

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>
</head>
<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: #3333;
        transition: 0.3s ease;
    }
</style>
```

```
<html lang="en">
<head>
    <style>
        p {
            font-size: 17px;
            color: #5555;
            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;
        }
    </style>
</head>
```

The screenshot shows a code editor interface with a sidebar containing a file tree. The file tree shows a folder named 'B.TECH_G' with a subfolder 'Kunal(RU-25-10725)' containing various HTML files like 'photo.jpg', 'practical_1.html', etc., and several CSS and JS files. The main editor area displays the content of 'Practical_15.html'. The code includes an HTML structure with a heading, a paragraph, and a button, along with a CSS style block for a class named 'success-box'. A detailed description of the code's purpose is provided in the comments.

```
<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 Manipulations</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>
  </body>
</html>
```

The screenshot shows a code editor interface with a sidebar containing a file tree. The file tree shows a folder named 'B.TECH_G' with a subfolder 'Kunal(RU-25-10725)' containing various HTML files like 'photo.jpg', 'practical_1.html', etc., and several CSS and JS files. The main editor area displays the content of 'Practical_15.html'. The code includes an HTML structure with a heading, a paragraph, and a button, along with a script block containing DOM manipulation logic. The script uses various methods like getElementById, querySelector, and innerHTML to update the page content.

```
<html lang="en">
  <head>
