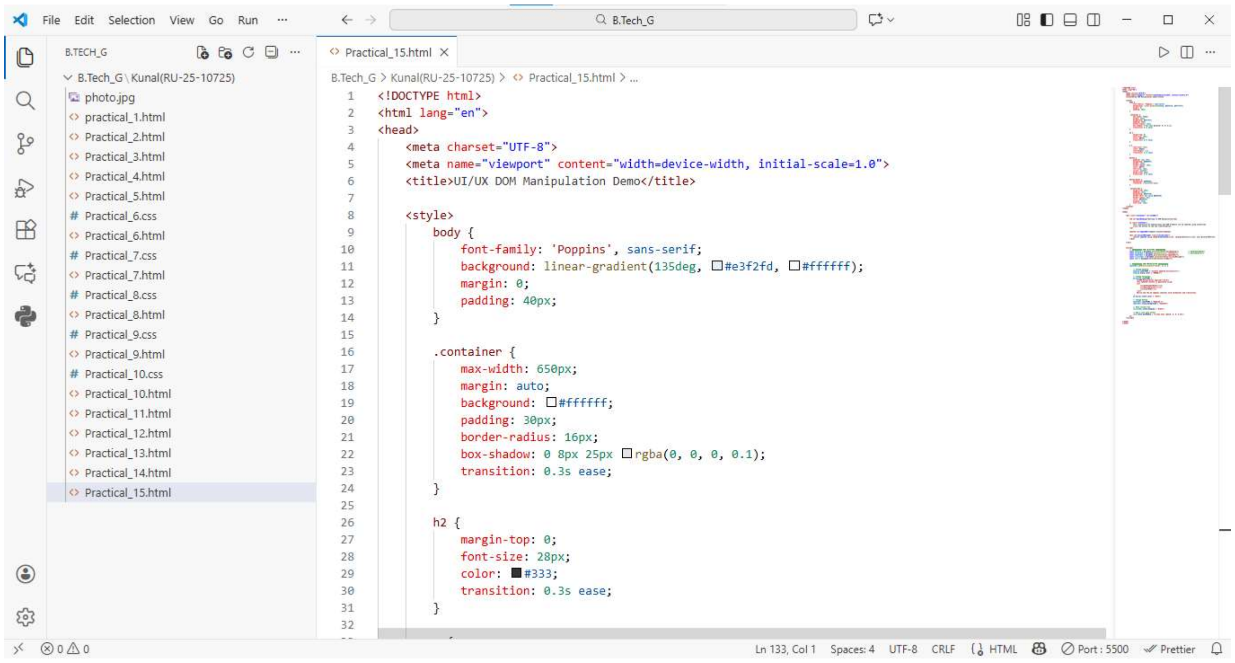
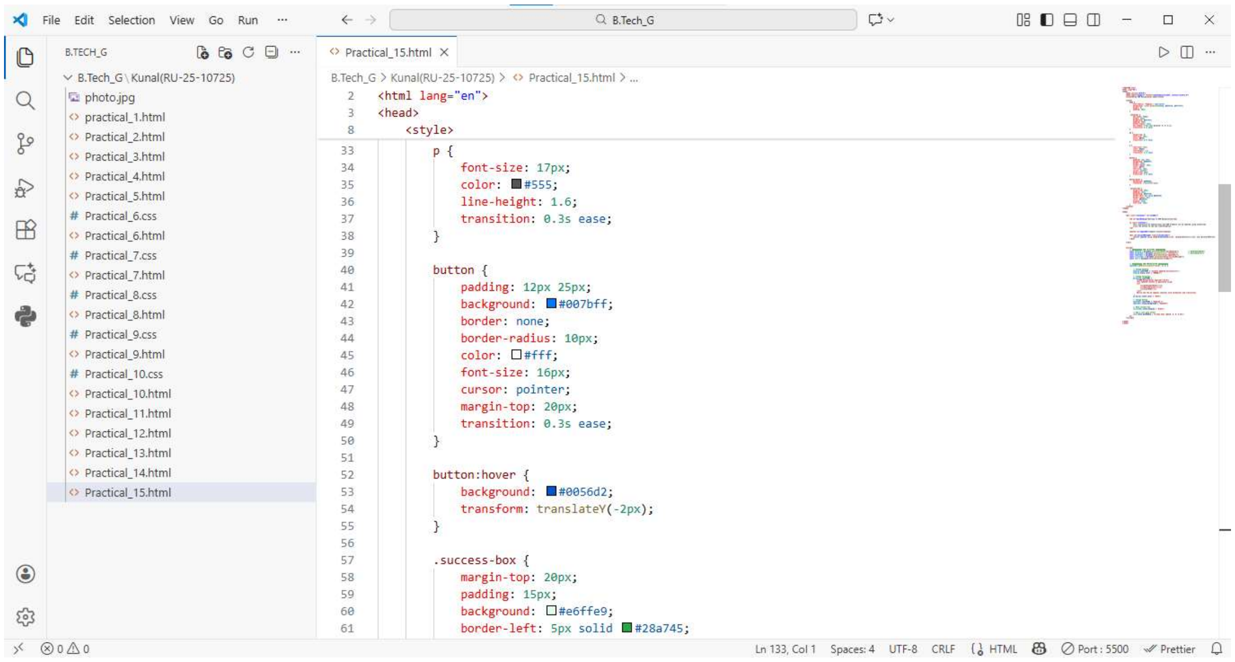
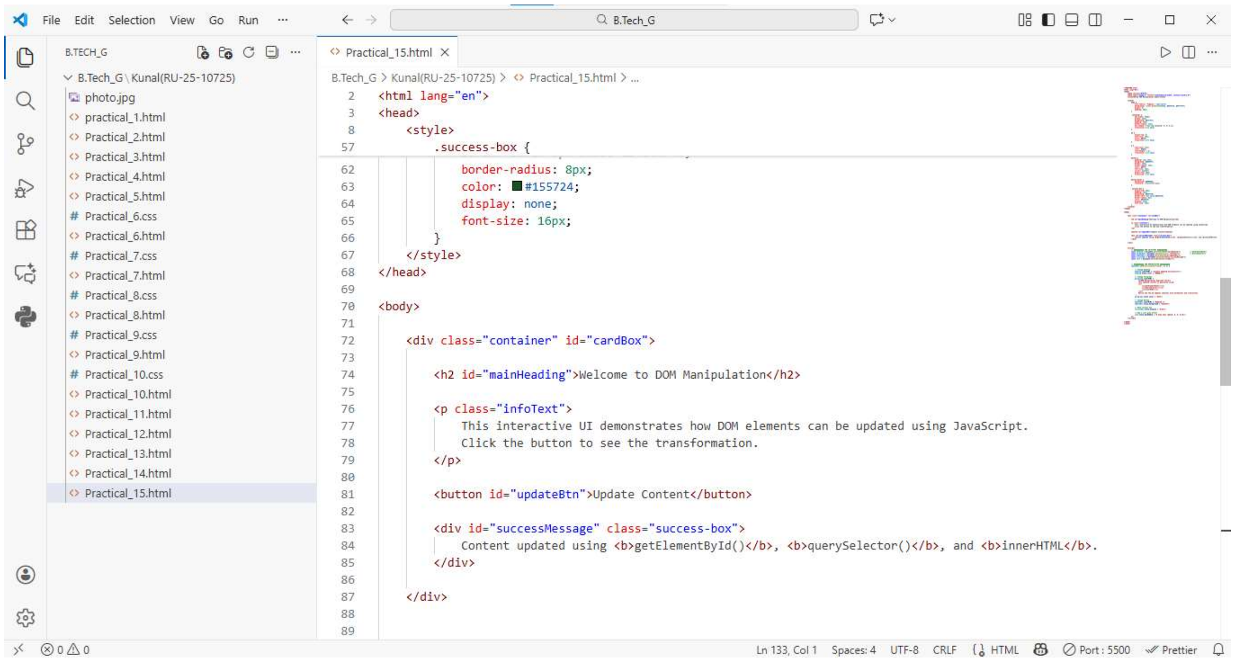
15. Demonstrate DOM manipulation using getElementById(), querySelector(), and innerHTML.



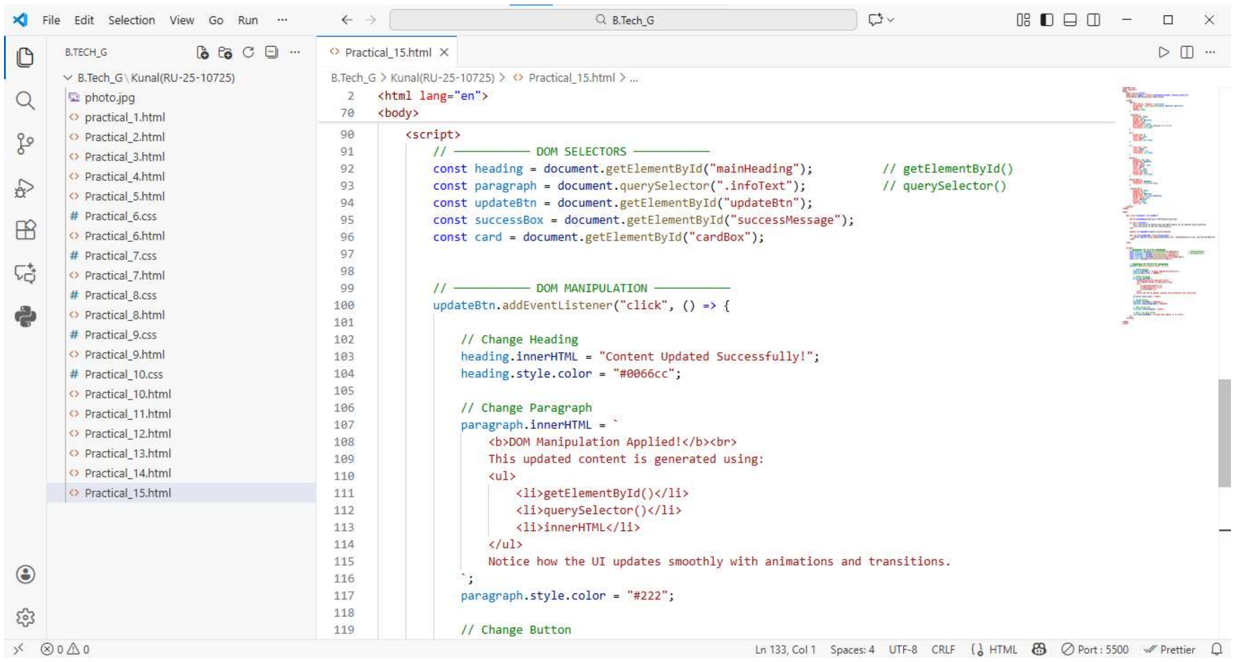
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>UI/UX DOM Manipulation Demo</title>

    <style>
        body {
            font-family: 'Poppins', sans-serif;
            background: linear-gradient(135deg, #e3f2fd, #ffffff);
            margin: 0;
            padding: 40px;
        }

        .container {
            max-width: 650px;
            margin: auto;
            background: #ffffff;
            padding: 30px;
            border-radius: 16px;
            box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
            transition: 0.3s ease;
        }

        h2 {
            margin-top: 0;
            font-size: 28px;
            color: #333;
            transition: 0.3s ease;
        }

```



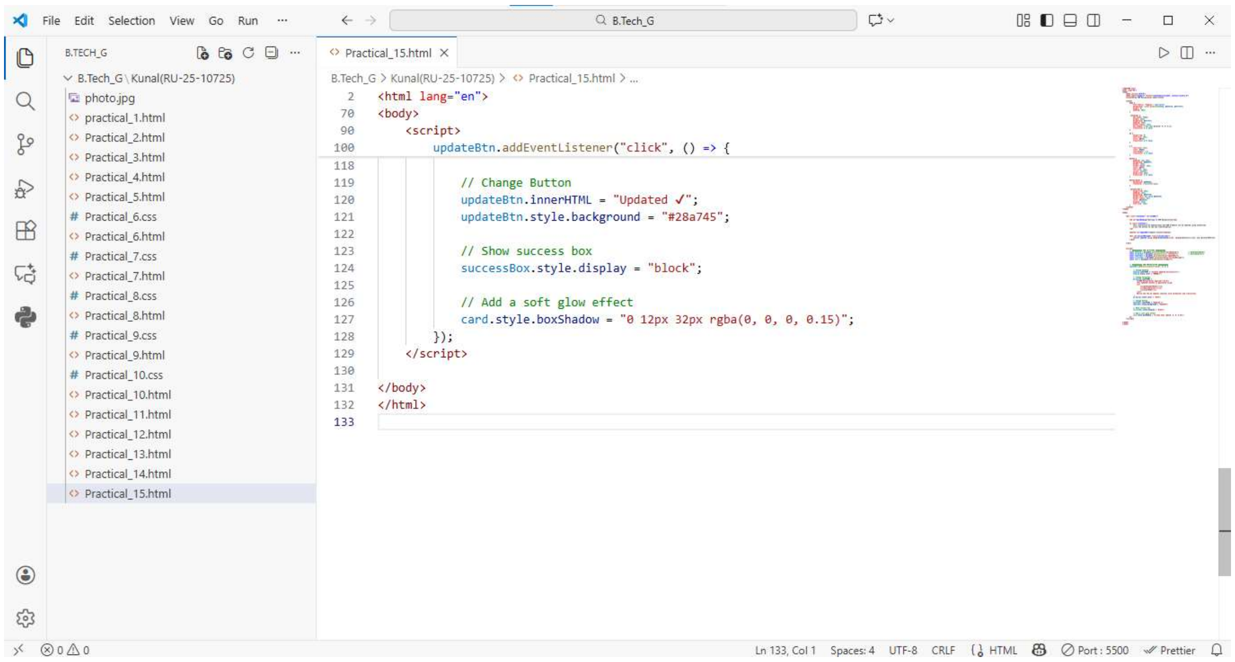
```
<html lang="en">
<head>
    <style>
        p {
            font-size: 17px;
            color: #555;
            line-height: 1.6;
            transition: 0.3s ease;
        }

        button {
            padding: 12px 25px;
            background: #007bff;
            border: none;
            border-radius: 10px;
            color: #fff;
            font-size: 16px;
            cursor: pointer;
            margin-top: 20px;
            transition: 0.3s ease;
        }

        button:hover {
            background: #0056d2;
            transform: translateY(-2px);
        }

