. ARQUIVOS HTML .

<!-- INDEX Comum para todos os steakholders (admin,cliente,funcionário)-->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../css/styles.css" />

  <link href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=swap" rel="stylesheet">

  <link href="https://fonts.googleapis.com/css2?family=Archivo+Black&display=swap" rel="stylesheet">

  <!-- Firebase (v8 compatível) -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-firestore.js"></script>

  <!-- Scripts personalizados -->

  <script defer src="../js/firebase-config.js"></script>

  <script defer src="../js/login.js"></script>

</head>

<body>

  <main class="main-wrap index">

    <!-- Marca / Logo -->

    <div class="brand">

      <img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

    </div>

    <!-- Texto de introdução -->

    <p class="login-intro">

      Para agendar seu horário, faça o login, ou crie sua conta.

    </p>

    <!-- FORMULÁRIOS COM FUNDO -->

  <div class="form-wrap">

    <!-- Caixa de LOGIN -->

    <div class="form-card" id="boxLogin">

      <form id="formLogin">

        <input type="email" id="loginEmail" class="input-field" placeholder="Digite seu email" required>

        <input type="password" id="loginSenha" class="input-field" placeholder="Digite sua senha" required>

        <!-- Botão EMAIL -->

        <button type="submit" id="btnLogin" class="btn-cta">

          <span class="icon">✉️</span>

          ENTRAR COM O EMAIL

        </button>

        <!-- Botão GOOGLE -->

        <button type="button" id="btnGoogle" class="btn-cta btn-google">

          <img class="g-logo" src="https://www.gstatic.com/firebasejs/ui/2.0.0/images/auth/google.svg" alt="Google">

          ENTRAR COM O Google

        </button>

      </form>

      <div class="login-links">

        <a href="#" id="esqueciSenha">Esqueci minha senha</a>

        <a href="#" id="mostrarCadastro">Criar conta</a>

      </div>

      <p id="loginErro"></p>

    </div>

    <!-- Caixa de CADASTRO (inicialmente escondida) -->

    <div id="boxCadastro" style="display:none;">

      <form id="formCadastro">

        <input type="text" id="cadNome" class="input-field" placeholder="Nome completo" required>

        <input type="tel" id="cadTelefone" class="input-field" placeholder="Telefone" required>

        <input type="email" id="cadEmail" class="input-field" placeholder="Email" required>

        <input type="password" id="cadSenha" class="input-field" placeholder="Senha" required>

        <input type="date" id="cadNascimento" class="input-field" placeholder="Data de nascimento (opcional)">

        <button type="button" id="btnCadastro" class="btn-cta">CADASTRAR</button>

      </form>

      <div class="login-links">

        <a href="#" id="mostrarLogin">Já tenho conta</a>

      </div>

      <p id="cadastroErro"></p>

    </div>

  </div>

     <!-- Rodapé -->

  <footer>

    <p>Rua Hercília 1160, Vila Emil - Mesquita/RJ</p>

    <p>&copy; 2025 Barbearia Carreiro</p>

  </footer>

  </main>

  <!-- Teste de Firestore (pode remover depois) -->

  <script>

    var firebaseConfig = {

        apiKey: "AIzaSyAEQMwHWQuPKybroJM4UK2\_mJWJESYm-a4",

        authDomain: "barbearia-carreiro.firebaseapp.com",

        projectId: "barbearia-carreiro",

        storageBucket: "barbearia-carreiro.appspot.com",

        messagingSenderId: "54089800722",

        appId: "1:54089800722:web:0221369454bfbb71c24ccb",

        measurementId: "G-4L0ZK72L4L"

     };

    firebase.initializeApp(firebaseConfig);

    const db = firebase.firestore();

    // teste de conexão

    db.collection("teste").add({

      nome:"Teste de conexão",

      criadoEm:new Date()

    }).then(()=>console.log("Banco conectado com sucesso!"))

      .catch(err=>console.error("Erro Firestore:", err));

  </script>

</body>

</html>

<!-- HOME do cliente -->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Home – Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../css/styles.css?v=4" />

  <link href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=swap" rel="stylesheet">

  <link href="https://fonts.googleapis.com/css2?family=Archivo+Black&display=swap" rel="stylesheet">

  <!-- Firebase (v8 compatível) -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-firestore.js"></script>

  <!-- Config e JS da página -->

  <script defer src="../js/firebase-config.js"></script>

  <script defer src="../js/home.js"></script>

</head>

<body>

  <main class="main-wrap home">

    <!-- Marca / Logo -->

    <div class="brand">

      <img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

    </div>

    <!-- Saudação -->

    <p class="home-greet">Olá, <span id="nomeCliente">(nome)</span>!</p>

    <!-- Texto introdutório -->

    <p class="home-text">

      Para melhor atendê-lo, trabalhamos com agendamento.

      Reserve o melhor horário!

    </p>

    <!-- Ações (sem card de fundo) -->

    <div class="home-actions">

    <button id="btnAgendar" class="btn-cta btn-primary">

        Agendar agora

    </button>

    <button id="btnHistorico" class="btn-cta btn-primary">

        Histórico

    </button>

    </div>

    <!-- Endereço -->

    <address class="home-address">

      Rua Hercília , 1160, Vila Emil - Mesquita/RJ

    </address>

    <!-- Rodapé dentro do main para manter alinhamento -->

    <footer>© Barbearia Carreiro</footer>

  </main>

</body>

</html>

<!-- historico.html (historico do cliente)-->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Histórico – Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../css/styles.css?v=7" />

  <link href="https://fonts.googleapis.com/css2?family=Open+Sans:wght@700&display=swap" rel="stylesheet">

  <link href="https://fonts.googleapis.com/css2?family=Archivo+Black&display=swap" rel="stylesheet">

  <!-- Firebase (v8 compatível) -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-firestore.js"></script>

  <!-- Config + JS da página -->

  <script defer src="../js/firebase-config.js"></script>

  <script defer src="../js/historico.js"></script>

</head>

<body>

  <main class="main-wrap history">

    <!-- include:brand -->

    <div class="brand">

      <img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

    </div>

    <h2 class="history-title">Histórico de agendamentos:</h2>

    <!-- Lista -->

    <ul id="historyList" class="history-list" aria-live="polite">

      <!-- itens inseridos pelo JS -->

    </ul>

    <!-- Botão voltar central -->

    <button id="btnVoltar" class="back-btn" aria-label="Voltar para a Home" title="Voltar">

      <!-- ícone padrão (seta) -->

      &#8592;

    </button>

    <!-- include:footer -->

    <footer>Rua Hercília , 1160, Vila Emil - Mesquita/RJ<br>© Barbearia Carreiro</footer>

  </main>

</body>

</html>

<!-- agendar-servico.html (Etapa 1 – serviço - cliente) -->

<!DOCTYPE html><html lang="pt-BR"><head>

<meta charset="UTF-8" /><meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Agendar – Serviço</title>

<link rel="stylesheet" href="../css/styles.css?v=8" />

<script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

<script defer src="../js/agendamento.js"></script>

</head><body>

<main class="main-wrap booking">

  <!-- include:brand -->

  <div class="brand"><img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

  <h2 class="booking-title">Escolha o serviço:</h2>

  <div class="booking-sub">APENAS 1 SERVIÇO</div>

  <ul id="serviceList" class="select-list"></ul>

  <div class="booking-nav">

    <button id="goBack"  class="btn-round" title="Voltar">&#8592;</button>

    <button id="goNext"  class="btn-round confirm" title="Confirmar">&#10003;</button>

  </div>

  <!-- include:footer -->

  <footer>Rua Hercília , 1160, Vila Emil - Mesquita/RJ<br>© Barbearia Carreiro</footer>

</main>

<script>document.addEventListener("DOMContentLoaded", BookingSteps.initStep1);</script>

</body></html>

<!-- agendar-profissinal.html (Etapa 2 – profissional - cliente) -->

<!DOCTYPE html><html lang="pt-BR"><head>

<meta charset="UTF-8" /><meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Agendar – Profissional</title>

<link rel="stylesheet" href="../css/styles.css?v=8" />

<script defer src="../js/agendamento.js"></script>

</head><body>

<main class="main-wrap booking">

  <!-- include:brand -->

  <div class="brand"><img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

  <h2 class="booking-title">Escolha o profissional:</h2>

  <div class="booking-sub">APENAS 1 OPÇÃO</div>

  <ul id="proList" class="select-list"></ul>

  <div class="booking-nav">

    <button id="goBack" class="btn-round" title="Voltar">&#8592;</button>

    <button id="goNext" class="btn-round confirm" title="Confirmar">&#10003;</button>

  </div>

  <!-- include:footer -->

  <footer>Rua Hercília , 1160, Vila Emil - Mesquita/RJ<br>© Barbearia Carreiro</footer>

</main>

<script>document.addEventListener("DOMContentLoaded", BookingSteps.initStep2);</script>

</body></html>

<!-- agendar-horario.html (Etapa 3 – dia/horário - cliente) -->

<!DOCTYPE html><html lang="pt-BR"><head>

<meta charset="UTF-8" /><meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Agendar – Dia e Horário</title>

<link rel="stylesheet" href="../css/styles.css?v=8" />

<script defer src="../js/agendamento.js"></script>

</head><body>

<main class="main-wrap booking">

  <!-- include:brand -->

  <div class="brand"><img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

  <h2 class="booking-title">Escolha o DIA e o HORÁRIO:</h2>

  <div class="calendar-wrap">

    <div id="monthLabel" class="month-label"></div>

    <div id="daysBar" class="days-bar"></div>

  </div>

  <div class="slots">

    <div class="slot-section">

      <div class="slot-header"><span>Manhã</span><span id="morningCount"></span></div>

      <div id="slotsMorning" class="slot-grid"></div>

    </div>

    <div class="slot-section">

      <div class="slot-header"><span>Tarde</span><span id="afternoonCount"></span></div>

      <div id="slotsAfternoon" class="slot-grid"></div>

    </div>

  </div>

  <div class="booking-nav">

    <button id="goBack" class="btn-round" title="Voltar">&#8592;</button>

    <button id="goNext" class="btn-round confirm" title="Confirmar">&#10003;</button>

  </div>

  <!-- include:footer -->

  <footer>Rua Hercília , 1160, Vila Emil - Mesquita/RJ<br>© Barbearia Carreiro</footer>

</main>

<script>document.addEventListener("DOMContentLoaded", BookingSteps.initStep3);</script>

</body></html>

<!-- agendar-confirmar.html (Etapa 4 – revisão + observação + modal - cliente) -->

<!DOCTYPE html><html lang="pt-BR"><head>

<meta charset="UTF-8" /><meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Agendar – Confirmar</title>

<link rel="stylesheet" href="../css/styles.css?v=8" />

<script defer src="../js/agendamento.js"></script>

</head><body>

<main class="main-wrap booking">

  <!-- include:brand -->

  <div class="brand"><img src="../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

  <p class="home-greet">Olá, <span id="nomeCliente">(nome)</span>!</p>

  <div class="confirm-card">

    <div class="confirm-row"><div class="key">Serviço</div>     <div id="cf-service" class="val"></div></div>

    <div class="confirm-row"><div class="key">Profissional</div><div id="cf-pro"     class="val"></div></div>

    <div class="confirm-row"><div class="key">Data</div>        <div id="cf-date"   class="val"></div></div>

    <div class="confirm-row"><div class="key">Horário</div>     <div id="cf-time"   class="val"></div></div>

    <div style="margin-top:8px;">

      <label><input type="checkbox" id="hasObs" /> Alguma observação?</label>

      <textarea id="obs" class="input-field" style="margin-top:8px; text-transform:none; display:none;"

        placeholder="Ex: Não precisa lavar, etc."></textarea>

      <div class="tolerance">⏱ <strong>15 min de tolerância</strong> após o horário marcado.</div>

    </div>

  </div>

  <div class="booking-nav">

    <button id="goBack" class="btn-round" title="Voltar">&#8592;</button>

    <button id="goNext" class="btn-round confirm" title="Confirmar">✔</button>

  </div>

  <!-- include:footer -->

  <footer>Rua Hercília , 1160, Vila Emil - Mesquita/RJ<br>© Barbearia Carreiro</footer>

</main>

<!-- Modal de sucesso -->

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

  <div class="box">

    <button class="close" title="Fechar">×</button>

    <div class="ok">✅</div>

    <h3 style="margin:6px 0 6px; font-size:22px;">Agendamento confirmado!</h3>

  </div>

</div>

<script>

  document.addEventListener("DOMContentLoaded", ()=>{

    BookingSteps.initStep4();

    const cb  = document.getElementById("hasObs");

    const txt = document.getElementById("obs");

    cb.addEventListener("change", ()=> txt.style.display = cb.checked? "block":"none");

  });

</script>

</body></html>

  <div id="modalConfig" class="modal hidden" aria-hidden="true">

<!-- Agenda Admin (pagina inicial do admin)-->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Agenda — Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../../css/admin.css?v=6" />

  <!-- Progressive Web App (PWA) -->

  <link rel="manifest" href="/manifest.json">

  <!-- (opcional) Firebase, se quiser saudar pelo Auth depois -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

<script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

<script defer src="../../js/firebase-config.js"></script>

  <!-- Marca item ativo do menu -->

  <script defer src="../../js/admin-nav.js"></script>

  <!-- Lógica da Agenda -->

  <script defer src="../../js/agenda.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#1E90FF" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

        <a class="nav-item" data-color="#F59E0B" href="./clientes.html"><span class="dot"></span>Clientes</a>

      </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <h2 class="view-title" style="font-size:32px; color:#e0b25b; text-transform:lowercase">agenda</h2>

      <!-- TOPO / FILTROS RÁPIDOS -->

      <section class="panel agenda-top">

        <div class="agenda-left">

          <button id="btnPrev" class="btn btn-ghost">◀ anterior</button>

          <button id="btnHoje" class="btn btn-blue">hoje</button>

          <button id="btnNext" class="btn btn-ghost">próximo ▶</button>

        </div>

        <div class="agenda-mid">

          <label>Data</label>

          <input id="agendaData" type="date" />

        </div>

        <div class="agenda-right">

          <label>Profissional</label>

          <select id="filtroProf"><option value="\*">Todos</option></select>

        </div>

      </section>

      <!-- GRID DE COLUNAS POR FUNCIONÁRIO -->

      <section id="agendaGrid" class="agenda-grid">

        <!-- preenchido via JS -->

      </section>

      <!-- BOTÃO FLUTUANTE “+” (novo agendamento futuro) -->

      <button id="btnNovoAg" class="fab" title="Novo agendamento">+</button>

    </main>

  </div>

  <!-- MODAL: CONFIGURAR AGENDA DO FUNCIONÁRIO -->

  <div id="modalConfig" class="modal hidden" aria-hidden="true">

    <div class="modal-card">

      <div class="modal-head">

        <h3 id="configTitle">Configurar agenda</h3>

        <button class="modal-close" id="closeConfig">✕</button>

      </div>

      <!-- Ajuda -->

      <p class="muted" style="margin:6px 0 10px">

        Clique nos horários para <strong>liberar</strong> (verde) ou <strong>bloquear</strong> (cinza).

        Intervalo fixo de 30min — de <strong>10:00</strong> até <strong>20:00</strong>, <em>segunda a domingo</em>.

      </p>

      <div id="dayMonth" class="day-month"></div>

      <div id="dayStrip" class="day-strip"></div>

      <div id="weekMatrix" class="week-matrix"></div>

      <!-- Barra de dias (30 dias) -->

      <div id="dayStrip" class="day-strip" role="tablist" aria-label="Selecionar dia específico"></div>

      <!-- Semana x Horas -->

      <div id="weekMatrix" class="week-matrix">

        <!-- JS monta a tabela: colunas = seg–dom, linhas = 10:00…20:00 -->

      </div>

      <div class="modal-actions">

        <button id="saveConfig" class="btn btn-green">Salvar</button>

        <button id="cancelConfig" class="btn btn-ghost">Cancelar</button>

      </div>

    </div>

  </div>

  <script>

  if ('serviceWorker' in navigator) {

    window.addEventListener('load', () => {

      navigator.serviceWorker.register('/sw.js').then(registration => {

        console.log('ServiceWorker registrado com sucesso: ', registration.scope);

      }).catch(error => {

        console.log('Falha no registro do ServiceWorker: ', error);

      });

    });

  }

</script>

</body>

</html>

<!-- cadastro de clientes( admin) -->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Clientes — Barbearia Carreiro</title>

  <!-- Estilos base do admin -->

  <link rel="stylesheet" href="../../css/admin.css?v=6" />

  <!-- (se quiser herdar algo global, pode incluir styles.css também) -->

  <!-- (Opcional) Firebase para futura integração de auth -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script defer src="../../js/firebase-config.js"></script>

  <!-- Marca o item ativo do menu pela URL -->

  <script defer src="../../js/admin-nav.js"></script>

  <!-- Lógica desta página -->

  <script defer src="../../js/clientes.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <!-- MENU (mantenha igual em todas as páginas do admin) -->

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

      </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>Clientes</h1>

      </header>

      <!-- Barra de busca + utilitários -->

      <section class="panel">

        <div class="svc-utils">

          <div class="search">

            <label for="cliQ" style="margin:0">Busca:</label>

            <input id="cliQ" type="text" placeholder="Nome, telefone, e-mail…" />

          </div>

          <div class="svc-util-btns">

            <button id="cliExport" class="btn btn-blue">Exportar JSON</button>

            <label class="btn btn-ghost">Importar JSON

              <input id="cliImport" type="file" accept="application/json" hidden />

            </label>

            <button id="cliZerar" class="btn btn-red">Zerar tudo</button>

          </div>

        </div>

        <div class="table-wrap">

          <table class="table" id="cliTable">

            <thead>

              <tr>

                <th>Foto</th>

                <th>Nome</th>

                <th>Telefone</th>

                <th class="hide-sm">E-mail</th>

                <th class="hide-sm">Nascimento</th>

                <th>Combos ativos</th>

                <th style="width:160px">Ações</th>

              </tr>

            </thead>

            <tbody id="cliTbody"></tbody>

          </table>

        </div>

      </section>

      <!-- FAB (novo cliente) -->

      <button id="btnFabAddCli" class="fab" title="Novo cliente" aria-label="Novo cliente">+</button>

      <!-- MODAL: Novo/Editar Cliente -->

      <div id="cliModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="cliTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="cliTitle">Novo cliente</h3>

          <!-- [CRUD CLIENTES] -->

          <form id="cliForm">

            <input type="hidden" id="cliId" />

            <div class="svc-row">

              <div>

                <label for="cNome">Nome \*</label>

                <input id="cNome" type="text" required />

              </div>

              <div>

                <label for="cTel">Telefone \*</label>

                <input id="cTel" type="text" placeholder="(21) 9XXXX-XXXX" required />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="cEmail">E-mail</label>

                <input id="cEmail" type="email" placeholder="nome@dominio.com" />

              </div>

              <div>

                <label for="cNasc">Nascimento</label>

                <input id="cNasc" type="date" />

              </div>

            </div>

            <!-- Upload de foto só aparece ao editar (novo não exige foto) -->

            <div id="fotoBox" class="svc-photo" hidden>

              <div class="svc-preview" id="fotoPreview">FOTO</div>

              <div style="flex:1">

                <label for="cFoto">Foto (PNG/JPG)</label>

                <input id="cFoto" type="file" accept="image/\*" />

              </div>

            </div>

            <p class="muted" style="margin-top:8px">

              Observação de segurança: senha não é exibida aqui. No backend/auth, marque <code>mustChangePassword=true</code> no primeiro acesso.