    <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");
    </script>
  <body>
    <h2 id="mainHeading">DOM Manipulation Application</h2>
    <p>
      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.
    </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>
  </body>
</html>
```

The screenshot shows a code editor interface with the following details:

- File Explorer:** On the left, it shows a file tree for "B.TECH_G\Kunal(RU-25-10725)". The "Practical_15.html" file is selected.
- Code Editor:** The main area displays the content of "Practical_15.html".

```
2 <html lang="en">
70 <body>
90 <script>
100 updateBtn.addEventListener("click", () => {
118     // Change Button
119     updateBtn.innerHTML = "Updated ✓";
120     updateBtn.style.backgroundColor = "#28a745";
121
122     // Show success box
123     successBox.style.display = "block";
124
125     // Add a soft glow effect
126     card.style.boxShadow = "0 12px 32px rgba(0, 0, 0, 0.15)";
127
128 });
129 </script>
130
131 </body>
132 </html>
133
```
- Status Bar:** At the bottom, it shows "Ln 133, Col 1 Spaces: 4 UTF-8 CRLF ⚡ HTML 🌐 Port: 5500 ✨ Prettier 🎨".

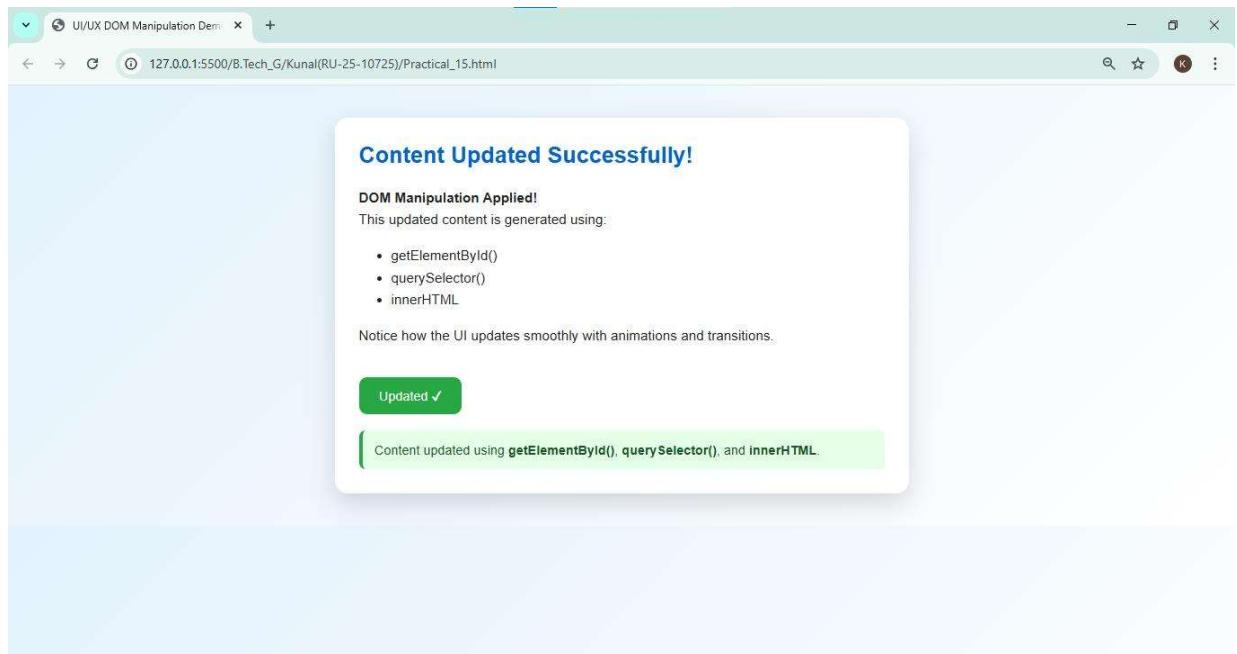
The screenshot shows a web browser window with the following details:

- Title Bar:** "UI/UX DOM Manipulation Demo" and the URL "127.0.0.1:5500/B.Tech_G/Kunal(RU-25-10725)/Practical_15.html".
- Content Area:** A light blue container with rounded corners contains the following text:

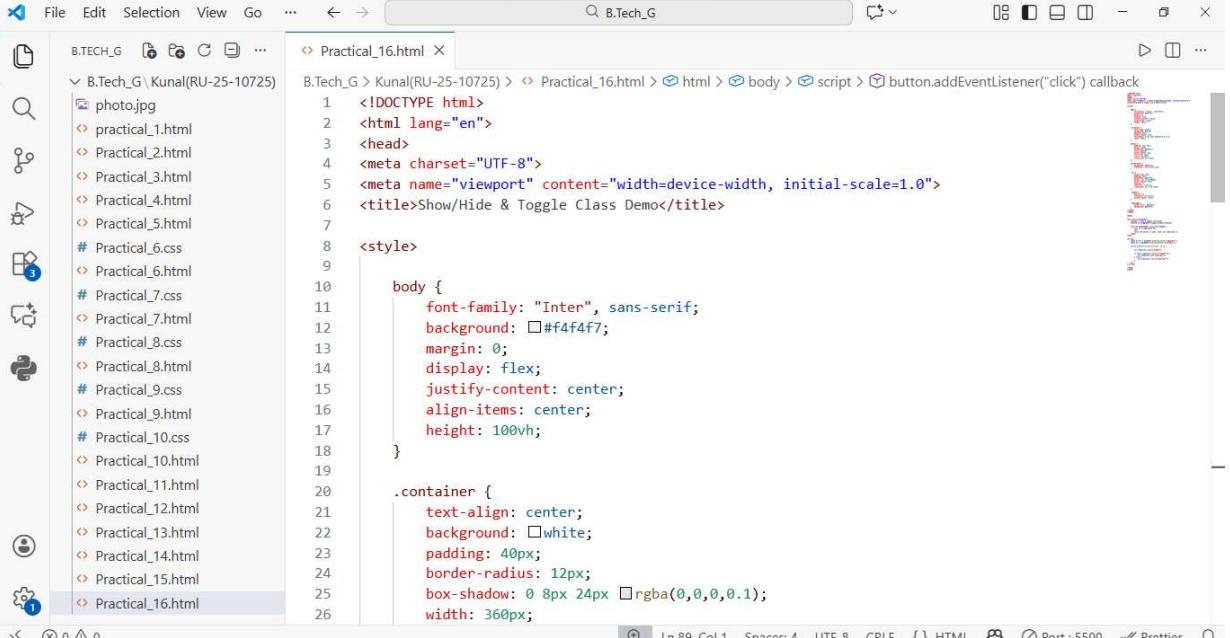
Welcome to DOM Manipulation

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

Update Content

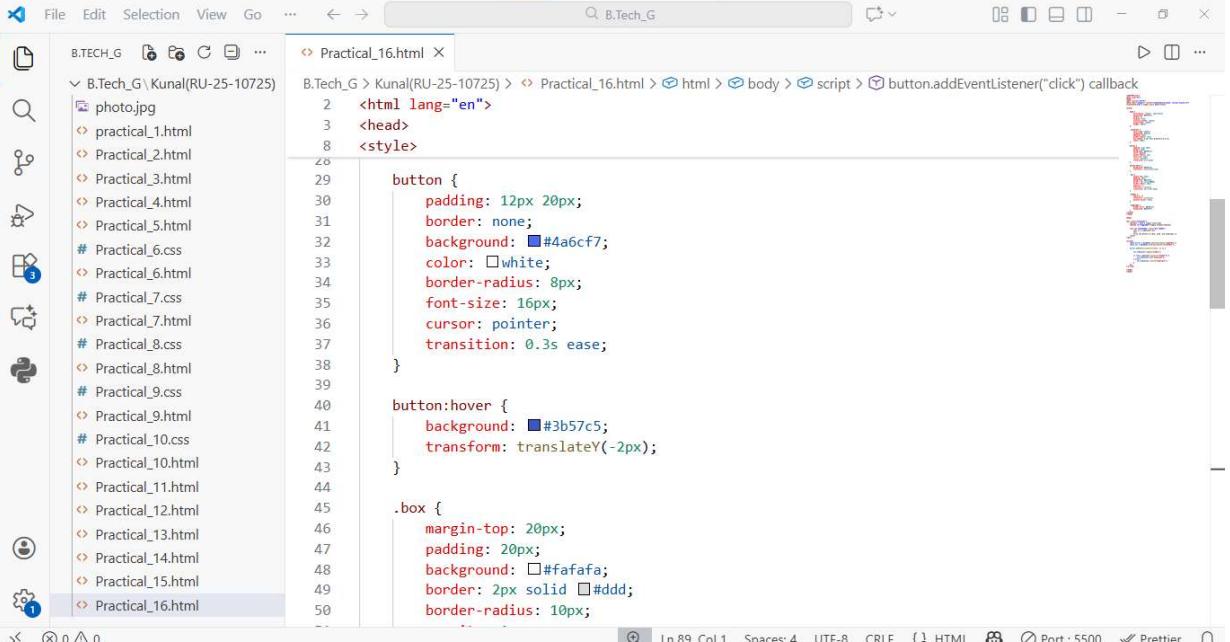


16. Show or hide HTML elements using JavaScript and toggle CSS classes dynamically.



