<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8"/>

<meta name="viewport" content="width=device-width, initial-scale=1"/>

<title>FinCoach Pro — Premium Personal Finance Coach</title>

<style>

:root{

--bg:#0f172a;

--card:#111827;

--muted:#94a3b8;

--text:#e5e7eb;

--accent:#22c55e;

--warn:#f59e0b;

--danger:#ef4444;

--premium:#f59e0b;

--pro:#8b5cf6;

--border:#2a3a55;

--shadow:rgba(0,0,0,0.1);

--shadow-hover:rgba(0,0,0,0.2);

}

\*{box-sizing:border-box}

html,body{margin:0;padding:0;background:var(--bg);color:var(--text);font-family:system-ui,-apple-system,Segoe UI,Roboto,Inter,Arial,sans-serif}

.wrap{max-width:1000px;margin:0 auto;padding:16px 14px 120px}

.auth-container{max-width:400px;margin:100px auto;padding:20px}

.auth-card{background:#0b1425;border:1px solid #1f2937;border-radius:14px;padding:30px}

.auth-title{text-align:center;margin-bottom:30px;color:var(--accent)}

.form-group{margin-bottom:20px}

.form-group label{display:block;margin-bottom:8px;color:var(--muted)}

.form-group input{width:100%;background:#0b1425;border:1px solid #243041;color:var(--text);border-radius:10px;padding:12px;font-size:16px;outline:none}

.form-group input:focus{border-color:var(--accent)}

.btn-primary{width:100%;background:var(--accent);border:none;color:#0b1425;font-weight:600;cursor:pointer;padding:14px;border-radius:10px;font-size:16px;margin-bottom:15px;transition:all 0.3s ease;box-shadow:0 4px 6px -1px rgba(34,197,94,0.2)}

.btn-primary:hover{transform:translateY(-2px);box-shadow:0 8px 15px -3px rgba(34,197,94,0.3)}

.btn-secondary{width:100%;background:transparent;border:1px solid #243041;color:var(--text);cursor:pointer;padding:14px;border-radius:10px;font-size:16px}

.auth-switch{text-align:center;margin-top:20px;color:var(--muted)}

.auth-switch a{color:var(--accent);text-decoration:none;cursor:pointer}

.error{color:var(--danger);text-align:center;margin-bottom:15px;display:none}

.success{color:var(--accent);text-align:center;margin-bottom:15px;display:none}

.top{display:flex;align-items:center;gap:10px;padding:12px}

.logo{width:36px;height:36px;border-radius:10px;background:linear-gradient(135deg,var(--accent),#60a5fa);display:grid;place-items:center;font-weight:900;color:#0b1425}

.sub{color:var(--muted);font-size:13px;margin-top:-6px}

.grid{display:grid;gap:12px}

.card{background:#0b1425;border:1px solid #1f2937;border-radius:14px;padding:14px}

.stat{display:flex;justify-content:space-between;align-items:flex-end}

.stat h3{margin:6px 0 2px;font-size:13px;color:var(--muted)}

.stat .v{font-size:22px;font-weight:800}

.badge{font-size:12px;padding:4px 8px;border-radius:999px;border:1px solid #2b3b55;color:#cbd5e1}

.badge.premium{background:var(--premium);color:#0b1425;border-color:var(--premium)}

.badge.pro{background:var(--pro);color:#0b1425;border-color:var(--pro)}

.progress{height:10px;background:#0b1425;border:1px solid #243041;border-radius:999px;overflow:hidden}

.bar{height:100%;background:linear-gradient(90deg,var(--accent),#60a5fa)}

.bar.warn{background:linear-gradient(90deg,var(--warn),#f472b6)}

.bar.danger{background:linear-gradient(90deg,var(--danger),#f59e0b)}

input,select,button{background:#0b1425;border:1px solid #243041;color:var(--text);border-radius:10px;padding:10px 12px;font-size:14px;outline:none}

.row{display:flex;gap:8px;flex-wrap:wrap}

.pill{display:inline-block;padding:4px 10px;border-radius:999px;font-size:12px;border:1px solid #243041;color:#cbd5e1}

.pill.premium{border-color:var(--premium);color:var(--premium)}

.pill.pro{border-color:var(--pro);color:var(--pro)}

.list{display:grid;gap:8px}

.item{display:flex;justify-content:space-between;align-items:center;padding:10px;border:1px solid #223047;border-radius:10px;background:#0b1425}

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

.coach{display:grid;gap:8px;max-height:200px;overflow:auto}

.msg{padding:10px;border:1px solid #223047;border-radius:10px;background:#0b1425}

.msg.premium{border-color:var(--premium);background:linear-gradient(135deg,#1a1a2e,#16213e)}

.msg.pro{border-color:var(--pro);background:linear-gradient(135deg,#1a1a2e,#16213e)}

.toast{position:fixed;left:50%;transform:translateX(-50%);bottom:18px;background:#0b1425;border:1px solid #2a3a55;padding:10px 12px;border-radius:10px;display:none}

.modal{position:fixed;top:0;left:0;right:0;bottom:0;background:rgba(0,0,0,0.8);display:none;z-index:1000}

.modal-content{position:absolute;top:50%;left:50%;transform:translate(-50%,-50%);background:linear-gradient(145deg,#0b1425,#1a1f2e);border:1px solid #2a3a55;border-radius:20px;padding:32px;max-width:1000px;width:95%;min-height:400px;box-shadow:0 25px 50px -12px rgba(0,0,0,0.25);backdrop-filter:blur(10px)}

.close{position:absolute;top:15px;right:20px;font-size:28px;cursor:pointer;color:var(--muted);width:32px;height:32px;display:flex;align-items:center;justify-content:center;border-radius:50%;transition:all 0.3s ease;background:rgba(148,163,184,0.1)}

.close:hover{background:rgba(148,163,184,0.2);color:var(--text);transform:scale(1.1)}

.pricing{display:flex;gap:20px;justify-content:center;padding:0 10px;margin:24px 0;max-width:100%;overflow-x:auto}

