Mini Project Report (KCS-554)

on

Open White Board

Submitted in partial fulfillment for award of

BACHELOR OF TECHNOLOGY

Degree

In

COMPUTER SCIENCE & ENGINEERING

2023-24

Submitted By:
Shivam Pandey (2100330100219)
Vansh Kabaria (2100330100245)
Shivam Kushwaha (2100330100218)

Under the Guidance of:
Mr. Vineet Shrivastava
Assistant Professor-CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJ KUMAR GOEL INSTITUTE OF TECHNOLOGY

DELHI-MEERUT ROAD, GHAZIABAD

Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION—ENVIRONMENT AND TOOLS USED:

No need to dive into much details to start using Teaching Board. After launching, simply take the pen and start writing on the screen. Do not be fearful go frankly, as if you were writing with a ballpoint pen on a sheet of paper. A simple look at the icons should allow you to quickly understand how Open Board works.

Using flexbox property of cascading sheet, user interface is easy to use and accessible. Every new user can interact with the teaching board in a convenient manner. In case we want to broadcast on multiple devices it can be done easily since it support real time accessibility.

We have added these functionalities in our project:

- We can also change width of pencil as well as eraser using dynamic range option.
- We are having 3 different colors for pen.
- Whatever stuff we write on our board can be saved on our device in .jpg format.
- For effective content delivery we can also upload from local device.
- Operations can be Undo/Redo using buttons.
- User Interface is very attractive.
  This is a real time board which means that same board can be run on multiple machines simultaneously.

1.1 Technologies Used

1.1.1 EXPRESS JS

Express.js is a small framework that works on top of Node.js web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middleware and routing. It adds helpful utilities to Node.js HTTP objects and facilitates the rendering of dynamic HTTP objects.

Why Express?

- Develops Node.js web applications quickly and easily.
- It's simple to set up and personalize.
- Allows you to define application routes using HTTP methods and URLs.
- Includes a number of middleware modules that can be used to execute additional requests and responses activities.
- Simple to interface with a variety of template engines, including Jade, Vash, and EJS.
- Allows you to specify a middleware for handling errors.

1.1.2 JavaScript

JavaScript, a versatile programming language, initially designed for client-side web development, now extends its reach to server-side applications through platforms like Node.js. Its syntax is approachable, employing variables with loose typing and supporting various data types. With functions as first-class citizens, JavaScript excels in asynchronous programming, using mechanisms like callbacks, Promises, and async/await. It manipulates the Document Object Model (DOM), enabling dynamic updates in web applications and responding to user actions through event-driven programming. The language thrives on a rich ecosystem of libraries and frameworks, including jQuery, React, and Express.js. JavaScript embraces modern ECMAScript features, such as arrow functions and template literals, offering developers a powerful and adaptable tool for building dynamic and interactive digital experiences.

1.1.3 NODE JS

Node JS is an open-source and cross-platform runtime environment built on Chrome’s V8 JavaScript engine for executing JavaScript code outside of a browser. It provides an event-driven, non-blocking (asynchronous) I/O and cross-platform runtime environment for building highly scalable server-side applications using JavaScript.

Features of Node JS:

- Easy Scalability
- Real time web apps
- Fast Suite
- Easy to learn and code
- Data Streaming
- Corporate Support

CHAPTER 2

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware: Hardware Specification Computer

- IBM PC/AT Computer or above
- Processor
- Intel Pentium Core 2 Duo or above
- Memory
- RAM 1 GB or above
- HDD 80.0 GB or above

OS: windows 7, 10 or higher**.**

Written in: Java language.

Platform: HTML, CSS, Java Script.

Software Specification:

- Front End: Visual Basic 6.0
- Back End: Microsoft Access 2003

CHAPTER 3

APPLICATION ARCHITECTURE

1. Client-Side (Frontend) – HTML, CSS, & JavaScript:

HTML (Structure and Layout):

Boilerplate Code: Common HTML structure is defined as a boilerplate, providing the foundational layout for all EJS files.
Modular Components: HTML is used to create modular components for different sections of the webpage, ensuring a consistent structure.

CSS (Styling and Design Enhancement):

CSS Folder: A dedicated folder contains stylesheets to enhance the visual appeal of the website.
Responsive Design: CSS is utilized for responsive design, ensuring a seamless experience across various devices and screen sizes.
Component Styling: Each modular component is styled independently, maintaining a clean separation of concerns.