            </p>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- MODAL: Confirmação de exclusão -->

      <div id="cliDelModal" class="modal" aria-hidden="true">

        <div class="modal-card">

          <button class="modal-close" data-close>&times;</button>

          <h3>Excluir cliente</h3>

          <p>Tem certeza que deseja excluir este cliente?</p>

          <div class="svc-actions">

            <button id="cliConfirmDel" class="btn btn-red">Excluir</button>

            <button class="btn btn-ghost" data-close>Cancelar</button>

          </div>

        </div>

      </div>

      <!-- MODAL: Histórico de agendamentos (somente leitura) -->

      <div id="histModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="histTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="histTitle">Histórico de agendamentos</h3>

          <div id="histList" class="table-wrap" style="margin-top:10px">

            <!-- Renderizado via JS como tabela simples -->

          </div>

        </div>

      </div>

    </main>

  </div>

</body>

</html>

<!-- dashboard (admin)-->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Painel — Barbearia Carreiro</title>

    <!-- Estilos -->

    <link rel="stylesheet" href="../../css/admin.css?v=3" />

    <!-- Se quiser herdar algo global do site, descomente: -->

    <!-- <link rel="stylesheet" href="../../css/styles.css?v=admin1" /> -->

    <!-- Firebase (opcional por enquanto) -->

    <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

    <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

    <script defer src="../../js/firebase-config.js"></script>

    <!-- Navegação do admin -->

    <script defer src="../../js/admin-nav.js"></script>

    <!-- Lógica do painel -->

    <script defer src="../../js/admin.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

        </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>Bem-vindo, <span id="adminName">Admin</span> 👋</h1>

      </header>

      <!-- DASHBOARD (apenas esta view nesta página) -->

      <section class="view show" id="view-dashboard" aria-labelledby="h-dashboard">

        <h2 id="h-dashboard" class="view-title">📊 Dashboard – Contabilidade Completa</h2>

        <!-- KPIs -->

        <div class="kpi-grid">

          <div class="kpi">

            <div class="kpi-label">Receita (mês/ano)</div>

            <div class="kpi-value" id="kpiReceita">R$ 0,00</div>

            <div class="kpi-sub">Serviços + Produtos</div>

          </div>

          <div class="kpi">

            <div class="kpi-label">Despesas (mês/ano)</div>

            <div class="kpi-value" id="kpiDespesas">R$ 0,00</div>

            <div class="kpi-sub">Fixas + Variáveis + Taxas</div>

          </div>

          <div class="kpi">

            <div class="kpi-label">Lucro líquido</div>

            <div class="kpi-value" id="kpiLucro">R$ 0,00</div>

            <div class="kpi-sub">Receita – Despesas</div>

          </div>

          <div class="kpi">

            <div class="kpi-label">Ticket médio</div>

            <div class="kpi-value" id="kpiTicket">R$ 0,00</div>

            <div class="kpi-sub">Receita ÷ Atendimentos</div>

          </div>

        </div>

        <!-- 3 colunas: gráficos / listas -->

        <div class="panel-grid">

          <div class="panel">

            <div class="panel-title">Formas de pagamento</div>

            <div class="chart-placeholder" id="chartPagamentos" aria-label="Gráfico de pizza"></div>

          </div>

          <div class="panel">

            <div class="panel-title">Top serviços</div>

            <ul class="rank-list" id="rankServicos"></ul>

          </div>

          <div class="panel">

            <div class="panel-title">Top produtos</div>

            <ul class="rank-list" id="rankProdutos"></ul>

          </div>

        </div>

        <div class="panel-grid">

          <div class="panel">

            <div class="panel-title">Entradas recentes</div>

            <div class="table-wrap">

              <table class="table" id="tabEntradas">

                <thead>

                  <tr><th>Data</th><th>Tipo</th><th>Descrição</th><th>Qtd</th><th>Forma</th><th>Taxa</th><th>Líquido</th></tr>

                </thead>

                <tbody></tbody>

              </table>

            </div>

          </div>

          <div class="panel">

            <div class="panel-title">Saídas recentes</div>

            <div class="table-wrap">

              <table class="table" id="tabSaidas">

                <thead>

                  <tr><th>Data</th><th>Descrição</th><th>Categoria</th><th>Valor</th><th>Forma</th><th>Situação</th></tr>

                </thead>

                <tbody></tbody>

              </table>

            </div>

          </div>

          <div class="panel">

            <div class="panel-title">Contas a pagar</div>

            <div class="table-wrap">

              <table class="table" id="tabPagar">

                <thead>

                  <tr><th>Vencimento</th><th>Descrição</th><th>Categoria</th><th>Valor</th><th>Status</th></tr>

                </thead>

                <tbody></tbody>

              </table>

            </div>

          </div>

        </div>

        <div class="panel">

          <div class="panel-title">Indicadores</div>

          <div class="indicators">

            <div><strong>Lucro bruto</strong>: <span id="indBruto">R$ 0,00</span></div>

            <div><strong>Lucro líquido</strong>: <span id="indLiquido">R$ 0,00</span></div>

            <div><strong>Margem de lucro</strong>: <span id="indMargem">0%</span></div>

            <div><strong>Saldo de caixa hoje</strong>: <span id="indSaldo">R$ 0,00</span></div>

          </div>

        </div>

        <div class="panel">

          <div class="panel-title">Relatórios</div>

          <div class="reports-grid">

            <div class="chart-placeholder" id="chartReceitaServicos" aria-label="Gráfico de barras"></div>

            <div class="chart-placeholder" id="chartEvolucao" aria-label="Linha do tempo"></div>

            <div class="table-wrap">

              <table class="table" id="tabRankProdutos">

                <thead><tr><th>Produto</th><th>Qtd</th><th>Receita</th></tr></thead>

                <tbody></tbody>

              </table>

            </div>

          </div>

        </div>

      </section>

      <!-- /DASHBOARD -->

    </main>

  </div>

</body>

</html>

<!-- funcionarios (admin)-->  
  
<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Funcionários — Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../../css/admin.css?v=4" />

  <!-- (Opcional) Firebase futuro -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script defer src="../../js/firebase-config.js"></script>

  <!-- Navegação do admin + Lógica da página -->

  <script defer src="../../js/admin-nav.js"></script>

  <script defer src="../../js/funcionarios.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

        </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>Funcionários</h1>

      </header>

      <!-- ===== Tabela + ações globais ===== -->

      <section class="panel">

        <div class="svc-utils">

          <div class="search">

            <label for="empQ" style="margin:0">Busca:</label>

            <input id="empQ" type="text" placeholder="Nome, função, telefone, contrato, acesso…" />

          </div>

          <div class="svc-util-btns">

            <button id="empExport" class="btn btn-blue">Exportar JSON</button>

            <label class="btn btn-ghost">Importar JSON

              <input id="empImportFile" type="file" accept="application/json" hidden />

            </label>

            <button id="empZerar" class="btn btn-red">Zerar tudo</button>

          </div>

        </div>

        <div class="table-wrap">

          <table id="empTable" class="table">

            <thead>

              <tr>

                <th>Foto</th>

                <th>Nome</th>

                <th>Função</th>

                <th>Telefone</th>

                <th class="hide-sm">Endereço</th>

                <th>Contrato</th>

                <th>Comissão</th>

                <th class="hide-sm">Salário fixo</th>

                <th>Acesso</th>

                <th>ID</th>

                <th>Senha</th>

                <th style="width:96px">Ações</th>

              </tr>

            </thead>

            <tbody id="empTbody"><!-- linhas via JS --></tbody>

          </table>

        </div>

      </section>

      <!-- FAB: adicionar novo funcionário -->

      <button id="btnFabAddEmp" class="fab" title="Novo funcionário" aria-label="Novo funcionário">+</button>

      <!-- ===== Modal: Novo/Editar Funcionário ===== -->

      <div id="empModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="empModalTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="empModalTitle">Novo funcionário</h3>

          <!-- Observação de segurança (demo) -->

          <p class="muted" style="margin-top:-6px">

            \* DEMO: senha padrão é salva em texto apenas para testes. Em produção, use Auth/Hash.

          </p>

          <form id="empForm">

            <input type="hidden" id="fId" />

            <div class="svc-row">

              <div>

                <label for="fName">Nome \*</label>

                <input id="fName" type="text" required />

              </div>

              <div>

                <label for="fRole">Função \*</label>

                <select id="fRole" required>

                  <option value="">Selecione…</option>

                  <option>Barbeiro</option>

                  <option>Recepção</option>

                  <option>Gerente</option>

                  <option>Auxiliar</option>

                  <option>Financeiro</option>

                </select>

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fPhone">Telefone \*</label>

                <input id="fPhone" type="text" placeholder="(21) 99999-9999" required />

              </div>

              <div>

                <label for="fAccess">Tipo de acesso \*</label>

                <select id="fAccess" required>

                  <option value="">Selecione…</option>

                  <option value="Lancamentos">Lançamentos</option>

                  <option value="Admin">Admin</option>

                  <option value="Leitor">Leitor</option>

                </select>

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fAddress">Endereço</label>

                <input id="fAddress" type="text" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fContract">Tipo de contrato \*</label>

                <select id="fContract" required>

                  <option value="">Selecione…</option>

                  <option>CLT</option>

                  <option>PJ</option>

                  <option>Freelancer</option>

                  <option>Comissionista</option>

                  <option>Estágio</option>

                </select>

              </div>

              <div>

                <label for="fFixedSalary">Salário fixo (R$)</label>

                <input id="fFixedSalary" type="number" step="0.01" min="0" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fComType">Tipo de comissão</label>

                <select id="fComType">

                  <option value="">(sem comissão)</option>

                  <option value="percentual">Percentual</option>

                  <option value="valor\_fixo">Valor fixo por serviço</option>

                </select>

              </div>

              <div>

                <label for="fComValue">Valor da comissão</label>

                <input id="fComValue" type="number" step="0.01" min="0" placeholder="Ex.: 40 (percentual) ou 5,00 (R$)" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fLoginId">Login (opcional)</label>

                <input id="fLoginId" type="text" placeholder="ex.: igor.sousa" />

              </div>

              <div>

                <label for="fGenId">ID (auto)</label>

                <input id="fGenId" type="text" readonly />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="fPassword">Senha (padrão)</label>

                <input id="fPassword" type="text" readonly />

              </div>

              <div style="display:flex; align-items:flex-end">

                <label style="display:flex; gap:8px; align-items:center">

                  <input id="fMustChange" type="checkbox" checked />

                  Forçar troca no 1º login

                </label>

              </div>

            </div>

            <!-- Box de Foto (só faz sentido na edição, mas deixamos aqui opcional) -->

            <div class="svc-row">

              <div>

                <label for="fPhoto">Foto (PNG/JPG)</label>

                <input id="fPhoto" type="file" accept="image/\*" />

                <div style="display:flex; align-items:center; gap:10px; margin-top:6px">

                  <div style="width:72px;height:72px;border-radius:50%;overflow:hidden;border:1px solid #232323;background:#0c1426;display:flex;align-items:center;justify-content:center;color:#94a3b8;font-weight:900" id="fPhotoPrev">FOTO</div>

                  <small class="muted">Pré-visualização (salva como base64)</small>

                </div>

              </div>

            </div>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- ===== Modal: Confirmar exclusão ===== -->

      <div id="empModalDelete" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="empModalDelTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="empModalDelTitle">Excluir funcionário</h3>

          <p>Tem certeza que deseja excluir <strong id="empDelNome">este funcionário</strong>?</p>

          <div class="svc-actions">

            <button id="btnEmpConfirmDelete" class="btn btn-red">Excluir</button>

            <button class="btn btn-ghost" data-close>Cancelar</button>

          </div>

        </div>

      </div>

    </main>

  </div>

</body>

</html>

<!-- lançamentos do admin -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Lançamentos — Barbearia Carreiro</title>

  <!-- CSS base do admin -->

  <link rel="stylesheet" href="../../css/admin.css?v=7" />

  <!-- (Opcional) herdar styles.css global -->

  <!-- <link rel="stylesheet" href="../../css/styles.css" /> -->

  <!-- (Opcional) Firebase para futura auth -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script defer src="../../js/firebase-config.js"></script>

  <!-- Marca o item ativo do menu pela URL -->

  <script defer src="../../js/admin-nav.js"></script>

  <!-- Lógica desta página -->

  <script defer src="../../js/lancamentos.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

      </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>Lançamentos</h1>

      </header>

      <!-- ================= FILTROS ================= -->

      <section class="panel" aria-labelledby="h-filtros">

        <h2 id="h-filtros" class="panel-title" style="margin-bottom:8px">Filtros</h2>

        <div class="svc-utils" style="row-gap:8px">

          <!-- Intervalo rápido -->

          <div class="btn-group" role="group" aria-label="Intervalo">

            <button class="btn btn-ghost" data-range="dia">Dia</button>

            <button class="btn btn-ghost" data-range="semana">Semana</button>

            <button class="btn btn-ghost" data-range="mes">Mês</button>

            <button class="btn btn-ghost" data-range="ano">Ano</button>

            <button class="btn btn-blue"  data-range="custom">Personalizado</button>

          </div>

          <!-- Datas -->

          <div class="svc-row">

            <div>

              <label for="fDataDe">De</label>

              <input type="date" id="fDataDe" />

            </div>

            <div>

              <label for="fDataAte">Até</label>

              <input type="date" id="fDataAte" />

            </div>

          </div>

          <!-- Seletor de profissional/cliente -->

          <div class="svc-row">

            <div>

              <label for="fProf">Profissional</label>

              <select id="fProf"></select>

            </div>

            <div>

              <label for="fCli">Cliente</label>

              <select id="fCli"></select>

            </div>

          </div>

          <!-- Forma de pagamento / Tipo / Ordenação -->

          <div class="svc-row">

            <div>

              <label for="fForma">Forma de pagamento</label>

              <select id="fForma">

                <option value="">Todos</option>

                <option>Dinheiro</option>

                <option>Pix</option>

                <option>Debito</option>

                <option>CreditoAVista</option>

                <option>CreditoParcelado</option>

              </select>

            </div>

            <div>

              <label for="fTipo">Tipo</label>

              <select id="fTipo">

                <option value="">Todos</option>

                <option value="servico">Serviço</option>

                <option value="combo\_uso">Combo (uso)</option>

                <option value="produto">Produto</option>

              </select>

            </div>

            <div>

              <label for="fOrd">Ordenação</label>

              <select id="fOrd">

                <option value="desc">Mais recente → Mais antigo</option>

                <option value="asc">Mais antigo → Mais recente</option>

              </select>

            </div>

          </div>

          <div class="svc-util-btns">

            <button id="btnAplicar" class="btn btn-green">Aplicar</button>

            <button id="btnLimpar"  class="btn btn-ghost">Limpar</button>

            <span class="spacer"></span>

            <button id="lanExport" class="btn btn-blue">Exportar JSON</button>

            <label class="btn btn-ghost">Importar JSON

              <input id="lanImport" type="file" accept="application/json" hidden />

            </label>

          </div>

        </div>

      </section>

      <!-- ================= KPIs ================= -->

      <section class="panel">

        <div class="kpi-grid">

          <div class="kpi"><div class="kpi-label">Receita bruta</div><div class="kpi-value" id="kpiReceitaBruta">R$ 0,00</div></div>

          <div class="kpi"><div class="kpi-label">Taxas</div><div class="kpi-value" id="kpiTaxas">R$ 0,00</div></div>

          <div class="kpi"><div class="kpi-label">Comissões</div><div class="kpi-value" id="kpiComissoes">R$ 0,00</div></div>

          <div class="kpi"><div class="kpi-label">Custos</div><div class="kpi-value" id="kpiCustos">R$ 0,00</div></div>

          <div class="kpi"><div class="kpi-label">Resultado líquido</div><div class="kpi-value" id="kpiResultado">R$ 0,00</div></div>

        </div>

      </section>

      <!-- ================= TABELA ================= -->

      <section class="panel">

        <div class="table-wrap">

          <table class="table" id="lanTable">

            <thead>

              <tr>

                <th>Data/Hora</th>

                <th>Profissional</th>

                <th>Cliente</th>

                <th>Serviço</th>

                <th>Produtos</th>

                <th>Forma pag.</th>

                <th>Preço bruto</th>

                <th>Taxa</th>

                <th>Comissão</th>

                <th>Custos</th>

                <th>Resultado</th>

                <th>Ações</th>

              </tr>

            </thead>

            <tbody id="lanTbody"></tbody>

          </table>

        </div>

      </section>

      <!-- FAB (Novo lançamento) -->

      <button id="btnFabAddLan" class="fab" title="Novo lançamento" aria-label="Novo lançamento">+</button>

      <!-- ================= MODAL: Novo/Editar ================= -->

      <div id="lanModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="lanTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="lanTitle">Novo lançamento</h3>

          <!-- [FORM: NOVO LANCAMENTO] -->

          <form id="lanForm">

            <!-- Profissional / Cliente -->

            <div class="svc-row">

              <div>

                <label for="lProf">Profissional \*</label>

                <select id="lProf" required></select>

              </div>

              <div>

                <label for="lCli">Cliente \*</label>

                <select id="lCli" required></select>

              </div>

            </div>

            <!-- Serviço ou Combo -->

            <div class="svc-row">

              <div>

                <label for="lTipoSrv">Tipo</label>

                <select id="lTipoSrv">

                  <option value="servico">Serviço</option>

                  <option value="combo\_uso">Combo (uso)</option>

                </select>

              </div>

              <div id="srvSelectBox">

                <label for="lSrv">Serviço</label>

                <select id="lSrv"></select>

              </div>

              <div id="comboSelectBox" hidden>

                <label for="lCombo">Combo do cliente</label>

                <select id="lCombo"></select>

              </div>

            </div>

            <!-- Preço/Desconto (só serviço) -->

            <div id="srvPrecoBox">

              <div class="svc-row">

                <div>

                  <label for="lPreco">Preço praticado (R$)</label>

                  <input id="lPreco" type="number" step="0.01" min="0" />

                </div>

                <div>

                  <label for="lDescTipo">Desconto</label>

                  <select id="lDescTipo">

                    <option value="">—</option>

                    <option value="valor">Valor (R$)</option>

                    <option value="percentual">% (%)</option>

                  </select>

                </div>

                <div>

                  <label for="lDescVal">Valor desc.</label>

                  <input id="lDescVal" type="number" step="0.01" min="0" />

                </div>

              </div>

            </div>

            <!-- Produtos -->

            <div class="hint" style="margin-top:8px">Produtos (opcional):</div>

            <div id="lanProdList"></div>

            <button id="btnAddProd" class="btn btn-ghost" type="button">+ Adicionar produto</button>

            <!-- Pagamento -->

            <div class="svc-row" style="margin-top:8px">

              <div>

                <label for="lForma">Forma de pagamento</label>

                <select id="lForma">

                  <option>Dinheiro</option>

                  <option>Pix</option>

                  <option>Debito</option>

                  <option>CreditoAVista</option>

                  <option>CreditoParcelado</option>

                </select>

              </div>

              <div>

                <label for="lParcelas">Parcelas (se cartão)</label>

                <input id="lParcelas" type="number" min="1" value="1" />

              </div>

            </div>

            <label for="lObs" class="svc-label-up" style="margin-top:6px">Observação</label>

            <textarea id="lObs" rows="3" placeholder="Opcional"></textarea>

            <!-- KPIs ao vivo -->

            <div class="svc-kpis" style="margin-top:10px">

              <div class="svc-kpi"><div class="lb">Receita</div><div class="vl" id="kpiLiveReceita">R$ 0,00</div></div>

              <div class="svc-kpi"><div class="lb">Taxa</div><div class="vl" id="kpiLiveTaxa">R$ 0,00</div></div>

              <div class="svc-kpi"><div class="lb">Comissão</div><div class="vl" id="kpiLiveComissao">R$ 0,00</div></div>

              <div class="svc-kpi"><div class="lb">Custos</div><div class="vl" id="kpiLiveCustos">R$ 0,00</div></div>

              <div class="svc-kpi"><div class="lb">Resultado</div><div class="vl" id="kpiLiveResultado">R$ 0,00</div></div>

            </div>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- MODAL: Confirmação de exclusão -->

      <div id="lanDelModal" class="modal" aria-hidden="true">

        <div class="modal-card">

          <button class="modal-close" data-close>&times;</button>

          <h3>Excluir lançamento</h3>

          <p>Recomendado: estornar ao invés de excluir. Deseja excluir mesmo assim?</p>

          <div class="svc-actions">

            <button id="lanConfirmDel" class="btn btn-red">Excluir</button>

            <button class="btn btn-ghost" data-close>Cancelar</button>

          </div>

        </div>

      </div>

    </main>

  </div>

</body>

</html>

<!—produtos(admin) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Produtos (Estoque) — Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../../css/admin.css?v=4" />

  <!-- (Opcional) Firebase futuro -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script defer src="../../js/firebase-config.js"></script>

  <!-- Navegação do admin -->

  <script defer src="../../js/admin-nav.js"></script>

  <!-- Lógica da página -->

  <script defer src="../../js/produtos.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

        </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>Produtos (Estoque)</h1>

      </header>

      <!-- Barra de ações -->

      <section class="panel">

        <div class="svc-utils">

          <div class="search">

            <label for="prdQ" style="margin:0">Busca:</label>

            <input id="prdQ" type="text" placeholder="Nome, SKU, categoria, fornecedor…" />

          </div>

          <div class="svc-util-btns">

            <button id="prdExport" class="btn btn-blue">Exportar JSON</button>

            <label class="btn btn-ghost">Importar JSON

              <input id="prdImportFile" type="file" accept="application/json" hidden />

            </label>

            <button id="prdZerar" class="btn btn-red">Zerar tudo</button>

          </div>

        </div>

        <!-- Tabela -->

        <div class="table-wrap">

          <table id="prdTable" class="table">

            <thead>

              <tr>

                <th>Produto</th>

                <th>SKU</th>

                <th>Categoria</th>

                <th>Un.</th>

                <th>Estoque</th>

                <th>Mín.</th>

                <th>Preço venda</th>

                <th>Custo (CMP)</th>

                <th>Lucro</th>

                <th>Status</th>

                <th class="hide-sm">Fornecedor</th>

                <th style="width:132px">Ações</th>

              </tr>

            </thead>

            <tbody id="prdTbody"><!-- via JS --></tbody>

          </table>

        </div>

      </section>

      <!-- FAB: adicionar produto -->

      <button id="btnFabAddPrd" class="fab" title="Novo produto" aria-label="Novo produto">+</button>

      <!-- ===== Modal: Novo/Editar Produto ===== -->

      <div id="prdModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="prdModalTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="prdModalTitle">Novo produto</h3>

          <form id="prdForm">

            <input type="hidden" id="pId" />

            <div class="svc-row">

              <div>

                <label for="pNome">Nome \*</label>

                <input id="pNome" type="text" required />

              </div>

              <div>

                <label for="pSKU">SKU/ID \*</label>

                <input id="pSKU" type="text" required />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="pCategoria">Categoria</label>

                <input id="pCategoria" type="text" placeholder="Pomadas, Shampoos…" />

              </div>

              <div>

                <label for="pUn">Unidade \*</label>

                <input id="pUn" type="text" placeholder="un, ml, g…" required />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="pPreco">Preço venda (R$) \*</label>

                <input id="pPreco" type="number" min="0" step="0.01" required />

              </div>

              <div>

                <label for="pFornecedor">Fornecedor principal</label>

                <input id="pFornecedor" type="text" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="pEstMin">Estoque mínimo \*</label>

                <input id="pEstMin" type="number" min="0" step="1" required />

              </div>

              <div>

                <label>Política de custo</label>

                <input type="text" id="pPolitica" value="CMP (padrão)" readonly />

              </div>

            </div>

            <label for="pObs" class="svc-label-up">Observação</label>

            <textarea id="pObs" placeholder="Notas internas, validade por lote será lançada nas movimentações."></textarea>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- ===== Modal: Movimentar Estoque ===== -->

      <div id="movModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="movTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="movTitle">Movimentar estoque</h3>

          <p class="muted" style="margin-top:-6px">\* Custo médio ponderado (CMP) é recalculado após cada ENTRADA.</p>

          <form id="movForm">

            <input type="hidden" id="mProductId" />

            <div class="svc-row">

              <div>

                <label for="mTipo">Tipo</label>

                <select id="mTipo">

                  <option value="entrada">Entrada (Compra)</option>

                  <option value="saida">Saída (Venda/Brinde/Consumo/Perda)</option>

                  <option value="ajuste">Ajuste</option>

                </select>

              </div>

              <div>

                <label for="mQtd">Quantidade \*</label>

                <input id="mQtd" type="number" step="1" />

              </div>

            </div>

            <!-- Campos específicos -->

            <div id="boxEntrada">

              <div class="svc-row">

                <div>

                  <label for="mCusto">Custo unitário (R$) \*</label>

                  <input id="mCusto" type="number" step="0.01" min="0" />

                </div>

                <div>

                  <label for="mFornecedor">Fornecedor</label>

                  <input id="mFornecedor" type="text" />

                </div>

              </div>

              <div class="svc-row">

                <div>

                  <label for="mLote">Lote (opcional)</label>

                  <input id="mLote" type="text" />

                </div>

                <div>

                  <label for="mValidade">Validade (opcional)</label>

                  <input id="mValidade" type="date" />

                </div>

              </div>

            </div>

            <div id="boxSaida" hidden>

              <div class="svc-row">

                <div>

                  <label for="mMotivo">Motivo</label>

                  <select id="mMotivo">

                    <option value="venda">Venda</option>

                    <option value="brinde">Brinde</option>

                    <option value="consumo">Consumo interno</option>

                    <option value="perda">Perda/Quebra</option>

                  </select>

                </div>

                <div>

                  <label for="mPreco">Preço aplicado (informativo)</label>

                  <input id="mPreco" type="number" step="0.01" min="0" />

                </div>

              </div>

            </div>

            <div id="boxAjuste" hidden>

              <label for="mJust">Justificativa</label>

              <input id="mJust" type="text" placeholder="Inventário, erro, etc." />

            </div>

            <label for="mObs" class="svc-label-up">Observação</label>

            <textarea id="mObs" placeholder="Notas da movimentação (opcional)"></textarea>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- ===== Modal: Confirmar exclusão ===== -->