        .success-box {
            margin-top: 20px;
            padding: 15px;
            background: #e6ffe9;
            border-left: 5px solid #28a745;
```



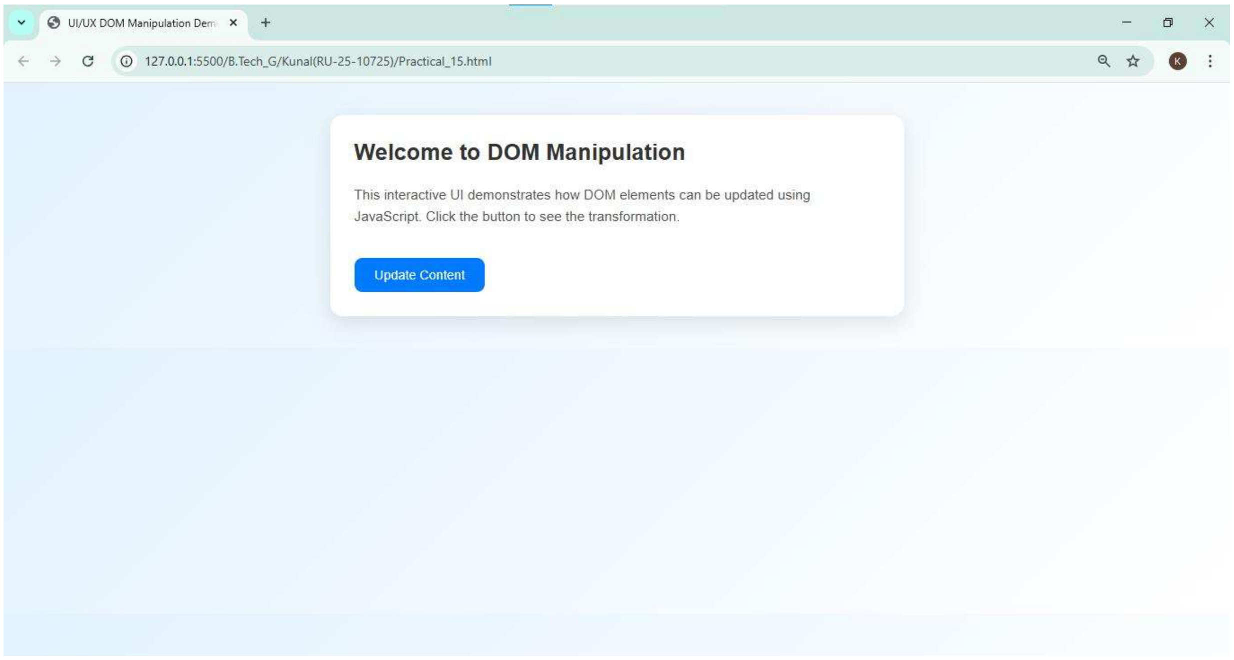
```html
<html lang="en">
<head>
    <style>
        .success-box {
            border-radius: 8px;
            color: #155724;
            display: none;
            font-size: 16px;
        }
    </style>
</head>

<body>

    <div class="container" id="cardBox">

        <h2 id="mainHeading">Welcome to DOM Manipulation</h2>

        <p class="infoText">
            This interactive UI demonstrates how DOM elements can be updated using JavaScript.
            Click the button to see the transformation.
        </p>

        <button id="updateBtn">Update Content</button>

        <div id="successMessage" class="success-box">
            Content updated using <b>getElementById()</b>, <b>querySelector()</b>, and <b>innerHTML</b>.
        </div>

    </div>


```



```html
<html lang="en">
<body>
    <script>
        // ---------- DOM SELECTORS ----------
        const heading = document.getElementById("mainHeading");          // getElementById()
        const paragraph = document.querySelector(".infoText");           // querySelector()
        const updateBtn = document.getElementById("updateBtn");
        const successBox = document.getElementById("successMessage");
        const card = document.getElementById("cardBox");


        // ---------- DOM MANIPULATION ----------
        updateBtn.addEventListener("click", () => {

            // Change Heading
            heading.innerHTML = "Content Updated Successfully!";
            heading.style.color = "#0066cc";

            // Change Paragraph
            paragraph.innerHTML = `
                <b>DOM Manipulation Applied!</b><br>
                This updated content is generated using:
                <ul>
                    <li>getElementById()</li>
                    <li>querySelector()</li>
                    <li>innerHTML</li>
                </ul>
                Notice how the UI updates smoothly with animations and transitions.
            `;
            paragraph.style.color = "#222";

            // Change Button
```

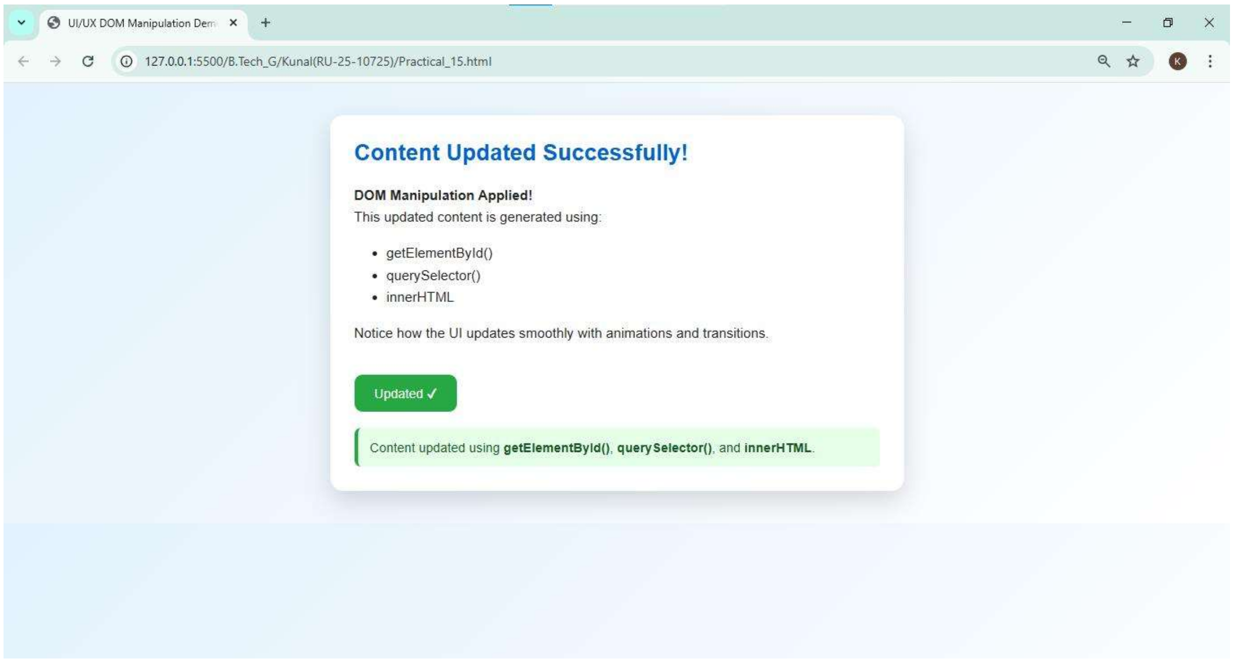


```
<html lang="en">
<body>
    <script>
        updateBtn.addEventListener("click", () => {

            // Change Button
            updateBtn.innerHTML = "Updated ✓";
            updateBtn.style.background = "#28a745";

            // Show success box
            successBox.style.display = "block";

            // Add a soft glow effect
            card.style.boxShadow = "0 12px 32px rgba(0, 0, 0, 0.15)";
        });
    </script>