.plan{background:linear-gradient(145deg,#0b1425,#1a1f2e);border:1px solid #2a3a55;border-radius:20px;padding:28px 24px;text-align:center;min-width:260px;max-width:300px;flex:1;flex-shrink:0;transition:all 0.4s cubic-bezier(0.4,0,0.2,1);box-shadow:0 4px 6px -1px rgba(0,0,0,0.1),0 2px 4px -1px rgba(0,0,0,0.06);overflow:hidden;position:relative}

.plan:hover{transform:translateY(-4px);box-shadow:0 20px 25px -5px rgba(0,0,0,0.1),0 10px 10px -5px rgba(0,0,0,0.04)}

.plan.premium{border-color:var(--premium);border-width:2px;background:linear-gradient(145deg,#1a1f2e,#1e2a3a);box-shadow:0 4px 6px -1px rgba(168,85,247,0.1),0 2px 4px -1px rgba(168,85,247,0.06)}

.plan.pro{border-color:var(--pro);border-width:2px;background:linear-gradient(145deg,#1a1f2e,#1e2a3a);box-shadow:0 4px 6px -1px rgba(34,197,94,0.1),0 2px 4px -1px rgba(34,197,94,0.06)}

.plan h3{color:var(--accent);margin:0 0 12px;font-size:22px;font-weight:800;letter-spacing:-0.025em;background:linear-gradient(135deg,var(--text),#cbd5e1);-webkit-background-clip:text;-webkit-text-fill-color:transparent;background-clip:text}

.plan .price{font-size:32px;font-weight:900;margin:20px 0;background:linear-gradient(135deg,var(--accent),#60a5fa);-webkit-background-clip:text;-webkit-text-fill-color:transparent;background-clip:text;letter-spacing:-0.02em}

.plan .price .period{font-size:16px;font-weight:500;color:var(--muted);-webkit-text-fill-color:var(--muted);background:none}

.plan .features{list-style:none;padding:0;margin:20px 0;text-align:left}

.plan .features li{padding:8px 0;color:var(--text);font-size:14px;line-height:1.6;display:flex;align-items:center}

.plan .features li:before{content:"✓";color:var(--accent);font-weight:900;margin-right:10px;font-size:16px;text-shadow:0 0 8px rgba(34,197,94,0.3);flex-shrink:0}

.plan .badge{position:absolute;top:-12px;right:-12px;background:linear-gradient(135deg,var(--accent),#60a5fa);color:#0b1425;padding:6px 12px;border-radius:16px;font-size:11px;font-weight:800;text-transform:uppercase;letter-spacing:0.5px;box-shadow:0 4px 8px rgba(0,0,0,0.2)}

.plan .badge.premium{background:linear-gradient(135deg,var(--premium),#f472b6)}

.plan .badge.pro{background:linear-gradient(135deg,var(--pro),#22c55e)}

.plan .description{color:var(--muted);font-size:13px;margin:8px 0 16px;line-height:1.5;font-style:italic}

.plan .popular{position:absolute;top:0;left:0;right:0;background:linear-gradient(135deg,var(--accent),#60a5fa);color:#0b1425;padding:8px;font-size:12px;font-weight:800;text-transform:uppercase;letter-spacing:0.5px;text-align:center}

.plan .features li:last-child{margin-bottom:0}

.plan .features li:first-child{margin-top:0}

.plan .price{margin-bottom:24px}

@media (max-width:768px){.pricing{flex-direction:column;align-items:center;gap:15px}.plan{min-width:280px;max-width:350px}.modal-content{max-width:95%;padding:15px}}

@media (max-width:1200px){.pricing{gap:15px}.plan{min-width:240px}}

@media (max-width:900px){.pricing{flex-direction:column;align-items:center;gap:15px}.plan{min-width:280px;max-width:350px}}

.price{font-size:32px;font-weight:900;margin:10px 0}

.features{text-align:left;margin:20px 0}

.features li{margin:8px 0;padding-left:20px;position:relative}

.features li:before{content:"✓";position:absolute;left:0;color:var(--accent)}

.btn-premium{background:var(--premium);border-color:var(--premium);color:#0b1425;font-weight:600;cursor:pointer;padding:12px 24px;border-radius:12px;font-size:14px;transition:all 0.3s ease;box-shadow:0 4px 6px -1px rgba(168,85,247,0.2)}

.btn-premium:hover{transform:translateY(-2px);box-shadow:0 8px 15px -3px rgba(168,85,247,0.3)}

.btn-pro{background:var(--pro);border-color:var(--pro);color:#0b1425;font-weight:600;cursor:pointer;padding:12px 24px;border-radius:12px;font-size:14px;transition:all 0.3s ease;box-shadow:0 4px 6px -1px rgba(34,197,94,0.2)}

.btn-pro:hover{transform:translateY(-2px);box-shadow:0 8px 15px -3px rgba(34,197,94,0.3)}

.upgrade-prompt{background:linear-gradient(135deg,var(--premium),var(--pro));border-radius:10px;padding:12px;margin:10px 0;text-align:center;color:#0b1425;font-weight:600}

.chart{height:200px;background:#0b1425;border:1px solid #2a3a55;border-radius:10px;display:flex;align-items:end;gap:2px;padding:20px 10px 10px;overflow:hidden}