      <div id="prdModalDelete" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="prdModalDelTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="prdModalDelTitle">Excluir produto</h3>

          <p>Tem certeza que deseja excluir <strong id="prdDelNome">este produto</strong>?</p>

          <div class="svc-actions">

            <button id="btnPrdConfirmDelete" class="btn btn-red">Excluir</button>

            <button class="btn btn-ghost" data-close>Cancelar</button>

          </div>

        </div>

      </div>

    </main>

  </div>

</body>

</html>

<!—serviços(admin) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

  <meta charset="UTF-8" />

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

  <title>Admin • Serviços — Barbearia Carreiro</title>

  <!-- Estilos -->

  <link rel="stylesheet" href="../../css/admin.css?v=4" />

  <!-- (Opcional) Firebase futuro -->

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-app.js"></script>

  <script src="https://www.gstatic.com/firebasejs/8.10.1/firebase-auth.js"></script>

  <script defer src="../../js/firebase-config.js"></script>

  <!-- Navegação do admin + Lógica de serviços -->

  <script defer src="../../js/admin-nav.js"></script>

  <script defer src="../../js/service.js"></script>

</head>

<body class="admin-body">

  <div class="admin-shell">

    <!-- SIDEBAR -->

    <aside class="admin-sidebar">

      <div class="admin-brand">

        <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

      </div>

      <nav class="admin-nav">

        <a class="nav-item" data-color="#2563EB" href="./agenda.html"><span class="dot"></span>Agenda</a>

        <a class="nav-item" data-color="#1E90FF" href="./caixa.html"><span class="dot"></span>Caixa</a>

        <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

        <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

        <a class="nav-item" data-color="#9B59B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

        <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span class="dot"></span>Clientes</a>

        <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

        </nav>

    </aside>

    <!-- MAIN -->

    <main class="admin-main">

      <header class="admin-topbar">

        <h1>serviços</h1>

      </header>

      <!-- ===== Tabela única ===== -->

      <section class="panel">

        <div class="svc-utils">

          <div class="search">

            <label for="svcQ" style="margin:0">Busca:</label>

            <input id="svcQ" type="text" placeholder="Nome, observação, tipo…" />

          </div>

          <div class="svc-util-btns">

            <button id="svcExport" class="btn btn-blue">Exportar JSON</button>

            <label class="btn btn-ghost">Importar JSON

              <input id="svcImportFile" type="file" accept="application/json" hidden />

            </label>

            <button id="svcZerar" class="btn btn-red">Zerar catálogo</button>

          </div>

        </div>

        <div class="table-wrap">

          <table id="svcTable" class="table">

            <thead>

              <tr>

                <th>ícone</th>

                <th>serviço</th>

                <th>duração(min)</th>

                <th>Preço (R$)</th>

                <th>Custo (R$)</th>

                <th>Lucro (R$)</th>

                <th>Margem (%)</th>

                <th class="hide-sm">observação</th>

                <th class="hide-sm">Materiais (lista resumida)</th>

                <th style="width:88px">Ações</th>

              </tr>

            </thead>

            <tbody id="svcTbody">

              <!-- linhas renderizadas via service.js -->

            </tbody>

          </table>

        </div>

      </section>

      <!-- FAB: adicionar novo serviço -->

      <button id="btnFabAddService" class="fab" title="Novo serviço" aria-label="Novo serviço">+</button>

      <!-- ===== Modal: Adicionar / Editar ===== -->

      <div id="svcModal" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="svcModalTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="svcModalTitle">Novo serviço</h3>

          <form id="svcForm">

            <input type="hidden" id="svcId" />

            <div class="svc-row">

              <div>

                <label for="mNome">Nome \*</label>

                <input id="mNome" type="text" required />

              </div>

              <div>

                <label for="mDuracao">Duração (min) \*</label>

                <input id="mDuracao" type="number" min="0" step="1" required />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="mPreco">Preço (R$) \*</label>

                <input id="mPreco" type="number" step="0.01" min="0" required />

              </div>

              <div>

                <label for="mCusto">Custo (R$)</label>

                <input id="mCusto" type="number" step="0.01" min="0" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="mObs">Observação</label>

                <input id="mObs" type="text" />

              </div>

            </div>

            <div class="svc-row">

              <div>

                <label for="mFoto">Foto (PNG/JPG)</label>

                <input id="mFoto" type="file" accept="image/\*" />

              </div>

            </div>

            <div class="svc-actions">

              <button type="submit" class="btn btn-green">Salvar</button>

              <button type="button" class="btn btn-ghost" data-close>Cancelar</button>

            </div>

          </form>

        </div>

      </div>

      <!-- ===== Modal: Confirmar exclusão ===== -->

      <div id="svcModalDelete" class="modal" aria-hidden="true">

        <div class="modal-card" role="dialog" aria-modal="true" aria-labelledby="svcModalDelTitle">

          <button class="modal-close" data-close>&times;</button>

          <h3 id="svcModalDelTitle">Excluir serviço</h3>

          <p>Tem certeza que deseja excluir <strong id="delNome">este serviço</strong>?</p>

          <div class="svc-actions">

            <button id="btnConfirmDelete" class="btn btn-red">Excluir</button>

            <button class="btn btn-ghost" data-close>Cancelar</button>

          </div>

        </div>

      </div>

    </main>

  </div>

</body>

</html>

<!—caixa (admin) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

    <meta charset="UTF-8" />

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

    <title>Admin | Caixa - Barbearia Carreiro</title>

    <link rel="stylesheet" href="../../css/admin.css?v=7" />

    <!-- Progressive Web App (PWA) -->

    <link rel="manifest" href="/manifest.json">

    <style>

        .caixa-container { max-width: 1000px; margin: 0 auto; font-family: "Open Sans", Arial, sans-serif; color: #fff; }

        .caixa-topo { display: grid; grid-template-columns: repeat(auto-fit, minmax(250px, 1fr)); gap: 15px; margin-bottom: 20px; }

        .caixa-topo > div { display: flex; flex-direction: column; }

        .caixa-section { margin-bottom: 25px; padding: 16px; border-radius: 14px; background: var(--card); border: 1px solid var(--border); }

        .caixa-section h3 { margin: 0 0 15px; font-size: 18px; color: var(--primary); border-bottom: 1px solid var(--border); padding-bottom: 8px; }

        .item-lista-row { display: grid; grid-template-columns: 2fr 1fr 1fr 1fr auto; gap: 10px; align-items: center; margin-bottom: 8px; }

        .item-lista-row select, .item-lista-row input { width: 100%; }

        .item-lista-row .del-btn { background: #3a0c0c; border:1px solid #ef4444; color:#fecaca; cursor: pointer; padding: 8px; border-radius: 8px; text-align: center; }

        .caixa-section .btn-add { background: #0f224a; border-color: #3b82f6; color: #dbeafe; }

        .resumo-grid { display: grid; grid-template-columns: 1fr 1fr; gap: 10px 20px; }

        .resumo-grid p { margin: 5px 0; font-size: 15px; display: flex; justify-content: space-between; }

        .resumo-grid strong { font-weight: 900; }

        .resumo-total { font-size: 22px !important; color: #4ade80; border-top: 1px solid var(--border); padding-top: 10px; margin-top: 10px; }

        .caixa-acoes { display: flex; justify-content: flex-end; gap: 10px; margin-top: 20px; }

    </style>

    <script defer src="../../js/admin-nav.js"></script>

    <script defer src="../../js/caixa.js"></script>

</head>

<body class="admin-body">

    <div class="admin-shell">

        <aside class="admin-sidebar">

            <div class="admin-brand">

                <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

            </div>

            <nav class="admin-nav">

                <a class="nav-item" data-color="#1E90FF" href="./agenda.html"><span class="dot"></span>Agenda</a>

                <a class="nav-item" data-color="#4ade80" href="./caixa.html"><span class="dot"></span>Caixa</a>

                <a class="nav-item" data-color="#1E90FF" href="./dashboard.html"><span class="dot"></span>Painel</a>

                <a class="nav-item" data-color="#34C759" href="./servicos.html"><span class="dot"></span>Serviços</a>

                <a class="nav-item" data-color="#9859B6" href="./funcionarios.html"><span class="dot"></span>Funcionários</a>

                <a class="nav-item" data-color="#0EA5E9" href="./lancamentos.html"><span class="dot"></span>Lançamentos</a>

                <a class="nav-item" data-color="#FF9500" href="./produtos.html"><span class="dot"></span>Produtos</a>

                <a class="nav-item" data-color="#F59E0B" href="./clientes.html"><span class="dot"></span>Clientes</a>

            </nav>

        </aside>

        <main class="admin-main">

            <header class="admin-topbar">

                <h1>Caixa / Ponto de Venda</h1>

            </header>

            <div class="caixa-container">

                <section class="panel caixa-topo">

                    <div>

                        <label for="caixaCliente">Cliente\*</label>

                        <select id="caixaCliente" required><option value="">Selecione...</option></select>

                    </div>

                    <div>

                        <label for="caixaProfissional">Profissional\*</label>

                        <select id="caixaProfissional" required><option value="">Selecione...</option></select>

                    </div>

                    <div>

                        <label for="caixaData">Data/Hora</label>

                        <input id="caixaData" type="datetime-local" />

                    </div>

                </section>

                <section class="caixa-section">

                    <h3>Serviços Realizados</h3>

                    <div id="servicosLista">

                        </div>

                    <button id="addServico" class="btn btn-ghost btn-add" style="margin-top: 10px;">+ Adicionar Serviço</button>

                </section>

                <section class="caixa-section">

                    <h3>Produtos Consumidos/Vendidos</h3>

                    <div id="produtosLista">

                        </div>

                    <button id="addProduto" class="btn btn-ghost btn-add" style="margin-top: 10px;">+ Adicionar Produto</button>

                </section>

                <section class="caixa-section">

                    <h3>Resumo e Pagamento</h3>

                    <div class="resumo-grid">

                        <p>Total Serviços: <strong id="resumoServicos">R$ 0,00</strong></p>

                        <p>Total Produtos: <strong id="resumoProdutos">R$ 0,00</strong></p>

                        <p>Subtotal: <strong id="resumoSubtotal">R$ 0,00</strong></p>

                        <div>

                            <label for="formaPagamento" style="display: block; margin-bottom: 5px;">Forma de Pagamento</label>

                            <select id="formaPagamento" style="width: 100%;">

                                <option>Dinheiro</option>

                                <option>Pix</option>

                                <option>Debito</option>

                                <option>CreditoAVista</option>

                                <option>CreditoParcelado</option>

                            </select>

                        </div>

                        <p>Taxa: <strong id="resumoTaxa">R$ 0,00</strong></p>

                        <p class="resumo-total">TOTAL A PAGAR: <strong id="resumoTotalFinal">R$ 0,00</strong></p>

                    </div>

                </section>

                <section class="caixa-acoes">

                    <button id="btnCancelar" class="btn btn-red">Cancelar</button>

                    <button id="btnFinalizar" class="btn btn-green">Finalizar Venda</button>

                </section>

            </div>

        </main>

    </div>

    <script>

  if ('serviceWorker' in navigator) {

    window.addEventListener('load', () => {

      navigator.serviceWorker.register('/sw.js').then(registration => {

        console.log('ServiceWorker registrado com sucesso: ', registration.scope);

      }).catch(error => {

        console.log('Falha no registro do ServiceWorker: ', error);

      });

    });

  }

</script>

</body>

</html>

<!-- AGENDA(FUNCIONARIO) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

    <meta charset="UTF-8" />

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

    <title>Funcionário | Agenda - Barbearia Carreiro</title>

    <link rel="stylesheet" href="../../css/admin.css?v=7" />

    <style>

        .agenda-item { background: var(--card); border: 1px solid var(--border); border-radius: 12px; padding: 15px; margin-bottom: 10px; display: grid; grid-template-columns: 1fr auto; gap: 15px; align-items: center; }

        .agenda-details h4 { margin: 0 0 5px; color: var(--primary); }

        .agenda-details p { margin: 0; color: #ccc; }

        .agenda-actions { display: flex; gap: 10px; }

    </style>

</head>

<body class="admin-body">

    <div class="admin-shell">

        <aside class="admin-sidebar">

            <div class="admin-brand"><img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

            <nav class="admin-nav">

                <a class="nav-item" href="./agenda.html"><span class="dot"></span>Agenda</a>

                <a class="nav-item" href="./caixa.html"><span class="dot"></span>Caixa</a>

                <a class="nav-item" href="./perfil.html"><span class="dot"></span>Minha Disponibilidade</a>

            </nav>

        </aside>

        <main class="admin-main">

            <header class="admin-topbar">

                <h1 id="header-title">Sua Agenda de Hoje</h1>

            </header>

            <section class="panel">

                <div id="lista-agenda">

                    <p>Carregando agenda...</p>

                </div>

            </section>

        </main>

    </div>

    <script defer src="../../js/agenda-funcionario.js"></script>

</body>

</html>

<!-- CAIXA(FUNCIONARIO) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

    <meta charset="UTF-8" />

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

    <title>Funcionário | Caixa - Barbearia Carreiro</title>

    <link rel="stylesheet" href="../../css/admin.css?v=7" />

    <style>

        .caixa-container { max-width: 1000px; margin: 0 auto; font-family: "Open Sans", Arial, sans-serif; color: #fff; }

        .caixa-topo { display: grid; grid-template-columns: repeat(auto-fit, minmax(250px, 1fr)); gap: 15px; margin-bottom: 20px; }

        .caixa-topo > div { display: flex; flex-direction: column; }

        .caixa-section { margin-bottom: 25px; padding: 16px; border-radius: 14px; background: var(--card); border: 1px solid var(--border); }

        .caixa-section h3 { margin: 0 0 15px; font-size: 18px; color: var(--primary); border-bottom: 1px solid var(--border); padding-bottom: 8px; }

        .item-lista-row { display: grid; grid-template-columns: 2fr 1fr 1fr 1fr auto; gap: 10px; align-items: center; margin-bottom: 8px; }

        .item-lista-row select, .item-lista-row input { width: 100%; }

        .item-lista-row .del-btn { background: #3a0c0c; border:1px solid #ef4444; color:#fecaca; cursor: pointer; padding: 8px; border-radius: 8px; text-align: center; }

        .caixa-section .btn-add { background: #0f224a; border-color: #3b82f6; color: #dbeafe; }

        .resumo-grid { display: grid; grid-template-columns: 1fr 1fr; gap: 10px 20px; }

        .resumo-grid p { margin: 5px 0; font-size: 15px; display: flex; justify-content: space-between; }

        .resumo-grid strong { font-weight: 900; }

        .resumo-total { font-size: 22px !important; color: #4ade80; border-top: 1px solid var(--border); padding-top: 10px; margin-top: 10px; }

        .caixa-acoes { display: flex; justify-content: flex-end; gap: 10px; margin-top: 20px; }

    </style>

</head>

<body class="admin-body">

    <div class="admin-shell">

        <aside class="admin-sidebar">

            <div class="admin-brand">

                <img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO">

            </div>

            <nav class="admin-nav">

                <a class="nav-item" href="./agenda.html"><span class="dot"></span>Agenda</a>

                <a class="nav-item" href="./caixa.html"><span class="dot"></span>Caixa</a>

                <a class="nav-item" href="./perfil.html"><span class="dot"></span>Minha Disponibilidade</a>

            </nav>

        </aside>

        <main class="admin-main">

            <header class="admin-topbar">

                <h1>Caixa / Registrar Atendimento</h1>

            </header>

            <div class="caixa-container">

                <section class="panel caixa-topo">

                    <div>

                        <label for="caixaCliente">Cliente\*</label>

                        <select id="caixaCliente" required><option value="">Selecione...</option></select>

                    </div>

                    <div>

                        <label for="caixaProfissional">Profissional\*</label>

                        <select id="caixaProfissional" required disabled><option value="">Carregando...</option></select>

                    </div>

                    <div>

                        <label for="caixaData">Data/Hora</label>

                        <input id="caixaData" type="datetime-local" />

                    </div>

                </section>

                <section class="caixa-section">

                    <h3>Serviços Realizados</h3>

                    <div id="servicosLista"></div>

                    <button id="addServico" class="btn btn-ghost btn-add" style="margin-top: 10px;">+ Adicionar Serviço</button>

                </section>

                <section class="caixa-section">

                    <h3>Produtos Consumidos/Vendidos</h3>

                    <div id="produtosLista"></div>

                    <button id="addProduto" class="btn btn-ghost btn-add" style="margin-top: 10px;">+ Adicionar Produto</button>

                </section>

                <section class="caixa-section">

                    <h3>Resumo e Pagamento</h3>

                    <div class="resumo-grid">

                        <p>Total Serviços: <strong id="resumoServicos">R$ 0,00</strong></p>

                        <p>Total Produtos: <strong id="resumoProdutos">R$ 0,00</strong></p>

                        <p>Subtotal: <strong id="resumoSubtotal">R$ 0,00</strong></p>

                        <div>

                            <label for="formaPagamento" style="display: block; margin-bottom: 5px;">Forma de Pagamento</label>

                            <select id="formaPagamento" style="width: 100%;">

                                <option>Dinheiro</option>

                                <option>Pix</option>

                                <option>Debito</option>

                                <option>CreditoAVista</option>

                                <option>CreditoParcelado</option>

                            </select>

                        </div>

                        <p>Taxa: <strong id="resumoTaxa">R$ 0,00</strong></p>

                        <p class="resumo-total">TOTAL A PAGAR: <strong id="resumoTotalFinal">R$ 0,00</strong></p>

                    </div>

                </section>

                <section class="caixa-acoes">

                    <button id="btnCancelar" class="btn btn-red">Cancelar</button>

                    <button id="btnFinalizar" class="btn btn-green">Finalizar Venda</button>

                </section>

            </div>

        </main>

    </div>

    <script defer src="../../js/caixa-funcionario.js"></script>

</body>

</html>

<!-- PERFIL(FUNCIONARIO) -->

<!DOCTYPE html>

<html lang="pt-BR">

<head>

    <meta charset="UTF-8" />

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

    <title>Funcionário | Minha Disponibilidade</title>

    <link rel="stylesheet" href="../../css/admin.css?v=8" />

    <style>

        /\* NOVOS ESTILOS PARA A NAVEGAÇÃO DOS DIAS \*/

        .day-selector {

            display: flex;

            align-items: center;

            justify-content: center;

            gap: 10px;

        }

        .day-strip-container {

            flex-grow: 1;

            overflow: hidden; /\* Esconde os dias que estão fora da área visível \*/

        }

        .day-strip {

            transition: transform 0.3s ease-in-out; /\* Adiciona uma animação suave de deslize \*/

        }

        .day-nav-btn {

            background: var(--card);

            border: 1px solid var(--border);

            color: #fff;

            width: 40px;

            height: 40px;

            border-radius: 50%;

            font-size: 20px;

            cursor: pointer;

        }

        .day-nav-btn:disabled {

            opacity: 0.3;

            cursor: not-allowed;

        }

    </style>

</head>

<body class="admin-body">

    <div class="admin-shell">

        <aside class="admin-sidebar">

            <div class="admin-brand"><img src="../../assets/icones-logo/sr.carreiro (4).png" alt="SR. CARREIRO"></div>

            <nav class="admin-nav">

                <a class="nav-item" href="./agenda.html"><span class="dot"></span>Agenda</a>

                <a class="nav-item" href="./caixa.html"><span class="dot"></span>Caixa</a>

                <a class="nav-item" href="./perfil.html"><span class="dot"></span>Minha Disponibilidade</a>

            </nav>

        </aside>

        <main class="admin-main">

            <header class="admin-topbar">

                <h1>Configure sua Disponibilidade</h1>

            </header>

            <section class="panel">

                <p class="muted" style="margin:6px 0 10px">

                    Clique nos horários para <strong>liberar</strong> (verde) ou <strong>bloquear</strong> (cinza). Navegue pelos dias usando as setas.

                </p>

                <div id="dayMonth" class="day-month"></div>

                <div class="day-selector">

                    <button id="prevDaysBtn" class="day-nav-btn">&lt;</button>

                    <div class="day-strip-container">

                        <div id="dayStrip" class="day-strip"></div>

                    </div>

                    <button id="nextDaysBtn" class="day-nav-btn">&gt;</button>

                </div>

                <div id="weekMatrix" class="week-matrix" style="margin-top: 15px;"></div>

                <div class="modal-actions" style="justify-content: flex-end; display: flex; margin-top: 20px;">

                    <button id="saveConfig" class="btn btn-green">Salvar Alterações</button>

                </div>

            </section>

        </main>

    </div>

    <script defer src="../../js/perfil-funcionario.js"></script>

</body>

</html>

. ARQUIVOS CSS .