JavaScript (Functionality and Interaction):

Interactive Features: JavaScript is responsible for implementing interactive features on the frontend, such as dynamic content updates and user interactions.
Event Handling: JavaScript handles user events, such as clicks and input, to trigger specific actions and maintain a responsive user interface.
Integration with Backend: Communicates with the backend, facilitating data retrieval and updates without page reloads.

2. Server-Side (Backend) - Node.js and Express.js:

Express.js Framework:

Serves as the backend framework for handling HTTP requests and responses.

Defines routes, controllers, and middleware for processing requests.

RESTful APIs:

RESTful API endpoints are designed for communication between the frontend and backend.

Endpoints handle requests for user data, wellness articles, authentication, etc.

Middleware:

Middleware functions handle tasks like authentication, logging, and error handling.

WebSocket (Optional):

Real-time features, such as chat or notifications, may use WebSockets for bidirectional communication.

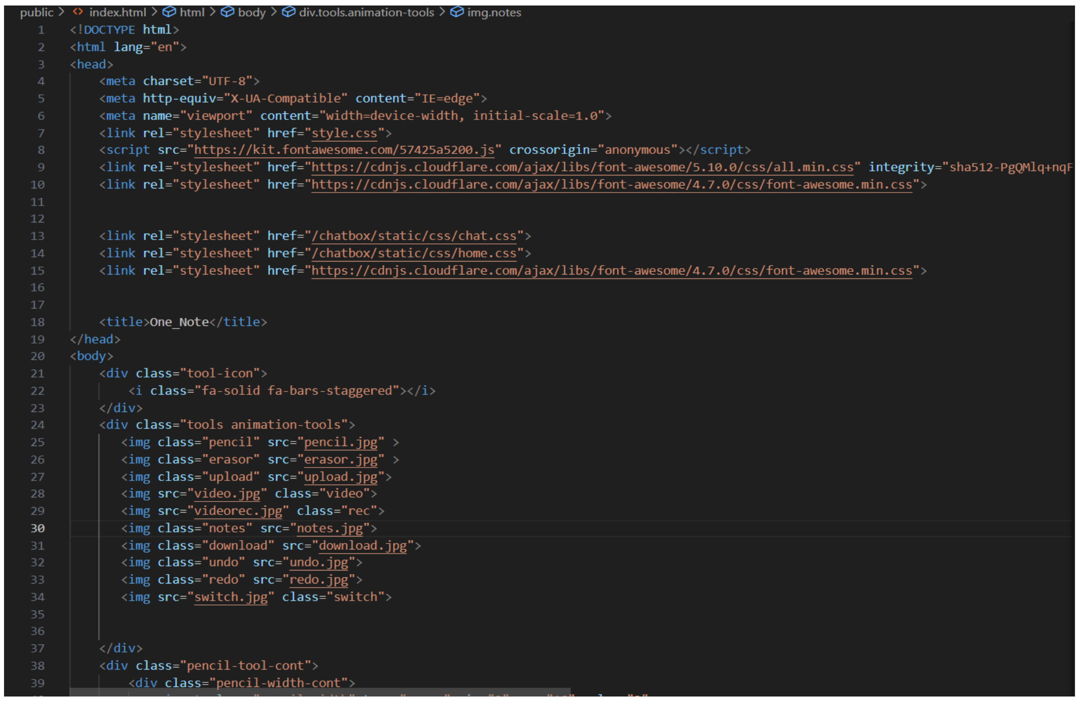
CHAPTER 4

PROJECT MODULES DESIGN

4.1 Html Modules

Firstly, we start with the main HTML portion.