.chart-bar{background:var(--accent);min-width:20px;border-radius:4px 4px 0 0}

.chart-bar.premium{background:var(--premium)}

.chart-bar.pro{background:var(--pro)}

.hidden{display:none}

</style>

</head>

<body>

<!-- Authentication Pages -->

<div id="authPages" class="auth-container">

<!-- Sign In Page -->

<div id="signInPage" class="auth-card">

<h1 class="auth-title">Welcome Back</h1>

<div id="signInError" class="error"></div>

<div id="signInSuccess" class="success"></div>

<form id="signInForm">

<div class="form-group">

<label for="signInEmail">Email</label>

<input type="email" id="signInEmail" required>

</div>

<div class="form-group">

<label for="signInPassword">Password</label>

<input type="password" id="signInPassword" required>

</div>

<button type="submit" class="btn-primary">Sign In</button>

</form>

<button class="btn-secondary" onclick="showSignUp()">Create New Account</button>

<div class="auth-switch">

Don't have an account? <a onclick="showSignUp()">Sign Up</a>

</div>

</div>

<!-- Sign Up Page -->

<div id="signUpPage" class="auth-card hidden">

<h1 class="auth-title">Create Account</h1>

<div id="signUpError" class="error"></div>

<div id="signUpSuccess" class="success"></div>

<form id="signUpForm">

<div class="form-group">

<label for="signUpUsername">Username</label>

<input type="text" id="signUpUsername" required>

</div>

<div class="form-group">

<label for="signUpEmail">Email</label>

<input type="email" id="signUpEmail" required>

</div>

<div class="form-group">

<label for="signUpPassword">Password</label>

<input type="password" id="signUpPassword" required minlength="8">

</div>

<div class="form-group">

<label for="signUpConfirmPassword">Confirm Password</label>

<input type="password" id="signUpConfirmPassword" required minlength="8">

</div>

<button type="submit" class="btn-primary">Create Account</button>

</form>

<button class="btn-secondary" onclick="showSignIn()">Back to Sign In</button>

<div class="auth-switch">

Already have an account? <a onclick="showSignIn()">Sign In</a>

</div>

</div>

</div>

<!-- Main App (Hidden until authenticated) -->

<div id="mainApp" class="hidden">

<header>

<div class="wrap">

<div class="top">

<div class="logo">₿</div>

<div>

<div style="font-weight:800">FinCoach Pro</div>

<div class="sub">Premium money coach with AI insights</div>

</div>

<div style="margin-left:auto">

<span class="pill" id="ageTag">Mode: Adult</span>

<span class="pill" id="planTag">Free Plan</span>

<select id="currencySelect" style="margin-left:8px;padding:6px 10px;border-radius:8px;font-size:13px">

<option value="USD">$ USD</option>

<option value="EUR">€ EUR</option>

<option value="GBP">£ GBP</option>

<option value="NGN">₦ NGN</option>

<option value="CAD">$ CAD</option>

<option value="AUD">$ AUD</option>

<option value="INR">₹ INR</option>

<option value="JPY">¥ JPY</option>

<option value="CNY">¥ CNY</option>

<option value="ZAR">R ZAR</option>

<option value="BRL">R$ BRL</option>

<option value="CHF">CHF</option>

<option value="SEK">kr SEK</option>

<option value="SGD">$ SGD</option>

<option value="HKD">$ HKD</option>

<option value="KRW">₩ KRW</option>

<option value="MXN">$ MXN</option>

<option value="RUB">₽ RUB</option>

<option value="TRY">₺ TRY</option>

<option value="PLN">zł PLN</option>

<option value="DKK">kr DKK</option>

<option value="NOK">kr NOK</option>

<option value="THB">฿ THB</option>

<option value="IDR">Rp IDR</option>

<option value="MYR">RM MYR</option>

<option value="PHP">₱ PHP</option>

<option value="TWD">NT$ TWD</option>

<option value="SAR">﷼ SAR</option>

<option value="AED">د.إ AED</option>

</select>

<button id="upgradeBtn" style="margin-left:8px">Upgrade</button>

<button id="logoutBtn" style="margin-left:8px">Logout</button>

</div>

</div>

</div>

</header>

<main class="wrap">

<!-- User Info -->

<div class="card">

<div class="row" style="justify-content:space-between;align-items:center">

<div>

<h3 style="margin:0">Welcome, <span id="userDisplayName">User</span>!</h3>

<div class="muted">Current Plan: <span id="currentPlan">Free</span></div>

</div>

<div class="row">

<button id="manageSub">Manage</button>

<button id="showPricing">View Plans</button>

</div>

</div>

</div>

<!-- Main Stats Grid -->

<div class="grid" style="grid-template-columns:repeat(3,1fr);gap:12px">

<div class="card">

<div class="stat">

<div>

<h3>Monthly Income</h3>

<div class="v" id="statIncome">$0</div>

</div>

<button id="editIncome">Edit</button>

</div>

<div class="muted" id="saveNote">Tip: Aim to save 20%+</div>

</div>

<div class="card">

<div class="stat">

<div>

<h3>Spent</h3>

<div class="v" id="statSpent">$0</div>

</div>

<span class="badge" id="spentPct">0%</span>

</div>

<div class="progress">

<div class="bar" id="spentBar" style="width:0%"></div>

</div>

</div>

<div class="card">

<div class="stat">

<div>

<h3>Estimated Savings</h3>

<div class="v" id="statSaved">$0</div>

</div>

<span class="badge" id="saveRate">0% rate</span>

</div>

<div class="muted">Income minus expenses</div>

</div>

</div>

<!-- Feature Grid -->

<div class="grid" style="margin-top:12px">

<!-- AI Coach -->

<div class="card">

<div class="row" style="justify-content:space-between;align-items:center">

<div class="pill">AI Coach</div>

<div class="row">

<button id="toggleAge">Switch to Teen Mode</button>

<button id="resetAll">Reset</button>

<button id="clearCoach">Clear</button>

</div>

</div>

<div id="coach" class="coach"></div>

</div>

<!-- Transaction Input -->

<div class="card">

<div class="pill">Add transaction</div>

<div class="row">

<input id="amt" type="number" min="0" step="0.01" placeholder="Amount" style="flex:1 1 120px"/>

