import React, { useState } from "react";

const Login = ({ handleLogin, setIsAdmin, isAdmin }) => {

  const [email, setEmail] = useState("");

  const [password, setPassword] = useState("");

  const [showPassword, setShowPassword] = useState(false);

  const submitHandler = (e) => {

    e.preventDefault();

    handleLogin(email, password);

    setEmail("");

    setPassword("");

  };

  return (

    <div className="flex w-screen h-screen">

      {/\* Right: Illustration \*/}

      <div className="w-1/2 hidden md:flex items-center justify-center bg-gradient-to-r from-blue-100 to-blue-200">

        {/\* Conditionally render the illustrations based on isAdmin \*/}

        <img

          src={

            isAdmin

              ? "/public/Admin\_illustration.png"

              : "/public/Employee\_illustration.png"

          }

          alt="Illustration"

          className="w-3/4 h-auto object-contain"

        />

      </div>

      {/\* Left: Login Form \*/}

      <div className="w-full md:w-1/2 flex items-center justify-center bg-white p-8">

        <div className="w-full max-w-md shadow-xl rounded-lg p-6 bg-white">

          <div className="flex justify-center mb-4">

            <img

              src="/public/Logo.svg"

              alt="Company Logo"

              className="h-20 w-auto object-contain"

            />

          </div>

          <h2 className="text-2xl font-bold text-center text-gray-800 mb-6">

            Employee Management System

          </h2>

          {/\* Role Toggle \*/}

          <div className="text-center mb-8">

            <div className="inline-flex items-center justify-center gap-10 text-lg font-medium">

              <button

                type="button"

                onClick={() => setIsAdmin(false)}

                className={`relative pb-1 transition-all duration-300 ${

                  !isAdmin

                    ? "text-blue-600 font-semibold after:absolute after:left-0 after:bottom-0 after:h-0.5 after:w-full after:bg-blue-600"

                    : "text-gray-400 hover:text-gray-600"

                }`}

              >

                Employee

              </button>

              <button

                type="button"

                onClick={() => setIsAdmin(true)}

                className={`relative pb-1 transition-all duration-300 ${

                  isAdmin

                    ? "text-blue-600 font-semibold after:absolute after:left-0 after:bottom-0 after:h-0.5 after:w-full after:bg-blue-600"

                    : "text-gray-400 hover:text-gray-600"

                }`}

              >

                Admin

              </button>

            </div>

          </div>

          {/\* Login Form \*/}

          <form onSubmit={submitHandler} className="space-y-5">

            <div>

              <label

                htmlFor="email"

                className="block text-sm font-medium text-gray-700"

              >

                Email Address

              </label>

              <input

                id="email"

                type="email"

                required

                value={email}

                onChange={(e) => setEmail(e.target.value)}

                className="text-black mt-1 block w-full px-4 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-blue-500 focus:border-blue-500"

                placeholder={

                  isAdmin ? "Enter admin email" : "Enter employee email"

                }

              />

            </div>

            <div>

              <label

                htmlFor="password"

                className="block text-sm font-medium text-gray-700"

              >

                Password

              </label>

              <div className="relative">

                <input

                  id="password"

                  type={showPassword ? "text" : "password"}

                  required

                  value={password}

                  onChange={(e) => setPassword(e.target.value)}

                  className="text-black mt-1 block w-full px-4 py-2 border border-gray-300 rounded-md shadow-sm focus:outline-none focus:ring-blue-500 focus:border-blue-500"

                  placeholder={

                    isAdmin ? "Enter admin password" : "Enter employee password"

                  }

                />

                <button

                  type="button"

                  onClick={() => setShowPassword(!showPassword)}

                  className="absolute right-3 top-1/2 transform -translate-y-1/2 text-sm text-blue-600 hover:text-blue-800 focus:outline-none"

                >

                  {showPassword ? "Hide" : "Show"}

                </button>

              </div>

            </div>

            <button

              type="submit"

              className="w-full bg-blue-600 text-white py-2 px-4 rounded-md hover:bg-blue-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-blue-500"