```
<!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;
}
</style>
<script>
button.addEventListener("click", () => {
    const box = document.querySelector(".box");
    if (box.classList.contains("show")) {
        box.classList.remove("show");
    } else {
        box.classList.add("show");
    }
});
</script>
```



```
<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;
}
</style>
<script>
button.addEventListener("click", () => {
    const box = document.querySelector(".box");
    if (box.classList.contains("show")) {
        box.classList.remove("show");
    } else {
        box.classList.add("show");
    }
});
</script>
```

The screenshot shows a code editor interface with a sidebar containing file navigation. The main area displays the content of `Practical_16.html`. The code includes a CSS block defining styles for `.hidden` and `.highlight` classes, and an HTML block with a button and a toggleable box.

```
<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>

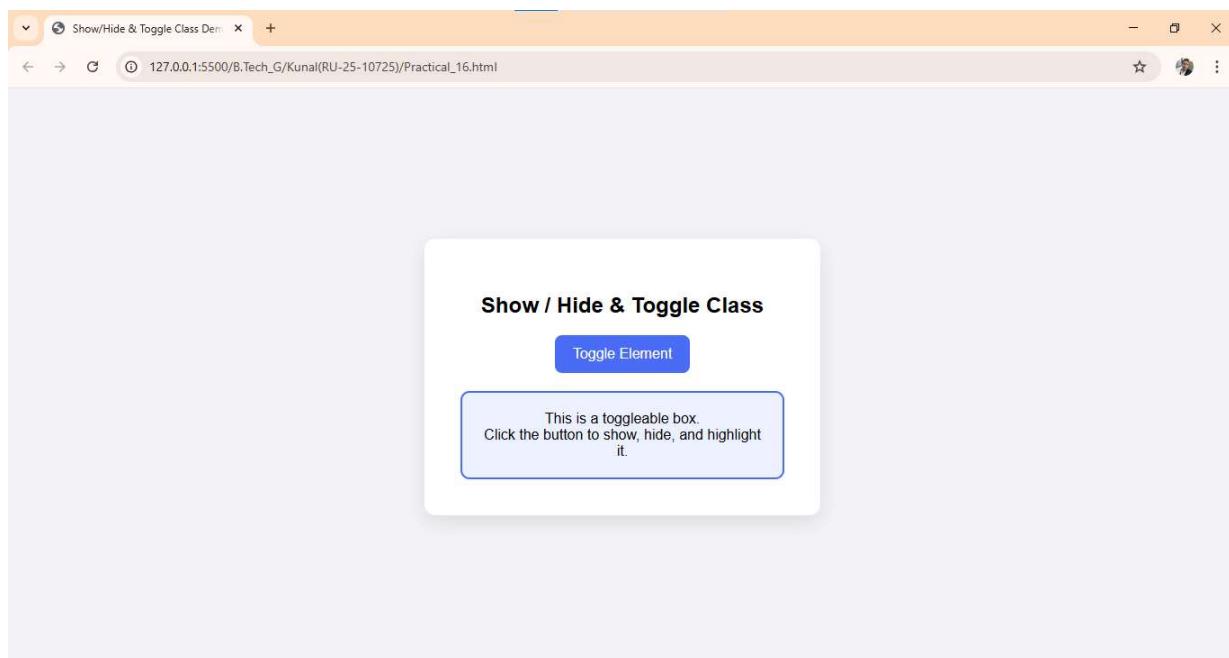
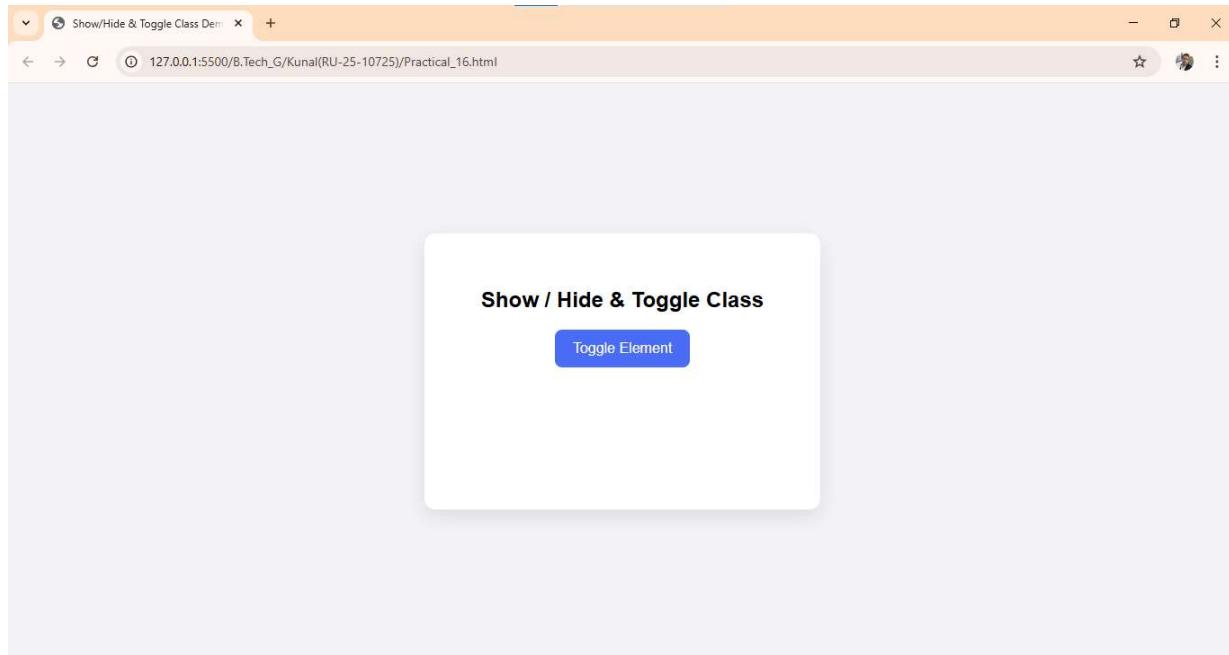
<div id="contentBox" class="box hidden">
    This is a toggleable box.
    <br>
</div>
</div>
</body>
</html>
```

The screenshot shows a code editor interface with a sidebar containing file navigation. The main area displays the content of `Practical_16.html`. The code includes a script block that adds an event listener to the button, which then toggles the `hidden` class on the `box` element.

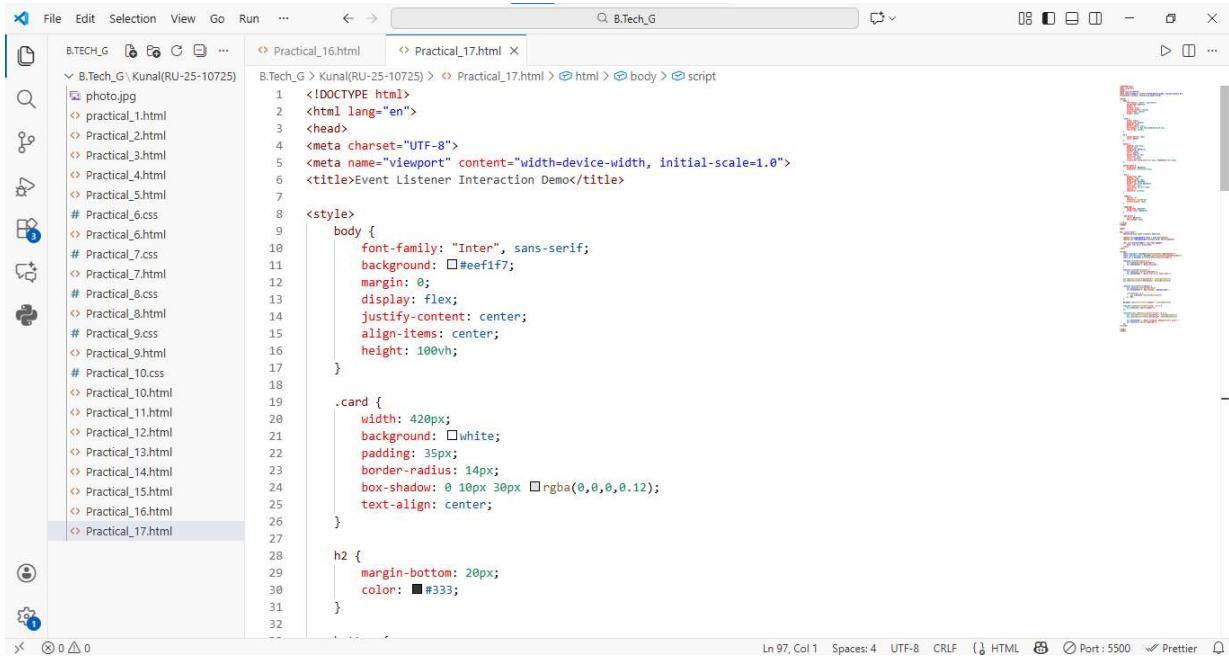
```
<html lang="en">
<body>
<div class="container">
</div>
<script>
const button = document.getElementById("toggleBtn");
const box = document.getElementById("contentBox");