</body>
</html>
```


UI/UX DOM Manipulation Dem
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_15.html

Welcome to DOM Manipulation

This interactive UI demonstrates how DOM elements can be updated using JavaScript. Click the button to see the transformation.


UI/UX DOM Manipulation Dem
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_15.html

Content Updated Successfully!

**DOM Manipulation Applied!**
This updated content is generated using:

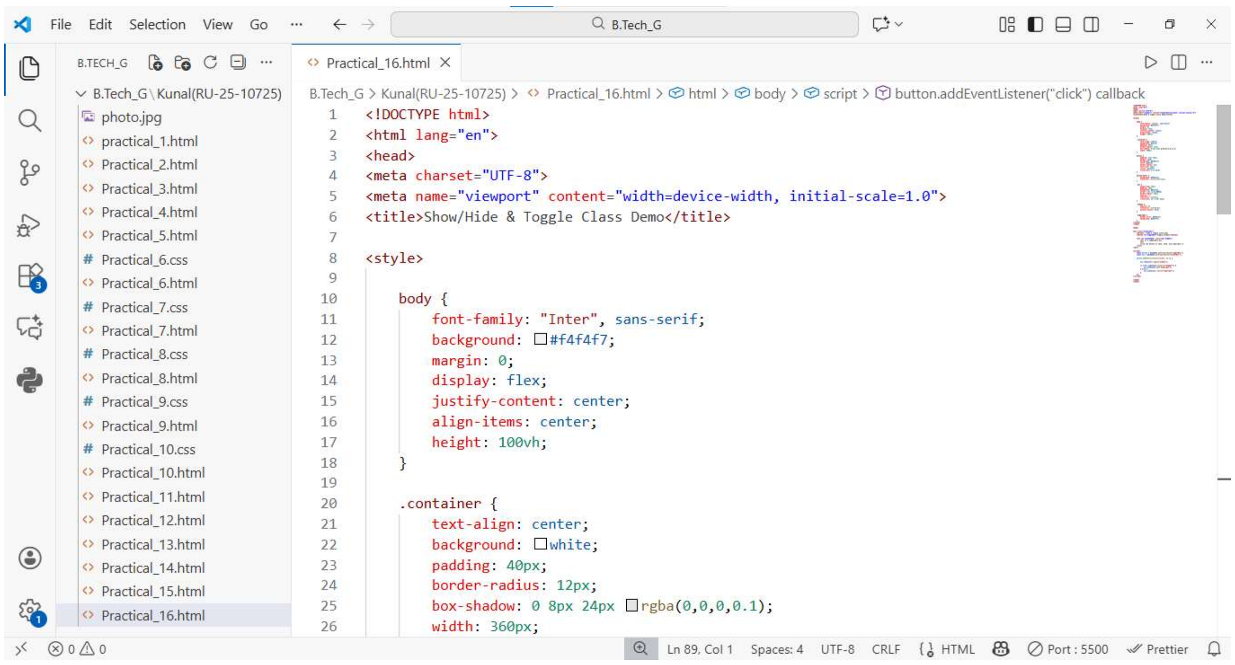
- getElementById()
- querySelector()
- innerHTML

Notice how the UI updates smoothly with animations and transitions.

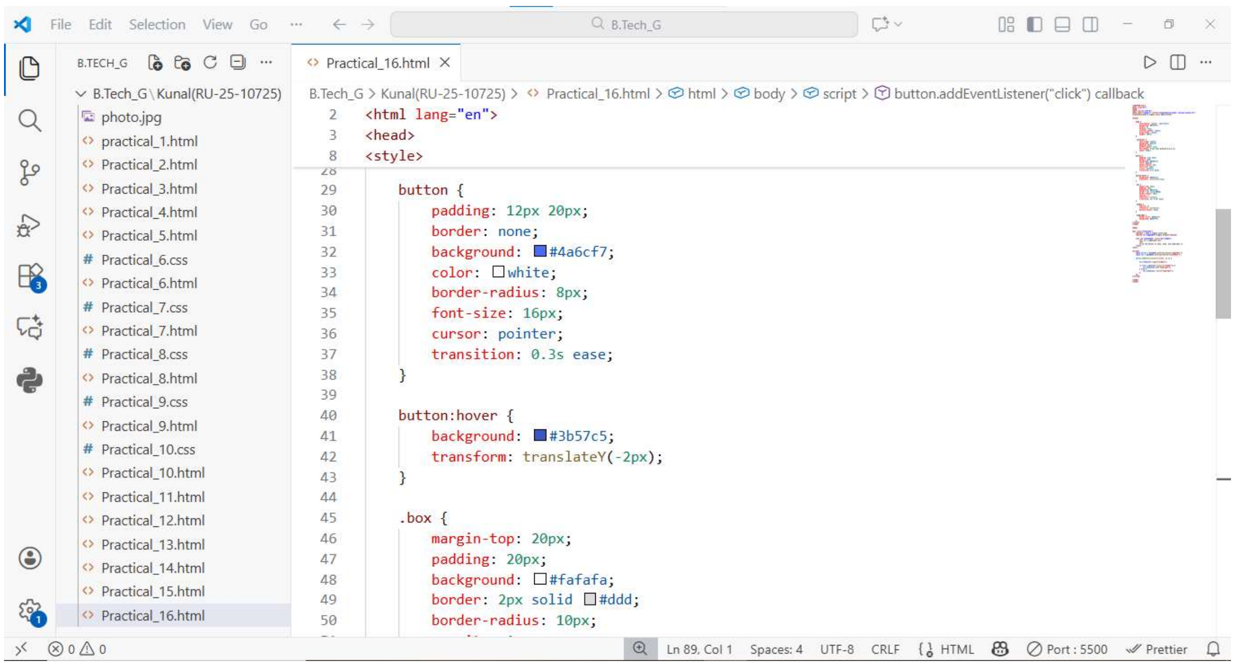
Updated ✓

Content updated using **getElementById()**, **querySelector()**, and **innerHTML**.

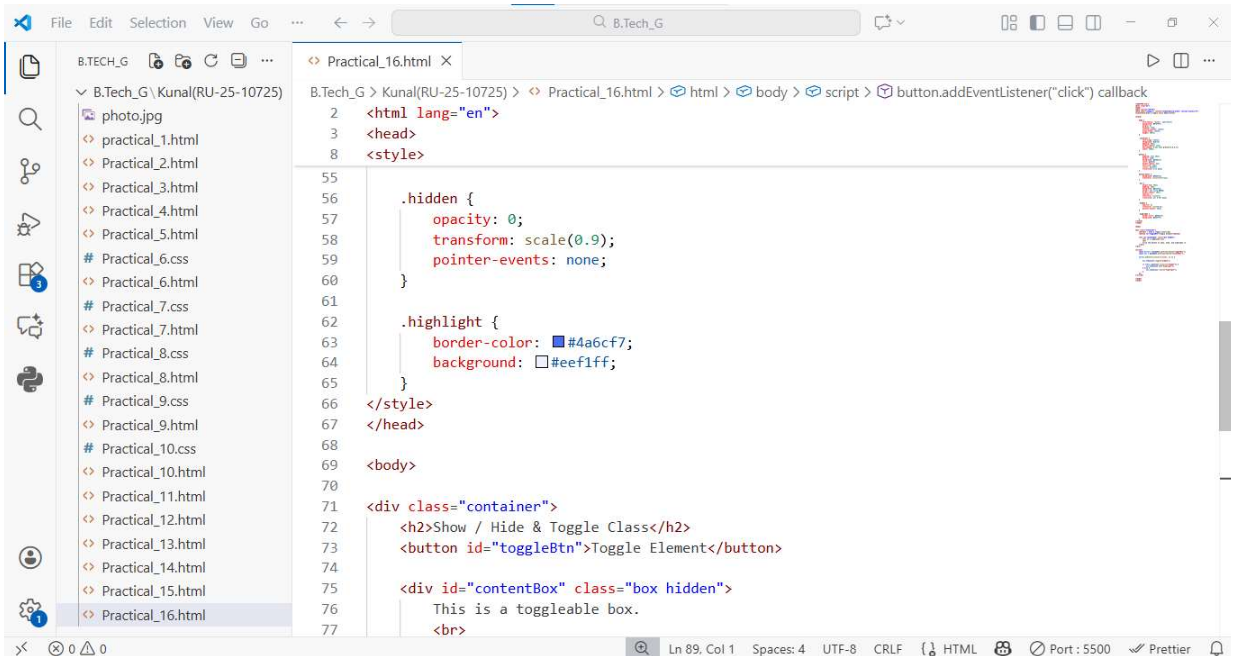
16. Show or hide HTML elements using JavaScript and toggle CSS classes dynamically.



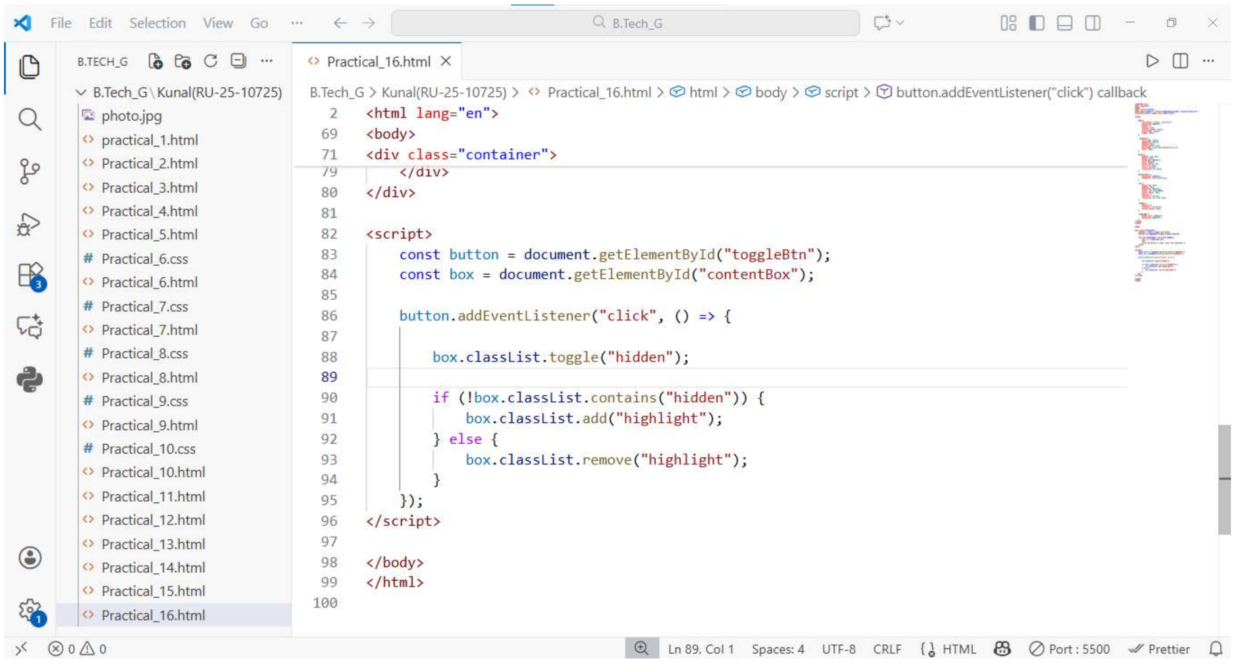
```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Show/Hide & Toggle Class Demo</title>

<style>

    body {
        font-family: "Inter", sans-serif;
        background: #f4f4f7;
        margin: 0;
        display: flex;
        justify-content: center;
        align-items: center;
        height: 100vh;
    }

    .container {
        text-align: center;
        background: white;
        padding: 40px;
        border-radius: 12px;
        box-shadow: 0 8px 24px rgba(0,0,0,0.1);
        width: 360px;
```


File Edit Selection View Go
B.Tech_G
Practical_16.html
B.Tech_G > Kunal(RU-25-10725) > Practical_16.html > html > body > script > button.addEventListener("click") callback
Ln 89, Col 1 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

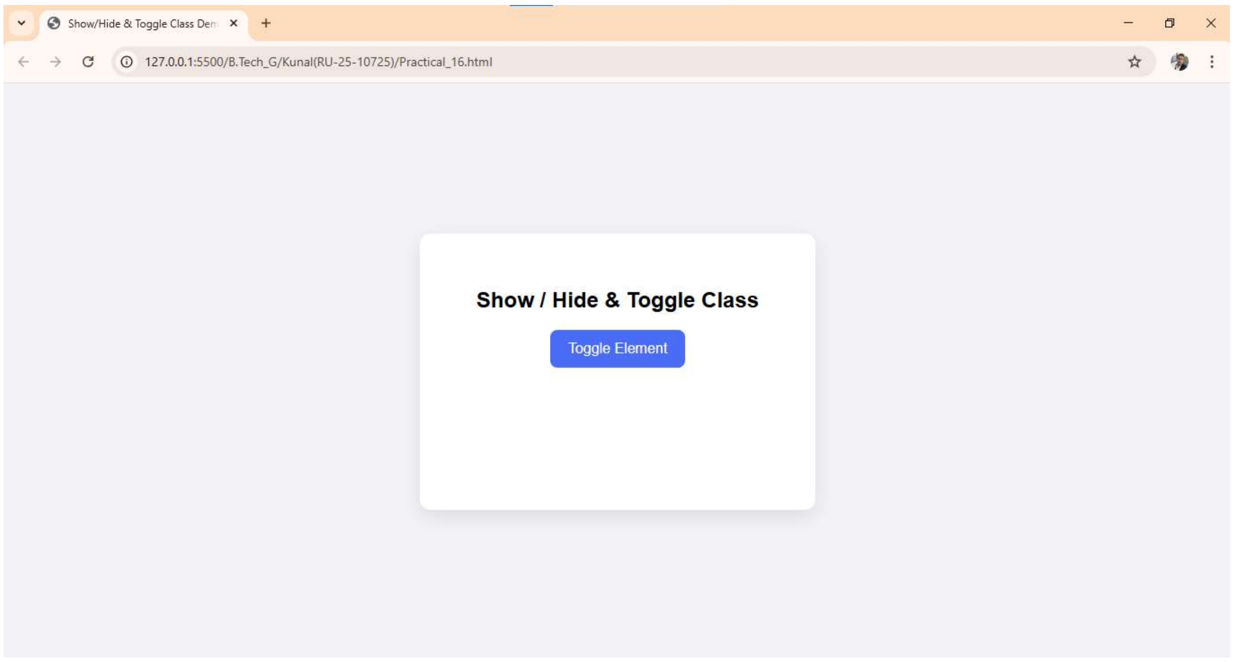
```html
<html lang="en">
<head>
<style>
    button {
        padding: 12px 20px;
        border: none;
        background: #4a6cf7;
        color: white;
        border-radius: 8px;
        font-size: 16px;
        cursor: pointer;
        transition: 0.3s ease;
    }

    button:hover {
        background: #3b57c5;
        transform: translateY(-2px);
    }

    .box {
        margin-top: 20px;
        padding: 20px;
        background: #fafafa;
        border: 2px solid #ddd;
        border-radius: 10px;
```

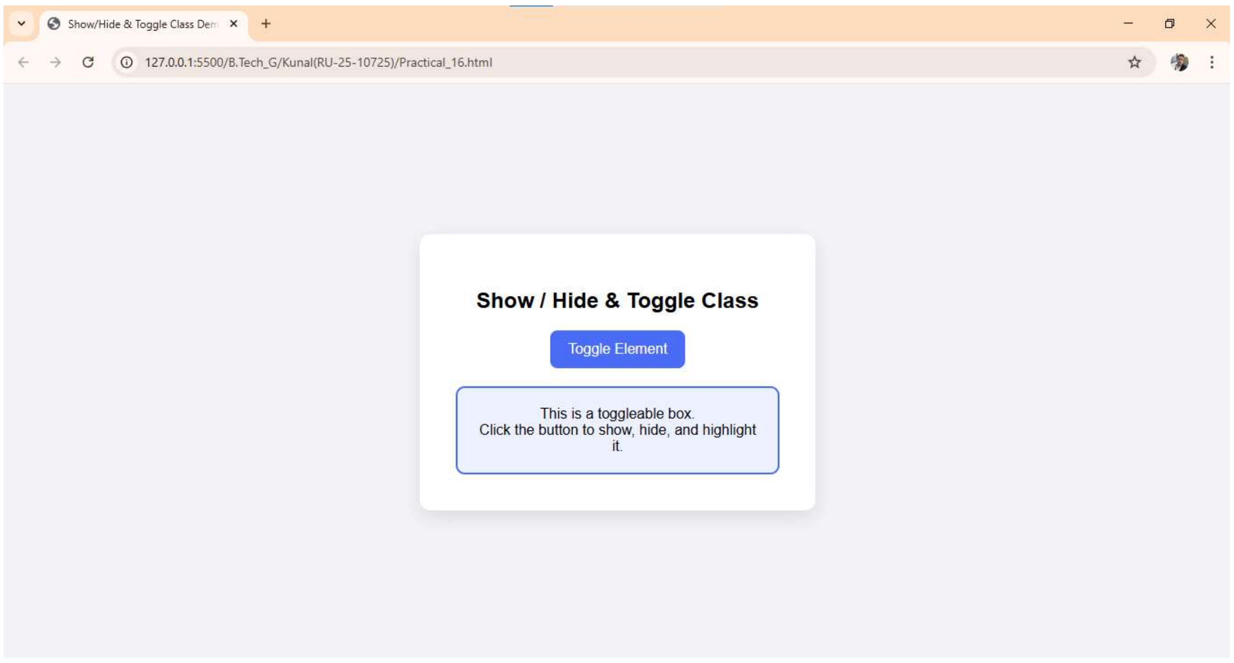


