Integrasi SortableJS, Drawflow, dan Interact.JS untuk Membangun Antarmuka Aplikasi Interaktif Berbasis Web

Sigit Wijanarko[1]

Abstract*—The development of web-based user interface (UI) technology has driven the need for interactive and user-friendly applications. This study aims to explore and implement the integration of three popular JavaScript libraries—SortableJS, Drawflow, and Interact.js—to build a flexible drag-and-drop interface. With this approach, users can arrange components from a sidebar, place them into a workspace (canvas), and dynamically adjust the position and size of nodes. Experimental results show that the three libraries can be integrated without major conflicts and are capable of supporting dynamic and extensible user interfaces. This study provides a foundation for developing visual applications such as form builders, workflow editors, or interactive dashboards.

***Intisari*—Perkembangan teknologi antarmuka pengguna (UI) berbasis web mendorong kebutuhan akan aplikasi yang interaktif dan mudah digunakan. Penelitian ini bertujuan untuk mengeksplorasi dan mengimplementasikan integrasi tiga pustaka JavaScript populer—SortableJS, Drawflow, dan Interact.js—untuk membangun antarmuka drag-and-drop yang fleksibel. Dengan pendekatan ini, pengguna dapat menyusun komponen dari sidebar, menempatkannya ke dalam area kerja (canvas), serta mengatur posisi dan ukuran node secara dinamis. Hasil eksperimen menunjukkan bahwa ketiga pustaka dapat diintegrasikan tanpa konflik besar dan mampu mendukung antarmuka pengguna yang dinamis serta dapat diperluas. Studi ini memberikan dasar bagi pengembangan aplikasi visual seperti form builder, workflow editor, atau dashboard interaktif. Kata kunci: drag-and-drop, JavaScript, Drawflow, SortableJS, Interact.js, UI interaktif.**

***Kata Kunci*— drag-and-drop, JavaScript, Drawflow, SortableJS, Interact.js, UI interaktif.**

I. PENDAHULUAN

Antarmuka pengguna (user interface/UI) yang interaktif menjadi elemen penting dalam pengembangan aplikasi modern, khususnya aplikasi berbasis web. Interaktivitas memungkinkan pengguna melakukan berbagai aksi seperti memindahkan elemen, menyusun ulang komponen, atau membangun alur kerja visual dengan cara yang lebih intuitif. Salah satu pendekatan yang umum digunakan untuk mendukung hal tersebut adalah dengan menerapkan mekanisme drag-and-drop.

Drag-and-drop menjadi teknik populer karena kemampuannya menyederhanakan interaksi pengguna tanpa perlu banyak klik atau input teks. Untuk mewujudkan fitur ini secara efisien, tersedia sejumlah pustaka JavaScript yang menawarkan kapabilitas berbeda. Tiga pustaka yang cukup relevan dan sering digunakan adalah SortableJS, Drawflow, dan Interact.js.

Penelitian ini mengeksplorasi integrasi ketiga pustaka tersebut dalam satu aplikasi web sederhana. Tujuannya adalah untuk menguji sejauh mana pustaka-pustaka ini dapat bekerja secara sinergis untuk membentuk antarmuka pengguna interaktif, sekaligus mengkaji potensi konflik, kelebihan, dan kekurangannya.

II. KAJIAN LITERATUR

SortableJS adalah pustaka ringan berbasis JavaScript yang memungkinkan pengguna menyusun ulang elemen dalam sebuah daftar melalui interaksi drag-and-drop. Pustaka ini mendukung fitur seperti cloning, multi-drag, dan koneksi antar daftar. ”Keunggulannya terletak pada kemudahan implementasi dan ukuran file yang kecil”[1].

Drawflow merupakan pustaka yang dirancang untuk membangun editor berbasis node, mirip dengan Node-RED atau flowchart builder. Drawflow menyediakan API sederhana untuk membuat node, koneksi antar node, serta ekspor dan impor data dalam format JSON. ”Ini menjadikannya cocok untuk membangun workflow editor, data processing pipelines, atau visual scripting tools”[2].

Interact.js adalah pustaka fleksibel yang memberikan fitur interaktif seperti drag, resize, dan gesture (sentuh). ”Pustaka ini sering digunakan untuk membuat elemen yang bisa dipindah atau diubah ukurannya secara real-time, dengan kontrol penuh terhadap batas, snapping, dan aturan gerakan”[4].