            >

              Log In

            </button>

          </form>

        </div>

      </div>

    </div>

  );

};

export default Login;

import React from "react";

import Header from "../other/Header";

import CreateTask from "../other/CreateTask";

import AllTask from "../other/AllTask";

const AdminDashboard = (props) => {

    const user = JSON.parse(localStorage.getItem("loggedInUser"));

  return (

    <div className="h-full bg-white w-full p-7">

      <Header changeUser={props.changeUser} data={props.data}/>

      <CreateTask />

      <AllTask />

    </div>

  );

};

export default AdminDashboard;

import React from 'react';

import Header from '../other/Header';

import TaskListNumbers from '../other/TaskListNumbers';

import TaskList from '../TaskList/TaskList';

const EmployeeDashboard = (props) => {

  return (

    <div className="min-h-screen bg-gray-50 p-8">

      <Header changeUser={props.changeUser} data={props.data}/>

      <TaskListNumbers data={props.data} />

      <TaskList data={props.data} />

    </div>

  );

};

export default EmployeeDashboard;

import React, { useContext } from "react";

import { AuthContext } from "../../context/AuthProvider";

const AllTask = () => {

  const [userData] = useContext(AuthContext);

  return (

    <div className="p-6 rounded-lg mt-6 shadow-lg border-2 border-gray-300">

      {/\* Table Header \*/}

      <div className="grid grid-cols-5 gap-4 bg-gradient-to-r from-blue-500 to-blue-600 text-white font-semibold text-sm py-3 px-4 rounded-lg mb-4">

        <h2 className="text-center">Employee Name</h2>

        <h3 className="text-center">New</h3>

        <h3 className="text-center">Active</h3>

        <h3 className="text-center">Completed</h3>

        <h3 className="text-center">Failed</h3>

      </div>

      {/\* Task Counts Per Employee \*/}

      <div className="space-y-3">

        {userData?.length > 0 ? (

          userData.map((user, index) => (

            <div

              key={index}

              className="grid grid-cols-5 gap-4 items-center text-sm text-gray-100 bg-gray-800 hover:bg-gray-700 transition-all px-4 py-3 rounded-lg border border-gray-700"

            >

              <h2 className="text-center font-medium text-white">

                {user.firstName}

              </h2>

              <h3 className="text-center text-blue-400 font-semibold">

                {user.taskCounts.newTask}

              </h3>

              <h3 className="text-center text-yellow-400 font-semibold">

                {user.taskCounts.active}

              </h3>

              <h3 className="text-center text-green-400 font-semibold">

                {user.taskCounts.completed}

              </h3>

              <h3 className="text-center text-red-500 font-semibold">

                {user.taskCounts.failed}

              </h3>

            </div>

          ))

        ) : (

          <div className="text-center text-gray-400 py-4">

            No task data found for employees.

          </div>

        )}

      </div>

    </div>

  );

};

export default AllTask;

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

import ReactQuill from "react-quill";

import 'react-quill/dist/quill.snow.css';

import { AuthContext } from "../../context/AuthProvider";

const CreateTask = () => {

  const [userData, setUserData] = useContext(AuthContext);

  const [taskTitle, setTaskTitle] = useState("");

  const [taskDescription, setTaskDescription] = useState(""); // Stores HTML from Quill

  const [taskDate, setTaskDate] = useState("");

  const [assignTo, setAssignTo] = useState("");

  const [category, setCategory] = useState("");

  const submitHandler = (e) => {

    e.preventDefault();

    const newTask = {

      taskTitle,

      taskDescription, // HTML string

      taskDate,

      category,

      active: false,

      newTask: true,

      failed: false,

      completed: false,

    };

    const updatedUserData = userData.map((user) => {

      if (user.firstName === assignTo) {

        return {

          ...user,

          tasks: [...user.tasks, newTask],

          taskCounts: {

            ...user.taskCounts,

            newTask: user.taskCounts.newTask + 1,

          },

        };