```
<html lang="en">
<head>
<style>

    .hidden {
        opacity: 0;
        transform: scale(0.9);
        pointer-events: none;
    }

    .highlight {
        border-color: #4a6cf7;
        background: #eef1ff;
    }
</style>
</head>

<body>

<div class="container">
    <h2>Show / Hide & Toggle Class</h2>
    <button id="toggleBtn">Toggle Element</button>

    <div id="contentBox" class="box hidden">
        This is a toggleable box.
        <br>
```



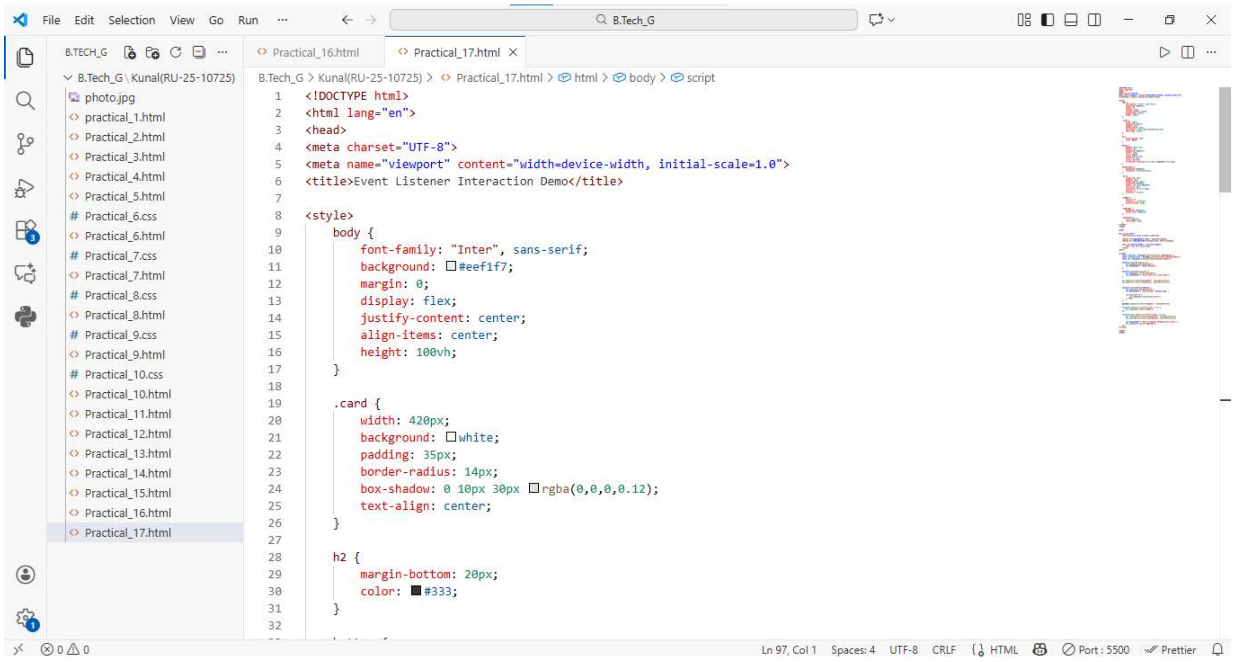
```
<html lang="en">
<body>
<div class="container">
    </div>
</div>

<script>
    const button = document.getElementById("toggleBtn");
    const box = document.getElementById("contentBox");

    button.addEventListener("click", () => {

        box.classList.toggle("hidden");

        if (!box.classList.contains("hidden")) {
            box.classList.add("highlight");
        } else {
            box.classList.remove("highlight");
        }
    });
</script>

</body>
</html>
```


Show/Hide & Toggle Class Dem
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_16.html
Show / Hide & Toggle Class
Toggle Element

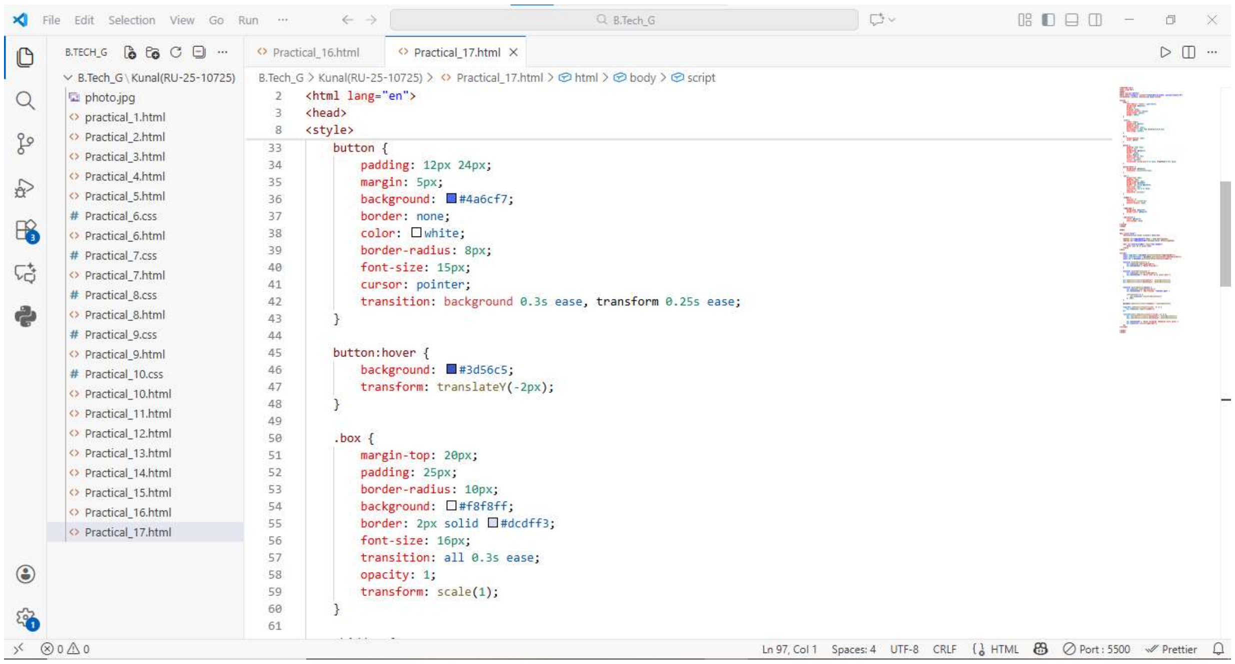

Show/Hide & Toggle Class Dem
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_16.html
Show / Hide & Toggle Class
Toggle Element

This is a toggleable box.
Click the button to show, hide, and highlight it.

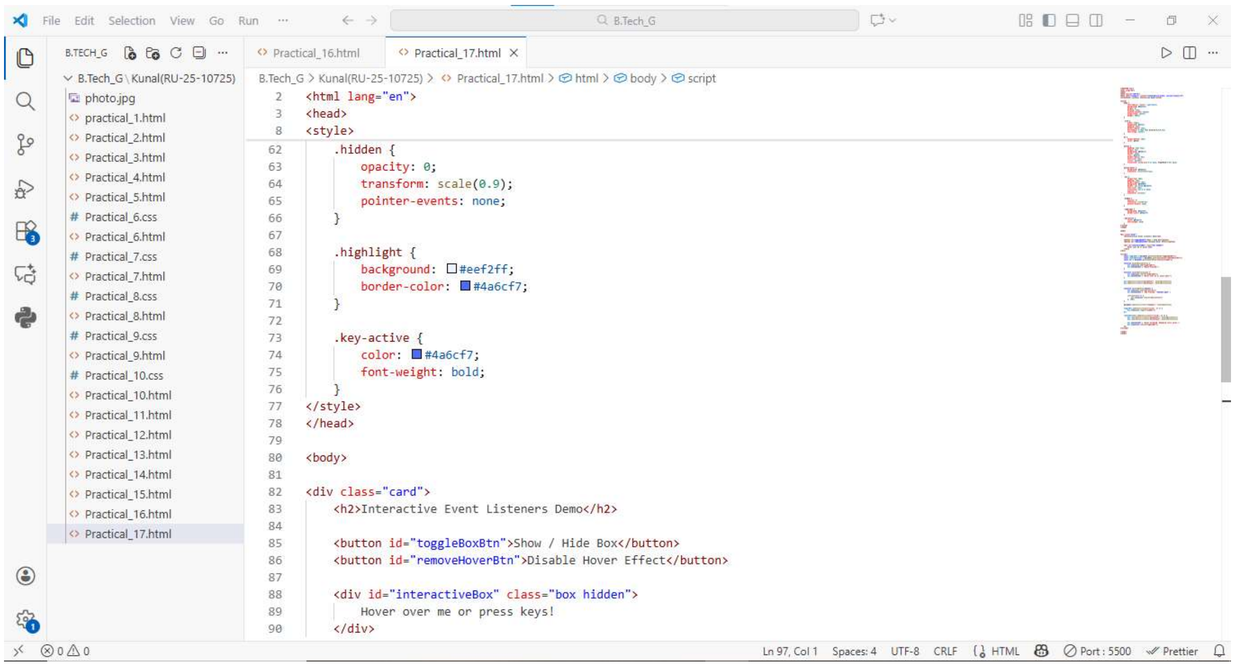
17. Add interactivity using event listeners (addEventListener, removeEventListener) for mouse/keyboard events.



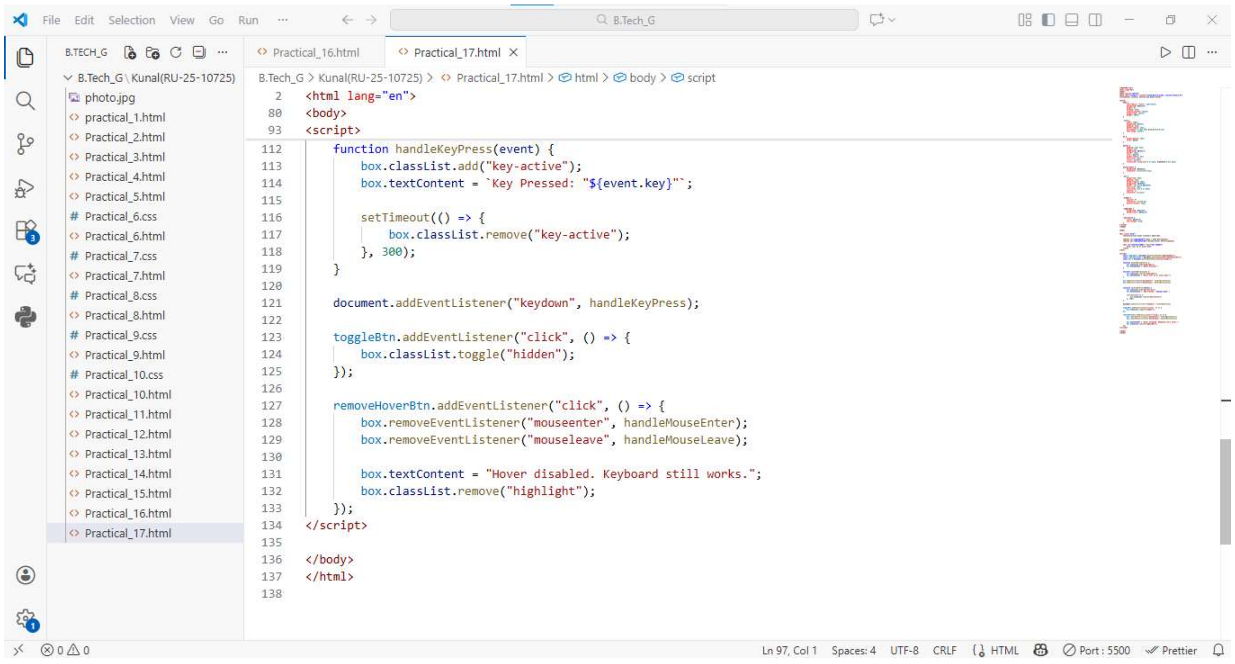
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Event Listener Interaction Demo</title>

<style>
    body {
        font-family: "Inter", sans-serif;
        background: #eef1f7;
        margin: 0;
        display: flex;
        justify-content: center;
        align-items: center;
        height: 100vh;
    }

    .card {
        width: 420px;
        background: white;
        padding: 35px;
        border-radius: 14px;
        box-shadow: 0 10px 30px rgba(0,0,0,0.12);
        text-align: center;
    }

    h2 {
        margin-bottom: 20px;
        color: #333;
    }

