











Code:

import React, { useState , useEffect } from "react";

import { jsPDF } from "jspdf";

import "jspdf-autotable";

function MaterialForm() {

  const options = ["Concrete", "Cement", "Sand and steel"]; // Dropdown options

  const [inputValue, setInputValue] = useState(""); // User input

  const [filteredOptions, setFilteredOptions] = useState(options); // Filtered dropdown options

  const [isDropdownOpen, setIsDropdownOpen] = useState(false); // Dropdown visibility

  const [selectedOption, setSelectedOption] = useState(null); // Selected option

  const [finalitemList, setfinalitemList] = useState([]);

  const [finalsceb, setfinalsceb] = useState([]);

  const [finalconstraint, setfinalconstraint] = useState([]);

  const itemListcon = [

    "Waste Ceramic",

    "Recycle Aggregate",

    "Blast Furnace Slag",

    "Brick Waste",

  ];

  const itemListcem = ["Fly ash", "Biomass Bottom ash", "Micro silica"];

  const itemListsteel = [

    "Blast furnace Slag",

    "Glass Powder",

    "Ceramic Powder",

    "Reinforce Bar",

  ];

  const constraintscon = [15, 15, 35, 15];

  const constraintscem = [20, 15, 7.5];

  const constraintsstell = [80, 20, 12.5, 15];

  const scebcon = [8.1, 8.1, 8.1, 8.1];

  const scebcem = [925, 925, 925];

  const scebsteel = [7.65, 7.65, 7.65, 1.85]; // Source Carbon Emission for Base Material (Kg/MT)

  // Update finalitemList based on the selectedOption using useEffect

  useEffect(() => {

    if (selectedOption === "Concrete") {

      setfinalitemList(itemListcon);

      setfinalsceb(scebcon);

      setfinalconstraint(constraintscon);

    } else if (selectedOption === "Cement") {

      setfinalitemList(itemListcem);

      setfinalsceb(scebcem)

      setfinalconstraint(constraintscem)

    } else if (selectedOption === "Sand and steel") {

      setfinalitemList(itemListsteel);

      setfinalsceb(scebsteel)

      setfinalconstraint(constraintsstell)

    }

  }, [selectedOption]); // Run effect when selectedOption changes

  // Handle input change and filter dropdown options

  const handleInputChange = (e) => {

    const value = e.target.value;

    setInputValue(value);

    // Filter options based on input value

    const filtered = options.filter((option) =>

      option.toLowerCase().includes(value.toLowerCase())

    );

    setFilteredOptions(filtered);

    setIsDropdownOpen(filtered.length > 0 && value.length > 0); // Show dropdown if matches exist

  };

  // Handle option selection

  const handleSelect = (option) => {

    setInputValue(option); // Set input to selected option

    setSelectedOption(option); // Save selected option

    setIsDropdownOpen(false); // Close dropdown

  };

  const handleMouseDown = (e) => {

    e.preventDefault();

  };

  const handleBlur = () => {

    setTimeout(() => {

      if (!options.includes(inputValue)) {

        setInputValue(""); // Reset input if invalid

        setSelectedOption(null); // Clear selected option

      }

      setIsDropdownOpen(false); // Close dropdown after validation

    }, 100);

  };

  const [items, setItems] = useState([]);

  const [formData, setFormData] = useState({});

  const [commonValues, setCommonValues] = useState({

    totalQuantity: "",

    distance: "",

    transportationBase: "",

  });

  const [results, setResults] = useState([]);

  const handleItemSelection = (index) => {

    setItems((prevItems) =>

      prevItems.includes(index)

        ? prevItems.filter((i) => i !== index)

        : [...prevItems, index]

    );

  };

  const handleChange = (e, index, fieldName) => {

    const { value } = e.target;

    setFormData({

      ...formData,

      [index]: {

        ...formData[index],

        [fieldName]: value,

      },

    });

  };

  const handleCommonChange = (e, fieldName) => {

    const { value } = e.target;

    setCommonValues({

      ...commonValues,

      [fieldName]: value,

    });

  };

  const handleReset = () => {

    setItems([]);

    setFormData({});

    setCommonValues({

      totalQuantity: "",

      distance: "",

      transportationBase: "",

    });

    setResults([]);