      }

      return user;

    });

    setUserData(updatedUserData);

    // Reset fields

    setTaskTitle("");

    setCategory("");

    setAssignTo("");

    setTaskDate("");

    setTaskDescription("");

  };

  // Define toolbar options

  const modules = {

    toolbar: [

      [{ header: [1, 2, false] }],

      ['bold', 'italic', 'underline', 'strike'],

      [{ list: 'ordered' }, { list: 'bullet' }],

      ['link', 'image'],

      ['clean'],

    ],

  };

  return (

    <div className="bg-white p-8 rounded-lg mt-6 shadow-lg border-2 border-gray-300">

      <h2 className="text-xl font-bold text-black mb-6">Create New Task</h2>

      <form

        onSubmit={submitHandler}

        className="flex flex-col md:flex-row md:justify-between gap-8"

      >

        <div className="w-full md:w-1/2 space-y-4">

          {/\* Task Title Input \*/}

          <div>

            <label className="block text-black font-medium mb-1 text-sm">

              Task Title

            </label>

            <input

              required

              value={taskTitle}

              onChange={(e) => setTaskTitle(e.target.value)}

              className="text-sm w-full py-3 px-3 rounded-lg border bg-gray-700 text-white font-medium focus:outline-none focus:ring-2 focus:ring-blue-500"

              type="text"

              placeholder="e.g., Design Login Page"

            />

          </div>

          {/\* Date Input \*/}

          <div>

            <label className="block text-black font-medium mb-1 text-sm">

              Date

            </label>

            <input

              required

              value={taskDate}

              onChange={(e) => setTaskDate(e.target.value)}

              className="text-sm w-full py-3 px-3 rounded-lg border bg-gray-700 text-white font-medium focus:outline-none focus:ring-2 focus:ring-blue-500"

              type="date"

            />

          </div>

          {/\* Assign To Dropdown \*/}

          <div>

            <label className="block text-gray-800 font-medium mb-1 text-sm">

              Assign To

            </label>

            <select

              required

              value={assignTo}

              onChange={(e) => setAssignTo(e.target.value)}

              className="text-sm w-full py-3 px-3 rounded-lg border bg-gray-800 text-white font-medium focus:outline-none focus:ring-2 focus:ring-blue-500"

            >

              <option value="" disabled>

                Select employee

              </option>

              {userData.map((user, index) => (

                <option key={index} value={user.firstName}>

                  {user.firstName}

                </option>

              ))}

            </select>

          </div>

          {/\* Category Dropdown \*/}

          <div>

            <label className="block text-black font-medium mb-1 text-sm">

              Category

            </label>

            <select

              required

              value={category}

              onChange={(e) => setCategory(e.target.value)}

              className="text-sm w-full py-3 px-3 rounded-lg border bg-gray-800 text-white font-medium focus:outline-none focus:ring-2 focus:ring-blue-500"

            >

              <option value="" disabled>

                Select category

              </option>

              <option value="UI/UX">UI/UX</option>

              <option value="Development">Development</option>

              <option value="Testing">Testing</option>

              <option value="DevOps">DevOps</option>

              <option value="Security">Security</option>

            </select>

          </div>

        </div>

        {/\* Description with ReactQuill \*/}

        <div className="w-full md:w-1/2 flex flex-col">

          <label className="block text-black font-medium mb-1 text-sm">

            Description

          </label>

          <ReactQuill

            theme="snow"

            value={taskDescription}

            onChange={setTaskDescription}

            modules={modules}

            className="bg-white rounded-lg h-[220px] mb-4"

          />

          <button

            type="submit"

            className="mt-[20px] bg-blue-500 text-white font-medium py-3 px-6 rounded-lg hover:bg-blue-600 transition duration-300 w-full"

          >

            Create Task

          </button>

        </div>

      </form>

    </div>

  );

};

export default CreateTask;

import React, { useState } from 'react'

import { setLocalStorage } from '../../utils/localStorage'

const Header = (props) => {

  const logOutUser = () => {

    localStorage.setItem('loggedInUser', '');

    props.changeUser('');