/\* ====== ADMIN ====== \*/

/\* ====== Base ====== \*/

:root{

  --bg:#000; --card:#111; --border:#232323; --text:#fff; --muted:#bdbdbd; --primary:#e0b25b;

}

\*{ box-sizing:border-box; }

html,body{ height:100%; }

body.admin-body{ margin:0; background:var(--bg); color:var(--text); font-family:"Open Sans", Arial, sans-serif; -webkit-font-smoothing:antialiased; -moz-osx-font-smoothing:grayscale; }

/\* ====== Shell ====== \*/

.admin-shell{

  display:grid;

  grid-template-columns: 260px 1fr;

  min-height: 100vh;

}

/\* ====== Sidebar ====== \*/

.admin-sidebar{

  border-right:1px solid var(--border);

  background:#0b0b0b;

  padding: 16px 12px;

  position: sticky; top:0; height:100dvh;

  overflow-y:auto;

}

.admin-brand img{ width: 160px; display:block; margin: 8px auto 16px; }

.admin-nav{ display:flex; flex-direction:column; gap:8px; }

.admin-nav .nav-item{

  display:flex; align-items:center; gap:10px;

  text-decoration:none;

  background:#121212;

  border:1px solid #1f1f1f;

  color:#fff;

  padding:12px 14px;

  border-radius:12px;

  font-weight:800;

  cursor:pointer;

}

.admin-nav .nav-item .dot{

  width:10px; height:10px; border-radius:50%;

  background: var(--c, #666);

  display:inline-block;

}

.admin-nav .nav-item.active{

  outline:2px solid var(--c, #fff);

  border-color: var(--c, #fff);

}

/\* ====== Main ====== \*/

.admin-main{ padding: 18px 22px 40px; }

.admin-topbar h1{ margin:0 0 12px; font-size:24px; font-weight:900; color:var(--primary); }

/\* ====== Seções / Painéis ====== \*/

.view-title{ margin: 8px 0 12px; font-size:20px; font-weight:900; }

.panel{

  background:var(--card); border:1px solid var(--border); border-radius:14px; padding:14px; margin:12px 0;

}

.panel-title{ font-weight:900; margin-bottom:10px; }

.muted{ color:var(--muted); }

/\* ====== KPIs ====== \*/

.kpi-grid{

  display:grid; grid-template-columns: repeat(4, minmax(180px, 1fr)); gap:12px;

}

.kpi{ background:var(--card); border:1px solid var(--border); border-radius:14px; padding:14px; }

.kpi-label{ color:#cfcfcf; font-size:12px; }

.kpi-value{ font-size:28px; font-weight:900; margin:6px 0; }

.kpi-sub{ color:#bdbdbd; font-size:12px; }

/\* ====== Grades de painéis ====== \*/

.panel-grid{ display:grid; gap:12px; grid-template-columns: repeat(3, 1fr); }

/\* ====== Tabelas ====== \*/

.table-wrap{ overflow:auto; }

.table{ width:100%; border-collapse: collapse; font-size:14px; }

.table th, .table td{ text-align:left; padding:8px 10px; border-bottom:1px solid var(--border); }

.table th{ color:#cfcfcf; font-weight:800; }

/\* ====== Indicadores ====== \*/

.indicators{ display:grid; grid-template-columns: repeat(4,1fr); gap:10px; }

/\* ====== Placeholders de gráfico ====== \*/

.chart-placeholder{

  height: 220px;

  background: repeating-linear-gradient(45deg,#161616,#161616 10px,#141414 10px,#141414 20px);

  border:1px dashed #2d2d2d; border-radius:10px;

  display:flex; align-items:center; justify-content:center; color:#9b9b9b; font-size:12px;

}

/\* ====== Responsivo mínimo ====== \*/

@media (max-width: 1100px){

  .kpi-grid{ grid-template-columns: repeat(2, 1fr); }

  .panel-grid{ grid-template-columns: 1fr; }

  .indicators{ grid-template-columns: repeat(2,1fr); }

  .admin-shell{ grid-template-columns: 220px 1fr; }

}

@media (max-width: 760px){

  .admin-shell{ grid-template-columns: 1fr; }

  .admin-sidebar{ position:relative; height:auto; }

}

/\* ====== Serviços (catálogo) – reutilizável em servicos.html também ====== \*/

.svc-grid{ display:grid; grid-template-columns: 420px 1fr; gap:12px; }

@media (max-width: 1200px){ .svc-grid{ grid-template-columns: 1fr; } }

.svc-card{ padding:16px; }

.svc-h3{ margin:0 0 10px; font-size:18px; }

.svc-row{ display:flex; gap:10px; } .svc-row > div{ flex:1; }

.svc-photo{ display:flex; align-items:center; gap:12px; margin:8px 0; }

.svc-preview{

  width:56px; height:56px; border-radius:50%; border:1px solid #252525; background:#0c1426;

  display:flex; align-items:center; justify-content:center; color:#94a3b8; font-weight:900; overflow:hidden;

}

.svc-preview img{ width:100%; height:100%; object-fit:cover; }

.svc-label-up{ margin-top:8px; display:block; }

.svc-switch{ display:flex; align-items:center; gap:10px; margin:6px 0; }

.svc-switch input{ transform:scale(1.1); }

.svc-mat-head, .svc-mat-row{

  display:grid; gap:8px; align-items:center;

  grid-template-columns: 1.2fr 0.9fr 0.6fr 0.8fr 0.7fr 0.4fr;

}

.svc-mat-head{ color:#cbd5e1; font-size:12px; margin-top:6px; }

.svc-mat-row{ margin:8px 0; }

.svc-mat-row .num{ text-align:right; }

.svc-mat-row .del{ background:#16213a; border:1px solid #22304f; border-radius:10px; padding:8px 0; text-align:center; cursor:pointer; }

.svc-add-mat{

  width:100%; margin-top:6px; background:#16213a; border:1px dashed #3b82f6; color:#3b82f6; padding:10px; border-radius:10px; font-weight:800; cursor:pointer;

}

/\* KPIs de serviço \*/

.svc-kpis{ display:grid; grid-template-columns: repeat(2,1fr); gap:10px; margin-top:8px; }

.svc-kpi{ background:#0f1a30; border:1px solid #252525; border-radius:12px; padding:10px 12px; }

.svc-kpi .lb{ color:#cbd5e1; font-size:12px; }

.svc-kpi .vl{ font-size:20px; font-weight:900; margin-top:2px; }

.svc-actions{ display:flex; gap:10px; margin-top:10px; flex-wrap:wrap; }

.svc-utils{ display:flex; justify-content:space-between; align-items:center; gap:8px; margin-bottom:8px; flex-wrap:wrap; }

.svc-util-btns{ display:flex; gap:8px; flex-wrap:wrap; }

/\* Inputs herdados do admin na página de serviços \*/

#view-servicos input[type="text"],

#view-servicos input[type="number"],

#view-servicos input[type="file"],

#view-servicos select,

#view-servicos textarea{

  width:100%; padding:10px 12px; border-radius:10px; border:1px solid #1f1f1f; background:#0c1426; color:#fff; font-weight:600;

}

#view-servicos textarea{ min-height:90px; resize:vertical; }

#view-servicos input[type="number"]{ text-align:right; }

#view-servicos .mini{ width:40px; height:40px; border-radius:50%; object-fit:cover; border:1px solid #232323; background:#0c1426; }

#view-servicos .cell-actions{ display:flex; gap:8px; }

/\* Botões utilitários \*/

.btn{ padding:10px 14px; border-radius:10px; font-weight:800; border:1px solid transparent; cursor:pointer; }

.btn-green{ background:#164e36; border-color:#2ea66b; color:#d4ffe9; }

.btn-blue{ background:#0f224a; border-color:#3b82f6; color:#dbeafe; }

.btn-yellow{ background:#3a320c; border-color:#eab308; color:#fef08a; }

.btn-red{ background:#3a0c0c; border-color:#ef4444; color:#fecaca; }

.btn-ghost{ background:#121212; border-color:#1f1f1f; color:#fff; }

/\* ABA SERVIÇOS \*/

/\* FAB (botão flutuante “+”) \*/

.fab{

  position: fixed;

  right: 32px;

  bottom: 32px;

  width: 58px; height: 58px; border-radius: 50%;

  background: #2ea66b; color:#0a1b12; font-size:34px; line-height:58px;

  border:2px solid #79d9aa; text-align:center; cursor:pointer;

  box-shadow: 0 10px 30px rgba(0,0,0,.45);

  font-weight: 900;

}

.fab:hover{ transform: translateY(-1px); }

/\* Modais genéricos \*/

.modal{

  position: fixed; inset:0; background: rgba(0,0,0,.55);

  display:none; align-items:center; justify-content:center; padding:20px;

  z-index: 50;

}

.modal.show{ display:flex; }

.modal-card{

  width: min(680px, 96vw);

  background: #111; border:1px solid #232323; border-radius:16px;

  padding: 16px;

}

.modal-card h3{ margin:0 0 10px; font-size:18px; font-weight:900; color:#e0b25b; }

.modal-close{

  position:absolute; margin:-8px -8px 0 0; right:20px;

  width:34px; height:34px; border-radius:50%;

  background:#1a1a1a; color:#fff; border:1px solid #2a2a2a; cursor:pointer;

}

/\* Campos no modal: aproveita inputs do admin \*/

.modal-card input[type="text"],

.modal-card input[type="number"],

.modal-card input[type="file"]{

  width:100%; padding:10px 12px; border-radius:10px;

  border:1px solid #1f1f1f; background:#0c1426; color:#fff; font-weight:600;

}

/\* ====== Barra de filtros da Agenda ====== \*/

.agenda-top{

  display:grid;

  grid-template-columns: 1fr auto 1fr; /\* esq | centro | dir \*/

  align-items:center;

  gap:12px;

  background:var(--card);

  border:1px solid var(--border);

  border-radius:14px;

  padding:10px 12px;

  margin:12px 0;

}

.agenda-left{ justify-self:start; display:flex; gap:8px; }

.agenda-mid{ justify-self:center; display:flex; align-items:center; gap:8px; }

.agenda-right{ justify-self:end; display:flex; align-items:center; gap:8px; }

/\* inputs/seletores bonitinhos dentro da barra \*/

.agenda-top input[type="date"],

.agenda-top select{

  padding:8px 10px;

  border:1px solid #1f1f1f;

  border-radius:10px;

  background:#0c1426;

  color:#fff;

  font-weight:600;

}

/\* responsivo: empilha os 3 blocos \*/

@media (max-width: 680px){

  .agenda-top{ grid-template-columns: 1fr; }

  .agenda-left{ justify-self:stretch; order:1; }

  .agenda-mid{ justify-self:stretch; order:2; }

  .agenda-right{ justify-self:stretch; order:3; }

}

/\* ===== Agenda em BARRAS HORIZONTAIS ===== \*/

.agenda-grid{

  display:flex;

  flex-direction: column;

  gap:22px;

  margin-top:8px;

}

.ag-row{

  display:flex;

  align-items:flex-start;

  gap:18px;

  background:#0d0d0d;

  border:1px solid var(--border);

  border-radius:16px;

  padding:16px;

}

/\* bloco à esquerda (avatar + nome + engrenagem) \*/

.ag-left{ width:140px; display:flex; flex-direction:column; align-items:center; gap:8px; }

.ag-avatar{

  width:96px; height:96px; border-radius:50%;

  background:#1f8a3b;                 /\* círculo verde \*/

  border:2px solid #9BE34E;            /\* borda verde viva \*/

  color:#0b1a0f; font-weight:900;

  display:flex; align-items:center; justify-content:center;

  overflow:hidden;

}

.ag-avatar img{ width:100%; height:100%; object-fit:cover; }

.ag-name{

  display:flex; align-items:center; gap:8px;

  font-weight:900; color:#e5e7eb; text-transform:lowercase;

}

.btn-gear{

  width:28px; height:28px; border-radius:50%;

  border:1px solid #2a2a2a; background:#141414; color:#fff; cursor:pointer;

}

/\* bloco à direita (manhã/tarde) \*/

.ag-right{ flex:1; }

.band-title{ font-weight:900; color:#e5e7eb; margin:2px 0 8px; }

.slot-row{ display:grid; grid-template-columns: repeat(4, 1fr); gap:12px; margin-bottom:14px; }

/\* cartõezinhos dos horários (look do mock) \*/

.slot{

  border:2px solid #9BE34E;            /\* verde \*/

  border-radius:16px;

  min-height:64px;

  background:transparent;

}

.slot.blocked{

  border-color:#2b2b2b;

  background:#151515;

}

/\* responsivo: reduzir colunas \*/

@media (max-width: 1100px){

  .slot-row{ grid-template-columns: repeat(3, 1fr); }

}

@media (max-width: 780px){

  .ag-left{ width:110px; }

  .slot-row{ grid-template-columns: repeat(2, 1fr); }

}

/\* Responsivo: em telas menores a grade vira 1 coluna \*/

@media (max-width: 1100px){

  .ag-slots{ grid-template-columns: 1fr; }

}

.week-table{ width:100%; border-collapse:collapse; font-size:14px; }

.week-table th, .week-table td{ border:1px solid var(--border); padding:6px; text-align:center; }

.week-time{ width:72px; color:#cbd5e1; font-weight:800; background:#0d0d0d; position:sticky; left:0; }

.week-table td.cell{ cursor:pointer; background:#111; }

.week-table td.cell.active{ background:#11361f; border-color:#2ea66b; box-shadow: inset 0 0 0 1px #2ea66b55; }

/\* ===== Barra de dias (30d) no modal ===== \*/

.day-strip{

  display:flex; gap:8px; overflow:auto; padding:6px 2px 10px;

  margin: 0 0 8px;

}

.day-pill{

  min-width:66px; text-align:center; padding:8px 10px; border-radius:12px;

  background:#131313; border:1px solid #232323; color:#e5e7eb; cursor:pointer;

  line-height:1.1;

}

.day-pill .dow{ display:block; font-size:11px; color:#cbd5e1; }

.day-pill .d{ font-weight:900; font-size:14px; }

.day-pill.active{

  border-color:#3b82f6; background:#0f224a; color:#dbeafe;

}

.day-pill.today{

  outline:2px solid #3b82f6;

}

.day-month{

  display:flex; justify-content:center; align-items:center;

  font-weight:900; color:#e0b25b;

  margin: 4px 0 6px;

}

.day-strip .day-pill{

  background:#151515; border:1px solid #2a2a2a; color:#e5e7eb;

  border-radius:10px; padding:6px 8px; cursor:pointer; font-weight:800;

}

.day-strip .day-pill .dow{ font-size:11px; display:block; color:#cbd5e1; }

.day-strip .day-pill .d{ font-size:14px; }

.day-strip .day-pill.today{ outline:2px solid #3b82f6; }

.day-strip .day-pill.active{ background:#0f224a; border-color:#3b82f6; color:#dbeafe; }

/\* células do grid diário/semana (só o visual do clique) \*/

.week-table{ width:100%; border-collapse: collapse; }

.week-table th, .week-table td{

  border:1px solid #222; padding:8px; font-size:13px;

}

.week-time{ width:64px; color:#cbd5e1; font-weight:800; background:#0f0f0f; }

.week-table td.cell{ background:#131313; cursor:pointer; }

.week-table td.cell.active{ background:#132a13; border-color:#1e3a1e; }

/\* Container geral \*/

.caixa-container {

  max-width: 1000px;

  margin: 20px auto;

  background: #1e1e1e;

  padding: 20px;

  border-radius: 12px;

  box-shadow: 0 4px 15px rgba(0,0,0,0.4);

  color: #fff;

  font-family: Arial, sans-serif;

}

/\* Topo \*/

.caixa-topo {

  display: grid;

  grid-template-columns: 1fr 1fr 1fr;

  gap: 15px;

  margin-bottom: 20px;

}

.caixa-topo label {

  font-size: 14px;

  color: #bbb;

}

.caixa-topo input,

.caixa-topo select {

  width: 100%;

  padding: 8px;

  border-radius: 6px;

  border: 1px solid #444;

  background: #2c2c2c;

  color: #fff;

}

/\* Sessões \*/

.caixa-section {

  margin-bottom: 25px;

  padding: 15px;

  border-radius: 10px;

  background: #252525;

}

.caixa-section h3 {

  margin-bottom: 15px;

  font-size: 18px;

  border-bottom: 1px solid #444;

  padding-bottom: 5px;

  color: #ffcc00;

}

/\* Lista de serviços/produtos \*/

#servicosLista div,

#produtosLista div {

  display: flex;

  align-items: center;

  gap: 10px;

  margin-bottom: 8px;

}

#servicosLista select,

#produtosLista select,

#produtosLista input {

  padding: 6px;

  border-radius: 6px;

  border: 1px solid #444;

  background: #333;

  color: #fff;

}

#produtosLista input[type="number"] {

  width: 60px;

  text-align: center;

}

#produtosLista label {

  font-size: 13px;

}

/\* Botões adicionar/remover \*/

button {

  cursor: pointer;

  border: none;

  border-radius: 6px;

  padding: 6px 12px;

  font-size: 14px;

  transition: 0.2s;

}

.caixa-section button {

  margin-top: 10px;

  background: #3b82f6;

  color: #fff;

}

.caixa-section button:hover {

  background: #2563eb;

}

#servicosLista button,

#produtosLista button {

  background: #dc2626;

  color: #fff;

  font-weight: bold;

}

#servicosLista button:hover,

#produtosLista button:hover {

  background: #b91c1c;

}

/\* Resumo \*/

.resumo p {

  margin: 5px 0;

  font-size: 15px;

}

.resumo b {

  font-size: 18px;

  color: #4ade80;

}

/\* Ações \*/

.caixa-acoes {

  display: flex;

  justify-content: space-between;

  margin-top: 20px;

}

.caixa-acoes .btn {

  flex: 1;

  margin: 0 5px;

  padding: 12px;

  font-size: 16px;

  font-weight: bold;

  border-radius: 8px;

}

.caixa-acoes .btn.cancelar {

  background: #dc2626;

  color: #fff;

}

.caixa-acoes .btn.cancelar:hover {

  background: #b91c1c;

}

.caixa-acoes .btn.rascunho {

  background: #facc15;

  color: #000;

}

.caixa-acoes .btn.rascunho:hover {

  background: #eab308;

}

.caixa-acoes .btn.finalizar {

  background: #22c55e;

  color: #fff;

}

.caixa-acoes .btn.finalizar:hover {

  background: #16a34a;

}

/\* ---------- STYLES ---------- \*/

/\* ===========================

   Barbearia Carreiro – Visual único (todas as telas)

   Mobile-first + upgrades progressivos

   Páginas escopadas: .index (login) e .home (pós-login)

   =========================== \*/

/\* ---------- Tema ---------- \*/

:root{

  --bg:#000;

  --card:#0e0e0e;

  --text:#ffffff;

  --muted:#bdbdbd;

  --primary:#e0b25b;

  --border:#2a2a2a;

}

/\* ---------- Base ---------- \*/

\*{ box-sizing:border-box; }

html,body{ height:100%; }

body{

  margin:0;

  font-family:"Open Sans", Arial, sans-serif;

  background:var(--bg);

  color:var(--text);

  -webkit-font-smoothing:antialiased;

  -moz-osx-font-smoothing:grayscale;

}

/\* Cabeçalho (oculto conforme mock) \*/

header{ display:none; }

/\* Container central padrão (ajustado por página se necessário) \*/

.main-wrap{

  width:100%;

  max-width:360px;        /\* base do mock para telas pequenas \*/

  margin:16px auto 28px;

  padding:0 8px;

  display:flex;

  flex-direction:column;

  align-items:center;

}

/\* ---------- Marca / Logo ---------- \*/

.brand{

  display:flex;

  flex-direction:column;

  align-items:center;

  gap:10px;

  margin-top:8px;

}

.brand img{ width:300px; height:auto; display:block; }

.brand .title{

  font-family:"Archivo Black", sans-serif;

  letter-spacing:2px;

  font-size:18px;

  color:var(--primary);

  text-align:center;

}

.brand .subtitle{

  font-size:12px;

  letter-spacing:3px;

  margin-top:-6px;

  color:var(--primary);

}

/\* ---------- Texto de chamada ---------- \*/

.login-intro{

  text-align:center;

  font-weight:800;

  font-size:18px;

  line-height:1.25;

  margin:18px 8px 16px;

}

/\* ---------- Formulários / Inputs ---------- \*/

.form-card{ width:100%; background:transparent; border-radius:16px; }

form{ display:flex; flex-direction:column; gap:10px; }

.input-field{

  width:100%;

  background:#1f1f1f;

  border:none;

  border-radius:16px;

  padding:14px 16px;

  color:var(--text);

  font-weight:800;

  font-size:14px;

  text-transform:uppercase; /\* placeholder MAIÚSCULO (apenas visual) \*/

}

.input-field::placeholder{ color:#9b9b9b; }

/\* ---------- Botões base ---------- \*/

.btn-cta{

  display:flex;

  align-items:center;

  justify-content:center;

  gap:12px;

  border-radius:16px;

  padding:12px 14px;

  background:#fff;

  color:#111;

  border:2px solid #fff;

  font-weight:900;

  letter-spacing:.5px;

  box-shadow:0 2px 0 rgba(255,255,255,.2);

}

.btn-cta .icon{

  width:28px; height:28px;

  display:flex; align-items:center; justify-content:center;

  border-radius:8px;

  border:2px solid #111;

  background:#fff; color:#111;

  font-size:14px;

}

.btn-google{ background:#fff; border-color:#fff; }

.btn-google .g-logo{ height:20px; }

/\* Links auxiliares \*/

.login-links{

  width:100%;

  display:flex;

  justify-content:space-between;

  margin-top:12px;

  padding:0 4px;

}

.login-links a{

  color:#dcdcdc;

  text-decoration:none;

  font-weight:800;

  font-size:14px;

}

/\* Mensagens de erro \*/

#loginErro, #cadastroErro{

  margin-top:8px;

  font-size:14px;

  color:#ff5b5b;

}

/\* Rodapé \*/

footer{

  margin-top:28px;

  text-align:center;

  color:#d8a94c;

  font-weight:700;

  font-size:14px;

}

/\* ---------- Interações (só onde há hover) ---------- \*/

@media (hover:hover){

  .btn-cta:hover{ transform:translateY(-1px); transition:.15s ease; }

  .login-links a:hover{ text-decoration:underline; }

}

/\* =====================================================

   PÁGINA: INDEX (login/cadastro) – escopo .index

   ===================================================== \*/

/\* Cartão de fundo apenas na página Index \*/

.index .form-wrap{

  width:100%;

  max-width:560px;

  margin:16px auto;

  padding:24px 18px;

  background:#161616;

  border:1px solid rgba(255,255,255,0.12);

  border-radius:20px;

  box-shadow:0 16px 48px rgba(0,0,0,0.75);

}

.index .form-wrap .form-card,

.index .form-wrap #boxCadastro{ width:100%; }

/\* Botões dos formulários do index ocupam 100% do card \*/

.index .btn-cta{ width:100%; }

@media (min-width: 900px){

  .index .form-wrap{ padding:28px 22px; border-radius:22px; }

}

/\* Espaçamento fino entre login/cadastro \*/

#boxCadastro{ margin-top:6px; }

/\* =====================================================

   PÁGINA: HOME (pós-login) – escopo .home

   ===================================================== \*/

.home-greet{

  text-align:center;

  font-weight:900;

  font-size:20px;

  margin:14px 8px 6px;

}

.home-greet #nomeCliente{ letter-spacing:.2px; }

.home-text{

  text-align:center;

  font-weight:900;

  font-size:16px;

  line-height:1.35;

  margin:8px 12px 14px;

  color:#eeeeee;

}

/\* Container da home um pouco mais largo \*/

main.main-wrap.home{

  max-width:min(92vw, 680px);

  padding-left:12px;

  padding-right:12px;

}

/\* Ações (sem card de fundo) \*/

.home .home-actions{

  display:flex;

  flex-direction:column;

  align-items:center;

  gap:18px;

  margin-top:20px;

}

/\* Mesma cor para os dois botões e largura responsiva

   - mobile: quase tela toda (90vw)

   - desktop: ~10–15 cm (máximo 580–600px) \*/

.home .btn-cta,

.home .btn-primary{

  width:clamp(260px, 90vw, 580px);

  padding:16px;

  font-size:20px;

  border-radius:14px;

  background:linear-gradient(180deg, #ffb24a, #e98622);

  color:#1a1205;

  border:2px solid #f6a23a;

  box-shadow:0 8px 22px rgba(233,134,34,.35);

}

/\* Endereço \*/

.home-address{

  margin-top:36px;

  text-align:center;

  font-style:normal;

  color:var(--primary);

  font-weight:900;

  font-size:20px;

}

/\* =====================================================

   Quebras progressivas (comuns às páginas)