Meskipun pustaka-pustaka ini banyak digunakan secara terpisah, studi mengenai integrasi ketiganya dalam satu aplikasi web masih terbatas. Oleh karena itu, penelitian ini *berkontribusi* dalam mengeksplorasi integrasi praktis ketiganya untuk membangun UI interaktif.

III.METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimental dengan metode studi kasus. Tahapan yang dilakukan adalah:

1. **Analisis kebutuhan**: Mendesain antarmuka aplikasi interaktif berbasis drag-and-drop.
2. **Implementasi awal**: Menggunakan SortableJS untuk membuat sidebar berisi komponen yang dapat di-drag.
3. **Integrasi dengan Drawflow**: Menyusun area kerja (canvas) menggunakan Drawflow, memungkinkan pengguna membuat node dari komponen yang di-drop.
4. **Penerapan Interact.js**: Menambahkan fitur resize pada node yang dibuat menggunakan interact.js, sebagai peningkatan fleksibilitas interaksi pengguna.
5. **Pengujian dan observasi**: Menguji fungsionalitas integrasi pustaka serta potensi konflik antar event.

Implementasi dilakukan dalam satu file HTML menggunakan CDN untuk masing-masing pustaka. Pengujian dilakukan secara manual menggunakan browser modern (Chrome, Firefox).

IV. HASIL DAN PEMBAHASAN

Hasil eksperimen menunjukkan bahwa SortableJS, Drawflow, dan Interact.js dapat diintegrasikan tanpa konflik signifikan selama setiap pustaka dikonfigurasikan dengan benar.

Pada tahap awal, sidebar yang dibuat dengan SortableJS mampu menghasilkan elemen yang bisa di-*drag* dan ditransfer menggunakan dataTransfer. Drawflow kemudian memproses elemen yang di-*drop* dan menghasilkan node baru yang dapat dikoneksikan antar satu sama lain.

HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>SortableJS + Drawflow</title>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/drawflow/dist/drawflow.min.css" />
</head>
<body>
  <div id="sidebar">
    <div class="draggable-item" data-node="input">Input Node</div>
    <div class="draggable-item" data-node="output">Output Node</div>
  </div>

  <div id="drawflow"></div>

  <script src="https://cdn.jsdelivr.net/npm/sortablejs@1.15.0/Sortable.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/drawflow/dist/drawflow.min.js"></script>

</body>
</html>
```

CSS

```css
body {
  display: flex;
  height: 100vh;
  margin: 0;
  font-family: sans-serif;
}
#sidebar {
  width: 200px;
  background: #f0f0f0;
  padding: 10px;
}
.draggable-item {
  background: #fff;
  padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  cursor: grab;
}
#drawflow {
  flex-grow: 1;
  position: relative;
  background: #fff;
}
```

JavaScript

```javascript
// 1. Inisialisasi SortableJS
new Sortable(document.getElementById("sidebar"), {
  group: {
    name: "shared",
    pull: "clone",
    put: false
  },
  sort: false,
  animation: 150
});

// 2. Inisialisasi Drawflow
const editor = new Drawflow(document.getElementById("drawflow"));
editor.reroute = true;
editor.start();

// 3. Tambahkan listener untuk drop
const drawflowContainer = document.getElementById("drawflow");

drawflowContainer.addEventListener("dragover", function (event) {
  event.preventDefault();
});

drawflowContainer.addEventListener("drop", function (event) {
  event.preventDefault();
  const nodeName = event.dataTransfer.getData("node");
  const rect = drawflowContainer.getBoundingClientRect();
  const x = event.clientX - rect.left;
  const y = event.clientY - rect.top;

  if (nodeName) {
    const nodeData = { name: nodeName };
    const html = `<div>${nodeName} Node</div>`;
    editor.addNode(nodeName, 1, 1, x, y, nodeName, nodeData, html);
  }
});

// 4. Siapkan drag manual pakai dataTransfer
document.querySelectorAll('.draggable-item').forEach(item => {
  item.setAttribute("draggable", true);
  item.addEventListener("dragstart", e => {
    e.dataTransfer.setData("node", item.dataset.node);
  });
});
```

Fitur resize pada node berhasil ditambahkan menggunakan Interact.js tanpa mempengaruhi fungsi drag node bawaan dari Drawflow. Untuk menghindari konflik, Interact.js hanya dikonfigurasikan untuk manipulasi ukuran (resizable), dan tidak menggunakan drag gesture bawaan.

HTML

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>SortableJS + Drawflow + Interact.js</title>
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/drawflow/dist/drawflow.min.css" />
</head>
<body>
  <div id="sidebar">
    <div class="draggable-item" data-node="input">Input Node</div>
    <div class="draggable-item" data-node="output">Output Node</div>
  </div>

  <div id="drawflow"></div>

  <script src="https://cdn.jsdelivr.net/npm/sortablejs@1.15.0/Sortable.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/drawflow/dist/drawflow.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/interactjs/dist/interact.min.js"></script>
</body>
</html>
```