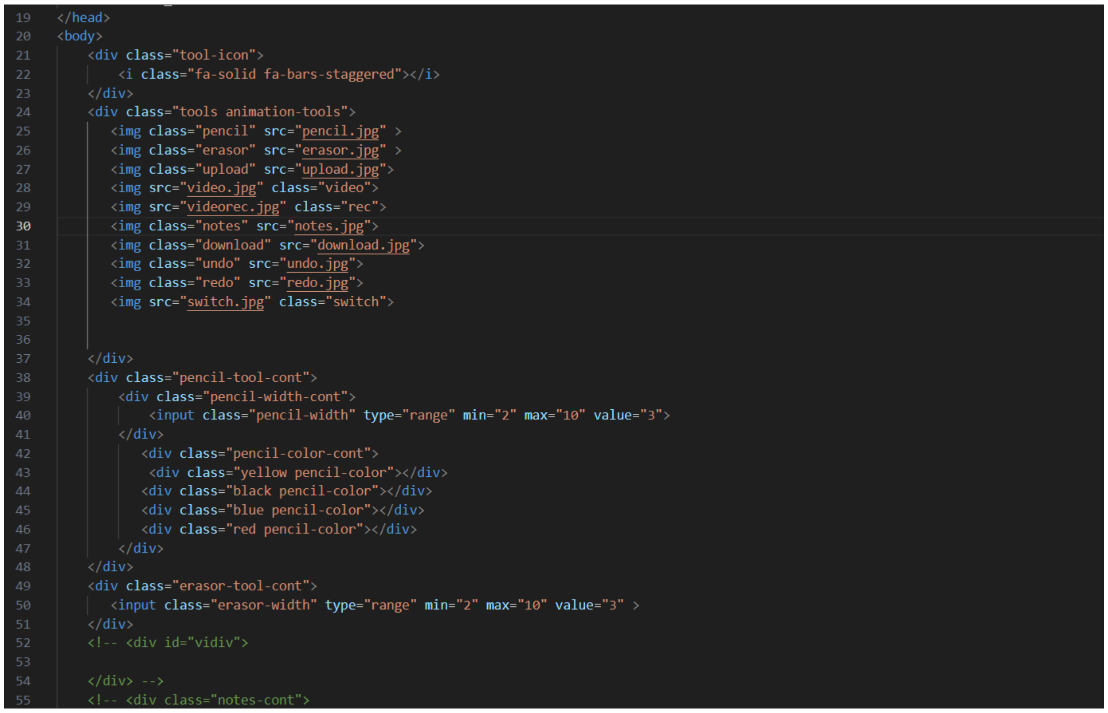
Initially we make the head portion in which we add the title of our project and we link all the scripts and design files.



```
public > index.html > html > body > div.tools.animation-tools > img.notes
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="style.css">
    <script src="https://kit.fontawesome.com/57425a5200.js" crossorigin="anonymous"></script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.10.0/css/all.min.css" integrity="sha512-PgQMlq+nqF
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">


    <link rel="stylesheet" href="/chatbox/static/css/chat.css">
    <link rel="stylesheet" href="/chatbox/static/css/home.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">


    <title>One_Note</title>
</head>
<body>
    <div class="tool-icon">
        <i class="fa-solid fa-bars-staggered"></i>
    </div>
    <div class="tools animation-tools">
      <img class="pencil" src="pencil.jpg" >
      <img class="erasor" src="erasor.jpg" >
      <img class="upload" src="upload.jpg">
      <img src="video.jpg" class="video">
      <img src="videorec.jpg" class="rec">
      <img class="notes" src="notes.jpg">
      <img class="download" src="download.jpg">
      <img class="undo" src="undo.jpg">
      <img class="redo" src="redo.jpg">
      <img src="switch.jpg" class="switch">


    </div>
    <div class="pencil-tool-cont">
        <div class="pencil-width-cont">
```