  };

  const user = props.data || {};

  return (

    <div className="flex justify-between items-center bg-white shadow-md p-6 rounded-lg mb-8">

      <div>

        <h1 className="text-2xl font-medium text-gray-800">Hello, <span className="text-3xl font-semibold text-blue-600">{user.firstName || "User"}👋</span></h1>

      </div>

      <button

        onClick={logOutUser}

        className="bg-red-600 text-white font-medium text-base px-6 py-3 rounded-lg hover:bg-red-700 transition duration-300"

      >

        Log Out

      </button>

    </div>

  );

};

export default Header

import React from "react";

const TaskListNumbers = ({ data }) => {

  const cards = [

    {

      label: "New Tasks",

      count: data.taskCounts.newTask,

      bg: "bg-blue-500",

      icon: "🆕",

    },

    {

      label: "Completed",

      count: data.taskCounts.completed,

      bg: "bg-green-500",

      icon: "✅",

    },

    {

      label: "Accepted",

      count: data.taskCounts.active,

      bg: "bg-yellow-500",

      icon: "📥",

    },

    {

      label: "Failed",

      count: data.taskCounts.failed,

      bg: "bg-red-500",

      icon: "❌",

    },

  ];

  return (

    <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 gap-6 mb-8">

      {cards.map((card, idx) => (

        <div

          key={idx}

          className={`p-6 rounded-lg shadow-lg text-white ${card.bg} flex items-center justify-between`}

        >

          <div>

            <h2 className="text-3xl font-semibold">{card.count}</h2>

            <p className="text-lg">{card.label}</p>

          </div>

          <div className="text-4xl">{card.icon}</div>

        </div>

      ))}

    </div>

  );

};

export default TaskListNumbers;

import React from 'react';

const AcceptTask = ({ data }) => {

  return (

    <div className="min-w-[300px] bg-yellow-100 p-6 rounded-lg shadow-lg transition transform hover:scale-105">

      <div className="flex justify-between items-center mb-4">

        <span className="bg-yellow-300 text-yellow-900 px-4 py-1 rounded-full text-sm font-medium">

          {data.category}

        </span>

        <span className="text-sm text-gray-600">{data.taskDate}</span>

      </div>

      <h3 className="text-xl font-semibold text-gray-800 mb-2">{data.taskTitle}</h3>

      <p className="text-sm text-gray-700 mb-4">{data.taskDescription}</p>

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

        <button className="bg-green-500 hover:bg-green-600 text-white px-5 py-2 rounded-lg text-sm w-full transition ease-in-out duration-200">

          Mark as Completed

        </button>

        <button className="bg-red-500 hover:bg-red-600 text-white px-5 py-2 rounded-lg text-sm w-full transition ease-in-out duration-200">

          Mark as Failed

        </button>

      </div>

    </div>

  );

};

export default AcceptTask;

import React from 'react';

const CompleteTask = ({ data }) => {

  return (

    <div className="min-w-[300px] bg-green-100 p-6 rounded-lg shadow-lg transition transform hover:scale-105">

      <div className="flex justify-between items-center mb-4">

        <span className="bg-green-300 text-green-900 px-4 py-1 rounded-full text-sm font-medium">

          {data.category}

        </span>

        <span className="text-sm text-gray-600">{data.taskDate}</span>

      </div>

      <h3 className="text-xl font-semibold text-gray-800 mb-2">{data.taskTitle}</h3>

      <p className="text-sm text-gray-700 mb-4">{data.taskDescription}</p>

      <button className="bg-green-600 hover:bg-green-700 text-white px-5 py-2 rounded-lg text-sm w-full transition ease-in-out duration-200">

        Completed

      </button>

    </div>

  );

};

export default CompleteTask;

import React from 'react';

const FailedTask = ({ data }) => {

  return (

    <div className="min-w-[300px] bg-red-100 p-6 rounded-lg shadow-lg transition transform hover:scale-105">

