27 Sep 2025, 00:13

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

// Simple single-file React app. // - Put this file into a CRA/Vite project as src/App.jsx (or src/App.js) // - Uses localStorage to persist registration and timer start

export default function App() { const [name, setName] = useState(""); const [password, setPassword] = useState(""); const [registered, setRegistered] = useState(false); const [message, setMessage] = useState(""); const [showPassword, setShowPassword] = useState(false); const [secondsLeft, setSecondsLeft] = useState(null);

const STORAGE_KEY = "registrationData"; // stores { name, startTs } const TOTAL_SECONDS = 24 * 60 * 60; // 24 hours

// On mount, check localStorage for existing
```

```
registration useEffect(() => { const raw =
localStorage.getItem(STORAGE_KEY); if (raw)
{ try { const data = JSON.parse(raw); if (data &&
data.name && data.startTs) { setRegistered(true);
setMessage(Registration successful. Welcome, $
{data.name}!); updateRemaining(data.startTs); } }
catch (e) { console.error("Failed to parse
storage", e); } }

// tick every second if registered
let timerId = null;
timerId = setInterval(() => {
  const raw2 =
localStorage.getItem(STORAGE_KEY);
  if (raw2) {
    try {
      const data2 = JSON.parse(raw2);
      if (data2 && data2.startTs)
updateRemaining(data2.startTs);
    } catch (e) {
      // ignore
    }
  }
}, 1000);
```

return () => clearInterval(timerId);
// eslint-disable-next-line react-hooks/
exhaustive-deps

}, []);

function updateRemaining(startTs) { const now = Date.now(); const elapsedSec = Math.floor((now - startTs) / 1000); const left = TOTAL_SECONDS - elapsedSec; setSecondsLeft(left > 0 ? left : 0); if (left <= 0) { // optionally clear registration or keep it and show expired // We'll keep the registration data but show timer 00:00:00 } }

function formatSeconds(s) { if (s === null) return "--:--:--"; if (s <= 0) return "00:00:00"; const hrs = Math.floor(s / 3600); const mins = Math.floor((s % 3600) / 60); const secs = s % 60; const pad = (n) => String(n).padStart(2, "0"); return ${pad(hrs)}:${pad(mins)}:${pad(secs)}; }

function handleRegister(e)
{ e.preventDefault(); // Simple validation if (!

```
name.trim()) { setMessage("Please enter your
name."); return; } if (password.length < 4)
{ setMessage("Password must be at least 4
characters."); return; }

const payload = { name: name.trim(), startTs:
Date.now() };
localStorage.setItem(STORAGE_KEY,
JSON.stringify(payload));
setRegistered(true);
setMessage(`Registration successful. Welcome,
${payload.name}!`);
updateRemaining(payload.startTs);

}

function handleLogout()
{ localStorage.removeItem(STORAGE_KEY);
setRegistered(false); setName("");
setPassword(""); setMessage("");
setSecondsLeft(null); }

return ( <div className="min-h-screen flex
items-center justify-center bg-gradient-to-br
```

```
from-slate-50 to-slate-100 p-6"> <div
className="w-full max-w-xl bg-white
rounded-2xl shadow-xl p-8"> <h1
className="text-2xl font-semibold
mb-4">Simple Registration & Dashboard</h1>

{!registered ? (
    <form onSubmit={handleRegister}
className="space-y-4">
      <div>
       <label className="block text-sm
font-medium mb-1">Name</label>
       <input
        value={name}
        onChange={(e) =>
setName(e.target.value)}
        className="w-full border rounded-md p-2"
        placeholder="Your name"
       />
      </div>

      <div>
       <label className="block text-sm
font-medium mb-1">Password</label>
```

```
        <div className="flex items-center gap-2">
          <input
            type={showPassword ? "text" :
"password"}
            value={password}
            onChange={(e) =>
setPassword(e.target.value)}
            className="flex-1 border rounded-md
p-2"
            placeholder="Choose a password"
          />
          <button
            type="button"
            onClick={() => setShowPassword((s) => !
s)}
            className="text-sm px-3 py-2 border
rounded-md"
          >
            {showPassword ? "Hide" : "Show"}
          </button>
        </div>
      </div>

      <div className="flex items-center gap-2">
```

```
        <button className="px-4 py-2 rounded-md
bg-sky-600 text-white">Register</button>
        <button
          type="button"
          onClick={() => {
            setName("Demo user");
            setPassword("demo1234");
          }}
          className="px-3 py-2 rounded-md border"
        >
          Fill demo
        </button>
      </div>

      {message && <p className="text-sm
text-emerald-700">{message}</p>}

      <p className="text-xs text-slate-500">This
demo stores registration in your browser
(localStorage).</p>
    </form>
  ) : (
    <div className="space-y-6">
      <div className="flex items-center
```

```
justify-between">
        <div>
          <h2 className="text-lg
font-medium">{message}</h2>
          <p className="text-sm
text-slate-500">You are registered locally on this
browser.</p>
        </div>

        <div>
          <button onClick={handleLogout}
className="px-3 py-2 rounded-md border">
            Clear / Logout
          </button>
        </div>
      </div>

      <div className="bg-slate-50 border
rounded-lg p-4">
        <h3 className="text-sm font-medium
mb-2">24-hour Timer</h3>
        <div className="text-3xl
font-mono">{formatSeconds(secondsLeft)}</
div>
```

```
        {secondsLeft === 0 && <p
className="text-sm text-rose-600 mt-2">The
24-hour period has expired.</p>}
      </div>

      <div className="grid grid-cols-1
sm:grid-cols-2 gap-4">
        <div className="p-4 border rounded-lg">
          <h4 className="font-medium">Quick
actions</h4>
          <p className="text-sm
text-slate-500">You can clear your registration
data or keep it for the full 24 hours.</p>
        </div>

        <div className="p-4 border rounded-lg">
          <h4 className="font-medium">Info</h4>
          <p className="text-sm
text-slate-500">Timer counts down from the
moment you registered.</p>
        </div>
      </div>
    </div>
  )}
```

```
  </div>

  {/* Tailwind CDN for quick demo: include in index.html head if using static build */}
</div>

); }
```