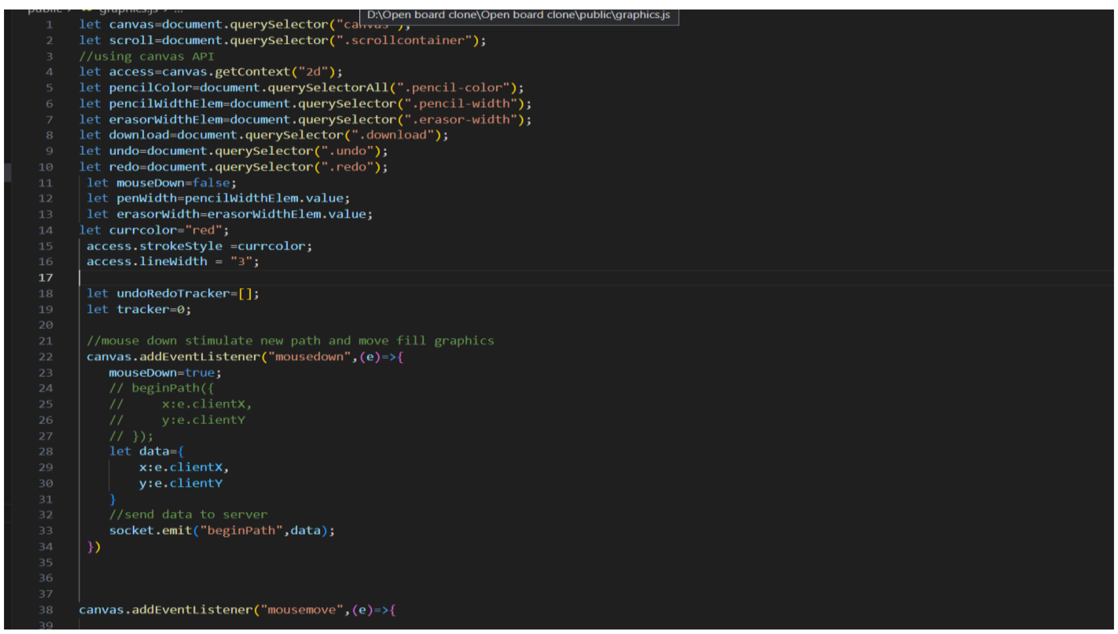
Fig 4.1 Boilerplate Code

Now we start off with the main body of our website. Initially we create the header using boilerplate in which we only try to focus on two elements.



```html
</head>
<body>
    <div class="tool-icon">
        <i class="fa-solid fa-bars-staggered"></i>
    </div>
    <div class="tools animation-tools">
       <img class="pencil" src="pencil.jpg" >
       <img class="erasor" src="erasor.jpg" >
       <img class="upload" src="upload.jpg">
       <img src="video.jpg" class="video">
       <img src="videorec.jpg" class="rec">
       <img class="notes" src="notes.jpg">
       <img class="download" src="download.jpg">
       <img class="undo" src="undo.jpg">
       <img class="redo" src="redo.jpg">
       <img src="switch.jpg" class="switch">


    </div>
    <div class="pencil-tool-cont">
        <div class="pencil-width-cont">
            <input class="pencil-width" type="range" min="2" max="10" value="3">
        </div>
           <div class="pencil-color-cont">
            <div class="yellow pencil-color"></div>
           <div class="black pencil-color"></div>
           <div class="blue pencil-color"></div>
           <div class="red pencil-color"></div>
        </div>
    </div>
    <div class="erasor-tool-cont">
       <input class="erasor-width" type="range" min="2" max="10" value="3" >
    </div>
    <!-- <div id="vidiv">

    </div> -->
    <!-- <div class="notes-cont">
```

Fig 4.2



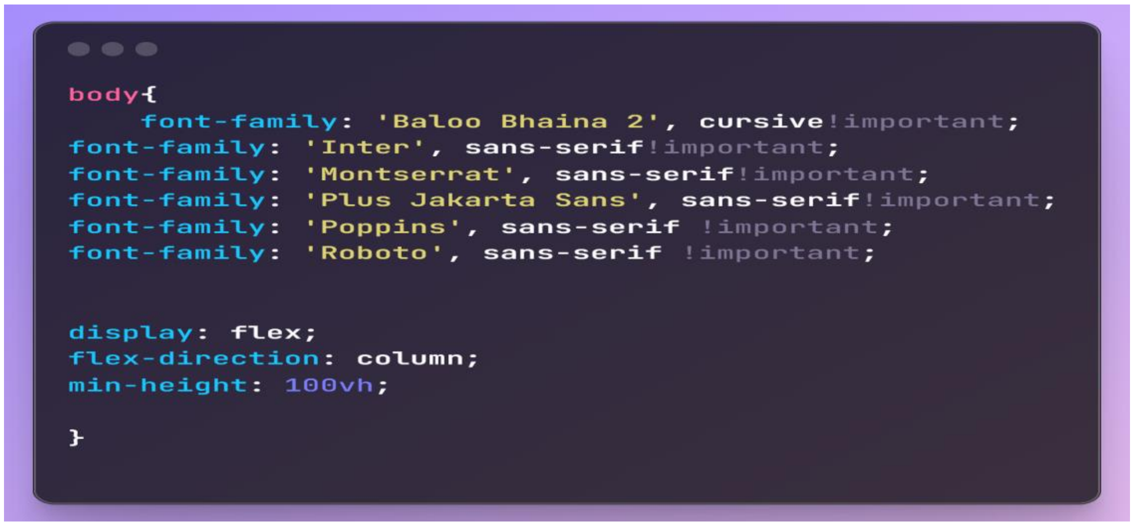
```javascript
let canvas=document.querySelector("canvas");
let scroll=document.querySelector(".scrollcontainer");
//using canvas API
let access=canvas.getContext("2d");
let pencilColor=document.querySelectorAll(".pencil-color");
let pencilWidthElem=document.querySelector(".pencil-width");
let erasorWidthElem=document.querySelector(".erasor-width");
let download=document.querySelector(".download");
let undo=document.querySelector(".undo");
let redo=document.querySelector(".redo");
 let mouseDown=false;
 let penWidth=pencilWidthElem.value;
 let erasorWidth=erasorWidthElem.value;
let currcolor="red";
 access.strokeStyle =currcolor;
 access.lineWidth = "3";

 let undoRedoTracker=[];
 let tracker=0;

 //mouse down stimulate new path and move fill graphics
 canvas.addEventListener("mousedown",(e)=>{
    mouseDown=true;
    // beginPath({
    //     x:e.clientX,
    //     y:e.clientY
    // });
    let data={
        x:e.clientX,
        y:e.clientY
    }
    //send data to server
    socket.emit("beginPath",data);
 })



canvas.addEventListener("mousemove",(e)=>{
```