   ===================================================== \*/

@media (min-width:420px){

  .main-wrap{ max-width:400px; }

  .brand img{ width:300px; }

}

@media (min-width:640px){

  .main-wrap{ max-width:460px; margin:28px auto 40px; }

  .brand img{ width:400px; }

  .login-intro{ font-size:20px; }

  .input-field{ padding:16px 18px; font-size:15px; }

  /\* mantém padding ligeiramente maior no botão base \*/

  .btn-cta{ padding:14px 18px; }

}

@media (min-width:900px){

  /\* removido o body:flex para evitar footer ao lado \*/

  .brand img{ width:450px; }

  .login-intro{ font-size:22px; }

}

@media (min-width:1200px){

  .main-wrap{ max-width:500px; }

  .brand img{

    width:450px;

    margin-top:50px; /\* espaço extra para não cortar o topo \*/

  }

  .login-intro{ font-size:24px; }

}

/\* =============== HISTÓRICO (página .history) =============== \*/

/\* container um pouco mais largo \*/

main.main-wrap.history{

  max-width: min(92vw, 700px);

  padding-left: 12px;

  padding-right: 12px;

}

.history-title{

  margin: 12px 0 8px;

  text-align:center;

  font-size: 22px;

  font-weight: 900;

}

/\* lista \*/

.history-list{

  list-style: none;

  margin: 10px auto 24px;

  padding: 0;

  width: clamp(280px, 90vw, 600px);   /\* ~10–15 cm no desktop \*/

}

/\* item \*/

.history-item{

  display: grid;

  grid-template-columns: 1fr auto;   /\* data/hora | status \*/

  align-items: center;

  gap: 12px;

  padding: 10px 8px;

  border-radius: 8px;

}

.history-item .when{

  font-weight: 900;

  font-size: 18px;

  letter-spacing: .2px;

}

/\* quadradinhos de status \*/

.status-box{

  width: 18px;

  height: 18px;

  border-radius: 3px;

  border: 2px solid rgba(255,255,255,.15);

}

/\* cores pedidas \*/

.status--ok        { background:#34C759; border-color:#2db750; } /\* verde: compareceu \*/

.status--no        { background:#E53935; border-color:#cf312e; } /\* vermelho: faltou \*/

.status--cancel    { background:#F7D35A; border-color:#e2be49; } /\* amarelo: cancelado \*/

.status--future    { background:#ffffff; border-color:#ffffff; } /\* branco: dia ainda não chegou \*/

/\* botão voltar (central, único) \*/

.back-btn{

  display:block;

  margin: 26px auto 12px;

  width: 64px;

  height: 64px;

  border-radius: 50%;

  border: 2px solid #fff;

  background: #0e0e0e;

  color: #fff;

  font-size: 28px;

  font-weight: 900;

  line-height: 1;

  display:flex; align-items:center; justify-content:center;

  box-shadow: 0 8px 22px rgba(0,0,0,.5);

  cursor: pointer;

}

@media (hover:hover){

  .back-btn:hover{ transform: translateY(-1px); transition:.15s ease; }

}

/\* responsivo \*/

@media (min-width: 900px){

  .history-title{ font-size: 24px; }

  .history-item .when{ font-size: 19px; }

}

/\* ====== BOOKING (todas as etapas) ====== \*/

main.main-wrap.booking{ max-width:min(92vw, 720px); padding:0 12px; }

.booking-title{

  text-align:center; font-size:22px; font-weight:900; margin:12px 0 4px;

}

.booking-sub{ text-align:center; font-size:12px; letter-spacing:.8px; color:#cfcfcf; margin-bottom:12px; }

/\* Lista base \*/

.select-list{ list-style:none; margin:8px auto 24px; padding:0; width:clamp(280px, 90vw, 640px); }

.select-item{

  display:grid; grid-template-columns: auto 1fr auto; align-items:center;

  gap:12px; padding:14px 10px; border-radius:12px;

}

.select-item + .select-item{ margin-top:6px; }

/\* Foto circular (placeholders .avatar/.thumb) \*/

.thumb, .avatar{

  width:52px; height:52px; border-radius:50%; background:#fff; color:#111;

  display:flex; align-items:center; justify-content:center; font-weight:900;

  border:2px solid rgba(255,255,255,.2);

}

/\* Texto da linha \*/

.item-main .title{ font-weight:900; font-size:18px; }

.item-main .price{ color:#9AF267; font-weight:900; margin-top:2px; }

.item-main .meta{ color:#dcdcdc; font-size:12px; margin-top:2px; }

/\* Checkbox visual (apenas aparência; o input fica escondido) \*/

.check{

  width:22px; height:22px; border-radius:3px; border:2px solid #fff; display:inline-block;

}

input[type="radio"].pick{ display:none; }

input[type="radio"].pick:checked + .check{ background:#0f0; border-color:#0f0; }

/\* Barra de navegação inferior (voltar/confirmar) \*/

.booking-nav{

  display:flex; justify-content:center; gap:22px; margin: 12px 0 6px;

}

.btn-round{

  width:64px; height:64px; border-radius:50%; border:2px solid #fff; background:#0e0e0e; color:#fff;

  font-size:28px; font-weight:900; display:flex; align-items:center; justify-content:center; cursor:pointer;

  box-shadow:0 8px 22px rgba(0,0,0,.5);

}

.btn-round.confirm{ background:linear-gradient(180deg, #9AF267, #5ad01d); color:#0a1800; border-color:#9AF267; }

/\* ====== Etapa 3: Calendário + horários ====== \*/

.calendar-wrap{ width:clamp(280px, 90vw, 700px); margin:8px auto 14px; }

.month-label{ text-align:center; font-weight:900; margin-bottom:6px; }

.days-bar{

  display:grid; grid-template-columns: repeat(5,1fr); gap:8px; background:#2b2b2b; padding:10px; border-radius:10px;

}

.day-btn{

  border:2px solid transparent; border-radius:10px; background:#1f1f1f; color:#fff; padding:10px 8px; text-align:center; cursor:pointer;

}

.day-btn .dw{ font-size:12px; color:#d1d1d1; }

.day-btn .dd{ font-size:18px; font-weight:900; }

.day-btn.active{ border-color:#1e90ff; box-shadow:0 0 0 2px rgba(30,144,255,.2) inset; }

.slots{

  width:clamp(280px, 90vw, 700px); margin: 12px auto;

}

.slot-section{ margin-top:10px; }

.slot-header{ display:flex; justify-content:space-between; color:#dcdcdc; margin-bottom:6px; font-weight:900; }

.slot-grid{

  display:grid; grid-template-columns: repeat(3,1fr); gap:10px;

}

.slot{

  border:2px solid #9AF267; border-radius:12px; padding:12px 8px; text-align:center; cursor:pointer; font-weight:900;

}

.slot.disabled{ opacity:.35; cursor:not-allowed; }

.slot.active{ background:#9AF267; color:#0a1800; }

/\* ====== Etapa 4: Quadro de confirmação + modal ====== \*/

.confirm-card{

  width:clamp(280px, 90vw, 700px); margin:8px auto; background:#1e1e1e; border:1px solid #3a3a3a; border-radius:12px; padding:14px;

}

.confirm-row{ display:flex; align-items:center; gap:10px; padding:10px 0; border-bottom:1px solid #2a2a2a; }

.confirm-row:last-child{ border-bottom:0; }

.confirm-row .key{ color:#d0d0d0; width:130px; }

.confirm-row .val{ font-weight:900; }

/\* Aviso tolerância \*/

.tolerance{ margin:8px 0 0; color:#cfcfcf; font-size:14px; }

/\* Modal simples \*/

.modal{

  position:fixed; inset:0; display:none; align-items:center; justify-content:center; background:rgba(0,0,0,.5); z-index:999;

}

.modal .box{

  background:#111; border:1px solid #3a3a3a; border-radius:14px; padding:22px; width:min(92vw, 420px); text-align:center;

}

.modal .box .ok{ font-size:42px; margin-bottom:6px; }

.modal .close{

  position:absolute; top:14px; right:14px; width:36px; height:36px; border-radius:50%; border:2px solid #fff; color:#fff; background:#111; cursor:pointer;

}

. ARQUIVOS JAVASCRIPT .

// ../js/admin-nav.js

document.addEventListener("DOMContentLoaded", () => {

  const links = Array.from(document.querySelectorAll(".admin-nav .nav-item"));

  if (!links.length) return;

  const current = location.pathname.split("/").pop().toLowerCase();

  let active = links.find(a => {

    const href = (a.getAttribute("href") || "").split("/").pop().toLowerCase();

    return current === href;

  }) || links[0];

  links.forEach(a => a.classList.remove("active"));

  active.classList.add("active");

  const color = active.dataset.color || "#fff";

  active.style.outline = `2px solid ${color}`;

  active.style.borderColor = color;

  const dot = active.querySelector(".dot");

  if (dot) dot.style.background = color;

});

// admin.js

document.addEventListener("DOMContentLoaded", () => {

  // ---- navegação entre views (SPA simples) ----

  const navBtns = document.querySelectorAll(".nav-item");

  navBtns.forEach(btn=>{

    btn.addEventListener("click", ()=>{

      navBtns.forEach(b=>b.classList.remove("active"));

      btn.classList.add("active");

      const view = btn.dataset.view;

      document.querySelectorAll(".view").forEach(v=>v.classList.remove("show"));

      document.getElementById(`view-${view}`).classList.add("show");

    });

  });

  // ---- saudação (se quiser ler do Firebase Auth depois) ----

  const nameEl = document.getElementById("adminName");

  try {

    if (window.firebase?.auth) {

      firebase.auth().onAuthStateChanged(u=>{

        if (u?.displayName) nameEl.textContent = u.displayName.split(" ")[0];

      });

    }

  } catch {}

  // ---- DASHBOARD (mock opcional) ----

  const useMock = new URLSearchParams(location.search).has("mock");

  if (useMock) {

    const mock = makeMock();

    renderDashboard(mock);

  } else {

    // deixa placeholders – depois plugamos em Firestore/Cloud Functions

    document.getElementById("chartPagamentos").textContent = "Gráfico – formas de pagamento";

    document.getElementById("chartReceitaServicos").textContent = "Gráfico – receita por serviço";

    document.getElementById("chartEvolucao").textContent = "Linha do tempo – receita/despesa";

  }

});

/\* ===== Mock de dados para visualizar o layout ===== \*/

function makeMock(){

  const entradas = [

    {data:"05/09/2025", tipo:"Serviço", desc:"Corte", qtd:1, forma:"Pix", taxa:0, liquido:30},

    {data:"05/09/2025", tipo:"Produto", desc:"Pomada", qtd:1, forma:"Crédito", taxa:2.1, liquido:27.9},

    {data:"06/09/2025", tipo:"Serviço", desc:"Barba", qtd:1, forma:"Débito", taxa:1.1, liquido:24.7},

    {data:"07/09/2025", tipo:"Serviço", desc:"Combo", qtd:1, forma:"Dinheiro", taxa:0, liquido:60},

  ];

  const saidas = [

    {data:"02/09/2025", desc:"Aluguel", cat:"Fixa", valor:1200, forma:"Pix", sit:"Pago"},

    {data:"03/09/2025", desc:"Compra Pomadas", cat:"Variável", valor:200, forma:"Pix", sit:"Pago"},

    {data:"04/09/2025", desc:"Energia", cat:"Fixa", valor:180, forma:"Boleto", sit:"Em aberto"},

  ];

  const pagar = [

    {venc:"10/09/2025", desc:"Internet", cat:"Fixa", valor:120, status:"Pendente"},

    {venc:"15/09/2025", desc:"Impostos", cat:"Financeira", valor:300, status:"Pendente"},

  ];

  const topServ = [

    {nome:"Corte", qtd:120, receita:3600},

    {nome:"Combo", qtd:40, receita:2400},

    {nome:"Barba", qtd:80, receita:2000},

  ];

  const topProd = [

    {nome:"Pomada", qtd:50, receita:1500},

    {nome:"Shampoo", qtd:20, receita:600},

  ];

  const formas = { Pix:45, Crédito:30, Débito:15, Dinheiro:10 };

  return { entradas, saidas, pagar, topServ, topProd, formas };

}

/\* ===== Render do dashboard com dados \*/

function renderDashboard(d){

  // KPIs

  const receita = sum(d.entradas.map(e=>e.liquido));

  const despesas= sum(d.saidas.map(s=>s.valor));

  const lucro   = receita - despesas;

  const atend   = sum(d.topServ.map(s=>s.qtd)); // aprox. nº de atendimentos

  const ticket  = atend ? receita/atend : 0;

  setText("kpiReceita", toBRL(receita));

  setText("kpiDespesas", toBRL(despesas));

  setText("kpiLucro", toBRL(lucro));

  setText("kpiTicket", toBRL(ticket));

  // ranks

  mountRank("rankServicos", d.topServ);

  mountRank("rankProdutos", d.topProd);

  // tabelas

  fillRows("tabEntradas", d.entradas.map(e=>[

    e.data, e.tipo, e.desc, e.qtd, e.forma, e.taxa ? e.taxa.toFixed(1)+"%" : "—", toBRL(e.liquido)

  ]));

  fillRows("tabSaidas", d.saidas.map(s=>[

    s.data, s.desc, s.cat, toBRL(s.valor), s.forma, s.sit

  ]));

  fillRows("tabPagar", d.pagar.map(p=>[

    p.venc, p.desc, p.cat, toBRL(p.valor), p.status

  ]));

  // indicadores

  const custosDiretos = 200; // mock de custo de mercadoria/insumo

  const lucroBruto = receita - custosDiretos;

  const margem = receita ? (lucro/receita)\*100 : 0;

  setText("indBruto", toBRL(lucroBruto));

  setText("indLiquido", toBRL(lucro));

  setText("indMargem", margem.toFixed(1)+"%");

  setText("indSaldo", toBRL(receita - despesas));

  // placeholders de gráficos (texto amigável)

  setText("chartPagamentos", pieText(d.formas));

  setText("chartReceitaServicos", barText(d.topServ));

  setText("chartEvolucao", "Receita/Despesas por mês (placeholder)");

}

/\* ===== helpers ===== \*/

function setText(id, v){ const el=document.getElementById(id); if(el) el.textContent=v; }

function sum(arr){ return arr.reduce((a,b)=>a+b,0); }

function toBRL(v){ return v.toLocaleString("pt-BR",{style:"currency", currency:"BRL"}); }

function mountRank(id, list){

  const ul = document.getElementById(id); if(!ul) return;

  ul.innerHTML = list.map(i=>`

    <li><span>${i.nome}</span><span>${i.qtd ?? ""} ${i.receita? "• "+toBRL(i.receita):""}</span></li>

  `).join("");

}

function fillRows(tableId, rows){

  const tb = document.querySelector(`#${tableId} tbody`); if(!tb) return;

  tb.innerHTML = rows.map(r=>`<tr>${r.map(c=>`<td>${c}</td>`).join("")}</tr>`).join("");

}

/\* textos simulando gráficos \*/

function pieText(obj){

  const total = Object.values(obj).reduce((a,b)=>a+b,0) || 1;

  return Object.entries(obj).map(([k,v])=>{

    const p = Math.round((v/total)\*100);

    return `${k}: ${p}%`;

  }).join(" • ");

}

function barText(list){

  const max = Math.max(...list.map(x=>x.receita));

  return list.map(x=>{

    const w = Math.round((x.receita/max)\*100);

    return `${x.nome} [${"█".repeat(Math.max(1,Math.round(w/10)))}] ${toBRL(x.receita)}`;

  }).join("\n");

}

// js/agenda-funcionario.js

// Lógica da página de Agenda para a visão restrita do Funcionário.

(function() {

    // --- SIMULAÇÃO DE LOGIN ---

    // Para testes locais, definimos qual funcionário está "logado".

    // Mude o 'id' aqui para o ID de um funcionário que você já tenha cadastrado.

    const mockFuncionarioLogado = { id: "EMP-IGOR", nome: "Igor" };

    // Quando integrar com Firebase, você substituirá isso pelo usuário autenticado real.

    // --- Chaves e Helpers ---

    const LS\_AGEND = "barbearia\_agendamentos\_v1";

    const read = (k, fb) => { try { const v = JSON.parse(localStorage.getItem(k)); return v ?? fb; } catch { return fb; } };

    const write = (k, v) => localStorage.setItem(k, JSON.stringify(v));

    // --- Referências do DOM ---

    const listaAgendaEl = document.getElementById('lista-agenda');

    const headerTitleEl = document.getElementById('header-title');

    // Função principal que carrega e exibe a agenda

    function carregarAgenda() {

        const hoje = new Date();

        const hojeYMD = `${hoje.getFullYear()}-${String(hoje.getMonth() + 1).padStart(2, '0')}-${String(hoje.getDate()).padStart(2, '0')}`;

        headerTitleEl.textContent = `Sua Agenda de Hoje (${hoje.toLocaleDateString('pt-BR')})`;

        // Busca todos os agendamentos do localStorage

        const todosAgendamentos = read(LS\_AGEND, []);

        // Filtra para mostrar apenas os agendamentos do funcionário logado E que são para o dia de hoje

        const agendamentosDoDia = todosAgendamentos.filter(ag => {

            const agendamentoData = new Date(ag.inicioISO);

            const agendamentoYMD = `${agendamentoData.getFullYear()}-${String(agendamentoData.getMonth() + 1).padStart(2, '0')}-${String(agendamentoData.getDate()).padStart(2, '0')}`;

            return ag.profissionalId === mockFuncionarioLogado.id && agendamentoYMD === hojeYMD;

        });

        // Ordena os agendamentos por hora

        agendamentosDoDia.sort((a, b) => new Date(a.inicioISO) - new Date(b.inicioISO));

        if (agendamentosDoDia.length === 0) {

            listaAgendaEl.innerHTML = '<p class="muted" style="text-align: center; padding: 20px;">Nenhum agendamento para hoje.</p>';

            return;

        }

        // Monta o HTML para cada agendamento

        listaAgendaEl.innerHTML = agendamentosDoDia.map(ag => {

            const hora = new Date(ag.inicioISO).toLocaleTimeString([], { hour: '2-digit', minute: '2-digit' });

            const status = ag.status || 'Pendente';

            const acoesVisiveis = status === 'Pendente'; // Só mostra botões se o status ainda for 'Pendente'

            return `

                <div class="agenda-item" data-id="${ag.id}">

                    <div class="agenda-details">

                        <h4>${hora} - ${ag.clienteNome}</h4>

                        <p>Serviço: ${ag.servicoNome}</p>

                        <p>Status: <strong class="status">${status}</strong></p>

                    </div>

                    <div class="agenda-actions">

                        ${acoesVisiveis ? `

                            <button class="btn btn-green btn-compareceu">Compareceu</button>

                            <button class="btn btn-red btn-nao-compareceu">Não Compareceu</button>

                        ` : `<strong>Status Definido!</strong>`}

                    </div>

                </div>

            `;

        }).join('');

        // Adiciona os eventos de clique aos botões de ação

        document.querySelectorAll('.agenda-item').forEach(item => {

            const agendamentoId = item.dataset.id;

            const btnCompareceu = item.querySelector('.btn-compareceu');

            const btnNaoCompareceu = item.querySelector('.btn-nao-compareceu');

            if (btnCompareceu) {

                btnCompareceu.addEventListener('click', () => atualizarStatus(agendamentoId, 'Compareceu', item));

            }

            if (btnNaoCompareceu) {

                btnNaoCompareceu.addEventListener('click', () => atualizarStatus(agendamentoId, 'Não Compareceu', item));

            }

        });

    }

    // Função para atualizar o status de um agendamento no localStorage

    function atualizarStatus(id, novoStatus, itemElement) {

        let ags = read(LS\_AGEND, []);

        const index = ags.findIndex(a => a.id === id);

        if (index > -1) {

            ags[index].status = novoStatus;

            write(LS\_AGEND, ags);

            // Atualiza a interface para dar feedback ao funcionário

            itemElement.querySelector('.status').textContent = novoStatus;

            itemElement.querySelector('.agenda-actions').innerHTML = `<strong>Status Atualizado!</strong>`;

        }

    }

    // Inicia o script quando a página carregar

    carregarAgenda();

})();

/\* ============================================

   AGENDA (Admin) – slots de 30 min + config semanal

   + Overrides por DIA (barra de 30 dias no modal)

   + Cabeçalho de mês acima da barra (Set/Out 2025)

   ============================================ \*/

(function(){

  const LS\_EMP   = 'barbearia\_funcionarios\_v1';

  const LS\_CFG   = 'barbearia\_agenda\_config\_v1';

  const SLOT\_MIN = 30;

  const START\_H  = 10;  // 10:00

  const END\_H    = 20;  // até 20:00 (último slot inicia às 19:30)

  // --------- helpers de data/hora

  const pad2 = n => (n<10?'0':'')+n;

  function toLocalDateInputValue(date){

    const d = new Date(date);

    return `${d.getFullYear()}-${pad2(d.getMonth()+1)}-${pad2(d.getDate())}`;

  }

  function dow1to7(date){ // 1=Seg ... 7=Dom

    const js = (new Date(date)).getDay(); // 0=Dom .. 6=Sab

    return js === 0 ? 7 : js;

  }

  function toYMD(d){ const x=new Date(d); return `${x.getFullYear()}-${pad2(x.getMonth()+1)}-${pad2(x.getDate())}`; }

  function addDays(d, n){ const x=new Date(d); x.setDate(x.getDate()+n); return x; }

  // Meses em PT + formatação do cabeçalho

  const MONTHS\_PT = [

    'janeiro','fevereiro','março','abril','maio','junho',

    'julho','agosto','setembro','outubro','novembro','dezembro'

  ];

  const cap = s => s ? s.charAt(0).toUpperCase() + s.slice(1) : s;

  function monthRangeLabel(startDate, days=30){

    const start = new Date(startDate);

    const end   = addDays(start, days-1);

    const m1 = MONTHS\_PT[start.getMonth()], y1 = start.getFullYear();

    const m2 = MONTHS\_PT[end.getMonth()],   y2 = end.getFullYear();

    if (y1 === y2){

      return (m1 === m2) ? `${cap(m1)} ${y1}` : `${cap(m1)}/${cap(m2)} ${y1}`;

    }

    return `${cap(m1)} ${y1} / ${cap(m2)} ${y2}`;

  }

  // gera array ["10:00","10:30",...,"19:30"]

  function buildTimes(){

    const times = [];

    for(let h=START\_H; h<END\_H; h++){

      times.push(`${pad2(h)}:00`);

      times.push(`${pad2(h)}:30`);

    }

    return times;

  }

  const TIMES = buildTimes();

  const DAYS  = ['Seg','Ter','Qua','Qui','Sex','Sáb','Dom'];

  const DOW\_PT= ['Seg','Ter','Qua','Qui','Sex','Sáb','Dom'];