<select id="cat" style="flex:1 1 160px">

<option>Dining</option>

<option>Groceries</option>

<option>Transport</option>

<option>Entertainment</option>

<option>Shopping</option>

<option>Bills</option>

<option>Other</option>

</select>

<input id="note" placeholder="Note (optional)" style="flex:2 1 200px"/>

</div>

<div class="row">

<button id="addExp">Add Expense</button>

<button id="addInc">Add Income</button>

</div>

<div id="liveTip" class="muted"></div>

<div class="list" id="tx"></div>

</div>

<!-- Budgets -->

<div class="card">

<div class="pill">Budgets</div>

<div id="budgets" class="list"></div>

<div class="row">

<button id="editBudgets">Edit Budgets</button>

<button id="resetBudgets">Reset Budgets</button>

</div>

</div>

<!-- Premium Analytics -->

<div class="card" id="analyticsCard">

<div class="pill premium">Premium Analytics</div>

<div id="upgradePrompt" class="upgrade-prompt">

Upgrade to Premium for advanced spending insights and trends

</div>

<div id="analyticsContent" style="display:none">

<div class="chart" id="spendingChart"></div>

<div class="muted">7-day spending trend</div>

</div>

</div>

<!-- Savings Challenges -->

<div class="card">

<div class="pill">Savings Challenges</div>

<div class="list">

<div class="item">

<div>

<strong>7-Day No Dining Out</strong>

<div class="muted">Cook at home for a week</div>

</div>

<div class="row">

<button class="cbtn" data-c="nodining">Log Day</button>

<span class="badge" id="c-nodining">0/7</span>

</div>

</div>

<div class="item">

<div>

<strong>Daily $5 Saver</strong>

<div class="muted">Save $5 per day (10 days)</div>

</div>

<div class="row">

<button class="cbtn" data-c="daily5">Log $5</button>

<span class="badge" id="c-daily5">$0/$50</span>

</div>

</div>

<div class="item">

<div>

<strong>Round-up Purchases</strong>

<div class="muted">Round each expense to next $</div>

</div>

<div class="row">

<button class="cbtn" data-c="roundup">Toggle</button>

<span class="badge" id="c-roundup">$0</span>

</div>

</div>

</div>

</div>

<!-- Investment Suggestions -->

<div class="card">

<div class="pill">Investment Suggestions</div>

<div class="row">

<input id="invIncome" type="number" min="0" step="1" placeholder="Monthly income" style="flex:1 1 160px"/>

<select id="risk" style="flex:1 1 140px">

<option value="low">Low</option>

<option value="medium" selected>Medium</option>

<option value="high">High</option>

</select>

<select id="horizon" style="flex:1 1 140px">

<option value="short">Short (0-2y)</option>

<option value="mid" selected>Mid (3-7y)</option>

<option value="long">Long (8y+)</option>

</select>

</div>

<div class="row">

<button id="ideasBtn">Get Suggestions</button>

</div>

<div id="ideas" class="list"></div>

<div class="muted">Educational only, not financial advice.</div>

</div>

</div>

<div class="muted" style="margin-top:8px">Data is stored locally in your browser. Premium features require subscription.</div>

</main>

</div>

<!-- Upgrade Modal -->

<div id="upgradeModal" class="modal">

<div class="modal-content">

<span class="close" onclick="closeModal('upgradeModal')">&times;</span>

<h2 style="text-align:center;margin-bottom:8px;font-size:28px;font-weight:800;background:linear-gradient(135deg,var(--text),#cbd5e1);-webkit-background-clip:text;-webkit-text-fill-color:transparent;background-clip:text">Choose Your Plan</h2>

<div class="muted" style="text-align:center;margin-bottom:24px;font-size:15px;line-height:1.6">All plans include core features. Premium and Pro add advanced capabilities.</div>

<div class="pricing">

<div style="text-align:center;margin-bottom:16px;color:var(--muted);font-size:13px;font-weight:500">← Scroll to see all plans →</div>

<div class="plan">

<h3>Free</h3>

<div class="description">Perfect for getting started</div>

<div class="price">$0<span class="period">/month</span></div>

<ul class="features">

<li>Basic expense tracking</li>

<li>Simple budgets</li>

<li>Basic savings challenges</li>

<li>General investment tips</li>

</ul>

<button class="btn-primary" onclick="selectPlan('free')" style="width:100%;padding:14px;font-weight:600;border-radius:12px;transition:all 0.3s ease">Current Plan</button>

</div>

<div class="plan premium">

<div class="badge premium">Popular</div>

<h3>Premium</h3>

<div class="description">For serious money management</div>

<div class="price">$9.99<span class="period">/month</span></div>

<ul class="features">

<li>Everything in Free</li>

<li>Advanced analytics & charts</li>

<li>AI-powered insights</li>

<li>Premium coaching</li>

<li>Spending trends</li>

<li>Priority support</li>

</ul>

<button class="btn-premium" onclick="selectPlan('premium')" style="width:100%;padding:14px;font-weight:600;border-radius:12px;transition:all 0.3s ease">Upgrade to Premium</button>

</div>

<div class="plan pro">

<div class="badge pro">Best Value</div>

<h3>Pro</h3>

<div class="description">Complete financial mastery</div>

<div class="price">$19.99<span class="period">/month</span></div>

<ul class="features">

<li>Everything in Premium</li>

<li>Tax optimization</li>

<li>Retirement planning</li>

<li>Portfolio tracking</li>

<li>Financial advisor access</li>

<li>Custom reports</li>

</ul>

<button class="btn-pro" onclick="selectPlan('pro')" style="width:100%;padding:14px;font-weight:600;border-radius:12px;transition:all 0.3s ease">Upgrade to Pro</button>

</div>

</div>

</div>

</div>

<!-- Payment Modal -->

<div id="paymentModal" class="modal">

<div class="modal-content">

<span class="close" onclick="closeModal('paymentModal')">&times;</span>

<h2>Complete Your Subscription</h2>

<div id="paymentForm">

<div class="row" style="flex-direction:column;gap:12px">

<input type="text" id="cardName" placeholder="Cardholder Name" required/>

<input type="text" id="cardNumber" placeholder="Card Number" maxlength="19" required/>

<div class="row">

<input type="text" id="expiry" placeholder="MM/YY" maxlength="5" required/>

<input type="text" id="cvv" placeholder="CVV" maxlength="4" required/>

</div>

<input type="email" id="email" placeholder="Email for receipt" required/>

<div class="form-group">

<label>

<input type="checkbox" id="termsCheck" required>

I agree to the <a href="#" onclick="showTerms()">Terms of Service</a> and <a href="#" onclick="showPrivacy()">Privacy Policy</a>

</label>

</div>

<button class="btn-premium" onclick="processPayment()" style="width:100%">Subscribe Now</button>

</div>

</div>

</div>

</div>

<div id="toast" class="toast"></div>

<script>

// Authentication System

let currentUser = null;

let isAuthenticated = false;

// Show/Hide pages

function showSignIn() {

document.getElementById('signInPage').classList.remove('hidden');

document.getElementById('signUpPage').classList.add('hidden');

}

function showSignUp() {

document.getElementById('signUpPage').classList.remove('hidden');

document.getElementById('signInPage').classList.add('hidden');