Fig 4.3

4.2 CSS Modules

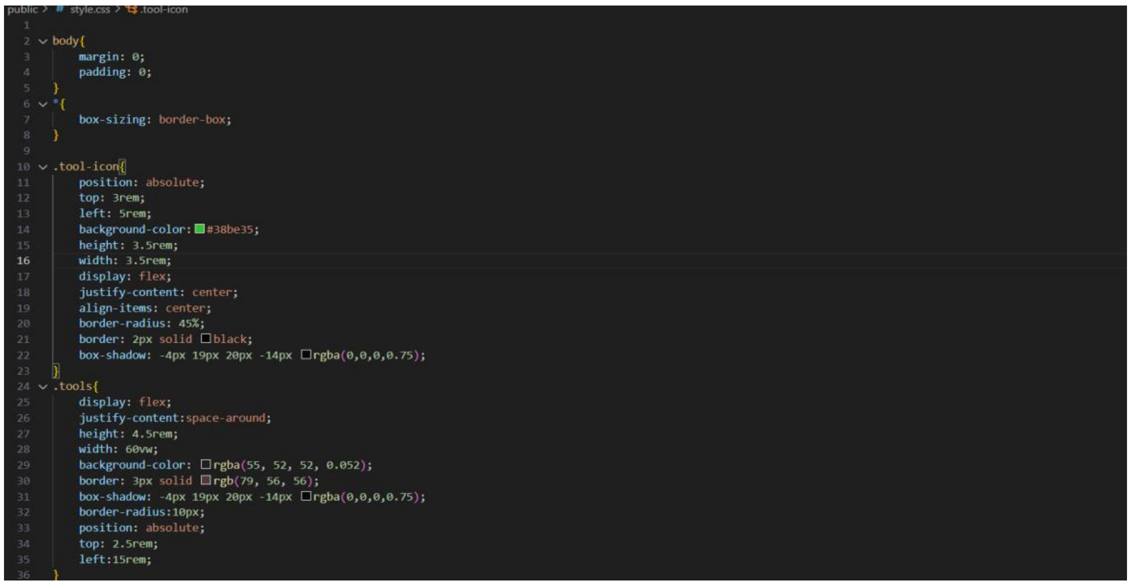
Now we start to design the page so that the page looks nice to navigate.

We start to declare some universal font and text colours.



```css
body{
    font-family: 'Baloo Bhaina 2', cursive!important;
font-family: 'Inter', sans-serif!important;
font-family: 'Montserrat', sans-serif!important;
font-family: 'Plus Jakarta Sans', sans-serif!important;
font-family: 'Poppins', sans-serif !important;
font-family: 'Roboto', sans-serif !important;


display: flex;
flex-direction: column;
min-height: 100vh;

}
```