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

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

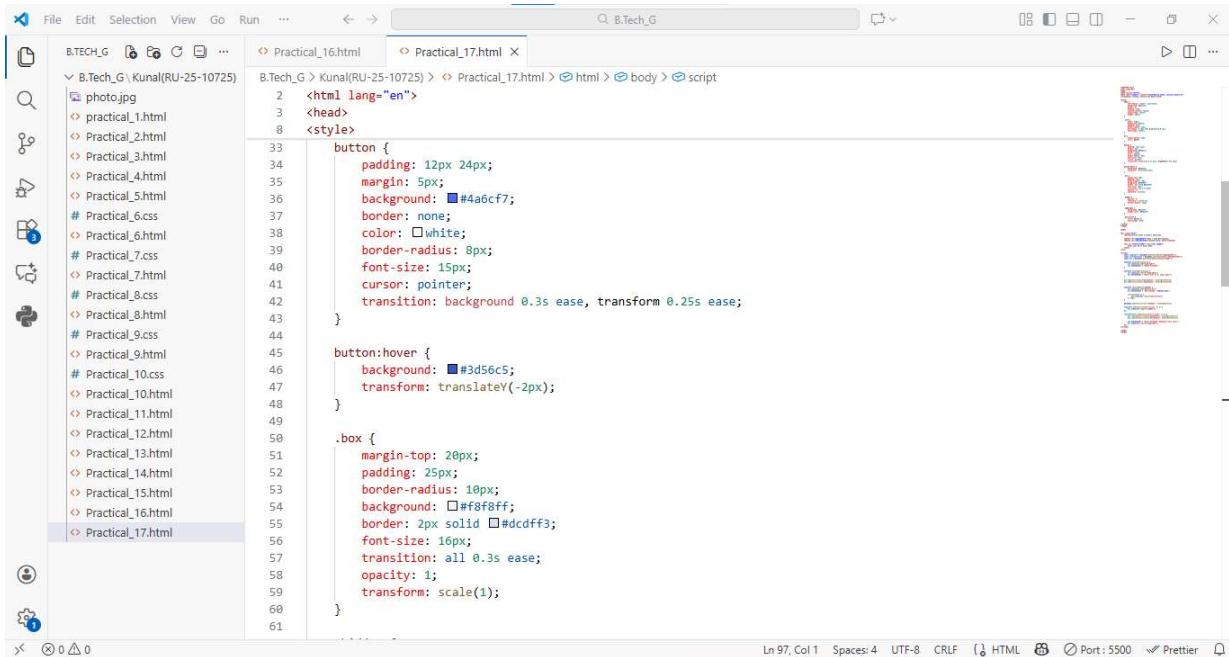


17. Add interactivity using event listeners (addEventListener, removeEventListener) for mouse/keyboard events.



```
B.TECH_G File Edit Selection View Go Run ... B.Tech_G Practical_16.html Practical_17.html
B.Tech_G > Kunal(RU-25-10725) > Practical_17.html > html > body > script
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Event Listener Interaction Demo</title>
7
8 <style>
9   body {
10     font-family: "Inter", sans-serif;
11     background: #eef1f7;
12     margin: 0;
13     display: flex;
14     justify-content: center;
15     align-items: center;
16     height: 100vh;
17   }
18
19   .card {
20     width: 420px;
21     background: white;
22     padding: 35px;
23     border-radius: 14px;
24     box-shadow: 0 10px 30px rgba(0,0,0,0.12);
25     text-align: center;
26   }
27
28   h2 {
29     margin-bottom: 20px;
30     color: #333;
31   }
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
```

Ln 97, Col 1 Spaces: 4 UTF-8 CRLF ⚡ HTML 🌐 Port: 5500 ✅ Prettier



```
B.TECH_G File Edit Selection View Go Run ... B.Tech_G Practical_16.html Practical_17.html
B.Tech_G > Kunal(RU-25-10725) > Practical_17.html > html > body > script
2 <html lang="en">
3 <head>
8 <style>
33   button {
34     padding: 12px 24px;
35     margin: 5px;
36     background: #4a6cf7;
37     border: none;
38     color: white;
39     border-radius: 8px;
40     font-size: 15px;
41     cursor: pointer;
42     transition: background 0.3s ease, transform 0.25s ease;
43   }
44
45   button:hover {
46     background: #3d56c5;
47     transform: translateY(-2px);
48   }
49
50   .box {
51     margin-top: 20px;
52     padding: 25px;
53     border-radius: 10px;
54     background: #f8f8ff;
55     border: 2px solid #dcdf3;
56     font-size: 16px;
57     transition: all 0.3s ease;
58     opacity: 1;
59     transform: scale(1);
60   }
61
```

Ln 97, Col 1 Spaces: 4 UTF-8 CRLF ⚡ HTML 🌐 Port: 5500 ✅ Prettier

The screenshot shows a code editor interface with the following details:

- Title Bar:** B.Tech_G
- File Explorer:** Shows a folder structure under B.Tech_G\Kunal(RU-25-10725) containing files like photo.jpg, practical_1.html, practical_2.html, practical_3.html, practical_4.html, practical_5.html, practical_6.css, practical_6.html, practical_7.css, practical_7.html, practical_8.css, practical_8.html, practical_9.css, practical_9.html, practical_10.css, practical_10.html, practical_11.html, practical_12.html, practical_13.html, practical_14.html, practical_15.html, practical_16.html, and practical_17.html.
- Code Editor:** The file practical_17.html is open, displaying the following code:

```
<html lang="en">
  <head>
    <script>
      .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>
    </div>
  </body>
</html>
```

Bottom status bar: Ln 97, Col 1 | Spaces: 4 | UTF-8 | CRLF | ⚡ HTML | Port: 5500 | Prettier

The screenshot shows a code editor interface with the following details:

