

CODING:

```

<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <title>Login — Example Authentication</title>
  <style>
    :root{
      --bg:#0f1724;
      --card:#0b1220;
      --accent:#6ee7b7;
      --muted:#9aa4b2;
      --danger:#ff6b6b;
      --glass: rgba(255,255,255,0.04);
      font-family: Inter, ui-sans-serif, system-ui, -apple-system, "Segoe UI", Roboto, "Helvetica Neue",
      Arial;
    }
    *{box-sizing:border-box}
    html,body{height:100%}
    body{
      margin:0;
      background: radial-gradient(1200px 600px at 10% 10%, rgba(110,231,183,0.06), transparent),
        radial-gradient(900px 500px at 90% 90%, rgba(99,102,241,0.04), transparent),
        var(--bg);
      color:#e6eef6;
      -webkit-font-smoothing:antialiased;
      -moz-osx-font-smoothing:grayscale;
      display:flex;
      align-items:center;
      justify-content:center;
      padding:24px;
    }

    .card{
      width:100%;
      max-width:420px;
      background:linear-gradient(180deg, rgba(255,255,255,0.02), rgba(255,255,255,0.01));
      border-radius:12px;
      padding:28px;
      box-shadow: 0 8px 30px rgba(2,6,23,0.7);
      border:1px solid rgba(255,255,255,0.03);
    }

    h1{margin:0 0 6px;font-size:20px}
    p.lead{margin:0 0 20px;color:var(--muted);font-size:13px}

    label{display:block;font-size:13px;margin-bottom:6px;color:#cbd6e3}
    .input{
      width:100%;

```

```

padding:12px 12px;
border-radius:8px;
background:var(--glass);
border:1px solid rgba(255,255,255,0.03);
color:inherit;
outline:none;
font-size:14px;
}
.input:focus{box-shadow:0 0 4px rgba(110,231,183,0.06);border-color:rgba(110,231,183,0.18)}
.row{display:flex;gap:10px}

.field{margin-bottom:14px}
.actions{display:flex;align-items:center;justify-content:space-between;margin-top:6px}

.btn{
  appearance:none;
  border:0;
  padding:10px 14px;
  border-radius:8px;
  background:linear-gradient(90deg,var(--accent), #60a5fa);
  color:#042027;
  font-weight:600;
  cursor:pointer;
}
.btn:disabled{opacity:0.6;cursor:not-allowed}

.link{background:none;border:0;color:var(--muted);cursor:pointer;font-size:13px}

.small{font-size:13px;color:var(--muted)}

.error{color:var(--danger);font-size:13px;margin-top:6px}
.success{color:#8ef0a8;font-size:13px;margin-top:6px}

.pw-wrap{position:relative}
.pw-toggle{
  position:absolute;right:8px;top:8px;padding:6px;border-
radius:6px;border:0;background:none;color:var(--muted);cursor:pointer;font-size:13px
}

.footer{margin-top:18px;text-align:center;color:var(--muted);font-size:13px}

@media (max-width:480px){.card{padding:18px}}
</style>
</head>
<body>
<main class="card" role="main" aria-labelledby="login-title">
  <h1 id="login-title">Welcome back</h1>
  <p class="lead">Sign in to continue to <strong>Example App</strong>.</p>

  <form id="loginForm" autocomplete="on" novalidate>
    <div class="field">

