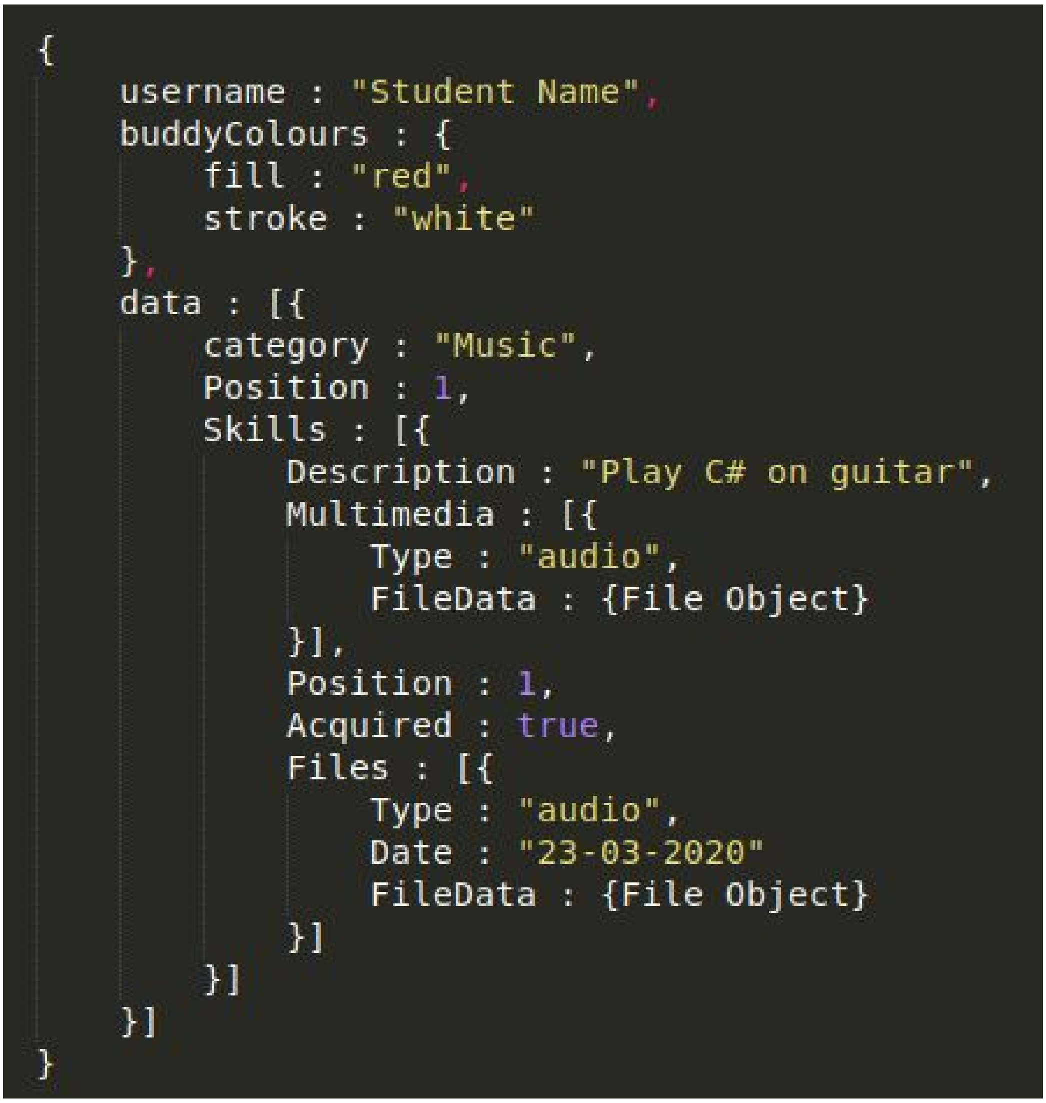
- **Journal** : The data about the student and it’s skills would be stored in the journal. The structure of the JSON data would be :



```
{
    username : "Student Name",
    data : [{
        category : "Music",
        Position : 1,
        Skills : [{
            Description : "Play C# on guitar",
            Multimedia : [{
                Type : "audio",
                FileData : {File Object}
            }],
            Position : 1,
            Acquired : true,
            Files : [{
                Type : "audio",
                Date : "23-03-2020"
                FileData : {File Object}
            }]
        }]
    }]
}
```

- **username** : This field stores the name of the student
- **data** : This field stores an array of category Javascript objects.
- **Category Javascript Objects** : This object is structured as follows:
    - **category** : This field stores the name of the category.