```



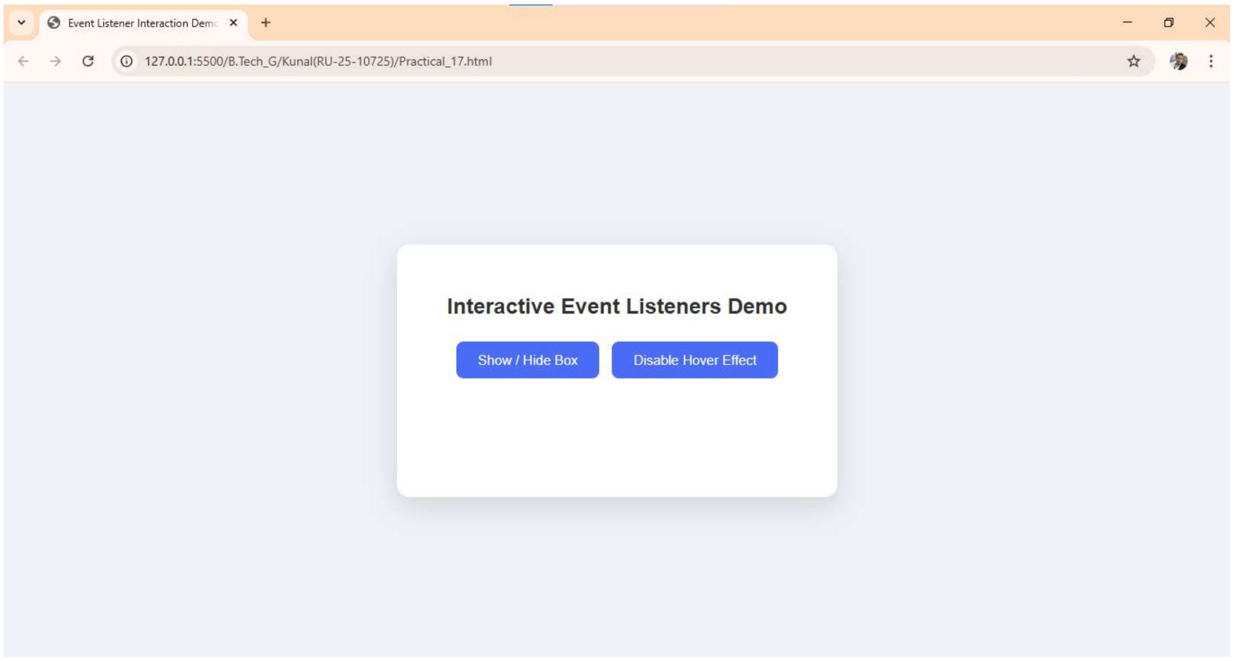
```
<html lang="en">
<head>
<style>
    button {
        padding: 12px 24px;
        margin: 5px;
        background: #4a6cf7;
        border: none;
        color: white;
        border-radius: 8px;
        font-size: 15px;
        cursor: pointer;
        transition: background 0.3s ease, transform 0.25s ease;
    }

    button:hover {
        background: #3d56c5;
        transform: translateY(-2px);
    }

    .box {
        margin-top: 20px;
        padding: 25px;
        border-radius: 10px;
        background: #f8f8ff;
        border: 2px solid #dcdff3;
        font-size: 16px;
        transition: all 0.3s ease;
        opacity: 1;
        transform: scale(1);
    }

```

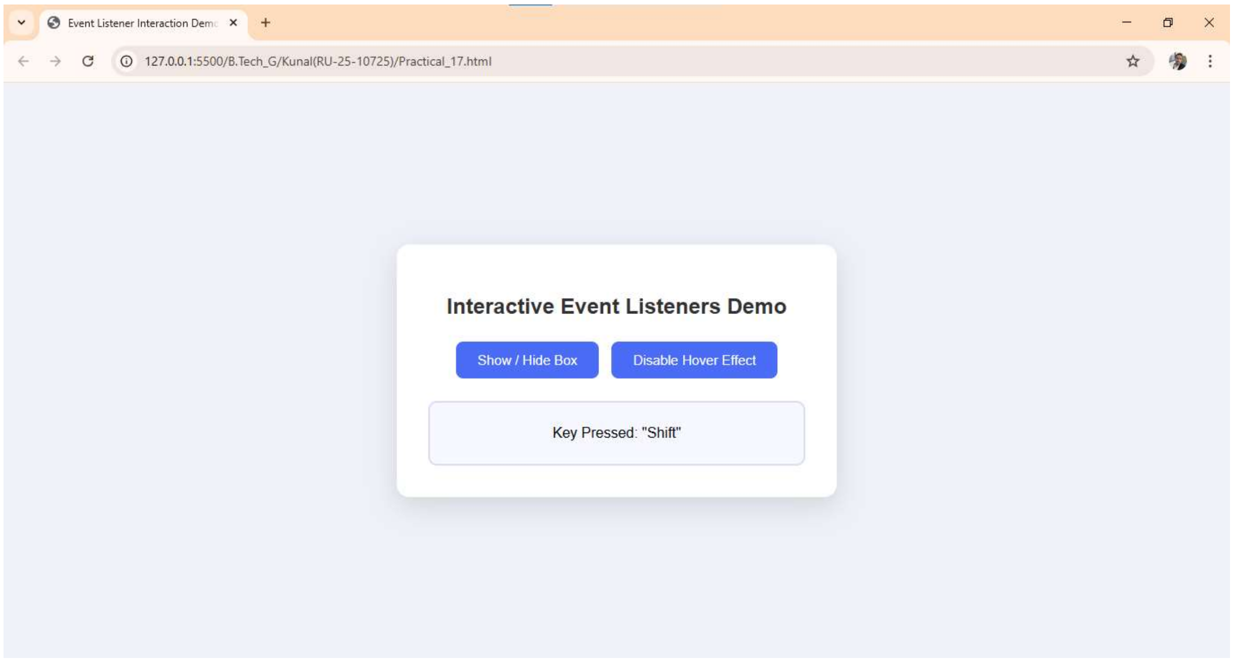


```
<html lang="en">
<head>
<style>
    .hidden {
        opacity: 0;
        transform: scale(0.9);
        pointer-events: none;
    }

    .highlight {
        background: #eef2ff;
        border-color: #4a6cf7;
    }

    .key-active {
        color: #4a6cf7;
        font-weight: bold;
    }
</style>
</head>

<body>

<div class="card">
    <h2>Interactive Event Listeners Demo</h2>

    <button id="toggleBoxBtn">Show / Hide Box</button>
    <button id="removeHoverBtn">Disable Hover Effect</button>

    <div id="interactiveBox" class="box hidden">
        Hover over me or press keys!
    </div>
```


File Edit Selection View Go Run
B.Tech_G
B.TECH_G
Practical_16.html
Practical_17.html
B.Tech_G \ Kunal(RU-25-10725)
B.Tech_G > Kunal(RU-25-10725) > Practical_17.html > html > body > script
Ln 97, Col 1 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

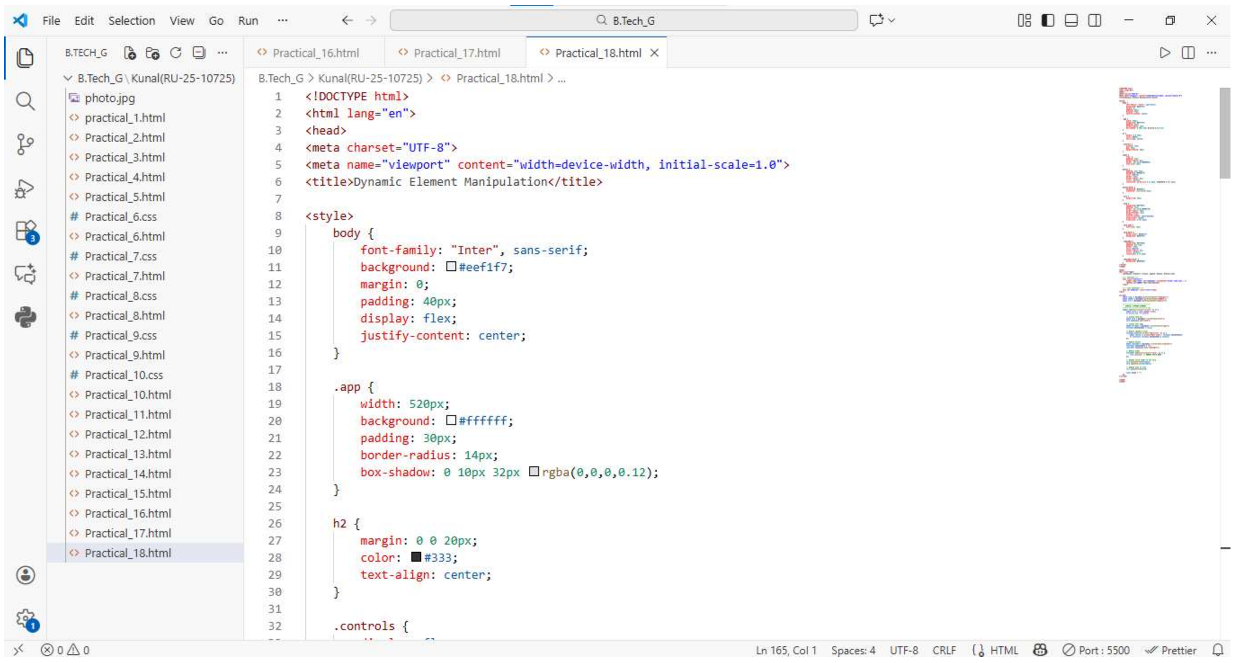
```
<html lang="en">
<body>
<script>
    function handleKeyPress(event) {
        box.classList.add("key-active");
        box.textContent = `Key Pressed: "${event.key}"`;

        setTimeout(() => {
            box.classList.remove("key-active");
        }, 300);
    }

    document.addEventListener("keydown", handleKeyPress);

    toggleBtn.addEventListener("click", () => {
        box.classList.toggle("hidden");
    });

    removeHoverBtn.addEventListener("click", () => {
        box.removeEventListener("mouseenter", handleMouseEnter);
        box.removeEventListener("mouseleave", handleMouseLeave);

        box.textContent = "Hover disabled. Keyboard still works.";
        box.classList.remove("highlight");
    });
</script>

</body>
</html>
```


Event Listener Interaction Demo
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_17.html
Interactive Event Listeners Demo
Show / Hide Box
Disable Hover Effect

