**Meet2Code**

**Introduction:**

In these desperate times, there is an undeniable need for online applications that allow people to interact with each other, whether it is for professional work purposes or just to hangout with friends. This application provides a collaborative learning platform for its users to discuss, develop and organise their work, especially for coding purposes, in real-time. Features like video call and chat are also included in order to make this a stand alone application and facilitate a feeling of connectedness among the users. The application is built to Host rooms and allow users to join to this room in order to access its features.

**Use Cases:**

* This can be used in a classroom environment to demonstrate projects and tackle errors that show up only during execution. The real time conferencing features like audio/ video sharing allow the students to make themselves clear about the programming concepts by voicing their doubts.
* Students may use this to edit their projects with their peers and test it simultaneously. They may also discuss using white boards, as well as document their Software Requirements along with their project code.
* Corporate companies can use this for their professional work too, for e.g. in order to collaborate among other employees, and managers can monitor the state of the projects under their supervision thereby understanding what needs to be done in order to complete the projects within the given deadlines.

**Real-time Collaborative Code Editing:**

Collaborative code editing in real time lets all the participants in a room to edit a file at the same time. The creator of a room can import a file from their own github repository or create a new one and push to their github. Any new participant who joins the room can fork the repository to their own git and commit to it. New files and folders can be created and will be added to the user’s github as well.

Participants also have the option to turn off collaborative editing and work on programs individually and can switch back to collaborative mode at any point later in the meeting.

**Code Execution support:**

Participants in a room can execute the program they’re working on at any point of time. Currently the code execution support applies only to programs written in C/C++, Java, Python and JavaScript.

**Chat Features:**

A dynamic chat api consisting of shared group messaging and private messaging among other individuals from the same room will be implemented. This is to facilitate common functionalities like sharing links among the room made easier. Beside this, the normal chat features can also be done.

**Real-time Video/audio Communication:**

Users can freely communicate with other participants by virtually seeing them and also vocally communicating with them. This provides the collaborators a feel of the offline environment.

**Collaborative Doc editing:**

The rooms participants can use this feature to start planning on their project’s documentation like SRS documentation. Apart from this they can also use it for any docx or text file editing. In this part of collaboration features like text styling can be done.

**Whiteboard:**

A realtime whiteboard can be implemented for the users to collaborate among co-participants to plan about their project. It is generally said diagrams can better exhibit a view than saying it orally or through documenting it. Also it can be used by tutor to take a step into whiteboard environment and explain as efficiently as it can be done through a offline environment

**Task Planner:**

Task planning is most important in any collaborative work. For example the project head can work on defining the set of tasks by adding those to the task planner as and when they feel like doing it. The tutors can add tasks related to the days plan, so they won't miss out any important message they planned on delivering

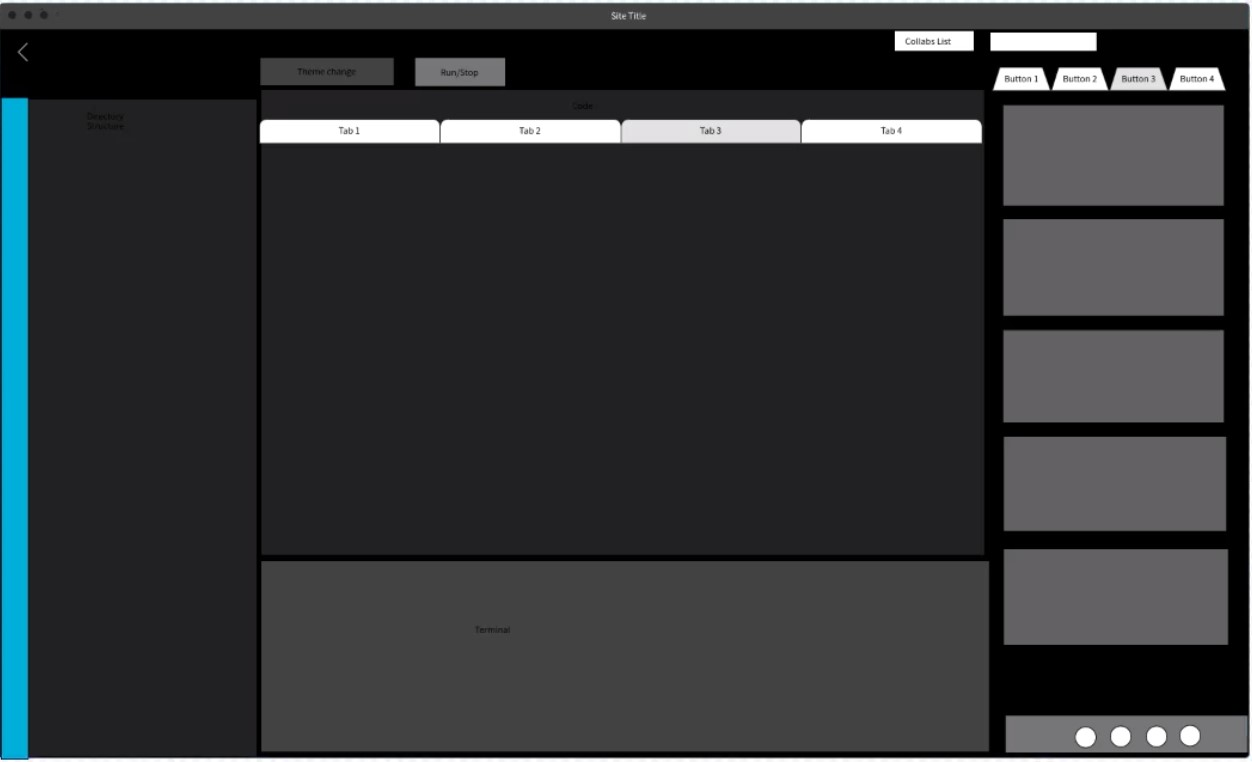
**Version Control Management of project using Git APIs:**

The main github repository created at the time of starting the room will be forked to any new participant’s github when they join the room if the user has edit access. When the participant has enabled collaborative mode they can only pull the changes that were committed to the main repository it was forked from. When the participant disables collaborative mode they can only commit to their own repository and fetch and merge changes from the main repository when they switch back to collaborative mode.

**Technology Used:**

* Monaco Editor : Gives user native VS code editor
* YJS : Real Time code collaboration using CRDT(Conflict-Free Replicated data type).
* Quill Editor : Doc Editing
* MongoDB : Shared database (Atlas)
* WebRTC : Video/Audio communication
* PeerJS: Peer to Peer data transfer.
* Socket.IO : Collaborate among users belonging to same room and separates one room from another ( Chat and WhiteBoard)
* Git Api’s: Version Control.
* React JS: Frontend.
* Node, Express JS: Server.
* Heroku: Deploy backend.
* Netlify: Deploy frontend.

**Sample UI:**

****