  };

  const handleSubmit = (e) => {

    e.preventDefault();

    const calculatedResults = [];

    items.forEach((index) => {

      const totalQuantity = parseFloat(commonValues.totalQuantity);

      const distance = parseFloat(commonValues.distance);

      const transportationBase = parseFloat(commonValues.transportationBase);

      const percentageAlternative = parseFloat(

        formData[index]?.percentageAlternative

      );

      const sourceCarbonAlternative = parseFloat(

        formData[index]?.sourceCarbonAlternative

      );

      const distanceAlternative = parseFloat(

        formData[index]?.distanceAlternative

      );

      const transportationAlternative = parseFloat(

        formData[index]?.transportationAlternative

      );

      const scebItem = finalsceb[index];

      console.log(scebItem)

      const constraint = finalconstraint[index];

      console.log(constraint)

      if (percentageAlternative <= constraint) {

        const totalCarbonBase =

          totalQuantity \* (scebItem + distance \* transportationBase);

        const quantityBaseMaterial =

          totalQuantity \* (1 - 0.01 \* percentageAlternative);

        const quantityAlternativeMaterial =

          totalQuantity \* (0.01 \* percentageAlternative);

        const maxAllowableDistanceAlternative =

          (scebItem - sourceCarbonAlternative) / transportationAlternative +

          (distance \* transportationBase) / transportationAlternative;

        let ceb =

          quantityBaseMaterial \* (scebItem + distance \* transportationBase);

        console.log("ceb:",ceb);

        let cea =

          quantityAlternativeMaterial \*

          (sourceCarbonAlternative +

            distanceAlternative \* transportationAlternative);

        let totalCarbonWithAlternative = ceb + cea;

        let carbonSaving = totalCarbonBase - totalCarbonWithAlternative;

        let acceptanceStatus = carbonSaving > 0 ? "Accepted" : "Not Accepted";

        calculatedResults.push({

          item: finalitemList[index],

          totalCarbonBase,

          totalCarbonWithAlternative,

          carbonSaving,

          acceptanceStatus,

          quantityBaseMaterial,

          quantityAlternativeMaterial,

          maxAllowableDistanceAlternative,

        });

        console.log(calculatedResults)

      } else {

        alert(

          `Percentage of alternative material use for ${finalitemList[index]} is invalid. It should be less than ${constraint}%.`

        );

      }

    });

    setResults(calculatedResults);

  };

const handleDownloadPDF = () => {

  const doc = new jsPDF("l", "mm", "a4"); // Set to landscape orientation

  // Set title styling

  doc.setFontSize(16);

  doc.setFont("helvetica", "bold");

  doc.text("Carbon Emission Comparison Results", 15, 15);

  // Add project details

  doc.setFontSize(10);

  doc.setFont("helvetica", "normal");

  doc.text(`Base Material: ${selectedOption}`, 15, 25);

  doc.text(`Total Quantity: ${commonValues.totalQuantity} MT`, 15, 30);

  doc.text(`Base Distance: ${commonValues.distance} Km`, 15, 35);

  // Format date

  const currentDate = new Date().toLocaleDateString();

  doc.text(`Report Generated: ${currentDate}`, 200, 25);

  // Prepare data for the main results table

  const mainTableData = results.map((result) => [

    result.item,

    result.totalCarbonBase.toFixed(2),

    result.totalCarbonWithAlternative.toFixed(2),

    result.carbonSaving.toFixed(2),

    result.acceptanceStatus,

  ]);

  // Add main results table

  doc.autoTable({

    head: [

      [

        "Alternative Material",

        "Base Carbon Emission (Kg)",

        "Alternative Carbon Emission (Kg)",

        "Carbon Saving (Kg)",

        "Status",

      ],

    ],

    body: mainTableData,

    startY: 45,

    headStyles: {

      fillColor: [41, 128, 185],

      textColor: 255,

      fontSize: 10,

      halign: "center",

    },

    bodyStyles: {

      fontSize: 9,

      halign: "center",

    },

    alternateRowStyles: {

      fillColor: [242, 242, 242],

    },

    margin: { top: 40 },

  });

  // Add detailed specifications table

  const detailedTableData = results.map((result) => [

    result.item,

    result.quantityBaseMaterial.toFixed(2),

    result.quantityAlternativeMaterial.toFixed(2),

    result.maxAllowableDistanceAlternative.toFixed(2),

  ]);