```

```

    <label for="email">Email</label>
    <input id="email" name="email" type="email" inputmode="email" class="input"
placeholder="you@example.com" required aria-required="true">
  </div>

  <div class="field pw-wrap">
    <label for="password">Password</label>
    <input id="password" name="password" type="password" class="input" placeholder="Enter your
password" required aria-required="true" minlength="6">
    <button type="button" id="togglePw" class="pw-toggle" aria-label="Show
password">Show</button>
  </div>

  <div class="field row" style="align-items:center;justify-content:space-between;margin-bottom:2px">
    <label style="display:flex;align-items:center;gap:8px">
      <input type="checkbox" id="remember" name="remember"> <span class="small">Remember
me</span>
    </label>
    <button type="button" class="link" id="forgotBtn">Forgot?</button>
  </div>

  <div class="field">
    <button class="btn" id="submitBtn" type="submit">Sign in</button>
    <div id="status" role="status" aria-live="polite"></div>
  </div>

  <div class="field" style="text-align:center;margin-top:8px">
    <div class="small">Or sign in with</div>
    <div class="row" style="margin-top:8px">
      <button type="button" class="btn" style="flex:1">Google</button>
      <button type="button" class="btn" style="flex:1;opacity:0.9">GitHub</button>
    </div>
  </div>
</form>

<div class="footer">Don't have an account? <button class="link" id="signupBtn">Create
one</button></div>
</main>

<script>
// ===== Simple front-end auth demo =====
// This file demonstrates an accessible, responsive login form with client-side validation
// and a fetch() call to a backend endpoint (/api/auth/login). The backend should perform
// real authentication (password hashing, rate-limits, issuing JWT or setting secure HttpOnly cookie).
//
// IMPORTANT security notes (backend responsibilities):
// - Store passwords using Argon2 / bcrypt with a strong work factor.
// - Use HTTPS only and set cookies with Secure; HttpOnly; SameSite=Strict (or Lax where
appropriate).
// - Implement rate-limiting and account lockouts for repeated failed attempts.
// - For SPAs, prefer setting refresh token in an HttpOnly cookie and returning short-lived access token.

```

```

const form = document.getElementById('loginForm');
const email = document.getElementById('email');
const pw = document.getElementById('password');
const toggle = document.getElementById('togglePw');
const status = document.getElementById('status');
const submitBtn = document.getElementById('submitBtn');

// Toggle password visibility (accessible)
toggle.addEventListener('click', () => {
  const isPw = pw.type === 'password';
  pw.type = isPw ? 'text' : 'password';
  toggle.textContent = isPw ? 'Hide' : 'Show';
  toggle.setAttribute('aria-pressed', String(isPw));
});

// Basic client-side validation helper
function validate() {
  status.textContent = "";
  if (!email.value) { status.textContent = 'Please enter your email.'; status.className='error'; return false }
  if (!pw.value) { status.textContent = 'Please enter your password.'; status.className='error'; return false }
  if (pw.value.length < 6) { status.textContent = 'Password must be at least 6 characters.'; status.className='error'; return false }
  return true;
}

// On submit -> send credentials to backend
form.addEventListener('submit', async (e) => {
  e.preventDefault();
  if (!validate()) return;

  submitBtn.disabled = true;
  const originalText = submitBtn.textContent;
  submitBtn.textContent = 'Signing in...';
  status.textContent = "";

  try {
    // Example payload. In production only send what is required.
    const payload = {
      email: email.value.trim().toLowerCase(),
      password: pw.value,
      remember: document.getElementById('remember').checked
    };

    // Use fetch to call your backend authentication endpoint
    const res = await fetch('/api/auth/login', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      credentials: 'include', // include cookies if your backend sets HttpOnly cookie
    });
  }
});

```



```

    body: JSON.stringify(payload)
  });

  const data = await res.json().catch(()=>({}));

  if (!res.ok) {
    // Backend should return useful error codes/messages like 401, 429, etc.
    status.className = 'error';
    status.textContent = data?.message || ('Login failed (' + res.status + ')');
    submitBtn.disabled = false;
    submitBtn.textContent = originalText;
    return;
  }

  // On success: backend may set HttpOnly refresh cookie and return a short-lived access token
  // For SPA: store access token in memory (not localStorage) and use it for API calls; refresh via
  cookie.
  status.className = 'success';
  status.textContent = 'Signed in successfully — redirecting...';

  // Optionally, backend returns { redirect: '/dashboard' }
  const redirectTo = data?.redirect || '/dashboard';
  setTimeout(()=>{ window.location.href = redirectTo }, 700);

} catch (err) {
  console.error(err);
  status.className = 'error';
  status.textContent = 'Network error. Please try again.';
  submitBtn.disabled = false;
  submitBtn.textContent = originalText;
}
});

// Small handlers for demo buttons (replace with real flows)
document.getElementById('forgotBtn').addEventListener('click', ()=>{ alert('Open forgot-password
flow (backend)') });
document.getElementById('signupBtn').addEventListener('click', ()=>{ window.location.href =
'/signup' });

// Accessibility: focus first field on load
window.addEventListener('load', ()=> email.focus());
</script>
</body>
</html>

```