      <div className="flex justify-between items-center mb-4">

        <span className="bg-red-300 text-red-900 px-4 py-1 rounded-full text-sm font-medium">

          {data.category}

        </span>

        <span className="text-sm text-gray-600">{data.taskDate}</span>

      </div>

      <h3 className="text-xl font-semibold text-gray-800 mb-2">{data.taskTitle}</h3>

      <p className="text-sm text-gray-700 mb-4">{data.taskDescription}</p>

      <button className="bg-red-600 hover:bg-red-700 text-white px-5 py-2 rounded-lg text-sm w-full transition ease-in-out duration-200">

        Failed

      </button>

    </div>

  );

};

export default FailedTask;

import React from 'react';

const NewTask = ({ data }) => {

  return (

    <div className="min-w-[300px] bg-blue-100 p-6 rounded-lg shadow-lg transition transform hover:scale-105">

      <div className="flex justify-between items-center mb-4">

        <span className="bg-blue-300 text-blue-900 px-4 py-1 rounded-full text-sm font-medium">

          {data.category}

        </span>

        <span className="text-sm text-gray-600">{data.taskDate}</span>

      </div>

      <h3 className="text-xl font-semibold text-gray-800 mb-2">{data.taskTitle}</h3>

      <p className="text-sm text-gray-700 mb-4">{data.taskDescription}</p>

      <button className="bg-indigo-500 hover:bg-indigo-600 text-white px-5 py-2 rounded-lg text-sm w-full transition ease-in-out duration-200">

        Accept Task

      </button>

    </div>

  );

};

export default NewTask;

import React from 'react';

import AcceptTask from './AcceptTask';

import NewTask from './NewTask';

import CompleteTask from './CompleteTask';

import FailedTask from './FailedTask';

const TaskList = ({ data }) => {

  return (

    <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6">

      {data.tasks.map((task, idx) => {

        if (task.active) return <AcceptTask key={idx} data={task} />;

        if (task.newTask) return <NewTask key={idx} data={task} />;

        if (task.completed) return <CompleteTask key={idx} data={task} />;

        if (task.failed) return <FailedTask key={idx} data={task} />;

        return null;

      })}

    </div>

  );

};

export default TaskList;

import React, { createContext, useEffect, useState } from 'react'

import { getLocalStorage, setLocalStorage } from '../utils/localStorage'

export const AuthContext = createContext()

const AuthProvider = ({ children }) => {

    // localStorage.clear()

    const [userData, setUserData] = useState(null)

    useEffect(() => {

        setLocalStorage()

        const {employees} = getLocalStorage()

        setUserData(employees)

    }, [])

    return (

        <div>

            <AuthContext.Provider value={[userData,setUserData]}>

                {children}

            </AuthContext.Provider>

        </div>

    )

}

export default AuthProvider

const employees = [

    {

        "id": 1,

        "firstName": "Arjun",

        "email": "e@e.com",

        "password": "123",

        "taskCounts": {

            "active": 2,

            "newTask": 1,

            "completed": 1,

            "failed": 0

        },

        "tasks": [

            {

                "active": true,

                "newTask": true,

                "completed": false,

                "failed": false,

                "taskTitle": "Update website",

                "taskDescription": "Revamp the homepage design",

                "taskDate": "2024-10-12",

                "category": "Design"

            },

            {

                "active": false,

                "newTask": false,

                "completed": true,

                "failed": false,

                "taskTitle": "Client meeting",

                "taskDescription": "Discuss project requirements",

                "taskDate": "2024-10-10",

                "category": "Meeting"

            },

            {

                "active": true,

                "newTask": false,

                "completed": false,

                "failed": false,

                "taskTitle": "Fix bugs",

                "taskDescription": "Resolve bugs reported in issue tracker",

                "taskDate": "2024-10-14",

                "category": "Development"

            }

        ]