  // Get the Y position after the first table

  const finalY = doc.previousAutoTable.finalY || 150;

  // Add second table title

  doc.setFontSize(12);

  doc.setFont("helvetica", "bold");

  doc.text("Detailed Specifications", 15, finalY + 15);

  // Add detailed specifications table

  doc.autoTable({

    head: [

      [

        "Alternative Material",

        "Base Material Quantity (MT)",

        "Alternative Material Quantity (MT)",

        "Max Allowable Distance (Km)",

      ],

    ],

    body: detailedTableData,

    startY: finalY + 20,

    headStyles: {

      fillColor: [41, 128, 185],

      textColor: 255,

      fontSize: 10,

      halign: "center",

    },

    bodyStyles: {

      fontSize: 9,

      halign: "center",

    },

    alternateRowStyles: {

      fillColor: [242, 242, 242],

    },

  });

  // Add footer

  const pageCount = doc.internal.getNumberOfPages();

  for (let i = 1; i <= pageCount; i++) {

    doc.setPage(i);

    doc.setFontSize(8);

    doc.setTextColor(128);

    doc.text(

      `Page ${i} of ${pageCount}`,

      doc.internal.pageSize.getWidth() - 20,

      doc.internal.pageSize.getHeight() - 10

    );

  }

  // Save the PDF

  doc.save("carbon\_emission\_analysis.pdf");

};

  return (

    <div className="max-w-2xl mx-auto bg-white p-8 rounded-lg shadow-md mt-6">

      <h2 className="text-3xl font-bold mb-6 text-center text-blue-700">

        Carbon Emission Comparison Form

      </h2>

      <form className="space-y-6" onSubmit={handleSubmit}>

        <div className="space-y-4 p-4 bg-gray-100 rounded-lg">

          <h4 className="text-lg font-medium">Base Material</h4>

          <div>

            <label className="block text-sm font-medium text-gray-700">

              Select Base Material

            </label>

            {/\* Input field for search and selection \*/}

            <input

              type="text"

              value={inputValue}

              onChange={handleInputChange}

              onBlur={handleBlur} // Validate on blur

              className="p-1 input input-bordered w-full my-1 max-w-xs"

              placeholder="Type to search..."

            />

            {/\* Conditionally render dropdown list \*/}

            {isDropdownOpen && (

              <ul className="dropdown-content menu bg-base-100 rounded-box z-[1] w-52 p-2 shadow">

                {filteredOptions.map((option, index) => (

                  <li key={index}>

                    <button

                      type="button"

                      onMouseDown={handleMouseDown} // Prevent immediate dropdown close

                      onClick={() => handleSelect(option)}

                      className="w-full text-left"

                    >

                      {option}

                    </button>

                  </li>

                ))}

              </ul>

            )}

            {/\* Display selected option or error \*/}

            {selectedOption ? (

              <p className="mt-2 text-sm text-green-600">

                Selected: {selectedOption}

              </p>

            ) : (

              inputValue &&

              !isDropdownOpen && (

                <p className="mt-2 text-sm text-red-600">Invalid selection</p>

              )

            )}

          </div>

          <div>

            <label className="block text-sm font-medium text-gray-700">

              Total Quantity (MT)

            </label>

            <input

              type="number"

              value={commonValues.totalQuantity}

              onChange={(e) => handleCommonChange(e, "totalQuantity")}

              className="w-full px-4 py-2 border rounded-md shadow-sm"

              required

            />

          </div>

          <div>

            <label className="block text-sm font-medium text-gray-700">

              Distance from Source (Km)

            </label>

            <input

              type="number"

              value={commonValues.distance}

              onChange={(e) => handleCommonChange(e, "distance")}

              className="w-full px-4 py-2 border rounded-md shadow-sm"

              required

            />

          </div>

          <div>

            <label className="block text-sm font-medium text-gray-700">

              Transportation Carbon Emission for Base Material (Kg/Km/MT)

            </label>

            <input

              type="number"

              value={commonValues.transportationBase}

              onChange={(e) => handleCommonChange(e, "transportationBase")}

              className="w-full px-4 py-2 border rounded-md shadow-sm"

              required

            />

          </div>

        </div>

        {selectedOption ? (

          <h3 className="text-lg font-medium">

            Select Alternative Items to Compare

          </h3>

        ) : (

          <h3 className="text-lg font-normal s"></h3>

        )}

        <div className="grid grid-cols-2 gap-4">

          {finalitemList.map((item, index) => (

            <div key={index} className="flex items-center">

              <input

                type="checkbox"

                id={`item-${index}`}

                checked={items.includes(index)}

                onChange={() => handleItemSelection(index)}

                className="h-4 w-4 text-blue-600 border-gray-300 rounded"