}

function showMainApp() {

document.getElementById('authPages').classList.add('hidden');

document.getElementById('mainApp').classList.remove('hidden');

}

function showAuth() {

document.getElementById('authPages').classList.remove('hidden');

document.getElementById('mainApp').classList.add('hidden');

}

// User Management

function createUser(username, email, password) {

const users = JSON.parse(localStorage.getItem('fc\_users') || '{}');

if (users[email]) {

throw new Error('User already exists');

}

const user = {

username,

email,

password: btoa(password), // Simple encoding (not secure for production)

plan: 'free',

subscription: null,

createdAt: new Date().toISOString()

};

users[email] = user;

localStorage.setItem('fc\_users', JSON.stringify(users));

return user;

}

function authenticateUser(email, password) {

const users = JSON.parse(localStorage.getItem('fc\_users') || '{}');

const user = users[email];

if (!user || user.password !== btoa(password)) {

throw new Error('Invalid credentials');

}

return user;

}

// Event Listeners

document.getElementById('signUpForm').addEventListener('submit', function(e) {

e.preventDefault();

const username = document.getElementById('signUpUsername').value;

const email = document.getElementById('signUpEmail').value;

const password = document.getElementById('signUpPassword').value;

const confirmPassword = document.getElementById('signUpConfirmPassword').value;

if (password !== confirmPassword) {

showError('signUpError', 'Passwords do not match');

return;

}

try {

const user = createUser(username, email, password);

currentUser = user;

isAuthenticated = true;

showSuccess('signUpSuccess', 'Account created successfully!');

setTimeout(() => {

showMainApp();

initializeApp();

}, 1000);

} catch (error) {

showError('signUpError', error.message);

}

});

document.getElementById('signInForm').addEventListener('submit', function(e) {

e.preventDefault();

const email = document.getElementById('signInEmail').value;

const password = document.getElementById('signInPassword').value;

try {

const user = authenticateUser(email, password);

currentUser = user;

isAuthenticated = true;

showSuccess('signInSuccess', 'Welcome back!');

setTimeout(() => {

showMainApp();

initializeApp();

}, 1000);

} catch (error) {

showError('signInError', error.message);

}

});

// Utility functions

function showError(elementId, message) {

const element = document.getElementById(elementId);

element.textContent = message;

element.style.display = 'block';

setTimeout(() => element.style.display = 'none', 3000);

}

function showSuccess(elementId, message) {

const element = document.getElementById(elementId);

element.textContent = message;

element.style.display = 'block';

setTimeout(() => element.style.display = 'none', 3000);

}

function logout() {

currentUser = null;

isAuthenticated = false;

showAuth();

// Clear form fields

document.getElementById('signInEmail').value = '';

document.getElementById('signInPassword').value = '';

document.getElementById('signUpUsername').value = '';

document.getElementById('signUpEmail').value = '';

document.getElementById('signUpPassword').value = '';

document.getElementById('signUpConfirmPassword').value = '';

}

// Check if user is already logged in

function checkAuthStatus() {

const savedUser = localStorage.getItem('fc\_currentUser');

if (savedUser) {

currentUser = JSON.parse(savedUser);

isAuthenticated = true;

showMainApp();

initializeApp();

}

}

// Initialize app when authenticated

function initializeApp() {

if (!currentUser) return;

// Update UI with user info

document.getElementById('userDisplayName').textContent = currentUser.username;

document.getElementById('currentPlan').textContent = currentUser.plan;

document.getElementById('planTag').textContent = currentUser.plan;

// Save current user to localStorage

localStorage.setItem('fc\_currentUser', JSON.stringify(currentUser));

// Initialize the main app functionality

initMainApp();

}

// Main App Logic

function initMainApp() {

// Initialize user-specific data

initUserData();

// Update UI with user info

document.getElementById('userDisplayName').textContent = currentUser.username;

document.getElementById('currentPlan').textContent = currentUser.plan;

document.getElementById('planTag').textContent = currentUser.plan;

// Initialize all the main app functionality

renderStats();

renderBudgets();

renderTx();

renderChallenges();

// Add event listeners for financial functionality

addFinancialEventListeners();

// Show welcome message

const coach = document.getElementById('coach');

if (coach) {

coach.innerHTML = `<div class="msg">Welcome to FinCoach Pro, ${currentUser.username}! Start by setting your income and adding some expenses.</div>`;

}

}

// Initialize user-specific data

function initUserData() {

const userKey = `fc\_${currentUser.email}`;

let userData = JSON.parse(localStorage.getItem(userKey) || '{}');

// Set default values if user data doesn't exist

if (!userData.month) {

const m = new Date();

const mk = m.getFullYear() + "-" + String(m.getMonth() + 1).padStart(2, "0");

userData = {

month: mk,

income: 0,

budgets: {Dining: 200, Groceries: 300, Transport: 120, Entertainment: 120, Shopping: 150, Bills: 400, Other: 100},

tx: [],

alerts: {},

age: "adult",

ch: {nodining: 0, daily5: 0, roundup: false, roundsaved: 0},

currency: "USD"

};

localStorage.setItem(userKey, JSON.stringify(userData));

}

if (!userData.currency) userData.currency = "USD";

// Store user data in global state

window.currentUserData = userData;

}

// Add all financial event listeners

function addFinancialEventListeners() {

// Edit Income

const editIncomeBtn = document.getElementById('editIncome');

if (editIncomeBtn) {

editIncomeBtn.onclick = function() {

const v = prompt("Your monthly income ($):", String(window.currentUserData.income || 0));

if (v === null) return;

window.currentUserData.income = Math.max(0, Number(v) || 0);

saveUserData();

renderStats();

coach("Income updated.");

};

}

// Add Expense

const addExpBtn = document.getElementById('addExp');

if (addExpBtn) {

addExpBtn.onclick = function() {

const a = Number(document.getElementById('amt').value || 0);

const c = document.getElementById('cat').value;

const n = document.getElementById('note').value.trim();

if (!(a > 0)) {

showToast("Enter an amount");

return;

}

window.currentUserData.tx.push({

type: "exp",

amount: a,

cat: c,

note: n,

date: new Date().toISOString()

});

recalcRoundup();

alertBudget(c);

saveUserData();

document.getElementById('amt').value = "";

document.getElementById('note').value = "";

rerender();