    },

    {

        "id": 2,

        "firstName": "Sneha",

        "email": "employee2@example.com",

        "password": "123",

        "taskCounts": {

            "active": 1,

            "newTask": 0,

            "completed": 1,

            "failed": 0

        },

        "tasks": [

            {

                "active": true,

                "newTask": false,

                "completed": false,

                "failed": false,

                "taskTitle": "Database optimization",

                "taskDescription": "Optimize queries for better performance",

                "taskDate": "2024-10-11",

                "category": "Database"

            },

            {

                "active": false,

                "newTask": false,

                "completed": true,

                "failed": false,

                "taskTitle": "Design new feature",

                "taskDescription": "Create mockups for the new feature",

                "taskDate": "2024-10-09",

                "category": "Design"

            }

        ]

    },

    {

        "id": 3,

        "firstName": "Ravi",

        "email": "employee3@example.com",

        "password": "123",

        "taskCounts": {

            "active": 2,

            "newTask": 1,

            "completed": 1,

            "failed": 0

        },

        "tasks": [

            {

                "active": true,

                "newTask": true,

                "completed": false,

                "failed": false,

                "taskTitle": "Prepare presentation",

                "taskDescription": "Prepare slides for upcoming client presentation",

                "taskDate": "2024-10-13",

                "category": "Presentation"

            },

            {

                "active": true,

                "newTask": false,

                "completed": false,

                "failed": false,

                "taskTitle": "Code review",

                "taskDescription": "Review the codebase for optimization",

                "taskDate": "2024-10-12",

                "category": "Development"

            },

            {

                "active": false,

                "newTask": false,

                "completed": true,

                "failed": false,

                "taskTitle": "Testing",

                "taskDescription": "Test the latest build for bugs",

                "taskDate": "2024-10-08",

                "category": "QA"

            }

        ]

    },

    {

        "id": 4,

        "firstName": "Priya",

        "email": "employee4@example.com",

        "password": "123",

        "taskCounts": {

            "active": 2,

            "newTask": 1,

            "completed": 0,

            "failed": 0

        },

        "tasks": [

            {

                "active": true,

                "newTask": true,

                "completed": false,

                "failed": false,

                "taskTitle": "Write documentation",

                "taskDescription": "Update the project documentation",

                "taskDate": "2024-10-13",

                "category": "Documentation"

            },

            {

                "active": true,

                "newTask": false,

                "completed": false,

                "failed": false,

                "taskTitle": "Set up CI/CD",

                "taskDescription": "Implement continuous integration pipeline",

                "taskDate": "2024-10-11",

                "category": "DevOps"

            }

        ]

    },

    {

        "id": 5,

        "firstName": "Karan",

        "email": "employee5@example.com",

        "password": "123",

        "taskCounts": {

            "active": 2,

            "newTask": 1,

            "completed": 1,

            "failed": 0

        },

        "tasks": [

            {

                "active": true,

                "newTask": true,

                "completed": false,

                "failed": false,

                "taskTitle": "UI redesign",

                "taskDescription": "Redesign the user interface for better UX",

                "taskDate": "2024-10-14",

                "category": "Design"

            },

            {

                "active": false,

                "newTask": false,

                "completed": true,

                "failed": false,

                "taskTitle": "Deploy new build",

                "taskDescription": "Deploy the latest build to production",

                "taskDate": "2024-10-09",

                "category": "DevOps"

            },

            {

                "active": true,

                "newTask": false,

                "completed": false,

                "failed": false,

                "taskTitle": "Client feedback",

                "taskDescription": "Gather feedback from clients after product launch",

                "taskDate": "2024-10-12",

                "category": "Support"

            }

        ]

    }

];

const admin = [{

    "id": 1,

    "firstName": "Karan",

    "email": "admin@example.com",

    "password": "123",

}];

export const setLocalStorage = ()=>{

    localStorage.setItem('employees',JSON.stringify(employees))

    localStorage.setItem('admin',JSON.stringify(admin))

}

export const getLocalStorage = ()=>{

    const employees = JSON.parse(localStorage.getItem('employees'))

    const admin = JSON.parse(localStorage.getItem('admin'))

    return {employees,admin}

}

#root {

  max-width: 1280px;

  margin: 0 auto;

  padding: 2rem;

  text-align: center;