        - Position : This field stores the position of category in the ordered view. Categories are ordered in the skill card according to this position.
        - Skills : This field stores an array of skill Javascript objects.
    - Skill Javascript Objects : This object is structured as follows:
        - Description : This field stores the description of the skill
        - Multimedia : Since all multimedia can be converted to Javascript objects, this field stores all such multimedia objects   related to the description of the skill in an array. Example : images can be converted to base64 data URL using *readAsDataURL()* function of the File Reader and stored as strings.
        - Position : This field stores the position of the skill in the ordered view. Skills are ordered inside the category in the skill card according to this position.
        - Acquired : This is a boolean which stores true if the skill is acquired and false if the skill is not acquired.
        - Files : Since all multimedia can be converted to Javascript objects, this field stores all such multimedia objects related to the validation of the skill in an array. Example : images can be converted to base64 data URL using *readAsDataURL()* function of the File Reader and stored as strings.
- Presence : Sharing or viewing a Skill Report over a network would be achieved using Presence and the JSON data to be shared would be the same as stored in the Journal with just one more field of the buddyColours
    - buddyColours : This field stored the Buddy Colours of the student profile.



```
{
    username : "Student Name",
    buddyColours : {
        fill : "red",
        stroke : "white"
    },
    data : [{
        category : "Music",
        Position : 1,
        Skills : [{
            Description : "Play C# on guitar",
            Multimedia : [{
                Type : "audio",
                FileData : {File Object}
            }],
            Position : 1,
            Acquired : true,
            Files : [{
                Type : "audio",
                Date : "23-03-2020"
                FileData : {File Object}
            }]
        }]
    }]
}
```

- **How will it impact SugarLabs :** The knowledge activity pack would help students understand some of the most essential qualities in life, skill acquisition, introspection, equality and freedom of speech. The vote activity will help to build an interactive ecosystem for teachers and children. The Vote activity ensures that everyone in a group has a say in decisions. It enables teachers to quickly take feedback from students. Voting can be used to create quick quizzes and it also acts as a conflict resolution technique in large groups. The Curriculum activity helps the student be aware of the skills they possess and need to work on. The self evaluation will motivate children to put

extra effort in their learning and will have less pressure. It is a fun activity to track the progress of the student regularly and easily share it with others in the network. It gives them a reason to learn new skills and help them understand the importance of introspection.

Timeline :

- **Community Bonding :** Explore the technologies and libraries that will be used in creating Curriculum and Vote Activities. Communicate with my mentors and other Organization members to finalize the features and UI of the Activity.
- **Week 1** : Start implementing the Vote Activity. Complete the design and implementation of the User Interface of the Vote Activity. Design templates for all parts of the vote activity and getting it reviewed and finalised by the mentors.
- **Week 2-3** : Integrating Journal with the current activity and implementing Presence in the activity to enable sharing of the activity. Testing activity sharing and debugging any errors.
- **Week 4** : Integrating Chart.js to draw graphs for the statistics. Documenting the code written so far and working on all the remarks on the implementation so far.
- **Week 5** : Phase I Evaluations :
  - **(Progress)** All the Main Features and UI of the Vote Activity completed.
- **Week 6** : Start implementing the Curriculum Activity. Complete the design and implementation of the User Interface of the Curriculum Activity. Design templates for all parts of the curriculum activity and getting it reviewed and finalised by the mentors.
- **Week 7-8** : Integrating Journal with the current activity and implementing Presence in the activity to enable sharing of the

activity. Testing activity sharing and debugging any errors. Documenting the code written so far.

- **Week 9** : Phase II Evaluations :
  - **(Progress)** All the Main Features and UI of the Curriculum Activity completed.
- **Week 10** : Implementing the drag and drop functionality of the settings mode using SortableJS and also implementing the feature for exporting skill report in form of Word/ODT document and pdf format. Taking feedback on the work done so far by the mentors.
- **Week 11-12** : Localization of both the activities, adding Tutorials for both the activities, working on all the remarks on the work done so far and documenting the code written so far.
- **Week 13** : Final Phase Evaluations :
  - **(Progress)** Export and drag and drop functionality added to curriculum activity.
  - **(Progress)** Localization of both the activities.
  - **(Progress)** Added Tutorials to both the activities.
  - **(Progress)** Implemented all the remarks suggested by the mentors.
- **Plans post GSoC Period** : Post GSoC, I will continue to maintain the activities and fix any bugs that show up. I would also keep working as an active member of the SugarLabs community and help new developers and contributors get familiar with the platform. I hope to mentor contributors for SugarLabs in GSoC and GCI in the upcoming years.

  At last, I would like to mention that as a newbie to OpenSource my experience has been awesome until now.All thanks to SugarLabs. The SugarLabs community is quite welcoming, responsive and helpful. Also, this project has helped me improve upon my design and coding skills many folds.