};

}

// Add Income

const addIncBtn = document.getElementById('addInc');

if (addIncBtn) {

addIncBtn.onclick = function() {

const v = prompt("Income amount to add ($):", "");

if (v === null) return;

const n = Number(v || 0);

if (!(n > 0)) {

showToast("Enter a positive number");

return;

}

window.currentUserData.income = (window.currentUserData.income || 0) + n;

saveUserData();

rerender();

coach("Income added. Consider moving 10–20% to savings first.");

};

}

// Edit Budgets

const editBudgetsBtn = document.getElementById('editBudgets');

if (editBudgetsBtn) {

editBudgetsBtn.onclick = function() {

const lines = Object.keys(window.currentUserData.budgets).map(c => c + "=" + window.currentUserData.budgets[c]).join("\n");

const v = prompt("Edit budgets (Category=Amount, one per line)", lines);

if (v === null) return;

const next = {};

v.split("\n").forEach(line => {

const p = line.split("=");

if (!p[0]) return;

const num = Number((p[1] || "").trim());

if (!isNaN(num)) next[p[0].trim()] = Math.max(0, num);

});

if (Object.keys(next).length) {

window.currentUserData.budgets = next;

window.currentUserData.alerts = {};

saveUserData();

renderBudgets();

renderStats();

coach("Budgets updated.");

}

};

}

// Reset Budgets

const resetBudgetsBtn = document.getElementById('resetBudgets');

if (resetBudgetsBtn) {

resetBudgetsBtn.onclick = function() {

if (!confirm("Reset budgets to defaults?")) return;

window.currentUserData.budgets = {Dining: 200, Groceries: 300, Transport: 120, Entertainment: 120, Shopping: 150, Bills: 400, Other: 100};

window.currentUserData.alerts = {};

saveUserData();

renderBudgets();

renderStats();

coach("Budgets reset.");

};

}

// Investment Suggestions

const ideasBtn = document.getElementById('ideasBtn');

if (ideasBtn) {

ideasBtn.onclick = function() {

ideas();

};

}

// Live tip for amount input

const amtInput = document.getElementById('amt');

if (amtInput) {

amtInput.addEventListener('input', live);

}

// Category change

const catSelect = document.getElementById('cat');

if (catSelect) {

catSelect.addEventListener('change', live);

}

// Challenge buttons

document.querySelectorAll('.cbtn').forEach(b => {

b.addEventListener('click', function() {

handleChallenge(b.getAttribute('data-c'));

});

});

// Toggle age mode

const toggleAgeBtn = document.getElementById('toggleAge');

if (toggleAgeBtn) {

toggleAgeBtn.onclick = function() {

window.currentUserData.age = window.currentUserData.age === "adult" ? "teen" : "adult";

saveUserData();

document.getElementById('ageTag').textContent = "Mode: " + (window.currentUserData.age === "adult" ? "Adult" : "Teen");

coach(window.currentUserData.age === "teen" ? "Teen mode: Short tips, small goals." : "Adult mode: Clear budgets and steady investing.");

};

}

// Reset all data

const resetAllBtn = document.getElementById('resetAll');

if (resetAllBtn) {

resetAllBtn.onclick = function() {

if (!confirm("This clears this month data. Continue?")) return;

window.currentUserData.tx = [];

window.currentUserData.alerts = {};

window.currentUserData.ch = {nodining: 0, daily5: 0, roundup: false, roundsaved: 0};

saveUserData();

rerender();

coach("Data cleared for this month.");

};

}

// Clear coach messages

const clearCoachBtn = document.getElementById('clearCoach');

if (clearCoachBtn) {

clearCoachBtn.onclick = function() {

const coachDiv = document.getElementById('coach');

if (coachDiv) {

coachDiv.innerHTML = '';

}

};

}

}

// Save user data to localStorage

function saveUserData() {

const userKey = `fc\_${currentUser.email}`;

localStorage.setItem(userKey, JSON.stringify(window.currentUserData));

}

// Financial calculation functions

function totals() {

const spent = window.currentUserData.tx.reduce((s, t) => s + (t.type === "exp" ? t.amount : 0), 0);

return {

spent,

income: window.currentUserData.income,

save: Math.max((window.currentUserData.income || 0) - spent, 0)

};

}

function spentByCat() {

const m = {};

Object.keys(window.currentUserData.budgets).forEach(c => m[c] = 0);

window.currentUserData.tx.forEach(t => {

if (t.type === "exp") m[t.cat] = (m[t.cat] || 0) + t.amount;

});

return m;

}

// Utility functions

const currencySymbols = {

USD: "$", EUR: "€", GBP: "£", NGN: "₦", CAD: "$", AUD: "$", INR: "₹", JPY: "¥", CNY: "¥", ZAR: "R", BRL: "R$", CHF: "CHF", SEK: "kr", SGD: "$", HKD: "$", KRW: "₩", MXN: "$", RUB: "₽", TRY: "₺", PLN: "zł", DKK: "kr", NOK: "kr", THB: "฿", IDR: "Rp", MYR: "RM", PHP: "₱", TWD: "NT$", SAR: "﷼", AED: "د.إ"

};

function dollars(n) {

const cur = window.currentUserData && window.currentUserData.currency ? window.currentUserData.currency : "USD";

const sym = currencySymbols[cur] || "$";

return sym + Number(n || 0).toFixed(2);

}

function coach(m) {

const d = document.createElement("div");

d.className = "msg";

d.textContent = m;

const coachDiv = document.getElementById('coach');

if (coachDiv) {

coachDiv.prepend(d);

}

}

// Render functions

function renderStats() {

const {spent, income, save} = totals();

const statIncome = document.getElementById('statIncome');

const statSpent = document.getElementById('statSpent');

const statSaved = document.getElementById('statSaved');

const spentPct = document.getElementById('spentPct');

const spentBar = document.getElementById('spentBar');

const saveRate = document.getElementById('saveRate');

const saveNote = document.getElementById('saveNote');

if (statIncome) statIncome.textContent = dollars(income);

if (statSpent) statSpent.textContent = dollars(spent);

if (statSaved) statSaved.textContent = dollars(save);

const rate = income ? Math.round(save / income \* 100) : 0;

if (saveRate) saveRate.textContent = rate + "% rate";

if (saveNote) saveNote.textContent = rate >= 20 ? "Great! You are at a healthy savings rate." : "Tip: Aim to save 20%+";