- Title Bar:** B.Tech_G
- File Explorer:** Shows a folder structure under B.Tech_G\Kunal(RU-25-10725) containing files like photo.jpg, practical_1.html, practical_2.html, practical_3.html, practical_4.html, practical_5.html, practical_6.css, practical_6.html, practical_7.css, practical_7.html, practical_8.css, practical_8.html, practical_9.css, practical_9.html, practical_10.css, practical_10.html, practical_11.html, practical_12.html, practical_13.html, practical_14.html, practical_15.html, practical_16.html, and practical_17.html.
- Code Editor:** The file practical_17.html is open, displaying the following code, which includes the script logic from the previous screenshot:

```
<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);
      }

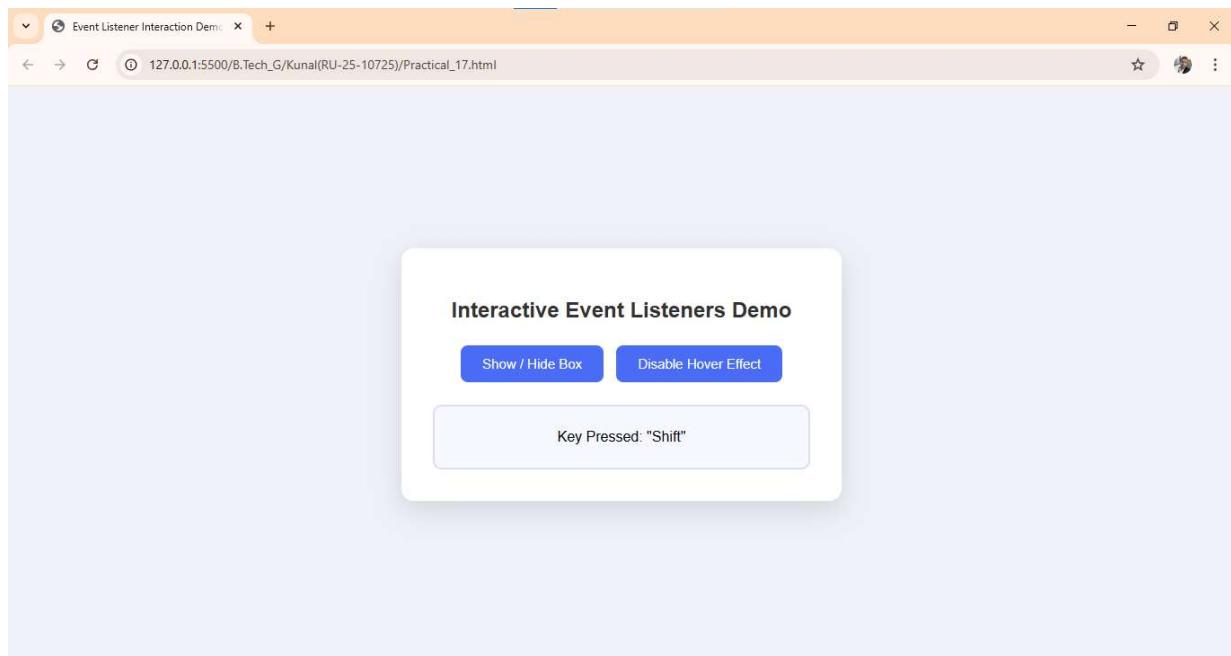
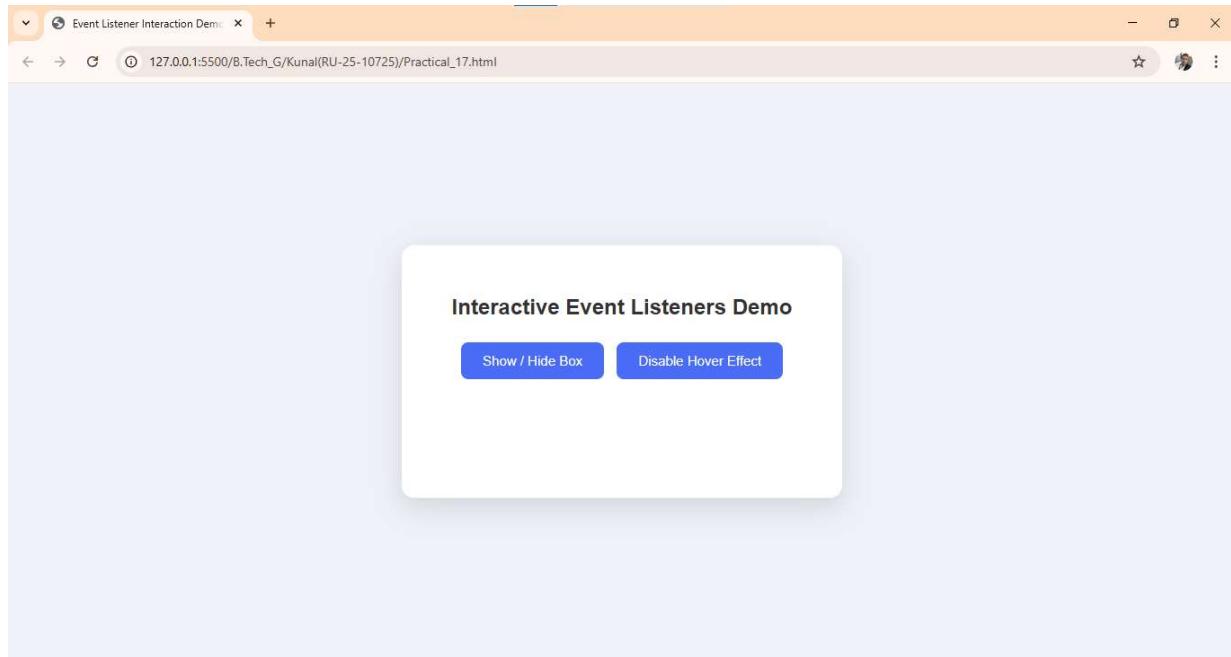
      document.addEventListener("keydown", handleKeyPress);

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

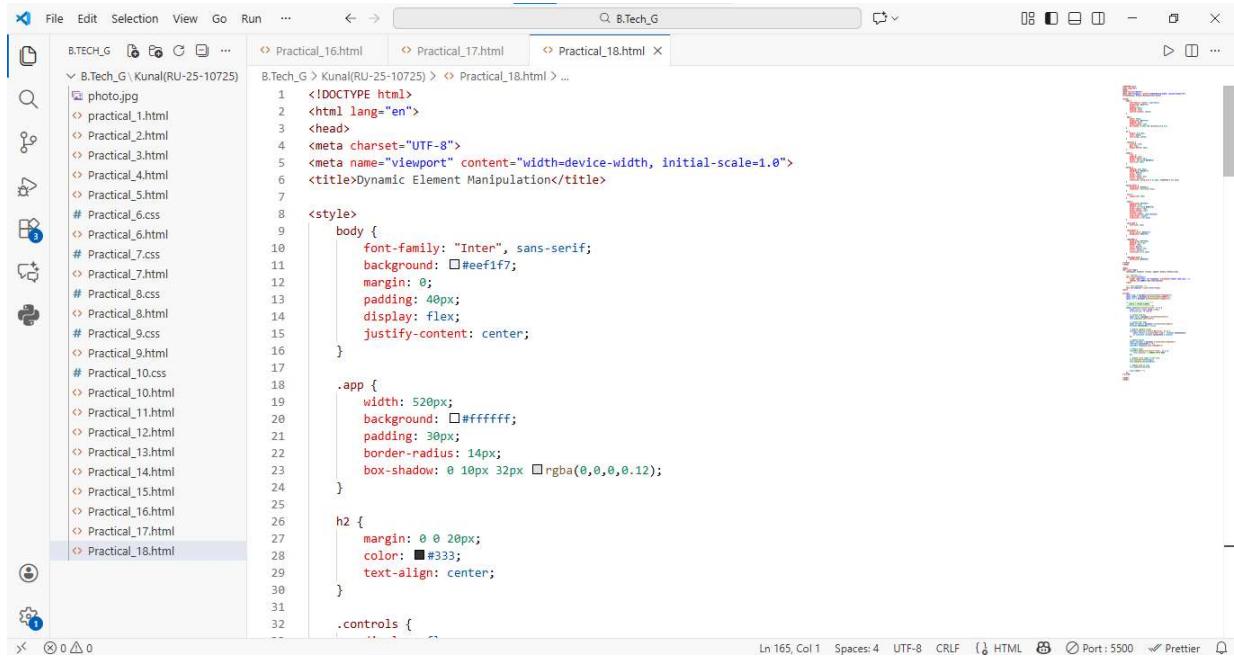
      removeHoverBtn.addEventListener("click", () => {
        box.removeEventListener("mouseenter", handleMouseEnter);
        box.removeEventListener("mouseleave", handleMouseLeave);

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

Bottom status bar: Ln 97, Col 1 | Spaces: 4 | UTF-8 | CRLF | ⚡ HTML | Port: 5500 | Prettier



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

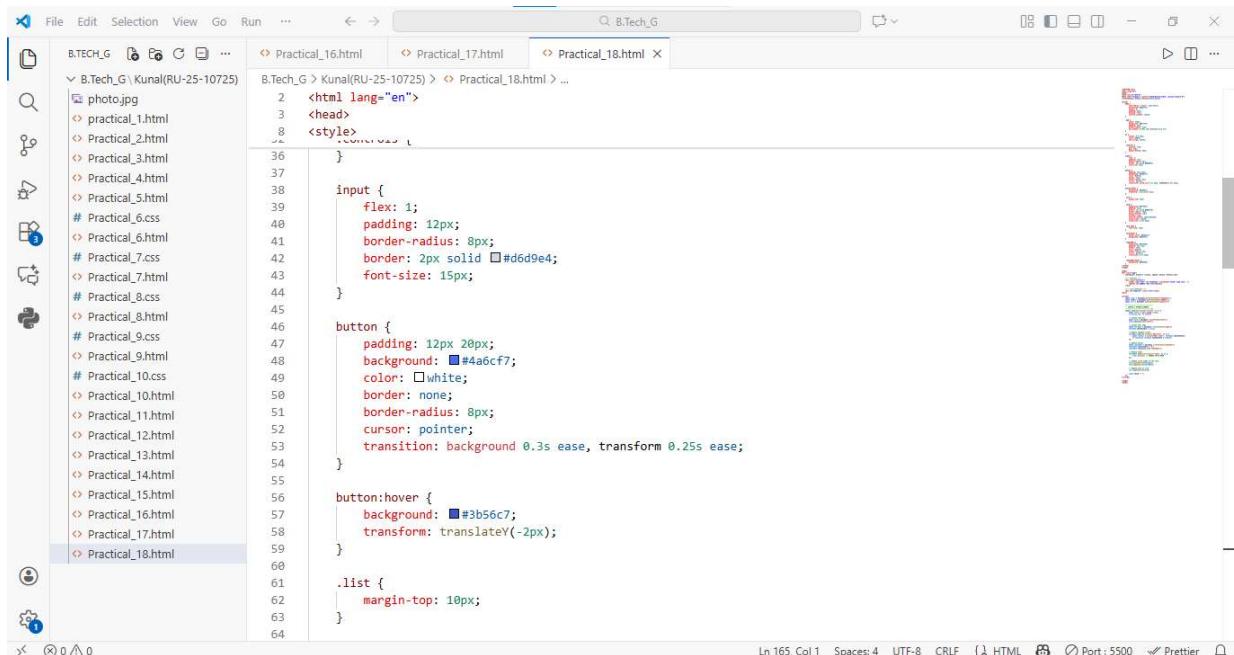