Fig 4.4

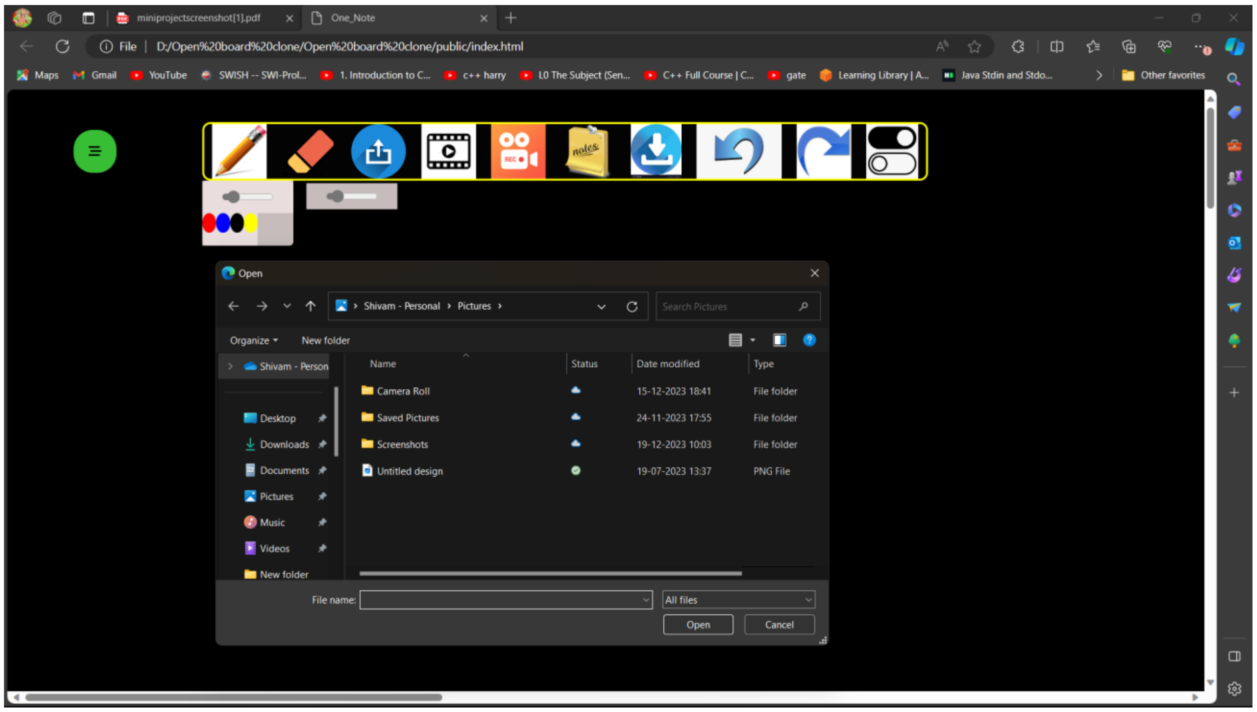


```css

body{
    margin: 0;
    padding: 0;
}
*{
    box-sizing: border-box;
}

.tool-icon{
    position: absolute;
    top: 3rem;
    left: 5rem;
    background-color: #38be35;
    height: 3.5rem;
    width: 3.5rem;
    display: flex;
    justify-content: center;
    align-items: center;
    border-radius: 45%;
    border: 2px solid black;
    box-shadow: -4px 19px 20px -14px rgba(0,0,0,0.75);
}
.tools{
    display: flex;
    justify-content:space-around;
    height: 4.5rem;
    width: 60vw;
    background-color: rgba(55, 52, 52, 0.052);
    border: 3px solid rgb(79, 56, 56);
    box-shadow: -4px 19px 20px -14px rgba(0,0,0,0.75);
    border-radius:10px;
    position: absolute;
    top: 2.5rem;
    left:15rem;
}
```

Fig 4.5

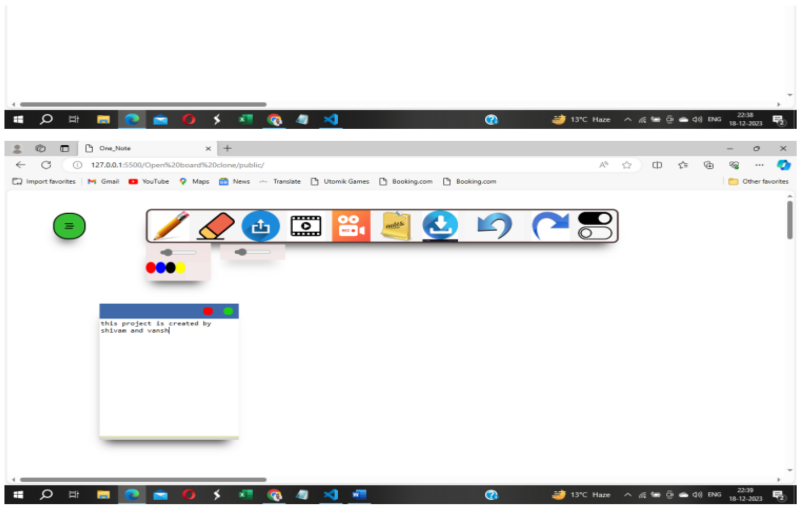
CHAPTER 5

PROJECT SNAPSHOT



Name	Status	Date modified	Type
Camera Roll		15-12-2023 18:41	File folder
Saved Pictures		24-11-2023 17:55	File folder
Screenshots		19-12-2023 10:03	File folder
Untitled design		19-07-2023 13:37	PNG File