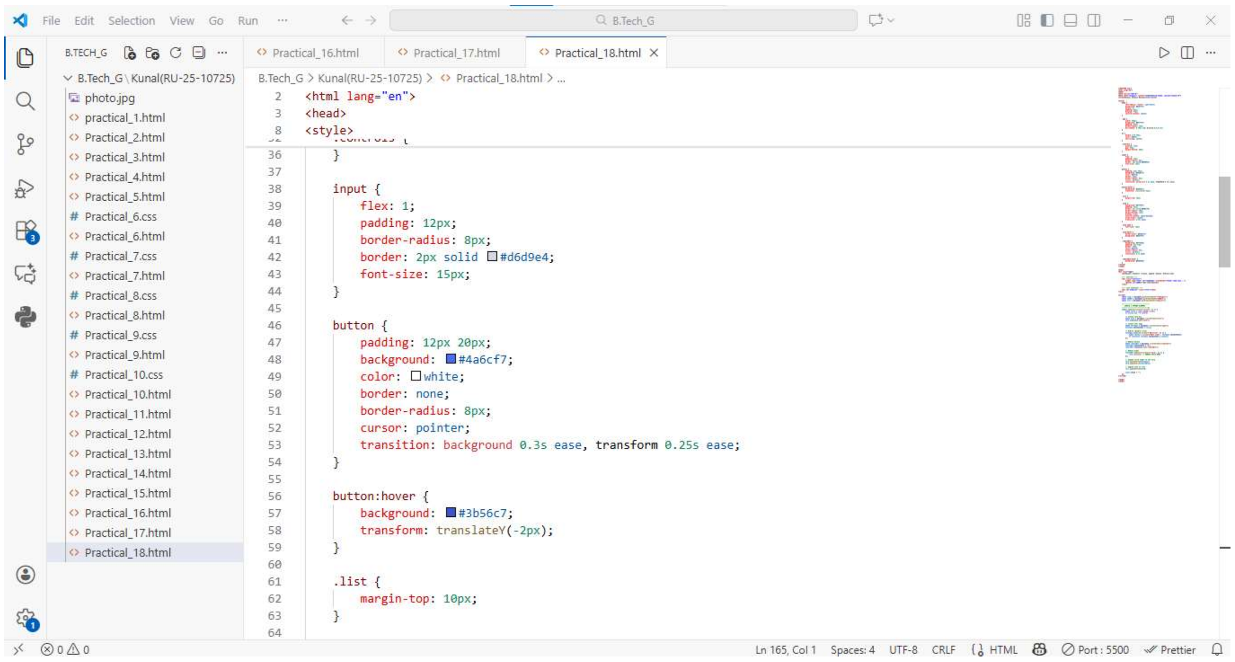

Event Listener Interaction Demo
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_17.html
Interactive Event Listeners Demo
Show / Hide Box
Disable Hover Effect

Key Pressed: "Shift"

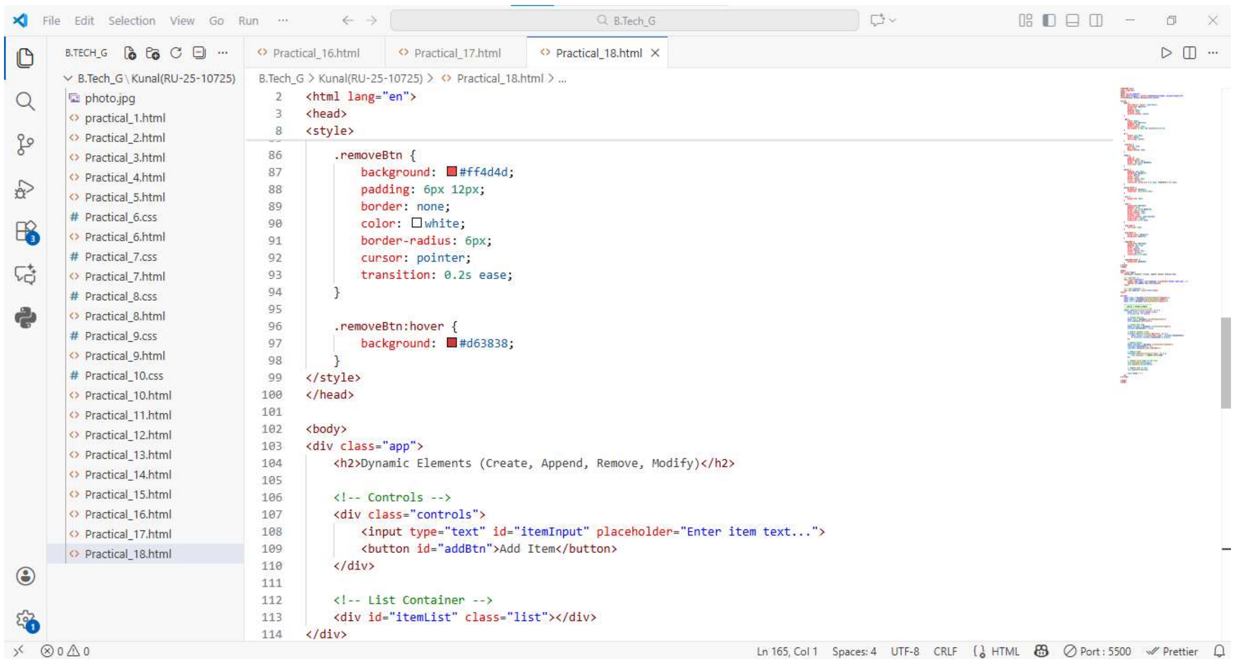
18. Create and manipulate elements using JavaScript: append, remove, or modify child nodes.



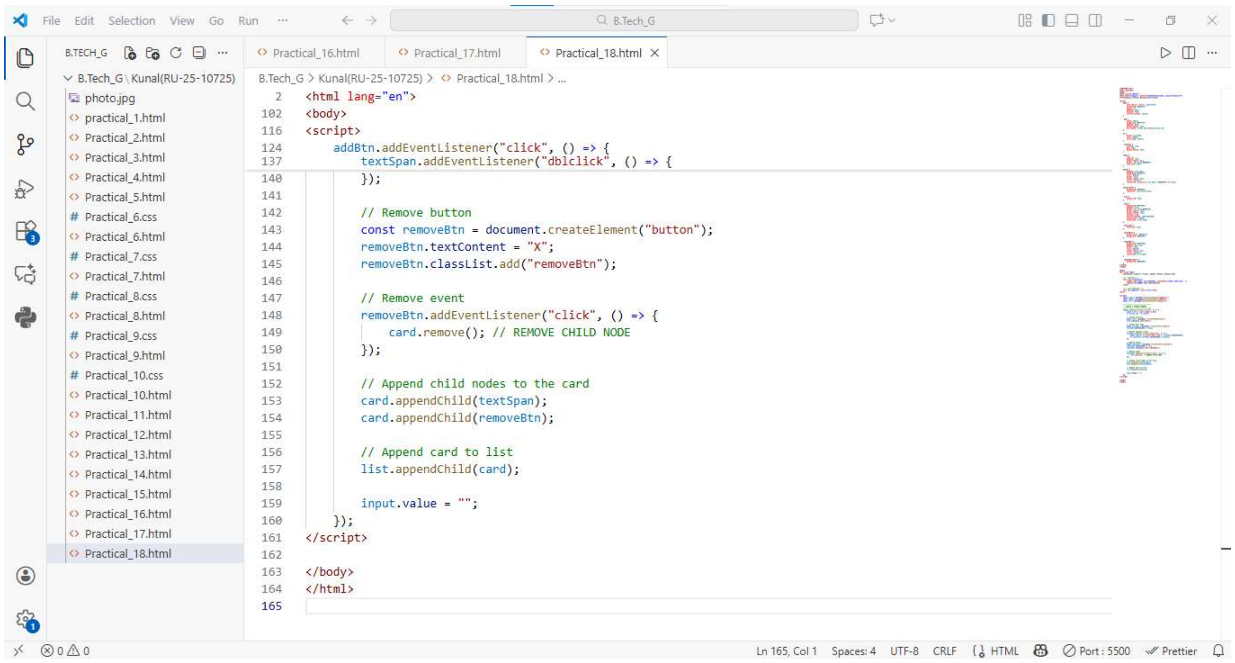
```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Dynamic Element Manipulation</title>

<style>
    body {
        font-family: "Inter", sans-serif;
        background: #eef1f7;
        margin: 0;
        padding: 40px;
        display: flex;
        justify-content: center;
    }

    .app {
        width: 520px;
        background: #ffffff;
        padding: 30px;
        border-radius: 14px;
        box-shadow: 0 10px 32px rgba(0,0,0,0.12);
    }

    h2 {
        margin: 0 0 20px;
        color: #333;
        text-align: center;
    }

    .controls {
```



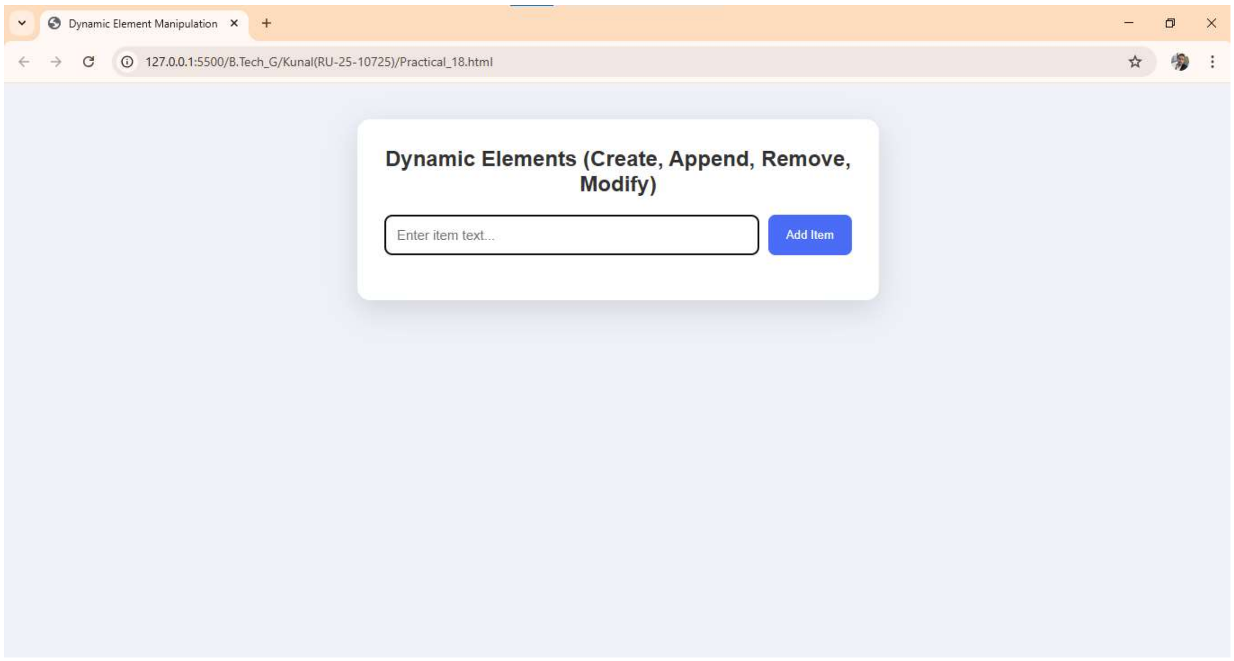
```html
<html lang="en">
<head>
<style>
    }

    input {
        flex: 1;
        padding: 12px;
        border-radius: 8px;
        border: 2px solid #d6d9e4;
        font-size: 15px;
    }

    button {
        padding: 12px 20px;
        background: #4a6cf7;
        color: white;
        border: none;
        border-radius: 8px;
        cursor: pointer;
        transition: background 0.3s ease, transform 0.25s ease;
    }

    button:hover {
        background: #3b56c7;
        transform: translateY(-2px);
    }

    .list {
        margin-top: 10px;
    }

```

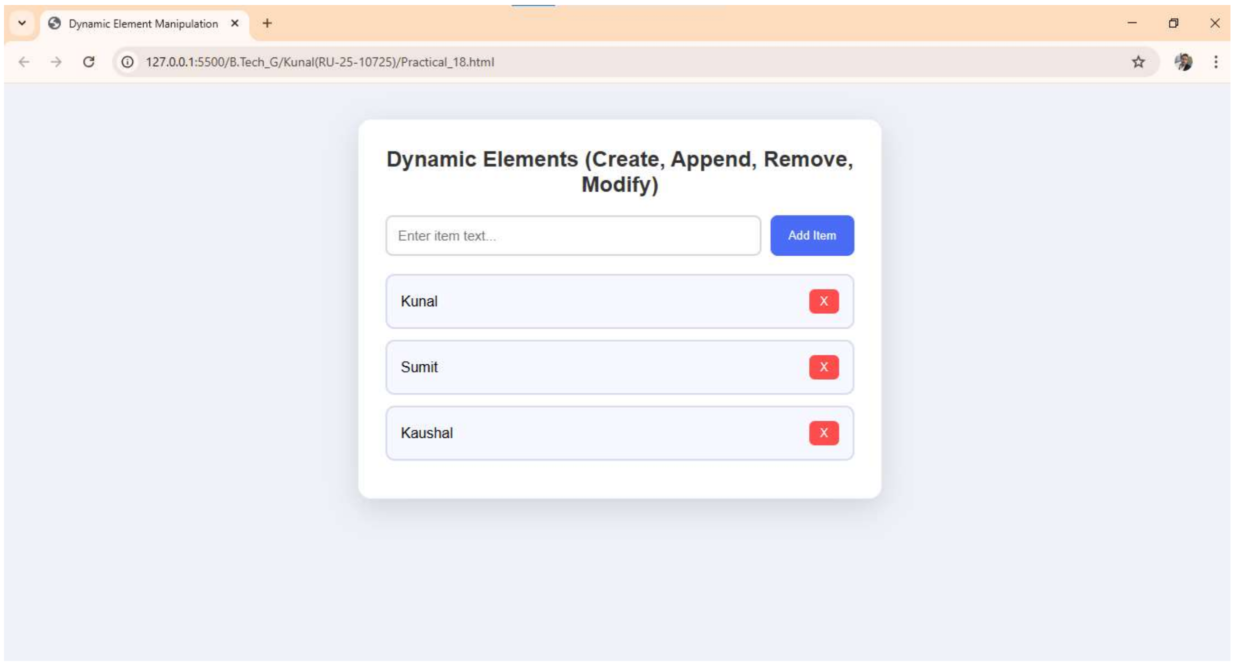

File Edit Selection View Go Run
B.Tech_G
Practical_16.html
Practical_17.html
Practical_18.html
B.Tech_G > Kunal(RU-25-10725) > Practical_18.html > ...
Ln 165, Col 1 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

```
<html lang="en">
<head>
<style>
    .removeBtn {
        background: #ff4d4d;
        padding: 6px 12px;
        border: none;
        color: white;
        border-radius: 6px;
        cursor: pointer;
        transition: 0.2s ease;
    }

    .removeBtn:hover {
        background: #d63838;
    }
</style>
</head>

<body>
<div class="app">
    <h2>Dynamic Elements (Create, Append, Remove, Modify)</h2>

    <!-- Controls -->
    <div class="controls">
        <input type="text" id="itemInput" placeholder="Enter item text...">
        <button id="addBtn">Add Item</button>
    </div>

    <!-- List Container -->
    <div id="itemList" class="list"></div>
</div>
```

