SketchPad! – Real Time Canvas Sharing Application

(Source Code available at: https://github.com/praveshganwani/sketch-pad)

Description:

Looking at the current pandemic situation, everything happens virtually over network connections. We have teams working virtually but simultaneously on a single project.

Such teams often face difficulties to express their ideas. This is because of restrictions imposed by virtual boundaries.

Our project "SketchPad!" tries to eliminate these virtual boundaries by bringing in real time canvas which can be shared amongst a team. This will help the team members to not only express their ideas or views virtually but also enhance other team members' ideas and collaborate on a project.

Technology Stack:

a) **Front End:** HTML, CSS, p5 JavaScript Library
b) **Back End:** Node.JS
c) **Database:** MySQL

How does it exactly work?

The p5 JS library provides us with a canvas over the front end. This canvas not only captures mouse strokes but also enables us to use metadata (i.e. width of the stroke, color of the stroke, etc.) along with primary X and Y axis co-ordinates.

The back end is implemented using Node.JS over Express Framework. Sockets are primarily used for establishing multi-client single server connections.

We have used Socket.IO to enable real-time, bidirectional and event-based communication. Socket.IO is a library that enables communication between the browser and the server. It consists of:

a) A Node.JS server (implemented using Socket APIs in back end).
b) A JavaScript Client Library for the browser (CDN in front end).

Whenever client logins to the portal, it establishes a socket connection with the back end. The client details get stored in the MySQL database along with the default stroke width and stroke color. The client is given a feature to change the stroke width and the stroke color of the brush. These values are stored as metadata along with X and Y co-ordinates. This metadata is broadcasted to other clients connected to the server. The metadata is interpreted by p5 JS and the same stroke is drawn for every client that is connected to the server.

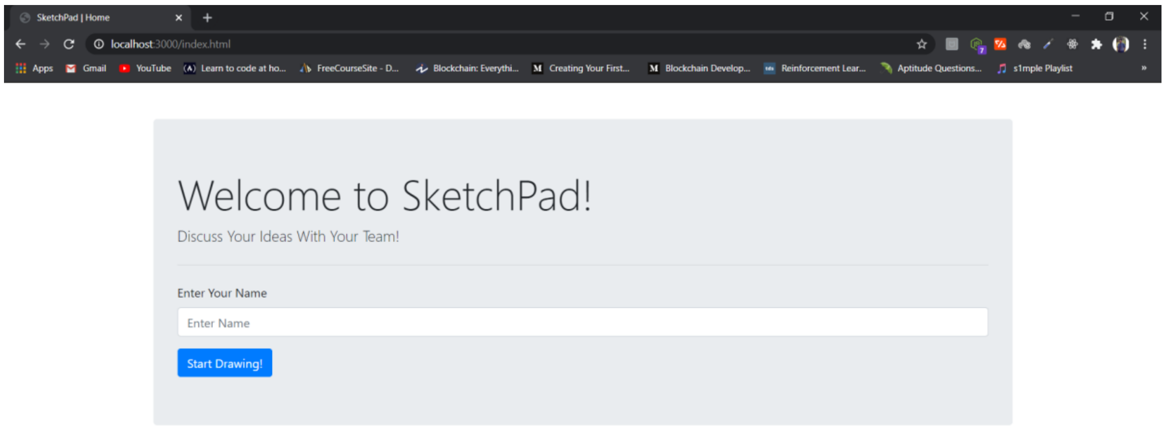
<u>**Understanding Socket.IO and its Advantages:**</u>

We know that web browser works over HTTP protocol. The HTTP protocol follows the request/response cycle. However, our application demands continuous responses (i.e. socket metadata) from the server. This can be achieved using polling. Polling strategy works in way such that the client keeps on pinging the server every defined time interval so as to receive continuous responses from the server. However, polling causes a network overhead and hence it is not preferred.

Socket.IO internally uses WebSockets. Hence, whenever a client connects to the server, it will try to establish a WebSocket connection if possible, and will fall back on HTTP long polling if not. WebSockets provide full-duplex mode of communication over the TCP protocol by exposing the underlying client TCP socket which will help us overcome the problem of maintaining a constant response flow from the server to the client even if there is no response from the client. They work as UDP providing the reliability of TCP. It also helps us store metadata (in our case, the stroke width and the stroke color) along with the socket connection. This is why, Socket.IO is a good choice for implementing Real Time Canvas Sharing application.

<u>**Screenshots of the Application:**</u>

P.T.O.