}

function renderBudgets() {

const budgetsDiv = document.getElementById('budgets');

if (!budgetsDiv) return;

budgetsDiv.innerHTML = "";

const used = spentByCat();

Object.keys(window.currentUserData.budgets).forEach(c => {

const wrap = document.createElement("div");

wrap.className = "item";

wrap.style.flexDirection = "column";

wrap.style.alignItems = "stretch";

const top = document.createElement("div");

top.style.display = "flex";

top.style.justifyContent = "space-between";

top.innerHTML = `<div><strong>${c}</strong> · <span class="muted">Budget ${dollars(window.currentUserData.budgets[c])}</span></div><span class="badge">${dollars(used[c] || 0)}</span>`;

const prog = document.createElement("div");

prog.className = "progress";

const inner = document.createElement("div");

inner.className = "bar";

const ratio = window.currentUserData.budgets[c] > 0 ? Math.min(100, Math.round((used[c] || 0) / window.currentUserData.budgets[c] \* 100)) : 100;

inner.style.width = ratio + "%";

if (ratio >= 80 && ratio < 100) inner.classList.add("warn");

if (ratio >= 100) inner.classList.add("danger");

prog.appendChild(inner);

wrap.appendChild(top);

wrap.appendChild(prog);

budgetsDiv.appendChild(wrap);

});

}

function renderTx() {

const txDiv = document.getElementById('tx');

if (!txDiv) return;

txDiv.innerHTML = "";

if (!window.currentUserData.tx.length) {

const i = document.createElement("div");

i.className = "muted";

i.textContent = "No transactions yet.";

txDiv.appendChild(i);

return;

}

window.currentUserData.tx.slice().reverse().forEach((t, i) => {

const it = document.createElement("div");

it.className = "item";

const left = document.createElement("div");

left.innerHTML = `<div><strong>${t.type === "exp" ? "− " : "+ "}${dollars(t.amount)}</strong> · ${t.cat}</div><div class="muted">${t.note || ""} · ${new Date(t.date).toLocaleDateString()}</div>`;

const r = document.createElement("div");

const del = document.createElement("button");

del.textContent = "Delete";

del.onclick = function() {

const idx = window.currentUserData.tx.length - 1 - i;

window.currentUserData.tx.splice(idx, 1);

recalcRoundup();

saveUserData();

rerender();

};

r.appendChild(del);

it.appendChild(left);

it.appendChild(r);

txDiv.appendChild(it);

});

}

function renderChallenges() {

const nodiningBadge = document.getElementById('c-nodining');

const daily5Badge = document.getElementById('c-daily5');

const roundupBadge = document.getElementById('c-roundup');

if (nodiningBadge) nodiningBadge.textContent = (window.currentUserData.ch.nodining || 0) + "/7";

if (daily5Badge) daily5Badge.textContent = dollars(window.currentUserData.ch.daily5 || 0) + `/${dollars(50)}`;

if (roundupBadge) roundupBadge.textContent = dollars(window.currentUserData.ch.roundsaved || 0);

}

function rerender() {

renderStats();

renderBudgets();

renderTx();

renderChallenges();

updatePlanCurrencySymbols();

if (document.getElementById('ageTag')) {

document.getElementById('ageTag').textContent = "Mode: " + (window.currentUserData.age === "adult" ? "Adult" : "Teen");

}

}

// Budget alert system

function alertBudget(cat) {

const used = spentByCat();

const budget = window.currentUserData.budgets[cat] || 0;

if (!budget) return;

const ratio = (used[cat] || 0) / budget;

const k80 = "80-" + cat;

const k100 = "100-" + cat;

if (ratio >= 1 && !window.currentUserData.alerts[k100]) {

coach(`You exceeded your ${cat} budget. Pause ${cat.toLowerCase()} for a week.`);

showToast("Budget exceeded: " + cat);

window.currentUserData.alerts[k100] = true;

} else if (ratio >= 0.8 && !window.currentUserData.alerts[k80]) {

coach(`You reached 80% of your ${cat} budget—consider cheaper options.`);

showToast("80% used: " + cat);

window.currentUserData.alerts[k80] = true;

}

}

// Live budget tip

function live() {

const a = Number(document.getElementById('amt').value || 0);

const c = document.getElementById('cat').value;

if (!a) {

const liveTip = document.getElementById('liveTip');

if (liveTip) liveTip.textContent = "";

return;

}

const used = spentByCat();

const budget = window.currentUserData.budgets[c] || 0;

const would = (used[c] || 0) + a;

if (!budget) {

const liveTip = document.getElementById('liveTip');

if (liveTip) liveTip.textContent = "No budget set yet for " + c;

return;

}

const r = would / budget;

const liveTip = document.getElementById('liveTip');

if (liveTip) {

if (r >= 1) liveTip.textContent = "This will exceed your " + c + " budget.";

else if (r >= 0.8) liveTip.textContent = "This puts you over 80% of " + c + " budget.";

else liveTip.textContent = "Looks good.";

}

}

// Roundup calculation

function recalcRoundup() {

if (!window.currentUserData.ch.roundup) {

window.currentUserData.ch.roundsaved = 0;

return;

}

let s = 0;

window.currentUserData.tx.forEach(t => {

if (t.type === "exp") {

const ceil = Math.ceil(t.amount);

s += ceil - t.amount;

}

});

window.currentUserData.ch.roundsaved = Number(s.toFixed(2));

}

// Challenge handling

function handleChallenge(code) {

if (code === "nodining") {

if (window.currentUserData.tx.some(t => t.cat === "Dining" && new Date(t.date).toDateString() === new Date().toDateString())) {

coach("You logged Dining today. Try again tomorrow.");

} else {

window.currentUserData.ch.nodining = Math.min(7, (window.currentUserData.ch.nodining || 0) + 1);

coach("Nice! No Dining today.");

}

}

if (code === "daily5") {

window.currentUserData.ch.daily5 = Math.min(50, (window.currentUserData.ch.daily5 || 0) + 5);

coach("Saved $5 (simulated). Keep it up!");