Figure 5.1



this project is created by shivam and vansh

Figure 5.2

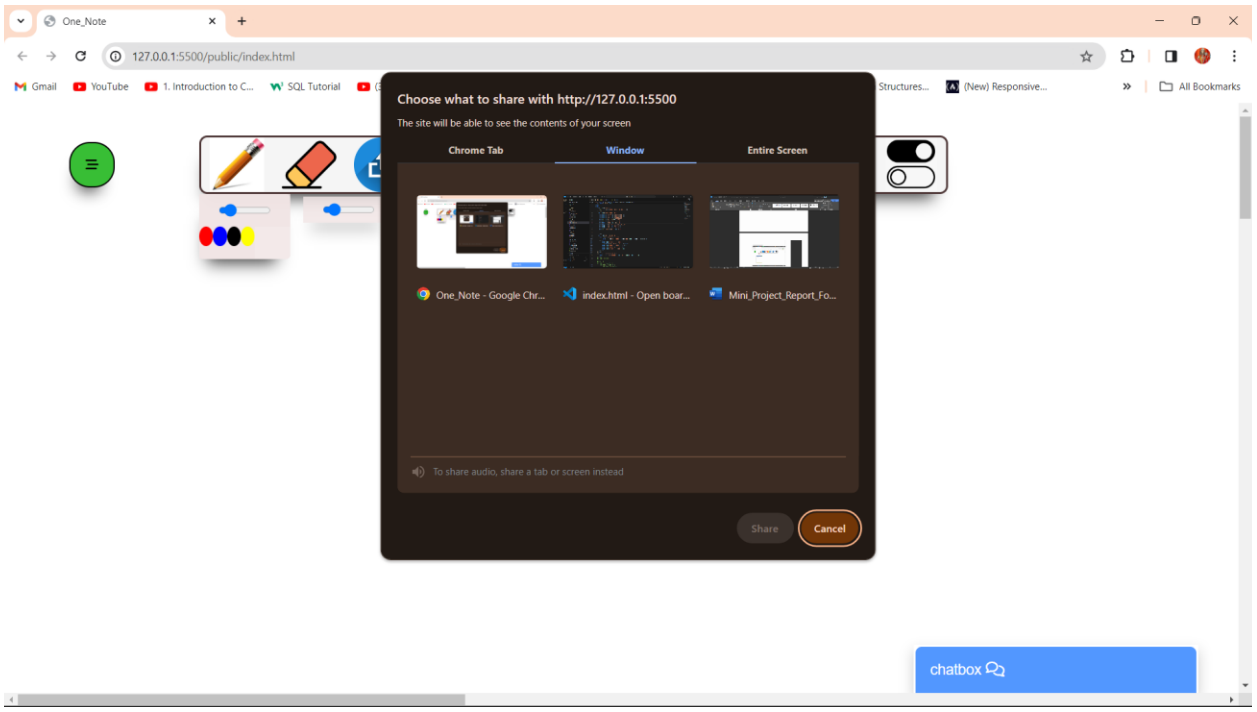

One_Note
127.0.0.1:5500/public/index.html
Gmail
YouTube
1. Introduction to C...
SQL Tutorial
Structures...
(New) Responsive...
All Bookmarks
Choose what to share with http://127.0.0.1:5500
The site will be able to see the contents of your screen
Chrome Tab
Window
Entire Screen
One_Note - Google Chr...
index.html - Open boar...
Mini_Project_Report_Fo...
To share audio, share a tab or screen instead
Share
Cancel
chatbox

Figure 5.3

CHAPTER 6

LIMITATIONS

External Resource Creation

As the software is used for the virtual online whiteboard can often be new within a school, there is often a requirement to produce additional resources for use within lessons. However, there is increasingly a range of content, such as presentations and interactive elements, which are available for free online, such as the TES, online teacher forums, and from subject-specific websites such as the Computing at schools (CAS) resources area.