CSS

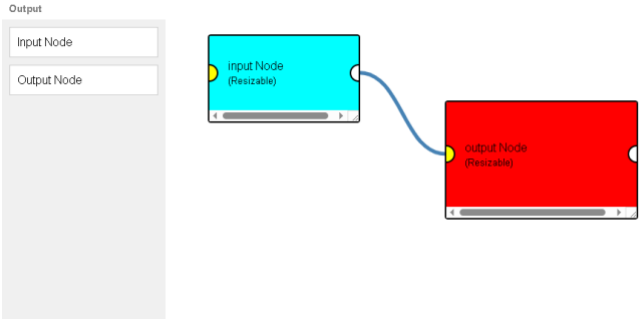
```css
body {
  display: flex;
  height: 100vh;
  margin: 0;
  font-family: sans-serif;
}

#sidebar {
  width: 200px;
  background: #f0f0f0;
  padding: 10px;
  overflow-y: auto;
}

.draggable-item {
  background: #fff;
  padding: 10px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
  cursor: grab;
}

#drawflow {
  flex-grow: 1;
  position: relative;
  background: #fff;
}

.drawflow-node {
  resize: both;
  overflow: auto;
  min-width: 120px;
  min-height: 60px;
}

.drawflow-node-content {
  padding: 10px;
}
```

JavaScript

```javascript
// 1. Inisialisasi SortableJS
new Sortable(document.getElementById("sidebar"), {
  group: {
    name: "shared",
    pull: "clone",
    put: false
  },
  sort: false,
  animation: 150
});

// 2. Inisialisasi Drawflow
const editor = new Drawflow(document.getElementById("drawflow"));
editor.reroute = true;
editor.start();

// 3. Tambahkan event drop dari sidebar ke canvas
const drawflowContainer = document.getElementById("drawflow");

drawflowContainer.addEventListener("dragover", function (event) {
  event.preventDefault();
});

drawflowContainer.addEventListener("drop", function (event) {
  event.preventDefault();
  const nodeName = event.dataTransfer.getData("node");
  const rect = drawflowContainer.getBoundingClientRect();
  const x = event.clientX - rect.left;
  const y = event.clientY - rect.top;

  if (nodeName) {
    const nodeData = { name: nodeName };
    const html = `<div class="drawflow-node-content">${nodeName} Node<br><small>(Resizable)</small></div>`;
    const id = editor.addNode(nodeName, 1, 1, x, y, nodeName, nodeData, html);

    // Terapkan interact.js ke node baru setelah render
    setTimeout(() => makeNodeResizable(id), 100);
  }
});
```

```
// 4. Siapkan drag manual dengan dataTransfer
document.querySelectorAll('.draggable-item').forEach(item => {
  item.setAttribute("draggable", true);
  item.addEventListener("dragstart", e => {
    e.dataTransfer.setData("node", item.dataset.node);
  });
});

// 5. Buat node resizable pakai interact.js
function makeNodeResizable(nodeId) {
  const nodeEl = document.querySelector(`#drawflow .drawflow-node[data-id="${nodeId}"]`);
  if (!nodeEl) return;

  interact(nodeEl).resizable({
    edges: { left: true, right: true, bottom: true, top: true },
    listeners: {
      move(event) {
        const { x, y } = event.target.dataset;
        let currentX = parseFloat(x) || 0;
        let currentY = parseFloat(y) || 0;

        Object.assign(event.target.style, {
          width: `${event.rect.width}px`,
          height: `${event.rect.height}px`
        });

        event.target.dataset.x = currentX;
        event.target.dataset.y = currentY;
      }
    },
    modifiers: [
      interact.modifiers.restrictSize({
        min: { width: 120, height: 60 }
      })
    ]
  });
}
```

Input dan Output node dapat dimanipulasi ukurannya (resizable).



