Experiment 6

Interactive SVG Drawing Tool with Mouse Event Handlers

1. HTML-

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>SVG Drawing Tool</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <div class="toolbar">
    <label for="colorPicker">Color:</label>
    <input type="color" id="colorPicker" value="#FFFFFF">

    <label for="strokeWidth">Width:</label>
    <input type="range" id="strokeWidth" min="2" max="50"
value="5">

    <button id="clearButton">Clear Canvas</button>
  </div>

  <svg id="drawingCanvas"></svg>

  <script src="script.js"></script>
</body>
</html>
```

1. CSS

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background-color: #333;
  color: white;
  display: flex;
  flex-direction: column;
```

```
  align-items: center;
  overflow: hidden; /* Prevents scrollbars */
}

.toolbar {
  padding: 15px;
  background-color: #222;
  width: 100%;
  display: flex;
  justify-content: center;
  align-items: center;
  gap: 20px;
  box-shadow: 0 4px 8px rgba(0,0,0,0.3);
}

.toolbar label {
  font-weight: bold;
}

.toolbar input[type="color"] {
  border: none;
  background: none;
  width: 40px;
  height: 40px;
  cursor: pointer;
}

.toolbar input[type="range"] {
  cursor: pointer;
}

.toolbar button {
  padding: 10px 20px;
  border: none;
  border-radius: 5px;
  background-color: #007BFF;
  color: white;
  font-size: 1rem;
  cursor: pointer;
  transition: background-color 0.2s;
}

.toolbar button:hover {
  background-color: #0056b3;
}

#drawingCanvas {
```

```
    width: 90vw;
    height: 80vh;
    margin-top: 20px;
    background-color: #f0f0f0;
    border: 2px solid #555;
    cursor: crosshair;
    touch-action: none; /* Prevents scrolling on touch devices */
  }
```

2. Javascript

```
document.addEventListener('DOMContentLoaded', () => {
  const canvas = document.getElementById('drawingCanvas');
  const colorPicker = document.getElementById('colorPicker');
  const strokeWidthPicker = document.getElementById('strokeWidth');
  const clearButton = document.getElementById('clearButton');

  // State variables
  let isDrawing = false;
  let currentPath = null;
  let currentColor = colorPicker.value;
  let currentStrokeWidth = strokeWidthPicker.value;

  // --- Event Listeners for UI Controls ---

  colorPicker.addEventListener('change', (e) => {
    currentColor = e.target.value;
  });

  strokeWidthPicker.addEventListener('input', (e) => {
    currentStrokeWidth = e.target.value;
  });

  clearButton.addEventListener('click', () => {
    // Remove all child elements (paths) from the SVG
    canvas.innerHTML = '';
  });

  // --- Event Listeners for Drawing ---

  // Start Drawing
  canvas.addEventListener('mousedown', (e) => {
    isDrawing = true;

    // Get mouse coordinates relative to the canvas
```

```
  const { x, y } = getMousePos(e);

  // Create a new SVG path element
  currentPath = createPathElement(x, y);
  canvas.appendChild(currentPath);
 });

 // Continue Drawing
 canvas.addEventListener('mousemove', (e) => {
  if (!isDrawing) return;

  const { x, y } = getMousePos(e);

  // Append a new point to the current path's 'd' attribute
  const newPoint = ` L ${x} ${y}`;
  currentPath.setAttribute('d', currentPath.getAttribute('d') + newPoint);
 });

 // Stop Drawing
 window.addEventListener('mouseup', () => {
  isDrawing = false;
  currentPath = null;
 });

 // --- Helper Functions ---

 function getMousePos(event) {
  // Get the bounding box of the canvas to handle offsets correctly
  const rect = canvas.getBoundingClientRect();
  return {
   x: event.clientX - rect.left,
   y: event.clientY - rect.top
  };
 }

 function createPathElement(x, y) {
  // SVG elements must be created with a namespace
  const path = document.createElementNS('http://www.w3.org/2000/svg',
'path');
  path.setAttribute('d', `M ${x} ${y}`);
  path.setAttribute('stroke', currentColor);
  path.setAttribute('stroke-width', currentStrokeWidth);
  path.setAttribute('stroke-linejoin', 'round');
  path.setAttribute('stroke-linecap', 'round');
  path.setAttribute('fill', 'none');
```

```
    return path;
  }
});
```

OUTPUT of experiment

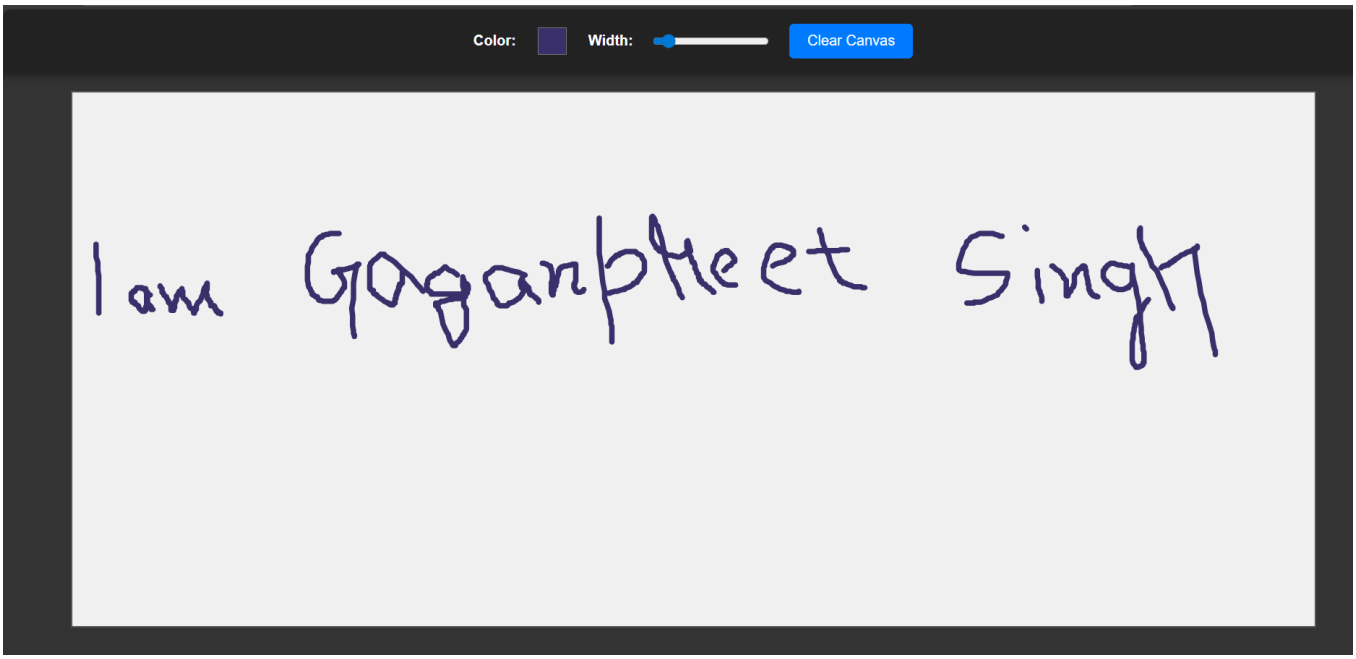

Figure 1 drawing tool with mouse