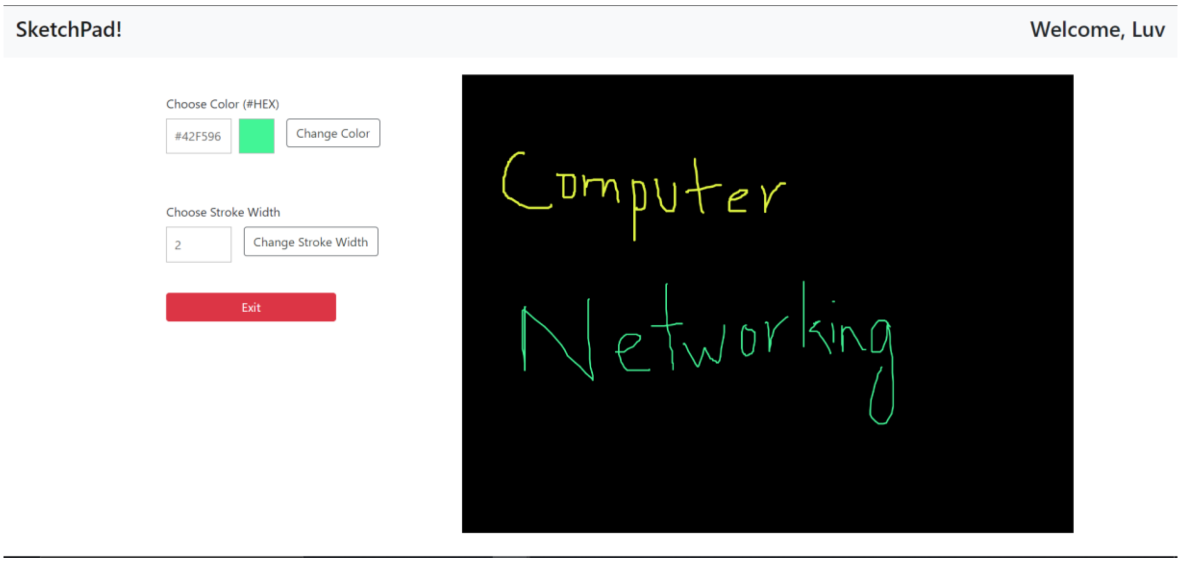
Welcome to SketchPad!

Discuss Your Ideas With Your Team!

Enter Your Name

Enter Name

Start Drawing!



SketchPad!

Welcome, Luv

Choose Color (#HEX)

#42F596

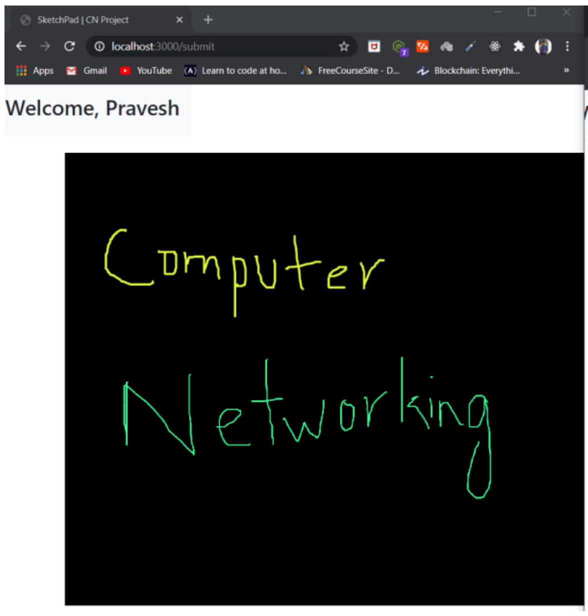
Change Color

Choose Stroke Width

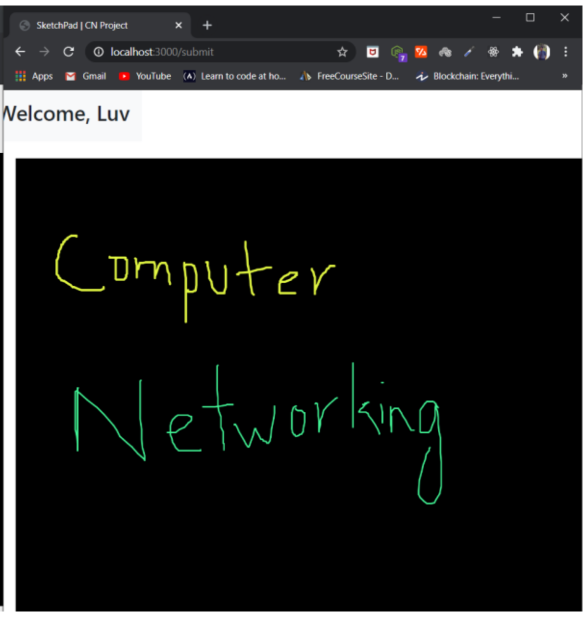
2

Change Stroke Width

Exit



Welcome, Pravesh



Welcome, Luv