  // --------- estado / LS

  function getEmpList(){

    try{

      const raw = localStorage.getItem(LS\_EMP);

      const arr = raw ? JSON.parse(raw) : null;

      if(Array.isArray(arr) && arr.length) return arr;

    }catch(e){}

    // fallback mock

    return [

      { id:'EMP-IGOR',   name:'Igor',   photo:'' },

      { id:'EMP-MOISES', name:'Moisés', photo:'' },

    ];

  }

  function loadCfg(){

    try{

      const raw = localStorage.getItem(LS\_CFG);

      const obj = raw ? JSON.parse(raw) : {};

      return (obj && typeof obj==='object') ? obj : {};

    }catch(e){ return {}; }

  }

  function saveCfg(cfg){

    localStorage.setItem(LS\_CFG, JSON.stringify(cfg));

  }

  // garante estrutura do funcionário (semanal + overrides)

  function ensureEmpCfg(cfg, empId){

    if(!cfg[empId]) cfg[empId] = {};

    for(let d=1; d<=7; d++){

      if(!cfg[empId][d]) cfg[empId][d] = {};

    }

    if(!cfg[empId].\_overrides) cfg[empId].\_overrides = {};

  }

  // --------- UI refs

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

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

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

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

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

  const grid       = document.getElementById('agendaGrid');

  const modal      = document.getElementById('modalConfig');

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

  const dayStrip   = document.getElementById('dayStrip'); // barra de dias (se não existir, será ignorado)

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

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

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

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

  // estado runtime

  const EMPs = getEmpList();

  const CFG  = loadCfg();

  let currentDate = new Date();

  let selectedEmp = null;        // {id, name}

  let workingCopy = null;        // estrutura mutável (inclui \_overrides)

  let editingDateYMD = null;     // quando != null, editando um DIA específico

  // --------- boot

  function init(){

    agendaData.value = toLocalDateInputValue(currentDate);

    // select de profissionais

    for(const e of EMPs){

      const opt = document.createElement('option');

      opt.value = e.id; opt.textContent = e.name;

      filtroProf.appendChild(opt);

    }

    // listeners

    agendaData.addEventListener('change', ()=> {

      currentDate = new Date(agendaData.value + 'T12:00:00');

      renderGrid();

    });

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

      currentDate = new Date();

      agendaData.value = toLocalDateInputValue(currentDate);

      renderGrid();

    });

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

      const d = new Date(agendaData.value + 'T12:00:00');

      d.setDate(d.getDate()-1);

      agendaData.value = toLocalDateInputValue(d);

      currentDate = d;

      renderGrid();

    });

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

      const d = new Date(agendaData.value + 'T12:00:00');

      d.setDate(d.getDate()+1);

      agendaData.value = toLocalDateInputValue(d);

      currentDate = d;

      renderGrid();

    });

    filtroProf.addEventListener('change', renderGrid);

    renderGrid();

  }

  // --------- render grid por funcionário (visual “manhã/tarde” como mock)

  function renderGrid(){

    grid.innerHTML = '';

    const filter = filtroProf.value;

    EMPs

      .filter(e => filter==='\*' ? true : e.id===filter)

      .forEach(emp => {

        ensureEmpCfg(CFG, emp.id);

        const row = buildStaffRow(emp);

        grid.appendChild(row);

      });

  }

  function buildStaffRow(emp){

    const row = document.createElement('div');

    row.className = 'ag-row';

    // ESQUERDA: avatar + nome + engrenagem

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

    left.className = 'ag-left';

    left.innerHTML = `

      <div class="ag-avatar">${emp.photo ? `<img src="${emp.photo}" alt="">` : 'foto'}</div>

      <div class="ag-name">

        ${emp.name}

        <button class="btn-gear" title="Configurar agenda" data-emp="${emp.id}">⚙</button>

      </div>

    `;

    row.appendChild(left);

    // DIREITA: faixas Manhã/Tarde com 4 caixinhas cada

    const right = document.createElement('div');

    right.className = 'ag-right';

    right.innerHTML = `

      <div>

        <div class="band-title">Manhã</div>

        <div class="slot-row">${'<div class="slot blocked"></div>'.repeat(4)}</div>

      </div>

      <div>

        <div class="band-title">Tarde</div>

        <div class="slot-row">${'<div class="slot blocked"></div>'.repeat(4)}</div>

      </div>

    `;

    row.appendChild(right);

    // “pinta” conforme disponibilidade do DIA (override > semanal)

    const dow = dow1to7(currentDate);

    const ymd = toYMD(currentDate);

    const baseDay = CFG[emp.id][dow] || {};

    const ovDay   = (CFG[emp.id].\_overrides && CFG[emp.id].\_overrides[ymd]) || {};

    const efetivo = (t)=> ovDay.hasOwnProperty(t) ? !!ovDay[t] : !!baseDay[t];

    const livres = TIMES.filter(t => efetivo(t)).length;

    const totalCaixas = 8; // 4 manhã + 4 tarde

    const pintar = Math.min(totalCaixas, Math.round(livres / TIMES.length \* totalCaixas));

    const boxes = right.querySelectorAll('.slot-row .slot');

    for (let i = 0; i < pintar; i++) boxes[i].classList.remove('blocked');

    // engrenagem abre o modal de configuração

    left.querySelector('.btn-gear').addEventListener('click', () => openConfig(emp));

    return row;

  }

  // --------- MODAL CONFIG (template semanal + overrides diários)

  function openConfig(emp){

    selectedEmp = emp;

    ensureEmpCfg(CFG, emp.id);

    workingCopy = JSON.parse(JSON.stringify(CFG[emp.id])); // inclui \_overrides

    configTitle.textContent = `Configurar agenda de ${emp.name}`;

    ensureDayMonthElement();           // garante cabeçalho de mês

    buildDayStrip();                   // barra 30 dias + mês

    selectDayForEdit(toYMD(currentDate));  // começa no dia atual da agenda

    modal.classList.add('show');

    modal.setAttribute('aria-hidden','false');

  }

  function closeModal(){

    modal.classList.remove('show');

    modal.setAttribute('aria-hidden','true');

    selectedEmp = null;

    workingCopy = null;

    editingDateYMD = null;

    if(weekMatrix) weekMatrix.innerHTML = '';

    if(dayStrip)   dayStrip.innerHTML = '';

    const m = document.getElementById('dayMonth');

    if (m) m.textContent = '';

  }

  closeConfig.addEventListener('click', closeModal);

  cancelConfig.addEventListener('click', closeModal);

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

    if(!selectedEmp || !workingCopy) return;

    CFG[selectedEmp.id] = workingCopy;   // grava do working para config real

    saveCfg(CFG);

    closeModal();

    renderGrid(); // reflete

  });

  // ----- Cria o cabeçalho do mês se não existir

  function ensureDayMonthElement(){

    if (document.getElementById('dayMonth')) return;

    if (!dayStrip || !dayStrip.parentNode) return;

    const div = document.createElement('div');

    div.id = 'dayMonth';

    div.className = 'day-month';

    // insere antes da barra de dias

    dayStrip.parentNode.insertBefore(div, dayStrip);

  }

  // ----- Atualiza o texto do cabeçalho do mês

  function setMonthHeader(baseDate){

    const el = document.getElementById('dayMonth');

    if (!el) return;

    el.textContent = monthRangeLabel(baseDate, 30);

  }

  // ----- Barra de dias (30d)

  function buildDayStrip(){

    if(!dayStrip) return;

    dayStrip.innerHTML = '';

    const base = new Date(currentDate);

    // Cabeçalho de mês

    setMonthHeader(base);

    for(let i=0; i<30; i++){

      const d = addDays(base, i);

      const ymd = toYMD(d);

      const pill = document.createElement('button');

      pill.type = 'button';

      pill.className = 'day-pill';

      pill.setAttribute('role','tab');

      pill.dataset.ymd = ymd;

      pill.innerHTML = `

        <span class="dow">${DOW\_PT[dow1to7(d)-1]}</span>

        <span class="d">${d.getDate()}</span>

      `;

      if(toYMD(new Date()) === ymd) pill.classList.add('today');

      pill.addEventListener('click', () => selectDayForEdit(ymd));

      dayStrip.appendChild(pill);

    }

  }

  function selectDayForEdit(ymd){

    editingDateYMD = ymd;

    // marca ativo

    document.querySelectorAll('.day-pill').forEach(p=>{

      p.classList.toggle('active', p.dataset.ymd===ymd);

    });

    // render grade do DIA

    buildWeekTable(true);

  }

  // ----- Tabela: semanal (7 colunas) ou dia (1 coluna)

  function buildWeekTable(isDay=false){

    const table = document.createElement('table');

    table.className = 'week-table';

    const thead = document.createElement('thead');

    const trh = document.createElement('tr');

    if(isDay){

      trh.innerHTML = `<th class="week-time"></th><th>${editingDateYMD || ''}</th>`;

    }else{

      trh.innerHTML = `<th class="week-time"></th>${DAYS.map(d=>`<th>${d}</th>`).join('')}`;

    }

    thead.appendChild(trh);

    table.appendChild(thead);

    const tbody = document.createElement('tbody');

    if(isDay){

      const d = new Date((editingDateYMD||toYMD(currentDate)) + 'T12:00:00');

      const dow = dow1to7(d);

      const baseDay = (workingCopy[dow] || {});

      const ov = (workingCopy.\_overrides && workingCopy.\_overrides[editingDateYMD]) || {};

      for(const time of TIMES){

        const tr = document.createElement('tr');

        tr.innerHTML = `<th class="week-time">${time}</th>`;

        const td = document.createElement('td');

        td.className = 'cell';

        const on = (ov.hasOwnProperty(time) ? !!ov[time] : !!baseDay[time]);

        if(on) td.classList.add('active');

        td.dataset.mode = 'day';

        td.dataset.time = time;

        td.addEventListener('click', toggleCell);

        tr.appendChild(td);

        tbody.appendChild(tr);

      }

    }else{

      for(const time of TIMES){

        const tr = document.createElement('tr');

        tr.innerHTML = `<th class="week-time">${time}</th>`;

        for(let dow=1; dow<=7; dow++){

          const cell = document.createElement('td');

          cell.className = 'cell';

          if(workingCopy[dow]?.[time]) cell.classList.add('active');

          cell.dataset.dow = String(dow);

          cell.dataset.time = time;

          cell.dataset.mode = 'week';

          cell.addEventListener('click', toggleCell);

          tr.appendChild(cell);

        }

        tbody.appendChild(tr);

      }

    }

    table.appendChild(tbody);

    weekMatrix.innerHTML = '';

    weekMatrix.appendChild(table);

  }

  // ----- Clique nas células

  function toggleCell(e){

    const td  = e.currentTarget;

    const mode= td.dataset.mode;     // 'day' | 'week'

    const tm  = td.dataset.time;

    if(mode === 'week'){

      const dow = Number(td.dataset.dow);

      if(!workingCopy[dow]) workingCopy[dow] = {};

      const cur = !!workingCopy[dow][tm];

      workingCopy[dow][tm] = !cur;

      td.classList.toggle('active', !cur);

    }else{

      // override do DIA

      if(!workingCopy.\_overrides) workingCopy.\_overrides = {};

      if(!workingCopy.\_overrides[editingDateYMD]) workingCopy.\_overrides[editingDateYMD] = {};

      const ov = workingCopy.\_overrides[editingDateYMD];

      const dow = dow1to7(new Date(editingDateYMD));

      const baseDay = (workingCopy[dow] || {});

      const curEffective = ov.hasOwnProperty(tm) ? !!ov[tm] : !!baseDay[tm];

      ov[tm] = !curEffective;

      td.classList.toggle('active', !curEffective);

    }

  }

  // start!

  document.addEventListener('DOMContentLoaded', init);

})();

// agendamento.js

// Estado compartilhado via sessionStorage + utilitários das 4 etapas

const BOOKING\_KEY = "booking";

const Services = [

  { id:"barba",      title:"BARBA COMUM",     price:25,  mins:30,  note:"",               img:"" },

  { id:"barbaterapia",title:"BARBATERAPIA",   price:30,  mins:30,  note:"toalha quente e fria", img:"" },

  { id:"corte",      title:"CORTE",           price:30,  mins:30,  note:"",               img:"" },

  { id:"combo",      title:"CORTE + BARBA + ALISANTE", price:60, mins:60, note:"", img:"" },

];

const Pros = [

  { id:"igor",   name:"Igor",   img:"" },

  { id:"moises", name:"Moisés", img:"" }

];

function saveBooking(patch){

  const cur = JSON.parse(sessionStorage.getItem(BOOKING\_KEY) || "{}");

  sessionStorage.setItem(BOOKING\_KEY, JSON.stringify({ ...cur, ...patch }));

}

function getBooking(){ return JSON.parse(sessionStorage.getItem(BOOKING\_KEY) || "{}"); }

function clearBooking(){ sessionStorage.removeItem(BOOKING\_KEY); }

function fmtMoney(v){ return "R$ " + v.toFixed(2).replace(".", ","); }

function fmtDate(d){

  const dd=String(d.getDate()).padStart(2,"0");

  const mm=String(d.getMonth()+1).padStart(2,"0");

  const yyyy=d.getFullYear();

  return `${dd}/${mm}/${yyyy}`;

}

function weekdayPT(d){

  return ["Dom","Seg","Ter","Qua","Qui","Sex","Sáb"][d.getDay()];

}

function monthLabel(d){

  // Ex.: Setembro/Outubro 2025

  const m1 = d.toLocaleString("pt-BR",{ month:"long" });

  const next = new Date(d); next.setDate(d.getDate()+4);

  const m2 = next.toLocaleString("pt-BR",{ month:"long" });

  const y  = next.getFullYear();

  return (m1===m2? m1 : `${m1}/${m2}`) + " " + y;

}

/\* ====== ETAPA 1: Serviços ====== \*/

function initStep1(){

  const ul = document.getElementById("serviceList");

  // monta lista

  ul.innerHTML = Services.map(s => `

    <li class="select-item">

      <div class="thumb">${s.img? `<img src="${s.img}" alt="" />` : 'FOTO'}</div>

      <div class="item-main">

        <div class="title">${s.title} ${s.note? `<span class="meta">– ${s.note}</span>`:''}</div>

        <div class="price">${fmtMoney(s.price)}</div>

        <div class="meta">⏱ ${s.mins} min</div>

      </div>

      <label>

        <input type="radio" name="service" class="pick" value="${s.id}">

        <span class="check"></span>

      </label>

    </li>

  `).join("");

  // seleção única

  ul.addEventListener("change", (e)=>{

    if(e.target && e.target.matches("input.pick")){

      const s = Services.find(x=>x.id===e.target.value);

      saveBooking({ serviceId:s.id, serviceTitle:s.title, servicePrice:s.price, serviceMins:s.mins });

    }

  });

  // navegação

  document.getElementById("goNext").addEventListener("click", ()=>{

    const b = getBooking();

    if(!b.serviceId){ alert("Selecione um serviço."); return; }

    window.location.href = "agendar-profissional.html";

  });

  document.getElementById("goBack").addEventListener("click", ()=> window.location.href="home.html");

}

/\* ====== ETAPA 2: Profissionais ====== \*/

function initStep2(){

  const ul = document.getElementById("proList");

  ul.innerHTML = Pros.map(p => `

    <li class="select-item">

      <div class="avatar">${p.img? `<img src="${p.img}" alt="" />` : 'FOTO'}</div>

      <div class="item-main"><div class="title">${p.name}</div></div>

      <label>

        <input type="radio" name="pro" class="pick" value="${p.id}">

        <span class="check"></span>

      </label>

    </li>

  `).join("");

  ul.addEventListener("change", (e)=>{

    if(e.target && e.target.matches("input.pick")){

      const p = Pros.find(x=>x.id===e.target.value);

      saveBooking({ proId:p.id, proName:p.name });

    }

  });

  document.getElementById("goNext").addEventListener("click", ()=>{

    const b = getBooking();

    if(!b.proId){ alert("Selecione um profissional."); return; }

    window.location.href = "agendar-horario.html";

  });

  document.getElementById("goBack").addEventListener("click", ()=> history.back());

}

/\* ====== ETAPA 3: Calendário e horários ====== \*/

function initStep3(){

  const start = new Date(); // hoje

  start.setHours(0,0,0,0);

  const monthEl = document.getElementById("monthLabel");

  const bar     = document.getElementById("daysBar");

  const manhaCt = document.getElementById("slotsMorning");

  const tardeCt = document.getElementById("slotsAfternoon");

  let activeDay = new Date(start);

  function renderBar(base){

    monthEl.textContent = monthLabel(base);

    bar.innerHTML = "";

    for(let i=0;i<5;i++){

      const d = new Date(base); d.setDate(base.getDate()+i);

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

      btn.className = "day-btn" + (i===0 ? " active": "");

      btn.innerHTML = `<div class="dw">${weekdayPT(d)}</div><div class="dd">${String(d.getDate()).padStart(2,"0")}</div>`;

      btn.addEventListener("click", ()=>{

        [...bar.children].forEach(x=>x.classList.remove("active"));

        btn.classList.add("active");

        activeDay = d;

        renderSlots();

      });

      bar.appendChild(btn);

    }

  }

  function renderSlots(){

    // MOCK de horários: manhã 09:00–11:30, tarde 13:00–17:30 (a cada 30 min)

    const mk = (hStart, hEnd)=>{

      const list=[]; const d=new Date(activeDay);

      for(let h=hStart; h<=hEnd; h+=0.5){

        const hh = Math.floor(h);

        const mm = (h%1)?30:0;

        const t = new Date(d); t.setHours(hh,mm,0,0);

        list.push({ label: `${String(hh).padStart(2,"0")}:${String(mm).padStart(2,"0")}`, disabled: false, date: t });

      }

      return list;

    };

    const morning = mk(9,11.5);

    const afternoon= mk(13,17.5);

    // Exemplo de alguns desabilitados (mock)

    morning[2].disabled = true;

    afternoon[1].disabled = true;

    const draw = (ct, arr)=>{

      ct.innerHTML = "";

      arr.forEach(s=>{

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

        div.className = "slot" + (s.disabled? " disabled": "");

        div.textContent = s.label;

        if(!s.disabled){

          div.addEventListener("click", ()=>{

            [...ct.children].forEach(x=>x.classList.remove("active"));

            div.classList.add("active");

            saveBooking({ dateISO: s.date.toISOString(), timeLabel: s.label, weekday: weekdayPT(s.date), dateLabel: fmtDate(s.date) });

          });

        }

        ct.appendChild(div);

      });

    };

    document.getElementById("morningCount").textContent  = morning.filter(x=>!x.disabled).length + " horários disponíveis";

    document.getElementById("afternoonCount").textContent= afternoon.filter(x=>!x.disabled).length + " horários disponíveis";

    draw(manhaCt, morning);

    draw(tardeCt, afternoon);

  }

  renderBar(start);

  renderSlots();

  document.getElementById("goNext").addEventListener("click", ()=>{

    const b = getBooking();

    if(!b.dateISO){ alert("Selecione um dia e um horário."); return; }

    window.location.href = "agendar-confirmar.html";

  });

  document.getElementById("goBack").addEventListener("click", ()=> history.back());

}

/\* ====== ETAPA 4: Confirmação + Modal ====== \*/

function initStep4(){

  const b = getBooking();

  if(!b.serviceId || !b.proId || !b.dateISO){ window.location.href="agendar-servico.html"; return; }

  document.getElementById("cf-service").textContent = `${b.serviceTitle} (${fmtMoney(b.servicePrice)})`;

  document.getElementById("cf-pro").textContent     = b.proName;

  document.getElementById("cf-date").textContent    = `${b.dateLabel} (${b.weekday})`;

  document.getElementById("cf-time").textContent    = `${b.timeLabel}  (15 min tolerância)`;

  document.getElementById("goBack").addEventListener("click", ()=> history.back());

  // confirmar → (no futuro: salvar no Firestore)

  document.getElementById("goNext").addEventListener("click", ()=>{

    const obs = (document.getElementById("obs")?.value || "").trim();

    saveBooking({ obs });

    // Abre modal

    const modal = document.getElementById("okModal");

    modal.style.display = "flex";

    modal.querySelector(".close").onclick = close;

    function close(){

      modal.style.display = "none";

      clearBooking();

      window.location.href = "home.html";

    }

  });

}

/\* Expose \*/

window.BookingSteps = { initStep1, initStep2, initStep3, initStep4 };

// js/caixa-funcionario.js

(function() {

    // --- SIMULAÇÃO DE LOGIN DO FUNCIONÁRIO ---

    // Para testes locais, definimos qual funcionário está "logado".

    // Mude o 'id' para testar com outros funcionários que você cadastrou.

    const mockFuncionarioLogado = { id: "EMP-IGOR", nome: "Igor" };

    // Quando integrar com Firebase, você substituirá isso pelo usuário autenticado real.

    // Chaves do localStorage

    const LS\_CLIENTES = "barbearia\_clientes\_v1";

    const LS\_FUNC = "barbearia\_funcionarios\_v1";

    const LS\_SERV = "barbearia\_servicos\_v2";

    const LS\_PROD = "barbearia\_produtos\_v1";

    const LS\_LANC = "barbearia\_lancamentos\_v1";

    const LS\_KARDEX = "barbearia\_kardex\_v1";

    const LS\_CONF = "barbearia\_config\_v1";

    // Helpers

    const read = (k, fb) => { try { const v = JSON.parse(localStorage.getItem(k)); return v ?? fb; } catch { return fb; } };

    const write = (k, v) => localStorage.setItem(k, JSON.stringify(v));

    const fmtBRL = n => new Intl.NumberFormat("pt-BR", { style: "currency", currency: "BRL" }).format(n || 0);

    // Estado e Dados

    let clientes = [];

    let funcionarios = [];

    let servicos = [];

    let produtos = [];

    let config = {};

    // Referências do DOM

    const selCliente = document.getElementById("caixaCliente");

    const selProfissional = document.getElementById("caixaProfissional");

    const inputData = document.getElementById("caixaData");

    const listaServicos = document.getElementById("servicosLista");

    const listaProdutos = document.getElementById("produtosLista");

    const selPagamento = document.getElementById("formaPagamento");

    function carregarDadosIniciais() {

        clientes = read(LS\_CLIENTES, []);

        funcionarios = read(LS\_FUNC, []);

        servicos = read(LS\_SERV, []).filter(s => s.tipoServico === 'unitario');

        produtos = read(LS\_PROD, []);

        const DEFAULT\_CONF = { taxas: { Dinheiro: 0, Pix: 0, Debito: 1.5, CreditoAVista: 3.0, CreditoParcelado: 4.0 } };

        config = { ...DEFAULT\_CONF, ...read(LS\_CONF, {}) };

        const preencherSelect = (select, dados, vKey = 'id', tKey = 'name') => {

            select.innerHTML = `<option value="">Selecione...</option>`;

            dados.forEach(item => {

                select.innerHTML += `<option value="${item[vKey]}">${item[tKey]}</option>`;

            });

        };

        preencherSelect(selCliente, clientes);

        preencherSelect(selProfissional, funcionarios);

        // --- LÓGICA PRINCIPAL DESTA PÁGINA ---

        // Trava o select de profissional no funcionário que está "logado"

        if (mockFuncionarioLogado && mockFuncionarioLogado.id) {

            selProfissional.value = mockFuncionarioLogado.id;

            // O 'disabled' já está no HTML, mas garantimos aqui também.

            selProfissional.disabled = true;

        } else {

            alert("ERRO: Não foi possível identificar o funcionário logado.");

        }

        // --- FIM DA LÓGICA ---

    }

    //

    // O RESTANTE DO CÓDIGO É IDÊNTICO AO 'caixa.js' DO ADMIN

    // (copie e cole as funções de adicionar linhas, recalcular, finalizar venda, etc.)