}

.logo {

  height: 6em;

  padding: 1.5em;

  will-change: filter;

  transition: filter 300ms;

}

.logo:hover {

  filter: drop-shadow(0 0 2em #646cffaa);

}

.logo.react:hover {

  filter: drop-shadow(0 0 2em #61dafbaa);

}

@keyframes logo-spin {

  from {

    transform: rotate(0deg);

  }

  to {

    transform: rotate(360deg);

  }

}

@media (prefers-reduced-motion: no-preference) {

  a:nth-of-type(2) .logo {

    animation: logo-spin infinite 20s linear;

  }

}

.card {

  padding: 2em;

}

.read-the-docs {

  color: #888;

}

import React, { useContext, useEffect, useState } from 'react';

import Login from './components/Auth/Login';

import EmployeeDashboard from './components/Dashboard/EmployeeDashboard';

import AdminDashboard from './components/Dashboard/AdminDashboard';

import { AuthContext } from './context/AuthProvider';

const App = () => {

  const [user, setUser] = useState(null);

  const [loggedInUserData, setLoggedInUserData] = useState(null);

  const [userData, SetUserData] = useContext(AuthContext);

  const [isAdmin, setIsAdmin] = useState(false);  // New state for toggle

  // Load user from local storage on initial render

  useEffect(() => {

    const loggedInUser = localStorage.getItem('loggedInUser');

    if (loggedInUser) {

      const userData = JSON.parse(loggedInUser);

      setUser(userData.role);

      setLoggedInUserData(userData.data);

    }

  }, []);

  // Handle login based on email and password

  const handleLogin = (email, password) => {

    if (isAdmin) {

      // Admin login

      if (email === 'admin@me.com' && password === '123') {

        setUser('admin');

        localStorage.setItem('loggedInUser', JSON.stringify({ role: 'admin' }));

      } else {

        alert("Invalid Admin Credentials");

      }

    } else {

      // Employee login

      if (userData) {

        const employee = userData.find((e) => email === e.email && e.password === password);

        if (employee) {

          setUser('employee');

          setLoggedInUserData(employee);

          localStorage.setItem('loggedInUser', JSON.stringify({ role: 'employee', data: employee }));

        } else {

          alert("Invalid Employee Credentials");

        }

      }

    }

  };

  return (

    <>

      {!user ? <Login handleLogin={handleLogin} setIsAdmin={setIsAdmin} isAdmin={isAdmin} /> : ''}

      {user === 'admin' ? (

        <AdminDashboard changeUser={setUser} />

      ) : user === 'employee' ? (

        <EmployeeDashboard changeUser={setUser} data={loggedInUserData} />

      ) : null}

    </>

  );

};

export default App;

@tailwind base;

@tailwind components;

@tailwind utilities;

#tasklist::-webkit-scrollbar{

    background-color: transparent;

    height: 5px;

}

#tasklist::-webkit-scrollbar-thumb{

  background-color: rgb(143, 143, 143);

  border-radius: 50px;

}

input[type="date"]::-webkit-datetime-edit {

  color: gray; /\* Tailwind's text-gray-400 \*/

  font-size: 14px;

}

input[type="date"]::-webkit-calendar-picker-indicator {

  filter: invert(60%) brightness(90%);

  cursor: pointer;

}

import React from "react";

import ReactDOM from "react-dom/client";

import App from "./App.jsx";

import "./index.css";

import AuthProvider from "./context/AuthProvider.jsx";

ReactDOM.createRoot(document.getElementById("root")).render(

  <React.StrictMode>

    <AuthProvider>

      <App />

    </AuthProvider>

  </React.StrictMode>

);

<!doctype html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <link rel="icon" type="image/svg+xml" href="/vite.svg" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Vite + React</title>

  </head>

  <body>

    <div id="root"></div>

    <script type="module" src="/src/main.jsx"></script>

  </body>

</html>