Limited Device Access

If content will be accessed using technology within the classroom, it is important devices are readily available and, where feasible, form part of pupils' everyday experiences in lessons. Where one-to-one device provision is not available, it may be necessary to book devices out in advance and work with both staff and pupils, such as Digital Leaders, to ensure they are charged and working as expected.

Ensure Content is Engaging for Pupils

Virtual and online whiteboard software has a number of inbuilt tools, which can be used to engage pupils on the task at hand. These often include animated objects, such as moving characters, which can interact with other objects on the screen.

CHAPTER 7

FUTURE SCOPE

Touch-Based Interface:

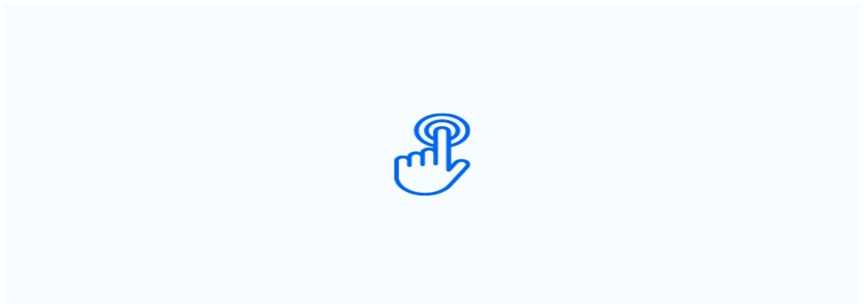


Fig. 7.1

One of the primary features of an interactive whiteboard is its touch-based interface. Teachers and students can directly interact with the content displayed on the board by using their fingers or a stylus. This hands-on approach enhances engagement and encourages active participation.

Multi-Touch Capability:

Fig. 7.2

Interactive whiteboards often support multi-touch gestures, allowing multiple users to interact with the board simultaneously. This collaborative feature promotes teamwork and facilitates group activities, fostering a cooperative learning environment.

Digital Annotation:

Fig. 7.3

With an interactive whiteboard, teachers can annotate and highlight content in real-time. They can write, draw, underline, or circle important points, making it easier to emphasize key concepts. This feature enhances the clarity of explanations and aids in capturing students' attention.

Multimedia Integration:

Fig. 7.4

Interactive whiteboards seamlessly integrate with multimedia resources such as images, videos, and audio. Teachers can incorporate visual and auditory elements into their lessons, making them more dynamic and engaging. This multimedia integration appeals to various learning styles and helps reinforce concepts effectively.

Interactive Software:

Fig. 7.5

Interactive whiteboards come with dedicated software that offers a wide range of tools and features. These software packages provide access to pre-made lesson templates, interactive games, and educational applications, saving teachers valuable time in lesson preparation.

References

- Hennessy, S. (2011). The role of digital artefacts on the interactive whiteboard in supporting classroom dialogue. *Journal of Computer Assisted Learning*, *27*(6), 463–489. https://doi.org/10.1111/j.1365-2729.2011.00416.x
- Murcia, K., & Sheffield, R. (2010). Talking about science in interactive whiteboard classrooms. *Australasian Journal of Educational Technology*, *26*(4). https://doi.org/10.14742/ajet.1062
- Smith, H., Higgins, S., Wall, K., & Miller, J. (2005). Interactive whiteboards: boon or bandwagon? A critical review of the literature. *Journal of Computer Assisted Learning*, *21*(2), 91–101. https://doi.org/10.1111/j.1365-2729.2005.00117.x
- Bidaki, M. Z., & Mobasheri, N. (2013). Teachers' views of the effects of the Interactive White Board (IWB) on teaching. *Procedia - Social and Behavioral Sciences*, *83*, 140–144.
  https://doi.org/10.1016/j.sbspro.2013.06.027
- Hall, I., & Higgins, S. (2005). Primary school students' perceptions of interactive whiteboards. *Journal of Computer Assisted Learning*, *21*(2), 102–117. https://doi.org/10.1111/j.1365-2729.2005.00118.x
- Primary reference of the Teaching Board Clone has taken from you tube channel Pep-coding. Here is the link to channel content: https://www.youtube.com/results?search_query=pepcoding+open+board.
- Reference also got selected from Git hub repository :
- https://github.com/OpenBoard-org/OpenBoard