```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Dynamic Element Manipulation</title>
<style>
body {
    font-family: "Inter", sans-serif;
    background: #eef1f7;
    margin: 0;
    padding: 40px;
    display: flex;
    justify-content: center;
}

.app {
    width: 520px;
    background: #ffffff;
    padding: 30px;
    border-radius: 14px;
    box-shadow: 0 10px 32px rgba(0,0,0,.12);
}

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

.controls {
    margin-top: 20px;
}
```



```
html lang="en">
<head>
<style>
input {
    flex: 1;
    padding: 12px;
    border-radius: 8px;
    border: 2px solid #d6d9e4;
    font-size: 15px;
}

button {
    padding: 12px 28px;
    background: #4aacf7;
    color: #fff;
    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;
}
```

B.Tech_G > Kunal(RU-25-10725) > Practical_18.html ...

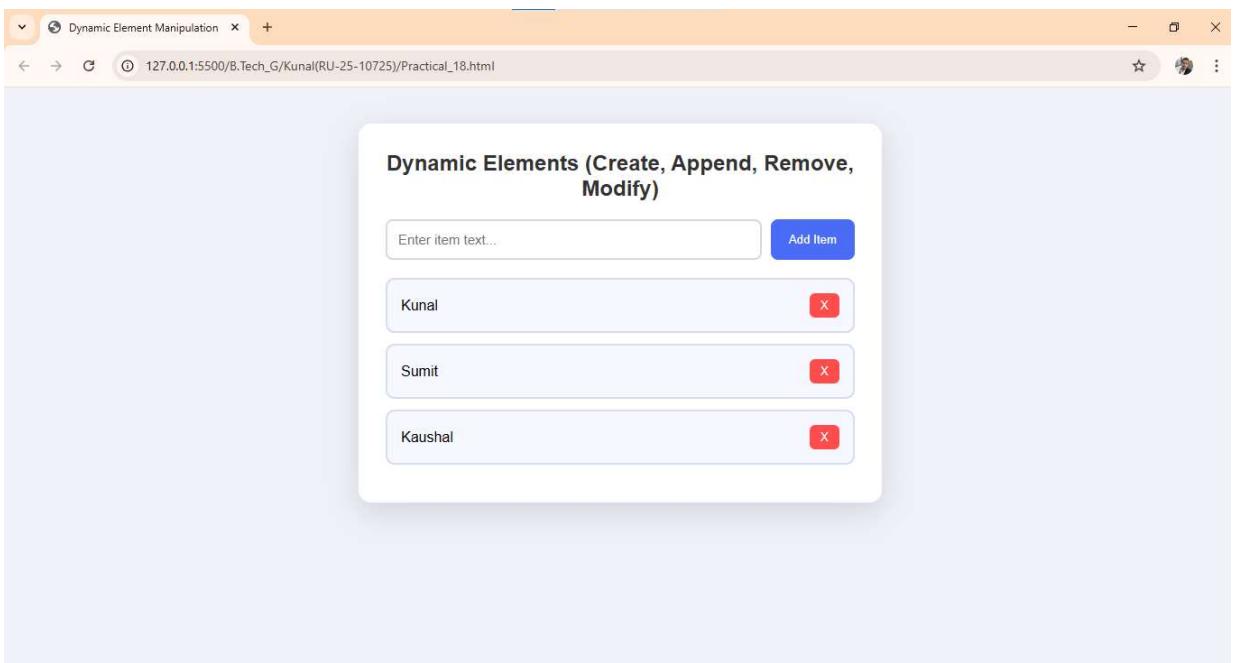
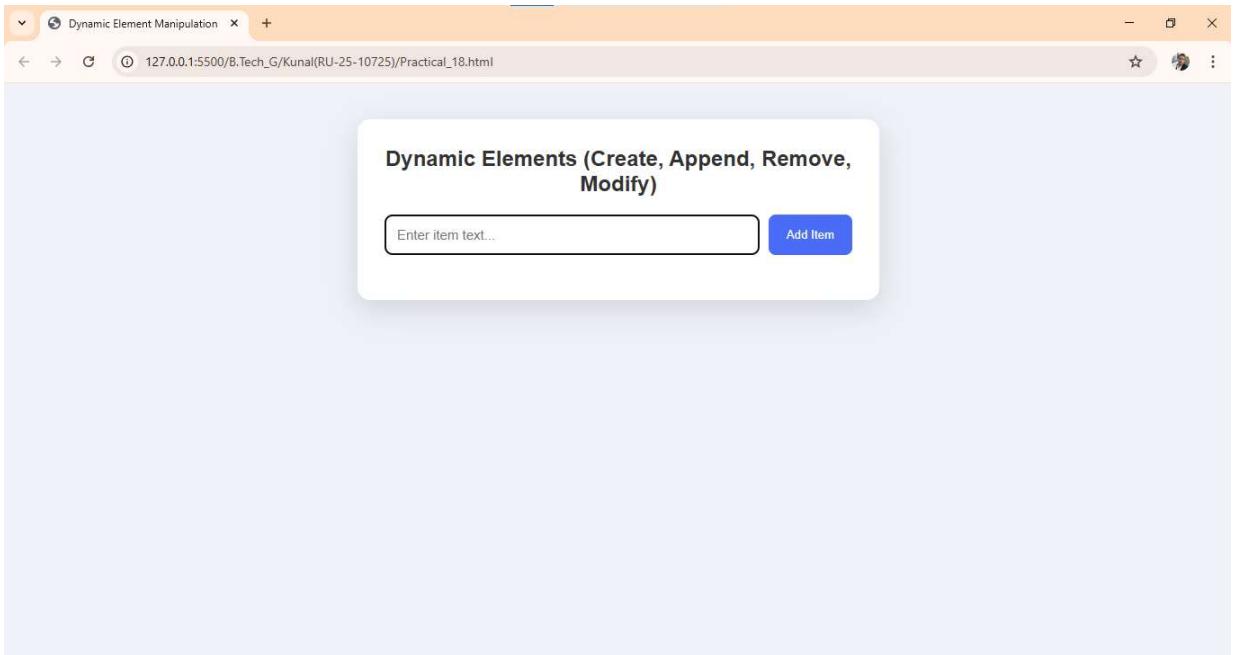
```
2 <html lang="en">
3 <head>
8 <style>
86 .removeBtn {
87 background: #ff4d4d;
88 padding: 6px 12px;
89 border: none;
90 color: white;
91 border-radius: 6px;
92 cursor: pointer;
93 transition: 0.2s ease;
94 }
95 .removeBtn:hover {
96 background: #d63838;
97 }
98 </style>
99 </head>
100 <body>
101 <div class="app">
102 <h2>Dynamic Elements (Create, Append, Remove, Modify)</h2>
103 <!-- Controls -->
104 <div class="controls">
105 <input type="text" id="itemInput" placeholder="Enter item text...">
106 <button id="addBtn">Add Item</button>
107 </div>
108 <!-- List Container -->
109 <div id="itemList" class="list"></div>
110 </div>
111 <!-- List Container -->
112 <div id="itemList" class="list"></div>
113 </div>
114 </div>
```

Ln 165, Col 1 Spaces: 4 UTF-8 CRLF ⚡ HTML 🌐 Port: 5500 ✨ Prettier

B.Tech_G > Kunal(RU-25-10725) > Practical_18.html ...