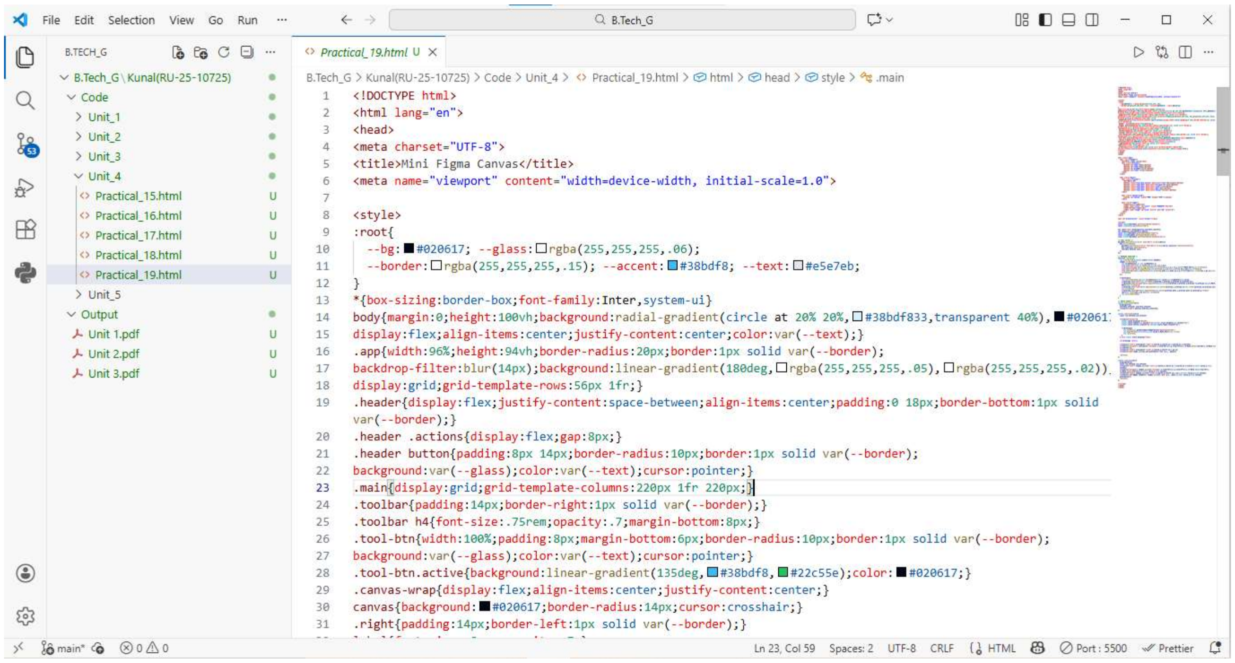

File Edit Selection View Go Run
B.Tech_G
Practical_16.html
Practical_17.html
Practical_18.html
B.Tech_G > Kunal(RU-25-10725) > Practical_18.html > ...
Ln 165, Col 1 Spaces: 4 UTF-8 CRLF HTML Port : 5500 Prettier

```
<html lang="en">
<body>
<script>
    addBtn.addEventListener("click", () => {
        textSpan.addEventListener("dblclick", () => {
        });

        // Remove button
        const removeBtn = document.createElement("button");
        removeBtn.textContent = "X";
        removeBtn.classList.add("removeBtn");

        // Remove event
        removeBtn.addEventListener("click", () => {
            card.remove(); // REMOVE CHILD NODE
        });

        // Append child nodes to the card
        card.appendChild(textSpan);
        card.appendChild(removeBtn);

        // Append card to list
        list.appendChild(card);

        input.value = "";
    });
</script>

</body>
</html>
```

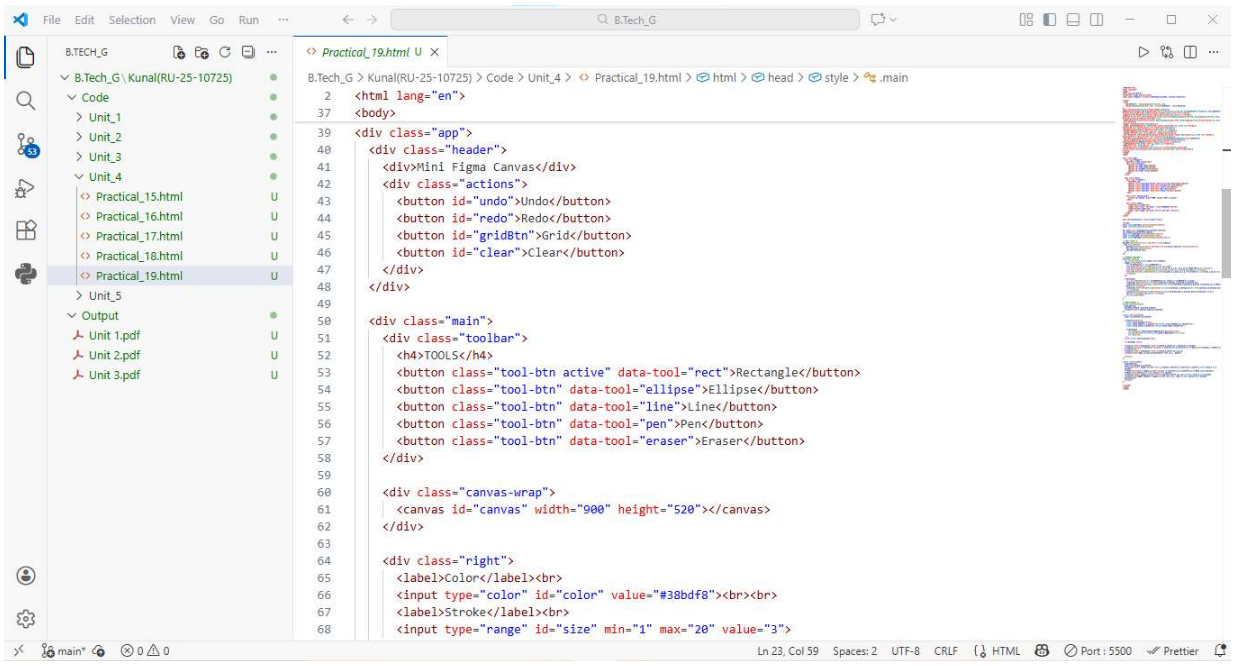

Dynamic Element Manipulation
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_18.html
Dynamic Elements (Create, Append, Remove, Modify)
Enter item text...
Add Item


Dynamic Element Manipulation
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_18.html
Dynamic Elements (Create, Append, Remove, Modify)
Enter item text...
Add Item
Kunal
X
Sumit
X
Kaushal
X

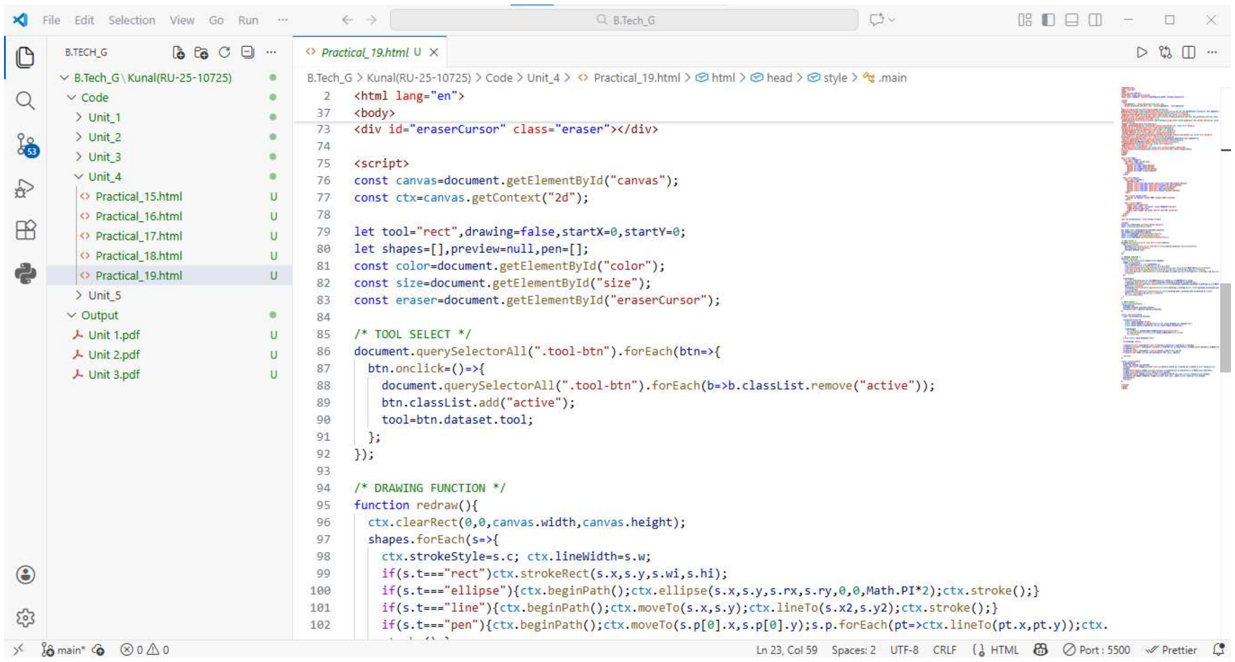
19. Create a simple drawing using the element: draw shapes and fill colors.