}

if (code === "roundup") {

window.currentUserData.ch.roundup = !window.currentUserData.ch.roundup;

recalcRoundup();

coach(window.currentUserData.ch.roundup ? "Round-ups enabled." : "Round-ups disabled.");

}

saveUserData();

renderChallenges();

renderStats();

}

// Investment suggestions

function ideas() {

const inc = Number(document.getElementById('invIncome').value || window.currentUserData.income || 0);

const risk = document.getElementById('risk').value;

const hor = document.getElementById('horizon').value;

const box = document.getElementById('ideas');

if (!box) return;

box.innerHTML = "";

function add(t) {

const d = document.createElement("div");

d.className = "msg";

d.textContent = t;

box.appendChild(d);

}

if (!(inc > 0)) {

add("Add your monthly income to tailor ideas.");

return;

}

add("1) Build an emergency fund (3–6 months) in high-yield savings.");

if (risk === "low") {

add("2) Consider broad market index funds with automatic contributions.");

} else if (risk === "medium") {

add("2) 80/20 mix: stock index + bonds/treasuries.");

} else {

add("2) Focus on diversified stock index funds.");

}

if (hor === "short") {

add("3) Short term goals: keep mostly in cash/T-bills.");

}

if (hor === "mid") {

add("3) Mid term: balanced index funds can smooth volatility.");

}

if (hor === "long") {

add("3) Long term: automate and stay consistent.");

}

coach("Investment ideas generated.");

}

// Modal functions

function showModal(id) {

document.getElementById(id).style.display = "block";

}

function closeModal(id) {

document.getElementById(id).style.display = "none";

}

function selectPlan(plan) {

if (plan === 'free') {

currentUser.plan = 'free';

localStorage.setItem('fc\_currentUser', JSON.stringify(currentUser));

closeModal('upgradeModal');

showToast('Plan updated to Free');

return;

}

// Show payment modal for paid plans

currentUser.plan = plan;

closeModal('upgradeModal');

showModal('paymentModal');

}

function processPayment() {

// Validate form

const cardName = document.getElementById('cardName').value;

const cardNumber = document.getElementById('cardNumber').value;

const expiry = document.getElementById('expiry').value;

const cvv = document.getElementById('cvv').value;

const email = document.getElementById('email').value;

const termsCheck = document.getElementById('termsCheck').checked;

if (!cardName || !cardNumber || !expiry || !cvv || !email || !termsCheck) {

showToast('Please fill in all fields and accept terms');

return;

}

// Simulate payment processing

showToast('Processing payment...');

setTimeout(() => {

// Simulate successful payment

currentUser.subscription = {

plan: currentUser.plan,

startDate: new Date().toISOString(),

status: 'active'

};

// Update user in localStorage

const users = JSON.parse(localStorage.getItem('fc\_users') || '{}');

users[currentUser.email] = currentUser;

localStorage.setItem('fc\_users', JSON.stringify(users));

localStorage.setItem('fc\_currentUser', JSON.stringify(currentUser));

closeModal('paymentModal');

showToast(`Welcome to ${currentUser.plan}! Your subscription is active.`);

// Update UI

document.getElementById('currentPlan').textContent = currentUser.plan;

document.getElementById('planTag').textContent = currentUser.plan;

// Clear form

document.getElementById('cardName').value = '';

document.getElementById('cardNumber').value = '';

document.getElementById('expiry').value = '';

document.getElementById('cvv').value = '';

document.getElementById('email').value = '';

document.getElementById('termsCheck').checked = false;

}, 2000);

}

function showToast(message) {

const toast = document.getElementById('toast');

toast.textContent = message;

toast.style.display = 'block';

setTimeout(() => toast.style.display = 'none', 3000);

}

// Event listeners for main app

document.addEventListener('DOMContentLoaded', function() {

// Check authentication status

checkAuthStatus();

// Add logout button listener

const logoutBtn = document.getElementById('logoutBtn');

if (logoutBtn) {

logoutBtn.addEventListener('click', logout);

}

// Add upgrade button listener

const upgradeBtn = document.getElementById('upgradeBtn');

if (upgradeBtn) {

upgradeBtn.addEventListener('click', () => showModal('upgradeModal'));

}

// Add pricing button listener

const showPricingBtn = document.getElementById('showPricing');

if (showPricingBtn) {

showPricingBtn.addEventListener('click', () => showModal('upgradeModal'));

}

// Add manage subscription button listener

const manageSubBtn = document.getElementById('manageSub');

if (manageSubBtn) {

manageSubBtn.addEventListener('click', () => {

if (currentUser.plan === 'free') {

showModal('upgradeModal');

} else {

alert(`Current plan: ${currentUser.plan}\nTo change plans, contact support.`);

}

});

}

});

// Close modals when clicking outside

window.onclick = function(event) {

if (event.target.classList.contains('modal')) {

event.target.style.display = "none";

}

};

// Terms and Privacy functions

function showTerms() {

alert('Terms of Service: This is a demo application. No real financial advice is provided.');

}

function showPrivacy() {

alert('Privacy Policy: Your data is stored locally in your browser. We do not collect personal information.');

}

// Add this to rerender() to update the plan/pricing area currency symbols

function updatePlanCurrencySymbols() {

const cur = window.currentUserData && window.currentUserData.currency ? window.currentUserData.currency : "USD";

const sym = currencySymbols[cur] || "$";

// Free plan

const freePrice = document.querySelector('.plan .price');

if (freePrice) freePrice.innerHTML = `${sym}0<span class="period">/month</span>`;

// Premium plan

const premiumPrice = document.querySelector('.plan.premium .price');

if (premiumPrice) premiumPrice.innerHTML = `${sym}9.99<span class="period">/month</span>`;

// Pro plan

const proPrice = document.querySelector('.plan.pro .price');

if (proPrice) proPrice.innerHTML = `${sym}19.99<span class="period">/month</span>`;

// Challenges (e.g., $50)

const daily5Badge = document.getElementById('c-daily5');

if (daily5Badge) daily5Badge.textContent = dollars(window.currentUserData.ch.daily5 || 0) + `/${sym}50`;

}

</script>

</body>

</html>