```
2 <html lang="en">
102 <body>
116 <script>
124 addBtn.addEventListener("click", () => {
125   textSpan.addEventListener("dblclick", () => {
126     });
127   });
128   // Remove button
129   const removeBtn = document.createElement("button");
130   removeBtn.textContent = "X";
131   removeBtn.classList.add("removeBtn");
132   // Remove event
133   removeBtn.addEventListener("click", () => {
134     card.remove(); // REMOVE CHILD NODE
135   });
136   // Append child nodes to the card
137   card.appendChild(textSpan);
138   card.appendChild(removeBtn);
139   // Append card to list
140   list.appendChild(card);
141   // Clear input field
142   input.value = "";
143   });
144 </script>
163 </body>
164 </html>
```

Ln 165, Col 1 Spaces: 4 UTF-8 CRLF ⚡ HTML 🌐 Port: 5500 ✨ Prettier



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

```
<!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%, #020617);
display:flex; align-items:center; justify-content:center; color:var(--text);}
.app{width:96%; height:94vh; border-radius:20px; border:1px solid var(--border);
background-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; border-right:1px solid var(--border);}
.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(315deg, #38bdf8, #2c255e); color: #020617;}
.canvas-wrap{display:flex; align-items:center; justify-content:center; border-radius:14px; cursor:crosshair;}
canvas{background: #020617; border-radius:14px; cursor:crosshair;}
.right{padding:14px; border-left:1px solid var(--border);}
```



The screenshot shows a browser-based code editor interface. On the left, there's a sidebar with icons for file operations like Open, Save, and Run, along with a file tree. The file tree shows a folder structure under 'B.TECH_G \ Kunal(RU-25-10725)'. Under 'Code', there are files: Unit_1, Unit_2, Unit_3, Unit_4, and several 'Practical_X.html' files (X from 15 to 19). Under 'Output', there are PDF files: Unit 1.pdf, Unit 2.pdf, and Unit 3.pdf. The main area is a code editor with the file 'Practical_19.html' open. The code is an HTML application with various components like a toolbar, canvas, and right panel controls.

```
<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">
      </div>
    </div>
  </body>

```



The screenshot shows a browser-based code editor interface. The left sidebar displays a file tree with the following structure:

- B.TECH_G
- B.Tech_G\Kunal(RU-25-10725)
 - Code
 - > Unit_1
 - > Unit_2
 - > Unit_3
 - Unit_4
 - ↳ Practical_15.html
 - ↳ Practical_16.html
 - ↳ Practical_17.html
 - ↳ Practical_18.html
 - Practical_19.html
 - > Unit_5
 - Output
 - ↳ Unit 1.pdf
 - ↳ Unit 2.pdf
 - ↳ Unit 3.pdf

The right pane shows the content of the selected file, Practical_19.html. The code is as follows:

```
B.Tech_G > Kunal(RU-25-10725) > Code > Unit_4 > Practical_19.html > html > head > style > .main  
2   <html lang="en">  
3   <body>  
4   <div id="eraserCursor" class="eraser"></div>  
5  
6   <script>  
7       const canvas=document.getElementById("canvas");  
8       const ctx=canvas.getContext("2d");  
9  
10      let tool="rect",drawing=false,startX=0,startY=0;  
11      let shapes=[],preview=null,pen=[];  
12      const color=document.getElementById("color");  
13      const size=document.getElementById("size");  
14      const eraser=document.getElementById("eraserCursor");  
15  
16      /* TOOL SELECT */  
17      document.querySelectorAll(".tool-btn").forEach(btn=>{  
18          btn.onclick=()=>{  
19              document.querySelectorAll(".tool-btn").forEach(b=>b.classList.remove("active"));  
20              btn.classList.add("active");  
21              tool=btn.dataset.tool;  
22          };  
23      });  
24  
25      /* DRAWING FUNCTION */  
26      function redraw(){  
27          ctx.clearRect(0,0,canvas.width,canvas.height);  
28          shapes.forEach(s=>{  
29              ctx.strokeStyle=s.c; ctx.lineWidth=s.w;  
30              if(s.t==="rect")ctx.strokeRect(s.x,s.y,s.w,s.h);  
31              if(s.t==="ellipse"){ctx.beginPath();ctx.ellipse(s.x,s.y,s.rx,s.ry,0,0,Math.PI*2);ctx.stroke();}  
32              if(s.t==="line"){ctx.beginPath();ctx.moveTo(s.x,s.y);ctx.lineTo(s.x2,s.y2);ctx.stroke();}  
33              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.  
34              .....
```

The screenshot shows a browser-based code editor interface. The left sidebar displays a file tree for a project named 'B.TECH_G' under 'B.Tech_G \ Kunal(RU-25-10725)'. The tree includes files like 'Unit_1', 'Unit_2', 'Unit_3', 'Unit_4' (which contains 'Practical_15.html', 'Practical_16.html', 'Practical_17.html', 'Practical_18.html', and 'Practical_19.html'), 'Unit_5', 'Output', and several PDF files ('Unit 1.pdf', 'Unit 2.pdf', 'Unit 3.pdf'). The right pane shows the content of 'Practical_19.html'. The code is written in JavaScript and uses canvas elements for drawing. It handles mouse events like 'mousedown' and 'mousemove' to draw shapes like rectangles, ellipses, and lines. It also handles a 'eraser' tool. The code uses global composite operations like 'destination-out' for drawing. The status bar at the bottom indicates the code length (Ln 23), file size (Col 59), and other details.

```
<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(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]}};

main* 0 □ 0 △ 0
```

A screenshot of a code editor window titled "Practical_19.html". The left sidebar shows a file tree with a project structure under "B.Tech_G \ Kunal(RU-25-10725) \ Code \ Unit_4". The main pane displays the following code:

```
2 <html lang="en">
37 <body>
75 <script>
142 </script>
147 canvas.onmouseup=e=>{
148   drawing=false;
149   const xe=e.offsetX,y=e.offsetY;
150   if(tool=="rect") shapes.push({t:"rect",x:startX,y:startY,wi:x-startX,hi:y-startY,c:color.value,w:size.value});
151   if(tool=="ellipse") shapes.push({t:"ellipse",x:(startX+x)/2,y:(startY+y)/2,rx:Math.abs(x-startX)/2,
152   ry:Math.abs(y-startY)/2,c:color.value,w:size.value});
153   if(tool=="line") shapes.push({t:"line",x:startX,y:startY,x2:x,y2:y,c:color.value,w:size.value});
154   if(tool=="pen"&&pen.length>1) shapes.push({t:"pen",p:[...pen],c:color.value,w:size.value});
155   preview=null;
156   redraw();
157 }
158 </script>
159 </body>
160 </html>
```