    //

    function adicionarLinhaServico() {

        const id = `serv\_${Math.random().toString(36).slice(2, 9)}`;

        const div = document.createElement('div');

        div.className = 'item-lista-row';

        div.id = id;

        div.innerHTML = `

            <select class="servico-select" style="grid-column: 1 / 4;">

                <option value="">Selecione um serviço...</option>

                ${servicos.map(s => `<option value="${s.id}" data-valor="${s.preco}">${s.nome}</option>`).join('')}

            </select>

            <input type="text" class="servico-valor" value="${fmtBRL(0)}" disabled />

            <div class="del-btn" title="Remover">🗑️</div>

        `;

        listaServicos.appendChild(div);

        div.querySelector('.servico-select').addEventListener('change', (e) => {

            const select = e.target;

            const valor = select.options[select.selectedIndex].dataset.valor || 0;

            div.querySelector('.servico-valor').value = fmtBRL(Number(valor));

            recalcularTudo();

        });

        div.querySelector('.del-btn').addEventListener('click', () => {

            div.remove();

            recalcularTudo();

        });

    }

    function adicionarLinhaProduto() {

        const id = `prod\_${Math.random().toString(36).slice(2, 9)}`;

        const div = document.createElement('div');

        div.className = 'item-lista-row';

        div.id = id;

        div.innerHTML = `

            <select class="produto-select">

                <option value="">Selecione um produto...</option>

                ${produtos.map(p => `<option value="${p.id}" data-valor="${p.precoVenda}">${p.nome} (Estoque: ${p.estoqueAtual || 0})</option>`).join('')}

            </select>

            <input type="number" class="produto-qtd" value="1" min="1" />

            <div>

                <input type="checkbox" id="brinde\_${id}" class="produto-brinde" />

                <label for="brinde\_${id}">Brinde</label>

            </div>

            <input type="text" class="produto-valor-total" value="${fmtBRL(0)}" disabled />

            <div class="del-btn" title="Remover">🗑️</div>

        `;

        listaProdutos.appendChild(div);

        const recalcularLinhaProduto = () => {

            const select = div.querySelector('.produto-select');

            const valorUnitario = select.options[select.selectedIndex].dataset.valor || 0;

            const qtd = div.querySelector('.produto-qtd').value || 1;

            const isBrinde = div.querySelector('.produto-brinde').checked;

            const total = isBrinde ? 0 : Number(valorUnitario) \* Number(qtd);

            div.querySelector('.produto-valor-total').value = fmtBRL(total);

            recalcularTudo();

        };

        div.querySelector('.produto-select').addEventListener('change', recalcularLinhaProduto);

        div.querySelector('.produto-qtd').addEventListener('input', recalcularLinhaProduto);

        div.querySelector('.produto-brinde').addEventListener('change', recalcularLinhaProduto);

        div.querySelector('.del-btn').addEventListener('click', () => {

            div.remove();

            recalcularTudo();

        });

        recalcularLinhaProduto();

    }

    function recalcularTudo() {

        let totalServicos = 0;

        document.querySelectorAll('#servicosLista .item-lista-row').forEach(row => {

            const select = row.querySelector('.servico-select');

            const valor = select.options[select.selectedIndex]?.dataset.valor || 0;

            totalServicos += Number(valor);

        });

        let totalProdutos = 0;

        document.querySelectorAll('#produtosLista .item-lista-row').forEach(row => {

            const valorTotalInput = row.querySelector('.produto-valor-total').value;

            totalProdutos += parseFloat(valorTotalInput.replace('R$', '').replace(/\./g, '').replace(',', '.')) || 0;

        });

        const subtotal = totalServicos + totalProdutos;

        const taxaPercentual = config.taxas[selPagamento.value] || 0;

        const valorTaxa = (subtotal \* taxaPercentual) / 100;

        const totalFinal = subtotal - valorTaxa;

        document.getElementById('resumoServicos').textContent = fmtBRL(totalServicos);

        document.getElementById('resumoProdutos').textContent = fmtBRL(totalProdutos);

        document.getElementById('resumoSubtotal').textContent = fmtBRL(subtotal);

        document.getElementById('resumoTaxa').textContent = `${fmtBRL(valorTaxa)} (${taxaPercentual}%)`;

        document.getElementById('resumoTotalFinal').textContent = fmtBRL(totalFinal);

    }

    function finalizarVenda() {

        if (!selCliente.value || !selProfissional.value) {

            return alert("Selecione um cliente e um profissional.");

        }

        const servicosAdicionados = Array.from(document.querySelectorAll('#servicosLista .item-lista-row'));

        const produtosAdicionados = Array.from(document.querySelectorAll('#produtosLista .item-lista-row'));

        if (servicosAdicionados.length === 0 && produtosAdicionados.length === 0) {

            return alert("Adicione pelo menos um serviço ou produto.");

        }

        const lancamentos = read(LS\_LANC, []);

        const funcionario = funcionarios.find(f => f.id === selProfissional.value);

        const novoLancamento = {

            id: `lan\_${Math.random().toString(16).slice(2, 8)}`,

            dataISO: new Date(inputData.value).toISOString(),

            profissionalId: selProfissional.value,

            profissionalNome: selProfissional.options[selProfissional.selectedIndex].text,

            clienteId: selCliente.value,

            clienteNome: selCliente.options[selCliente.selectedIndex].text,

            tipo: "venda\_pdv",

            servico: [],

            produtos: [],

            pagamento: { forma: selPagamento.value, parcelas: 1 },

            valores: {},

            obs: "Venda registrada via PDV"

        };

        const servicosDaVenda = servicosAdicionados.map(row => {

            const select = row.querySelector('.servico-select');

            const servicoInfo = servicos.find(s => s.id === select.value);

            return { id: servicoInfo.id, nome: servicoInfo.nome, precoFinal: Number(servicoInfo.preco), custoMateriais: Number(servicoInfo.custoTotal || 0) };

        });

        const produtosDaVenda = produtosAdicionados.map(row => {

            const select = row.querySelector('.produto-select');

            const produtoInfo = produtos.find(p => p.id === select.value);

            const qtd = Number(row.querySelector('.produto-qtd').value);

            const isBrinde = row.querySelector('.produto-brinde').checked;

            return { produtoId: produtoInfo.id, nome: produtoInfo.nome, quantidade: qtd, brinde: isBrinde, precoUnitarioAplicado: isBrinde ? 0 : Number(produtoInfo.precoVenda), custoUnitarioAplicado: Number(produtoInfo.cmp || 0) };

        });

        novoLancamento.servico = servicosDaVenda;

        novoLancamento.produtos = produtosDaVenda;

        const receitaServ = servicosDaVenda.reduce((acc, s) => acc + s.precoFinal, 0);

        const receitaProd = produtosDaVenda.reduce((acc, p) => acc + (p.precoUnitarioAplicado \* p.quantidade), 0);

        const receitaBruta = receitaServ + receitaProd;

        const taxa = (receitaBruta \* (config.taxas[selPagamento.value] || 0)) / 100;

        let comissao = 0;

        if (funcionario?.commissionType === "percentual") { comissao = receitaServ \* (Number(funcionario.commissionValue || 0) / 100); }

        else if (funcionario?.commissionType === "valor\_fixo") { comissao = servicosDaVenda.length \* Number(funcionario.commissionValue || 0); }

        const custoServ = servicosDaVenda.reduce((acc, s) => acc + s.custoMateriais, 0);

        const custoProd = produtosDaVenda.reduce((acc, p) => acc + (p.custoUnitarioAplicado \* p.quantidade), 0);

        const custoTotal = custoServ + custoProd;

        novoLancamento.valores = { receitaBruta, taxaPagamento: taxa, comissao, custoTotal, resultadoLiquido: receitaBruta - taxa - comissao - custoTotal };

        lancamentos.push(novoLancamento);

        write(LS\_LANC, lancamentos);

        const kardex = read(LS\_KARDEX, []);

        produtosDaVenda.forEach(itemVendido => {

            const produtoDB = produtos.find(p => p.id === itemVendido.produtoId);

            if (produtoDB) {

                produtoDB.estoqueAtual = (produtoDB.estoqueAtual || 0) - itemVendido.quantidade;

                kardex.push({ id: `mov\_${Math.random().toString(16).slice(2, 8)}`, productId: itemVendido.produtoId, tipo: "saida", motivo: itemVendido.brinde ? "brinde" : "venda", dataISO: new Date().toISOString(), quantidade: itemVendido.quantidade, custoUnitario: itemVendido.custoUnitarioAplicado });

            }

        });

        write(LS\_PROD, produtos);

        write(LS\_KARDEX, kardex);

        alert("Venda finalizada com sucesso!");

        resetarFormulario();

    }

    function resetarFormulario() {

        selCliente.value = "";

        listaServicos.innerHTML = "";

        listaProdutos.innerHTML = "";

        selPagamento.value = "Dinheiro";

        definirDataHoraAtual();

        recalcularTudo();

        // Não reseta o profissional, pois ele é fixo

    }

    function definirDataHoraAtual() {

        const agora = new Date();

        agora.setMinutes(agora.getMinutes() - agora.getTimezoneOffset());

        inputData.value = agora.toISOString().slice(0, 16);

    }

    document.addEventListener("DOMContentLoaded", () => {

        carregarDadosIniciais();

        resetarFormulario();

        setInterval(definirDataHoraAtual, 1000);

        document.getElementById('addServico').addEventListener('click', adicionarLinhaServico);

        document.getElementById('addProduto').addEventListener('click', adicionarLinhaProduto);

        selPagamento.addEventListener('change', recalcularTudo);

        document.getElementById('btnFinalizar').addEventListener('click', finalizarVenda);

        document.getElementById('btnCancelar').addEventListener('click', resetarFormulario);

    });

})();

// js/caixa.js

(function() {

    // Chaves do localStorage (consistente com outros módulos)

    const LS\_CLIENTES = "barbearia\_clientes\_v1";

    const LS\_FUNC = "barbearia\_funcionarios\_v1";

    const LS\_SERV = "barbearia\_servicos\_v2";

    const LS\_PROD = "barbearia\_produtos\_v1";

    const LS\_LANC = "barbearia\_lancamentos\_v1";

    const LS\_KARDEX = "barbearia\_kardex\_v1";

    const LS\_CONF = "barbearia\_config\_v1";

    // Helpers

    const read = (k, fb) => { try { const v = JSON.parse(localStorage.getItem(k)); return v ?? fb; } catch { return fb; } };

    const write = (k, v) => localStorage.setItem(k, JSON.stringify(v));

    const fmtBRL = n => new Intl.NumberFormat("pt-BR", { style: "currency", currency: "BRL" }).format(n || 0);

    // Estado e Dados

    let clientes = [];

    let funcionarios = [];

    let servicos = [];

    let produtos = [];

    let config = {};

    // Referências do DOM

    const selCliente = document.getElementById("caixaCliente");

    const selProfissional = document.getElementById("caixaProfissional");

    const inputData = document.getElementById("caixaData");

    const listaServicos = document.getElementById("servicosLista");

    const listaProdutos = document.getElementById("produtosLista");

    const selPagamento = document.getElementById("formaPagamento");

    function carregarDadosIniciais() {

        clientes = read(LS\_CLIENTES, []);

        funcionarios = read(LS\_FUNC, []);

        servicos = read(LS\_SERV, []).filter(s => s.tipoServico === 'unitario'); // Apenas serviços unitários no PDV

        produtos = read(LS\_PROD, []);

        const DEFAULT\_CONF = { taxas: { Dinheiro: 0, Pix: 0, Debito: 1.5, CreditoAVista: 3.0, CreditoParcelado: 4.0 } };

        config = { ...DEFAULT\_CONF, ...read(LS\_CONF, {}) };

        const preencherSelect = (select, dados, vKey = 'id', tKey = 'name') => {

            select.innerHTML = `<option value="">Selecione...</option>`;

            dados.forEach(item => {

                select.innerHTML += `<option value="${item[vKey]}">${item[tKey]}</option>`;

            });

        };

        preencherSelect(selCliente, clientes);

        preencherSelect(selProfissional, funcionarios);

    }

    function adicionarLinhaServico() {

        const id = `serv\_${Math.random().toString(36).slice(2, 9)}`;

        const div = document.createElement('div');

        div.className = 'item-lista-row';

        div.id = id;

        div.innerHTML = `

            <select class="servico-select" style="grid-column: 1 / 4;">

                <option value="">Selecione um serviço...</option>

                ${servicos.map(s => `<option value="${s.id}" data-valor="${s.preco}">${s.nome}</option>`).join('')}

            </select>

            <input type="text" class="servico-valor" value="${fmtBRL(0)}" disabled />

            <div class="del-btn" title="Remover">🗑️</div>

        `;

        listaServicos.appendChild(div);

        div.querySelector('.servico-select').addEventListener('change', (e) => {

            const select = e.target;

            const valor = select.options[select.selectedIndex].dataset.valor || 0;

            div.querySelector('.servico-valor').value = fmtBRL(Number(valor));

            recalcularTudo();

        });

        div.querySelector('.del-btn').addEventListener('click', () => {

            div.remove();

            recalcularTudo();

        });

    }

    function adicionarLinhaProduto() {

        const id = `prod\_${Math.random().toString(36).slice(2, 9)}`;

        const div = document.createElement('div');

        div.className = 'item-lista-row';

        div.id = id;

        div.innerHTML = `

            <select class="produto-select">

                <option value="">Selecione um produto...</option>

                ${produtos.map(p => `<option value="${p.id}" data-valor="${p.precoVenda}">${p.nome} (Estoque: ${p.estoqueAtual || 0})</option>`).join('')}

            </select>

            <input type="number" class="produto-qtd" value="1" min="1" />

            <div>

                <input type="checkbox" id="brinde\_${id}" class="produto-brinde" />

                <label for="brinde\_${id}">Brinde</label>

            </div>

            <input type="text" class="produto-valor-total" value="${fmtBRL(0)}" disabled />

            <div class="del-btn" title="Remover">🗑️</div>

        `;

        listaProdutos.appendChild(div);

        const recalcularLinhaProduto = () => {

            const select = div.querySelector('.produto-select');

            const valorUnitario = select.options[select.selectedIndex].dataset.valor || 0;

            const qtd = div.querySelector('.produto-qtd').value || 1;

            const isBrinde = div.querySelector('.produto-brinde').checked;

            const total = isBrinde ? 0 : Number(valorUnitario) \* Number(qtd);

            div.querySelector('.produto-valor-total').value = fmtBRL(total);

            recalcularTudo();

        };

        div.querySelector('.produto-select').addEventListener('change', recalcularLinhaProduto);

        div.querySelector('.produto-qtd').addEventListener('input', recalcularLinhaProduto);

        div.querySelector('.produto-brinde').addEventListener('change', recalcularLinhaProduto);

        div.querySelector('.del-btn').addEventListener('click', () => {

            div.remove();

            recalcularTudo();

        });

        // Trigger a primeira vez para preencher o valor

        recalcularLinhaProduto();

    }

    function recalcularTudo() {

        let totalServicos = 0;

        document.querySelectorAll('#servicosLista .item-lista-row').forEach(row => {

            const select = row.querySelector('.servico-select');

            const valor = select.options[select.selectedIndex]?.dataset.valor || 0;

            totalServicos += Number(valor);

        });

        let totalProdutos = 0;

        document.querySelectorAll('#produtosLista .item-lista-row').forEach(row => {

            const valorTotalInput = row.querySelector('.produto-valor-total').value;

            totalProdutos += parseFloat(valorTotalInput.replace('R$', '').replace('.', '').replace(',', '.')) || 0;

        });

        const subtotal = totalServicos + totalProdutos;

        const taxaPercentual = config.taxas[selPagamento.value] || 0;

        const valorTaxa = (subtotal \* taxaPercentual) / 100;

        const totalFinal = subtotal - valorTaxa;

        document.getElementById('resumoServicos').textContent = fmtBRL(totalServicos);

        document.getElementById('resumoProdutos').textContent = fmtBRL(totalProdutos);

        document.getElementById('resumoSubtotal').textContent = fmtBRL(subtotal);

        document.getElementById('resumoTaxa').textContent = `${fmtBRL(valorTaxa)} (${taxaPercentual}%)`;

        document.getElementById('resumoTotalFinal').textContent = fmtBRL(totalFinal);

    }

    function finalizarVenda() {

        // Validações

        if (!selCliente.value || !selProfissional.value) {

            return alert("Selecione um cliente e um profissional.");

        }

        const servicosAdicionados = Array.from(document.querySelectorAll('#servicosLista .item-lista-row'));

        const produtosAdicionados = Array.from(document.querySelectorAll('#produtosLista .item-lista-row'));

        if (servicosAdicionados.length === 0 && produtosAdicionados.length === 0) {

            return alert("Adicione pelo menos um serviço ou produto.");

        }

        const lancamentos = read(LS\_LANC, []);

        const funcionario = funcionarios.find(f => f.id === selProfissional.value);

        // Estrutura baseada no `lancamentos.js` e na sua solicitação

        const novoLancamento = {

            id: `lan\_${Math.random().toString(16).slice(2, 8)}`,

            dataISO: new Date(inputData.value).toISOString(),

            profissionalId: selProfissional.value,

            profissionalNome: selProfissional.options[selProfissional.selectedIndex].text,

            clienteId: selCliente.value,

            clienteNome: selCliente.options[selCliente.selectedIndex].text,

            tipo: "venda\_pdv",

            servico: null, // Simplificado para o PDV; lógica de múltiplos serviços abaixo

            produtos: [],

            pagamento: { forma: selPagamento.value, parcelas: 1 },

            valores: {},

            obs: "Venda registrada via PDV"

        };

        // Coletar serviços

        const servicosDaVenda = servicosAdicionados.map(row => {

            const select = row.querySelector('.servico-select');

            const servicoInfo = servicos.find(s => s.id === select.value);

            return {

                id: servicoInfo.id,

                nome: servicoInfo.nome,

                precoFinal: Number(servicoInfo.preco),

                custoMateriais: Number(servicoInfo.custoTotal || 0)

            };

        });

        // Coletar produtos

        const produtosDaVenda = produtosAdicionados.map(row => {

            const select = row.querySelector('.produto-select');

            const produtoInfo = produtos.find(p => p.id === select.value);

            const qtd = Number(row.querySelector('.produto-qtd').value);

            const isBrinde = row.querySelector('.produto-brinde').checked;

            return {

                produtoId: produtoInfo.id,

                nome: produtoInfo.nome,

                quantidade: qtd,

                brinde: isBrinde,

                precoUnitarioAplicado: isBrinde ? 0 : Number(produtoInfo.precoVenda),

                custoUnitarioAplicado: Number(produtoInfo.cmp || 0)

            };

        });

        novoLancamento.servico = servicosDaVenda; // Usando o campo 'servico' para a lista

        novoLancamento.produtos = produtosDaVenda;

        // Recalcular valores para o objeto de lançamento (consistente com `lancamentos.js`)

        const receitaServ = servicosDaVenda.reduce((acc, s) => acc + s.precoFinal, 0);

        const receitaProd = produtosDaVenda.reduce((acc, p) => acc + (p.precoUnitarioAplicado \* p.quantidade), 0);

        const receitaBruta = receitaServ + receitaProd;

        const taxa = (receitaBruta \* (config.taxas[selPagamento.value] || 0)) / 100;

        let comissao = 0;

        if (funcionario?.commissionType === "percentual") {

            comissao = receitaServ \* (Number(funcionario.commissionValue || 0) / 100);

        } else if (funcionario?.commissionType === "valor\_fixo") {

            comissao = servicosDaVenda.length \* Number(funcionario.commissionValue || 0);

        }

        const custoServ = servicosDaVenda.reduce((acc, s) => acc + s.custoMateriais, 0);

        const custoProd = produtosDaVenda.reduce((acc, p) => acc + (p.custoUnitarioAplicado \* p.quantidade), 0);

        const custoTotal = custoServ + custoProd;

        novoLancamento.valores = {

            receitaBruta: receitaBruta,

            taxaPagamento: taxa,

            comissao: comissao,

            custoTotal: custoTotal,

            resultadoLiquido: receitaBruta - taxa - comissao - custoTotal

        };

        // Salvar Lançamento

        lancamentos.push(novoLancamento);

        write(LS\_LANC, lancamentos);

        // Atualizar Estoque e Kardex

        const kardex = read(LS\_KARDEX, []);

        produtosDaVenda.forEach(itemVendido => {

            const produtoDB = produtos.find(p => p.id === itemVendido.produtoId);

            if (produtoDB) {

                produtoDB.estoqueAtual = (produtoDB.estoqueAtual || 0) - itemVendido.quantidade;

                kardex.push({

                    id: `mov\_${Math.random().toString(16).slice(2, 8)}`,

                    productId: itemVendido.produtoId,

                    tipo: "saida",

                    motivo: itemVendido.brinde ? "brinde" : "venda",

                    dataISO: new Date().toISOString(),

                    quantidade: itemVendido.quantidade,

                    custoUnitario: itemVendido.custoUnitarioAplicado

                });

            }

        });

        write(LS\_PROD, produtos);

        write(LS\_KARDEX, kardex);

        alert("Venda finalizada com sucesso! O registro foi criado em Lançamentos.");

        resetarFormulario();

    }

    function resetarFormulario() {

        selCliente.value = "";

        selProfissional.value = "";

        listaServicos.innerHTML = "";

        listaProdutos.innerHTML = "";

        selPagamento.value = "Dinheiro";

        definirDataHoraAtual();

        recalcularTudo();

    }

    function definirDataHoraAtual() {

        const agora = new Date();

        // Ajusta para o fuso horário local para o input type="datetime-local"

        agora.setMinutes(agora.getMinutes() - agora.getTimezoneOffset());

        inputData.value = agora.toISOString().slice(0, 16);

    }

    // Inicialização e Event Listeners

    document.addEventListener("DOMContentLoaded", () => {

        carregarDadosIniciais();

        resetarFormulario();