              />

              <label

                htmlFor={`item-${index}`}

                className="ml-2 text-sm text-gray-900"

              >

                {item}

              </label>

            </div>

          ))}

        </div>

        {items.map((index) => (

          <div key={index} className="space-y-4 p-4 bg-gray-100 rounded-lg">

            <h4 className="text-lg font-medium">{finalitemList[index]}</h4>

            <div>

              <label className="block text-sm font-medium text-gray-700">

                Percentage Alternative Material Used (%)

              </label>

              <input

                type="number"

                value={formData[index]?.percentageAlternative || ""}

                onChange={(e) =>

                  handleChange(e, index, "percentageAlternative")

                }

                className="w-full px-4 py-2 border rounded-md shadow-sm"

                required

              />

            </div>

            <div>

              <label className="block text-sm font-medium text-gray-700">

                Source Carbon Emission of Alternative (Kg/MT)

              </label>

              <input

                type="number"

                value={formData[index]?.sourceCarbonAlternative || ""}

                onChange={(e) =>

                  handleChange(e, index, "sourceCarbonAlternative")

                }

                className="w-full px-4 py-2 border rounded-md shadow-sm"

                required

              />

            </div>

            <div>

              <label className="block text-sm font-medium text-gray-700">

                Distance from Source for Alternative (Km)

              </label>

              <input

                type="number"

                value={formData[index]?.distanceAlternative || ""}

                onChange={(e) => handleChange(e, index, "distanceAlternative")}

                className="w-full px-4 py-2 border rounded-md shadow-sm"

                required

              />

            </div>

            <div>

              <label className="block text-sm font-medium text-gray-700">

                Transportation Carbon Emission for Alternative (Kg/Km/MT)

              </label>

              <input

                type="number"

                value={formData[index]?.transportationAlternative || ""}

                onChange={(e) =>

                  handleChange(e, index, "transportationAlternative")

                }

                className="w-full px-4 py-2 border rounded-md shadow-sm"

                required

              />

            </div>

          </div>

        ))}

        <div className="flex justify-between mt-4">

          <button

            type="button"

            onClick={handleReset}

            className="py-2 px-4 bg-red-600 text-white rounded-md hover:bg-red-700 transition duration-200"

          >

            Reset

          </button>

          <button

            type="submit"

            className="py-2 px-4 bg-blue-600 text-white rounded-md hover:bg-blue-700 transition duration-200"

          >

            Calculate

          </button>

        </div>

      </form>

      {results.length > 0 && (

        <div className="mt-6">

          <h3 className="text-lg font-medium mb-4">Results</h3>

          <table className="min-w-full bg-white border border-gray-300 rounded-lg shadow-md">

            <thead>

              <tr className="bg-gray-200">

                <th className="px-4 py-2">Item</th>

                <th className="px-4 py-2">Total Carbon (Base)</th>

                <th className="px-4 py-2">Total Carbon (Alternative)</th>

                <th className="px-4 py-2">Carbon Saving</th>

                <th className="px-4 py-2">Status</th>

              </tr>

            </thead>

            <tbody>

              {results.map((result, index) => (

                <tr key={index} className="border-b">

                  <td className="px-4 py-2">{result.item}</td>

                  <td className="px-4 py-2">

                    {result.totalCarbonBase.toFixed(2)} Kg

                  </td>

                  <td className="px-4 py-2">

                    {result.totalCarbonWithAlternative.toFixed(2)} Kg

                  </td>

                  <td className="px-4 py-2">

                    {result.carbonSaving.toFixed(2)} Kg

                  </td>

                  <td className="px-4 py-2">{result.acceptanceStatus}</td>

                </tr>

              ))}

            </tbody>

          </table>

          <button

            onClick={handleDownloadPDF}

            className="mt-4 py-2 px-4 bg-green-600 text-white rounded-md hover:bg-green-700 transition duration-200"

          >

            Download PDF

          </button>

        </div>

      )}

    </div>

  );

}

export default MaterialForm;