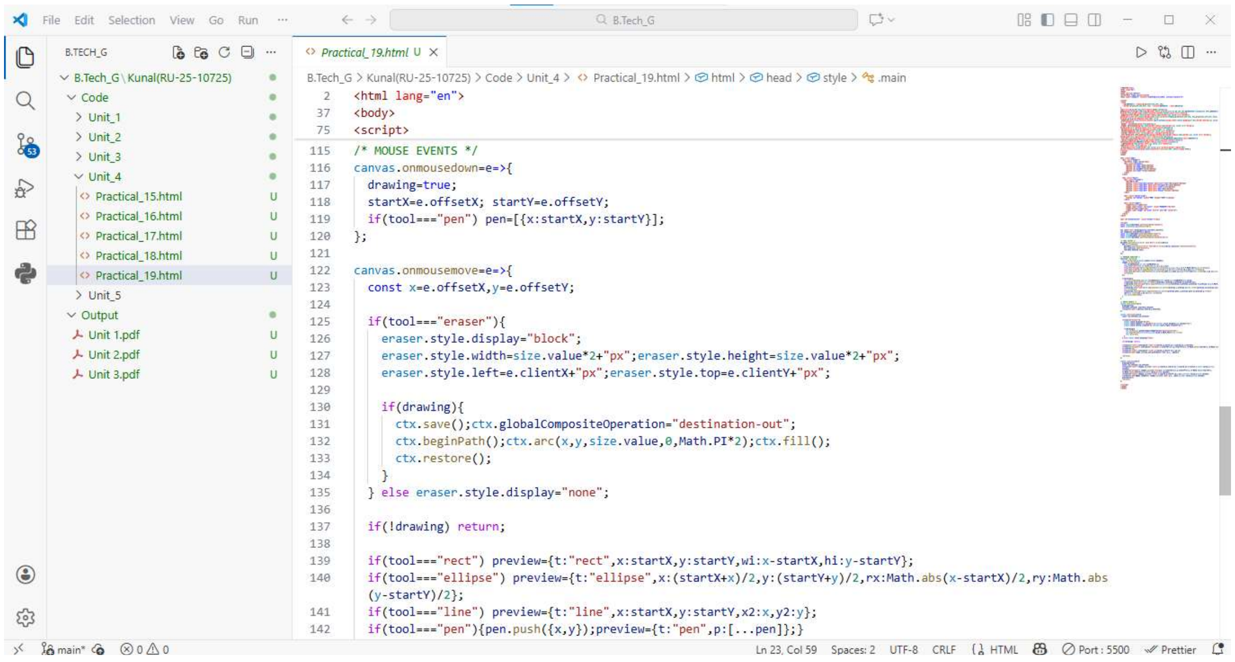
```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Mini Figma Canvas</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<style>
:root{
  --bg:#020617; --glass:rgba(255,255,255,.06);
  --border:rgba(255,255,255,.15); --accent:#38bdf8; --text:#e5e7eb;
}
*{box-sizing:border-box;font-family:Inter,system-ui}
body{margin:0;height:100vh;background:radial-gradient(circle at 20% 20%,#38bdf833,transparent 40%),#02061
display:flex;align-items:center;justify-content:center;color:var(--text);}
.app{width:96%;height:94vh;border-radius:20px;border:1px solid var(--border);
backdrop-filter:blur(14px);background:linear-gradient(180deg,rgba(255,255,255,.05),rgba(255,255,255,.02))
display:grid;grid-template-rows:56px 1fr;}
.header{display:flex;justify-content:space-between;align-items:center;padding:0 18px;border-bottom:1px solid
var(--border);}
.header .actions{display:flex;gap:8px;}
.header button{padding:8px 14px;border-radius:10px;border:1px solid var(--border);
background:var(--glass);color:var(--text);cursor:pointer;}
.main{display:grid;grid-template-columns:220px 1fr 220px;}
.toolbar{padding:14px;border-right:1px solid var(--border);}
.toolbar h4{font-size:.75rem;opacity:.7;margin-bottom:8px;}
.tool-btn{width:100%;padding:8px;margin-bottom:6px;border-radius:10px;border:1px solid var(--border);
background:var(--glass);color:var(--text);cursor:pointer;}
.tool-btn.active{background:linear-gradient(135deg,#38bdf8,#22c55e);color:#020617;}
.canvas-wrap{display:flex;align-items:center;justify-content:center;}
canvas{background:#020617;border-radius:14px;cursor:crosshair;}
.right{padding:14px;border-left:1px solid var(--border);}
```



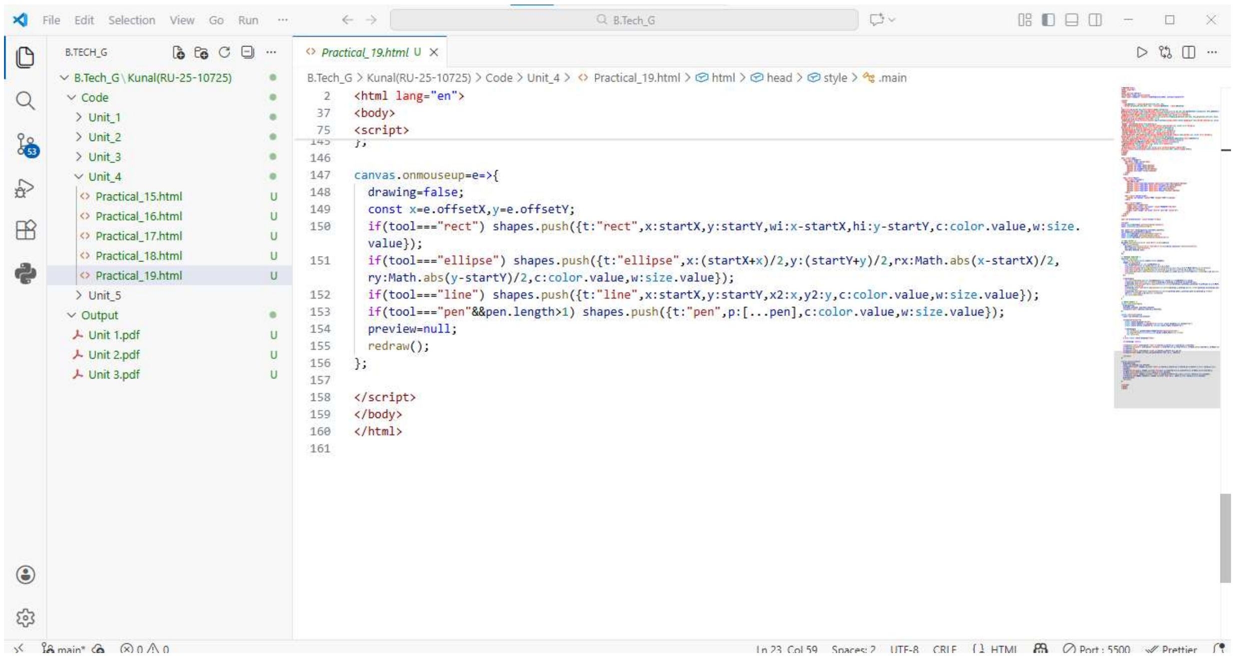
```html
<html lang="en">
<body>
<div class="app">
  <div class="header">
    <div>Mini Figma Canvas</div>
    <div class="actions">
      <button id="undo">Undo</button>
      <button id="redo">Redo</button>
      <button id="gridBtn">Grid</button>
      <button id="clear">Clear</button>
    </div>
  </div>

  <div class="main">
    <div class="toolbar">
      <h4>TOOLS</h4>
      <button class="tool-btn active" data-tool="rect">Rectangle</button>
      <button class="tool-btn" data-tool="ellipse">Ellipse</button>
      <button class="tool-btn" data-tool="line">Line</button>
      <button class="tool-btn" data-tool="pen">Pen</button>
      <button class="tool-btn" data-tool="eraser">Eraser</button>
    </div>

    <div class="canvas-wrap">
      <canvas id="canvas" width="900" height="520"></canvas>
    </div>

    <div class="right">
      <label>Color</label><br>
      <input type="color" id="color" value="#38bdf8"><br><br>
      <label>Stroke</label><br>
      <input type="range" id="size" min="1" max="20" value="3">
```



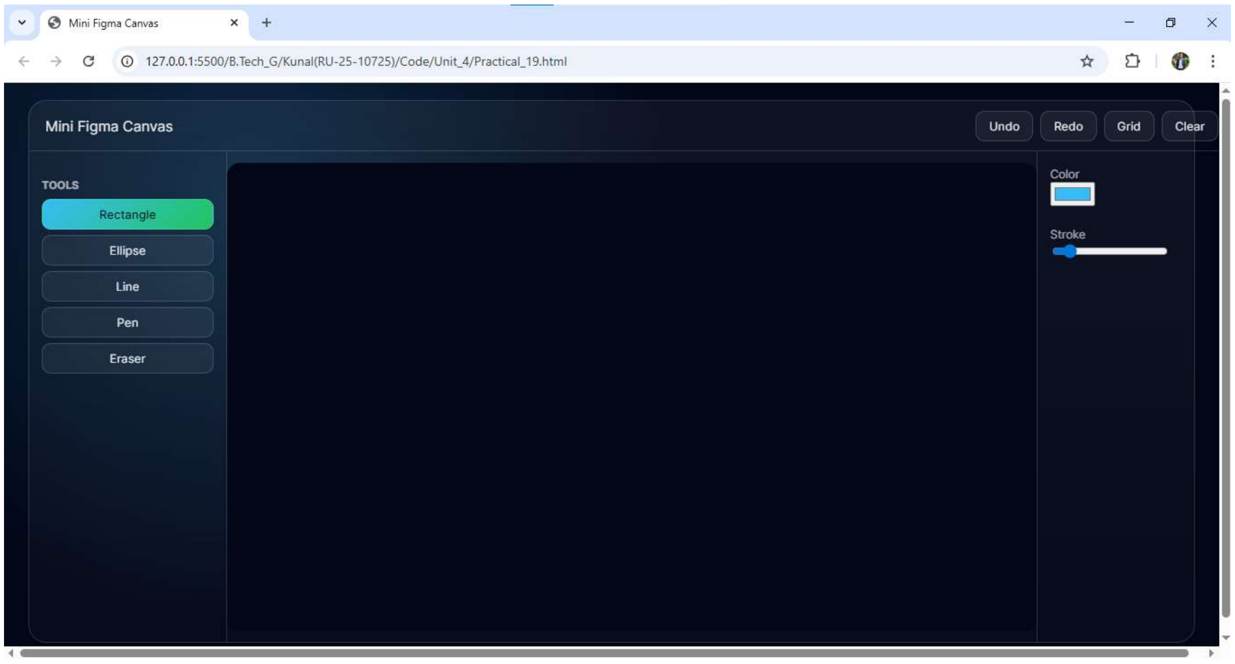
```
<html lang="en">
<body>
<div id="eraserCursor" class="eraser"></div>

<script>
const canvas=document.getElementById("canvas");
const ctx=canvas.getContext("2d");

let tool="rect",drawing=false,startX=0,startY=0;
let shapes=[],preview=null,pen=[];
const color=document.getElementById("color");
const size=document.getElementById("size");
const eraser=document.getElementById("eraserCursor");

/* TOOL SELECT */
document.querySelectorAll(".tool-btn").forEach(btn=>{
  btn.onclick=()=>{
    document.querySelectorAll(".tool-btn").forEach(b=>b.classList.remove("active"));
    btn.classList.add("active");
    tool=btn.dataset.tool;
  };
});

/* DRAWING FUNCTION */
function redraw(){
  ctx.clearRect(0,0,canvas.width,canvas.height);
  shapes.forEach(s=>{
    ctx.strokeStyle=s.c; ctx.lineWidth=s.w;
    if(s.t==="rect")ctx.strokeRect(s.x,s.y,s.wi,s.hi);
    if(s.t==="ellipse"){ctx.beginPath();ctx.ellipse(s.x,s.y,s.rx,s.ry,0,0,Math.PI*2);ctx.stroke();}
    if(s.t==="line"){ctx.beginPath();ctx.moveTo(s.x,s.y);ctx.lineTo(s.x2,s.y2);ctx.stroke();}
    if(s.t==="pen"){ctx.beginPath();ctx.moveTo(s.p[0].x,s.p[0].y);s.p.forEach(pt=>ctx.lineTo(pt.x,pt.y));ctx.
```



```
<html lang="en">
<body>
<script>
/* MOUSE EVENTS */
canvas.onmousedown=e=>{
  drawing=true;
  startX=e.offsetX; startY=e.offsetY;
  if(tool==="pen") pen=[{x:startX,y:startY}];
};

canvas.onmousemove=e=>{
  const x=e.offsetX,y=e.offsetY;

  if(tool==="eraser"){
    eraser.style.display="block";
    eraser.style.width=size.value*2+"px";eraser.style.height=size.value*2+"px";
    eraser.style.left=e.clientX+"px";eraser.style.top=e.clientY+"px";

    if(drawing){
      ctx.save();ctx.globalCompositeOperation="destination-out";
      ctx.beginPath();ctx.arc(x,y,size.value,0,Math.PI*2);ctx.fill();
      ctx.restore();
    }
  } else eraser.style.display="none";

  if(!drawing) return;

  if(tool==="rect") preview={t:"rect",x:startX,y:startY,wi:x-startX,hi:y-startY};
  if(tool==="ellipse") preview={t:"ellipse",x:(startX+x)/2,y:(startY+y)/2,rx:Math.abs(x-startX)/2,ry:Math.abs
  (y-startY)/2};
  if(tool==="line") preview={t:"line",x:startX,y:startY,x2:x,y2:y};
  if(tool==="pen"){pen.push({x,y});preview={t:"pen",p:[...pen]};}
```



```
<html lang="en">
<body>
<script>
};

canvas.onmouseup=e=>{
  drawing=false;
  const x=e.offsetX,y=e.offsetY;
  if(tool==="rect") shapes.push({t:"rect",x:startX,y:startY,wi:x-startX,hi:y-startY,c:color.value,w:size.
  value});
  if(tool==="ellipse") shapes.push({t:"ellipse",x:(startX+x)/2,y:(startY+y)/2,rx:Math.abs(x-startX)/2,
  ry:Math.abs(y-startY)/2,c:color.value,w:size.value});
  if(tool==="line") shapes.push({t:"line",x:startX,y:startY,x2:x,y2:y,c:color.value,w:size.value});
  if(tool==="pen"&&pen.length>1) shapes.push({t:"pen",p:[...pen],c:color.value,w:size.value});
  preview=null;
  redraw();
};

</script>
</body>
</html>
```


Mini Figma Canvas
127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Code/Unit_4/Practical_19.html
Mini Figma Canvas
Undo
Redo
Grid
Clear
TOOLS
Rectangle
Ellipse
Line
Pen
Eraser
Color
Stroke