Penggunaan kombinasi ini memungkinkan pengembangan antarmuka seperti:

a. Form builder dinamis (drag input form dari sidebar)
b. Workflow builder (alur proses logis)
c. Visual editor sederhana

Namun, terdapat beberapa batasan:

a. Drawflow memiliki gaya khusus yang harus dipatuhi untuk customisasi tampilan node
b. Interact.js perlu diikat ke node secara manual setelah node selesai dibuat
c. Sinkronisasi posisi atau ukuran perlu dijaga agar tetap kompatibel saat canvas diperbesar atau diskalakan

V. KESIMPULAN

Integrasi SortableJS, Drawflow, dan Interact.js terbukti dapat digunakan secara efektif untuk membangun antarmuka pengguna interaktif berbasis web. Ketiganya memiliki kekuatan unik: SortableJS untuk manajemen daftar, Drawflow untuk sistem node dan koneksi, serta Interact.js untuk fleksibilitas dalam manipulasi ukuran elemen. Penggunaan kombinasi ini dapat diterapkan pada berbagai jenis aplikasi seperti editor visual, form builder, dan workflow manager.

Penelitian ini membuka peluang untuk pengembangan lebih lanjut, seperti penambahan fitur ekspor data, dukungan multi-tab canvas, atau penyimpanan lokal/remote alur kerja. Pengujian lebih luas pada perangkat mobile dan skenario dengan jumlah node yang besar juga direkomendasikan sebagai studi lanjutan.

UCAPAN TERIMA KASIH

Penulis bersyukur kepada Tuhan Yang Maha Esa, Pencipta alam semesta, dan ingin mengungkapkan rasa terima kasih kepada semua individu dan kelompok yang telah memberikan bantuan, dukungan, dan arahan berharga selama pelaksanaan penelitian ini.

REFERENSI / REFERNCES

[1] J. Ruba, *Getting Started with SortableJS, 2020.* [Online]. Available: https://github.com/SortableJS/Sortable [Accessed: Jul. 28, 2025]

[2] A. Dominguez, *Drawflow: A simple and lightweight library for flow programming*, 2021. [Online]. Available: https://github.com/jerosoler/Drawflow [Accessed: Jul. 28, 2025]

[3] Mozilla Developer Network, *Using the HTML Drag and Drop API*, [Online]. Available: https://developer.mozilla.org [Accessed: Jul. 28, 2025]

[4] T. Kovacs, *Interact.js Documentation*, 2019. [Online]. Available: https://interactjs.io [Accessed: Jul. 28, 2025]

[5] W. Barendregt and M. M. Bekker, "Children may expect drag-and-drop instead of point-and-click," in *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, New York, NY, USA: ACM, 2011, pp. 1297–1302. doi: 10.1145/1979742.1979764.

[6] H. R. Ponce, R. E. Mayer, and M. S. Loyola, "Effects on test performance and efficiency of technology-enhanced items: An analysis of drag-and-drop response interactions," *Journal of Educational Computing Research*, vol. 59, no. 4, pp. 713–739, 2020. doi: 10.1177/0735633120969666.

[7] M. Raposo, S. Eloy, and M. S. Dias, "Bridging the gap in customised housing design: Integrating a graphic user interface for user collaboration," *PLOS ONE*, vol. 19, no. 12, Art. no. e0313291, 2024. doi: 10.1371/journal.pone.0313291.

[8] C. Crawshaw, *Drag & Drop UX Design Best Practices*, Pencil & Paper, Apr. 25, 2024. [Online]. Available: https://www.pencilandpaper.io/articles/ux-pattern-drag-and-drop. [Accessed: Jul. 28, 2025]. .

[9] J.-Y. Jeong and J.-W. Jeong, "Understanding user behavior in window selection using dragging for multiple targets," in *Proc. 2025 CHI Conf. on Human Factors in Computing Systems (CHI '25)*, New York, NY, USA: ACM, 2025, Art. no. 855, pp. 1–21. doi: 10.1145/3706598.3713410.

[10] B. Jalender, G. Govardhan, P. Premchand, C. Kiranmai, and G. Suresh Reddy, "Drag and drop: Influences on the design of reusable software components," *International Journal on Computer Science and Engineering*, vol. 2, 2010.

Sigit Wijanarko lahir di Sintang pada tanggal 23 Juni 1980. Lulus S1 Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Pembangunan Nasional – Jakarta pada tahun 2004. Lulus Magister Ilmu Komputer Program Pascasarjana Universitas Budi Luhur dengan konsentrasi Rekayasa Komputasi Terapan pada tahun 2019. Saat ini aktif sebagai Dosen tetap di STMIK Antar Bangsa dan praktisi IT di perusahaan swasta.