        // NOVO: Inicia o relógio para atualizar o campo de data/hora a cada segundo

        setInterval(definirDataHoraAtual, 1000);

        document.getElementById('addServico').addEventListener('click', adicionarLinhaServico);

        document.getElementById('addProduto').addEventListener('click', adicionarLinhaProduto);

        selPagamento.addEventListener('change', recalcularTudo);

        document.getElementById('btnFinalizar').addEventListener('click', finalizarVenda);

        document.getElementById('btnCancelar').addEventListener('click', resetarFormulario);

    });

})();

// ===============================

// Admin • Clientes – Barbearia

// Tabela + CRUD + Combos ativos + Histórico (read-only)

// Persistência: localStorage (v1) – troque depois pelo seu backend

// Âncoras: [UTIL], [LS], [STATE], [CRUD], [UPLOAD FOTO], [RENDER], [COMBOS], [HIST], [EXPORT]

// ===============================

(function(){

  // [UTIL]

  const $ = (s, r=document) => r.querySelector(s);

  const $$ = (s, r=document) => Array.from(r.querySelectorAll(s));

  const fmtBR = {

    tel(v){

      const d = String(v||"").replace(/\D/g,"");

      if (d.length >= 11) return d.replace(/(\d{2})(\d{5})(\d{4}).\*/, "($1) $2-$3");

      if (d.length >= 10) return d.replace(/(\d{2})(\d{4})(\d{4}).\*/, "($1) $2-$3");

      return v || "—";

    },

    dateISOtoBR(iso){

      if (!iso) return "—";

      const d = new Date(iso);

      if (isNaN(d)) return "—";

      return d.toLocaleDateString("pt-BR");

    },

    dtISOtoBRhm(iso){

      if (!iso) return "—";

      const d = new Date(iso);

      if (isNaN(d)) return "—";

      const dd = d.toLocaleDateString("pt-BR");

      const hm = d.toLocaleTimeString("pt-BR",{hour:"2-digit",minute:"2-digit"});

      return `${dd} ${hm}`;

    },

    emailValid(e){ return !e || /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(e); }

  };

  // [LS] – chaves de storage

  const LS\_CLIENTES = "barbearia\_clientes\_v1";

  const LS\_COMBOS   = "barbearia\_clientes\_creditos\_v1";   // somente leitura aqui

  const LS\_AGEND    = "barbearia\_agendamentos\_v1";        // somente leitura aqui

  const read = (k, fallback) => {

    try { const v = JSON.parse(localStorage.getItem(k)); return v ?? fallback; }

    catch { return fallback; }

  };

  const write = (k, v) => localStorage.setItem(k, JSON.stringify(v));

  // [STATE]

  let clientes = read(LS\_CLIENTES, []);

  let combos   = read(LS\_COMBOS,   []);  // formato da spec (array)

  let agends   = read(LS\_AGEND,    []);  // formato da spec (array)

  let q = "";             // termo de busca

  let toDelete = null;    // id para exclusão

  let editingId = null;   // id em edição

  // Seletores base

  const tbody = $("#cliTbody");

  const inpQ  = $("#cliQ");

  // ====== Helpers de ID e Foto ======

  const genId = () => {

    const id = "C-" + Math.random().toString(16).slice(2,6).toUpperCase();

    return clientes.some(c=>c.id===id) ? genId() : id;

  };

  async function fileToDataURL(file){

    if (!file) return "";

    if (!/^image\//.test(file.type)) { alert("Selecione uma imagem válida."); return ""; }

    return new Promise((res,rej)=>{

      const fr = new FileReader();

      fr.onload = () => res(fr.result);

      fr.onerror = () => rej();

      fr.readAsDataURL(file);

    });

  }

  // ====== [RENDER] Tabela ======

  function render(){

    const term = (q||"").toLowerCase();

    const list = clientes.filter(c=>{

      const hay = [c.name, c.phone, c.email].filter(Boolean).join(" ").toLowerCase();

      return !term || hay.includes(term);

    });

    tbody.innerHTML = list.map(c=>{

      const foto = c.photo ? `<img class="mini" src="${c.photo}" alt="">` : `<div class="mini mini-badge">sem foto</div>`;

      return `

        <tr>

          <td>${foto}</td>

          <td>${c.name || "—"}</td>

          <td>${fmtBR.tel(c.phone)}</td>

          <td class="hide-sm">${c.email || "—"}</td>

          <td class="hide-sm">${fmtBR.dateISOtoBR(c.birthISO)}</td>

          <td>${comboResumo(c.id)}</td>

          <td>

            <div class="cell-actions">

              <button class="btn btn-blue"   data-act="hist" data-id="${c.id}">📜 Histórico</button>

              <button class="btn btn-yellow" data-act="edit" data-id="${c.id}">✏️</button>

              <button class="btn btn-red"    data-act="del"  data-id="${c.id}">🗑️</button>

            </div>

          </td>

        </tr>

      `;

    }).join("");

  }

  // ====== [COMBOS] Resumo por cliente (linha da tabela) ======

  function comboResumo(clienteId){

    const mine = combos.filter(x=>x.clienteId===clienteId);

    if (!mine.length) return "—";

    const parts = mine.map(x=>{

      const label = x.comboNome || "Combo";

      const quota = `${x.qtdConsumidos ?? 0}/${x.qtdTotal ?? 0}`;

      const exp = x.dataExpiraISO ? fmtBR.dateISOtoBR(x.dataExpiraISO) : "sem validade";

      return `<span class="badge badge-green" title="${label}">${label} • ${quota} • expira ${exp}</span>`;

    });

    return parts.join(" ");

  }

  // ====== [HIST] Modal de histórico de agendamentos ======

  const histModal = $("#histModal");

  const histList  = $("#histList");

  function open(el){ el.classList.add("show"); }

  function close(el){ el.classList.remove("show"); }

  $$(".modal [data-close]").forEach(b=> b.addEventListener("click", e=> close(e.target.closest(".modal")) ));

  function openHistorico(cliente){

    $("#histTitle").textContent = `Histórico – ${cliente.name}`;

    // filtra e ordena (mais recentes primeiro)

    const rows = agends

      .filter(a=>a.clienteId===cliente.id)

      .sort((a,b)=> new Date(b.inicioISO) - new Date(a.inicioISO));

    if (!rows.length){

      histList.innerHTML = `<p class="muted">Nenhum agendamento encontrado para este cliente.</p>`;

      open(histModal); return;

    }

    const html = `

      <table class="table">

        <thead>

          <tr>

            <th>Data/Hora</th><th>Serviço</th><th>Profissional</th><th>Status</th><th>Obs.</th>

          </tr>

        </thead>

        <tbody>

          ${rows.map(a=>`

            <tr>

              <td>${fmtBR.dtISOtoBRhm(a.inicioISO)}</td>

              <td>${a.servicoNome || "—"}</td>

              <td>${a.profissionalNome || "—"}</td>

              <td>${a.status || "—"}</td>

              <td>${a.observacao || "—"}</td>

            </tr>

          `).join("")}

        </tbody>

      </table>

    `;

    histList.innerHTML = html;

    open(histModal);

  }

  // ====== [CRUD] Modal Novo/Editar ======

  const formModal = $("#cliModal");

  const form      = $("#cliForm");

  const idField = $("#cliId");

  const nome    = $("#cNome");

  const tel     = $("#cTel");

  const email   = $("#cEmail");

  const nasc    = $("#cNasc");

  const fotoBox     = $("#fotoBox");

  const fotoPreview = $("#fotoPreview");

  const fotoInput   = $("#cFoto");

  $("#btnFabAddCli")?.addEventListener("click", ()=>{

    editingId = null;

    $("#cliTitle").textContent = "Novo cliente";

    form.reset();

    idField.value = "";

    fotoBox.hidden = true;  // sem upload no cadastro inicial

    open(formModal);

  });

  // Preenche form p/ edição

  function fillForm(c){

    idField.value = c.id;

    nome.value    = c.name || "";

    tel.value     = c.phone || "";

    email.value   = c.email || "";

    nasc.value    = c.birthISO || "";

    if (c.photo){

      fotoPreview.innerHTML = `<img src="${c.photo}" alt="" style="width:100%;height:100%;object-fit:cover" />`;

    }else{

      fotoPreview.textContent = "FOTO";

    }

    fotoInput.value = "";

  }

  // Enviar (criar/atualizar)

  form?.addEventListener("submit", async (e)=>{

    e.preventDefault();

    // Validações básicas

    if (!nome.value.trim()) return alert("Informe o nome.");

    const dOnly = tel.value.replace(/\D/g,"");

    if (dOnly.length < 10)   return alert("Telefone inválido.");

    if (!fmtBR.emailValid(email.value)) return alert("E-mail inválido.");

    // Data de nascimento futura?

    if (nasc.value){

      const nd = new Date(nasc.value);

      if (nd > new Date()) return alert("Data de nascimento não pode ser futura.");

    }

    let c = clientes.find(x=>x.id === idField.value);

    const nowISO = new Date().toISOString();

    if (!c){

      // Novo

      const id = genId();

      c = {

        id,

        name: nome.value.trim(),

        phone: tel.value.trim(),

        email: email.value.trim() || null,

        birthISO: nasc.value || null,

        photo: "",

        createdAtISO: nowISO,

        updatedAtISO: nowISO,

        mustChangePassword: true   // tratado no backend de autenticação

      };

      clientes.push(c);

    }else{

      // Update

      c.name = nome.value.trim();

      c.phone = tel.value.trim();

      c.email = email.value.trim() || null;

      c.birthISO = nasc.value || null;

      c.updatedAtISO = nowISO;

    }

    // Upload de foto (apenas quando o box está visível = edição)

    if (!fotoBox.hidden && fotoInput.files?.[0]){

      try{

        c.photo = await fileToDataURL(fotoInput.files[0]);

      }catch{}

    }

    write(LS\_CLIENTES, clientes);

    render();

    close(formModal);

  });

  // Clique em ações da tabela

  tbody?.addEventListener("click", (e)=>{

    const btn = e.target.closest("[data-act]");

    if (!btn) return;

    const id = btn.dataset.id;

    const c = clientes.find(x=>x.id===id);

    if (!c) return;

    if (btn.dataset.act === "edit"){

      editingId = id;

      $("#cliTitle").textContent = "Editar cliente";

      fillForm(c);

      fotoBox.hidden = false;  // na edição, habilita upload

      open(formModal);

    }

    else if (btn.dataset.act === "del"){

      toDelete = id; open($("#cliDelModal"));

    }

    else if (btn.dataset.act === "hist"){

      openHistorico(c);

    }

  });

  // Confirma exclusão

  $("#cliConfirmDel")?.addEventListener("click", ()=>{

    if (!toDelete) return;

    clientes = clientes.filter(c=>c.id !== toDelete);

    write(LS\_CLIENTES, clientes);

    render();

    toDelete = null;

    close($("#cliDelModal"));

  });

  // ====== Busca / Export / Import / Zerar ======

  inpQ?.addEventListener("input", e=>{ q = e.target.value||""; render(); });

  $("#cliExport")?.addEventListener("click", ()=>{

    const blob = new Blob([JSON.stringify({ clientes }, null, 2)], { type:"application/json" });

    const url  = URL.createObjectURL(blob);

    const a = document.createElement("a");

    a.href = url; a.download = "clientes\_barbearia.json"; a.click();

    URL.revokeObjectURL(url);

  });

  $("#cliImport")?.addEventListener("change", async e=>{

    const f = e.target.files?.[0]; if (!f) return;

    try{

      const obj = JSON.parse(await f.text());

      if (!obj || !Array.isArray(obj.clientes)) throw 0;

      clientes = obj.clientes;

      write(LS\_CLIENTES, clientes);

      render();

    }catch{

      alert("JSON inválido.");

    }

    e.target.value = "";

  });

  $("#cliZerar")?.addEventListener("click", ()=>{

    if (!confirm("Apagar TODOS os clientes?")) return;

    clientes = [];

    write(LS\_CLIENTES, clientes);

    render();

  });

  // ====== Seeds opcionais (para visualizar de primeira) ======

  if (!clientes.length){

    clientes = [

      { id:"C-8A3F", name:"Rafael Brandão", phone:"(21) 99999-9999", email:"rafael@email.com",

        birthISO:"1991-05-12", photo:"", createdAtISO:new Date().toISOString(), updatedAtISO:new Date().toISOString(),

        mustChangePassword:true }

    ];

    write(LS\_CLIENTES, clientes);

  }

  if (!combos.length){

    combos = [

      { clienteId:"C-8A3F", comboServicoId:"svc\_combo\_4cortes", comboNome:"Combo 4 Cortes",

        qtdTotal:4, qtdConsumidos:2, qtdDisponiveis:2, dataCompraISO:"2025-08-10T12:00:00.000Z",

        dataExpiraISO:"2025-12-10T12:00:00.000Z", comboTipo:"fixo", servicoBaseId:"svc\_corte\_classico", servicosElegiveisIds:[] }

    ];

    write(LS\_COMBOS, combos);

  }

  if (!agends.length){

    agends = [

      { id:"ag\_1", clienteId:"C-8A3F", servicoId:"svc\_corte\_classico", servicoNome:"Corte clássico",

        profissionalId:"EMP-77BC", profissionalNome:"Carlos", inicioISO:new Date().toISOString(),

        fimISO:new Date(Date.now()+1800000).toISOString(), status:"compareceu", observacao:"" }

    ];

    write(LS\_AGEND, agends);

  }

  // Inicialização

  render();

})();

// firebase-config.js

// Import the functions you need from the SDKs you need

import { initializeApp } from "firebase/app";

import { getAnalytics } from "firebase/analytics";

// TODO: Add SDKs for Firebase products that you want to use

// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration

// For Firebase JS SDK v7.20.0 and later, measurementId is optional

const firebaseConfig = {

  apiKey: "AIzaSyAEQMwHWQuPKybroJM4UK2\_mJWJESYm-a4",

  authDomain: "barbearia-carreiro.firebaseapp.com",

  projectId: "barbearia-carreiro",

  storageBucket: "barbearia-carreiro.firebasestorage.app",

  messagingSenderId: "54089800722",

  appId: "1:54089800722:web:0221369454bfbb71c24ccb",

  measurementId: "G-4L0ZK72L4L"

};

// Inicializa os serviços que serão usados

const auth = firebase.auth();

const db = firebase.firestore();

// Inicializa os serviços que serão usados

const auth = firebase.auth();

const db = firebase.firestore();

/\*  Funcionários – CRUD em localStorage (barbearia\_funcionarios\_v1)

    - Lista, busca, criar/editar/excluir

    - Exportar/Importar JSON

    - Gera ID EMP-XXXX

    - Senha padrão (DEMO): "mudar123" + mustChangePassword=true

    - Comentários em PT-BR para facilitar manutenção

\*/

(function(){

  const LS\_KEY = "barbearia\_funcionarios\_v1";

  const PASSWORD\_DEFAULT = "mudar123";

  // ---------- Helpers ----------

  const fmtBRL = (v) => {

    if (v === null || v === undefined || v === "") return "—";

    return new Intl.NumberFormat("pt-BR", { style: "currency", currency: "BRL" }).format(Number(v));

  };

  const fmtPercent = (v) => (v === null || v === undefined || v === "" ? "—" : `${Number(v)}%`);

  const maskPhone = (s="") => {

    const d = s.replace(/\D/g, "").slice(0, 11);

    const p1 = d.slice(0,2), p2 = d.length>10? d.slice(2,7) : d.slice(2,6), p3 = d.length>10? d.slice(7,11) : d.slice(6,10);

    if (!p1) return s;

    if (!p2) return `(${p1}`;

    if (!p3) return `(${p1}) ${p2}`;

    return `(${p1}) ${p2}-${p3}`;

  };

  const genId = (existing) => {

    const has = new Set(existing.map(e => e.id));

    let id;

    do { id = "EMP-" + Math.random().toString(16).slice(2, 6).toUpperCase(); } while (has.has(id));

    return id;

  };

  const readLS = () => {

    try {

      const raw = localStorage.getItem(LS\_KEY);

      if (!raw) return [];

      const arr = JSON.parse(raw);

      return Array.isArray(arr) ? arr : [];

    } catch { return []; }

  };

  const writeLS = (arr) => localStorage.setItem(LS\_KEY, JSON.stringify(arr));

  // ---------- Estado ----------

  let state = readLS();

  let q = ""; // query de busca

  let deletingId = null;

  // ---------- Renderização ----------

  const tbody = () => document.getElementById("empTbody");

  function render(){

    const tb = tbody();

    if (!tb) return;

    const term = q.trim().toLowerCase();

    const filtered = state.filter(emp => {

      if (!term) return true;

      const hay = [

        emp.name, emp.role, emp.phone, emp.contractType,

        emp.accessType, emp.address

      ].join(" ").toLowerCase();

      return hay.includes(term);

    });

    tb.innerHTML = filtered.map(emp => {

      const foto = emp.photo

        ? `<img src="${emp.photo}" alt="" style="width:72px;height:72px;border-radius:50%;object-fit:cover;border:1px solid #232323;background:#0c1426" />`

        : `<div style="width:72px;height:72px;border-radius:50%;display:flex;align-items:center;justify-content:center;border:1px solid #232323;background:#0c1426;color:#94a3b8;font-weight:900">sem foto</div>`;

      // comissão (percentual/valor\_fixo/—)

      let comStr = "—";

      if (emp.commissionType === "percentual") comStr = fmtPercent(emp.commissionValue);

      if (emp.commissionType === "valor\_fixo") comStr = fmtBRL(emp.commissionValue);

      const salaryStr = emp.fixedSalary ? fmtBRL(emp.fixedSalary) : "—";

      return `

        <tr>

          <td>${foto}</td>

          <td>${emp.name || ""}</td>

          <td>${emp.role || ""}</td>

          <td>${emp.phone || ""}</td>

          <td class="hide-sm">${emp.address || "—"}</td>

          <td>${emp.contractType || ""}</td>

          <td>${comStr}</td>

          <td class="hide-sm">${salaryStr}</td>

          <td>${emp.accessType || ""}</td>

          <td>${emp.id}</td>

          <td>••••••</td>

          <td>

            <div class="cell-actions">

              <button class="btn btn-yellow" data-action="edit" data-id="${emp.id}" title="Editar">✏️</button>

              <button class="btn btn-red" data-action="delete" data-id="${emp.id}" title="Excluir">🗑️</button>

            </div>

          </td>

        </tr>

      `;

    }).join("");

  }

  // ---------- Modais ----------

  const modal = document.getElementById("empModal");

  const modalDel = document.getElementById("empModalDelete");

  const delName = document.getElementById("empDelNome");

  const open = (el) => { el.classList.add("show"); el.setAttribute("aria-hidden","false"); };

  const close = (el) => { el.classList.remove("show"); el.setAttribute("aria-hidden","true"); };

  document.querySelectorAll("[data-close]").forEach(b => b.addEventListener("click", () => {

    close(modal); close(modalDel);

  }));

  // FAB: novo

  document.getElementById("btnFabAddEmp")?.addEventListener("click", () => {

    resetForm();

    document.getElementById("empModalTitle").textContent = "Novo funcionário";

    open(modal);

  });

  // ---------- Form ----------

  const form = document.getElementById("empForm");

  const fId = document.getElementById("fId");

  const fName = document.getElementById("fName");

  const fRole = document.getElementById("fRole");

  const fPhone = document.getElementById("fPhone");

  const fAddress = document.getElementById("fAddress");

  const fContract = document.getElementById("fContract");

  const fFixedSalary = document.getElementById("fFixedSalary");

  const fComType = document.getElementById("fComType");

  const fComValue = document.getElementById("fComValue");

  const fAccess = document.getElementById("fAccess");

  const fLoginId = document.getElementById("fLoginId");

  const fGenId = document.getElementById("fGenId");

  const fPassword = document.getElementById("fPassword");

  const fMustChange = document.getElementById("fMustChange");

  const fPhoto = document.getElementById("fPhoto");

  const fPhotoPrev = document.getElementById("fPhotoPrev");

  fPhone?.addEventListener("input", () => fPhone.value = maskPhone(fPhone.value));

  function resetForm(){

    form.reset();

    fId.value = "";

    fGenId.value = genId(state);

    fPassword.value = PASSWORD\_DEFAULT;

    fMustChange.checked = true;

    fPhotoPrev.textContent = "FOTO";

    fPhotoPrev.style.backgroundImage = "";

    fPhotoPrev.style.backgroundSize = "";

  }

  fPhoto?.addEventListener("change", async (e) => {

    const file = e.target.files?.[0];

    if (!file) return;

    const dataUrl = await fileToDataURL(file);

    fPhotoPrev.textContent = "";

    fPhotoPrev.style.backgroundImage = `url('${dataUrl}')`;

    fPhotoPrev.style.backgroundSize = "cover";

    fPhotoPrev.dataset.dataurl = dataUrl; // guardamos aqui, salvamos no submit

  });

  function fileToDataURL(file){

    return new Promise((res, rej) => {

      const fr = new FileReader();

      fr.onload = () => res(fr.result);

      fr.onerror = rej;

      fr.readAsDataURL(file);

    });

  }

  form?.addEventListener("submit", (e) => {

    e.preventDefault();

    // validações básicas

    if (!fName.value.trim() || !fRole.value || !fPhone.value.trim() || !fContract.value || !fAccess.value){

      alert("Preencha Nome, Função, Telefone, Contrato e Acesso.");

      return;

    }

    // regra de comissão obrigatória para Barbeiro ou Comissionista

    const needsCom = fRole.value === "Barbeiro" || fContract.value === "Comissionista";

    if (needsCom && !fComType.value){

      alert("Para Barbeiro/Comissionista, informe o tipo de comissão.");

      return;

    }

    if (needsCom && fComType.value && (fComValue.value === "" || Number(fComValue.value) < 0)){

      alert("Informe um valor de comissão válido.");

      return;

    }

    if (fComType.value === "percentual" && Number(fComValue.value) > 100){

      alert("Percentual de comissão não pode passar de 100%.");

      return;

    }

    if (fFixedSalary.value !== "" && Number(fFixedSalary.value) < 0){

      alert("Salário fixo deve ser ≥ 0.");

      return;

    }

    const existing = readLS();

    const payload = {

      id: fId.value || fGenId.value,

      name: fName.value.trim(),

      role: fRole.value,

      phone: fPhone.value.trim(),

      address: fAddress.value.trim() || "",

      contractType: fContract.value,

      commissionType: fComType.value || null,

      commissionValue: fComValue.value === "" ? null : Number(fComValue.value),

      fixedSalary: fFixedSalary.value === "" ? null : Number(fFixedSalary.value),

      accessType: fAccess.value,

      loginId: fLoginId.value.trim() || null,

      password: fPassword.value,           // DEMO apenas

      mustChangePassword: !!fMustChange.checked,

      photo: fPhotoPrev.dataset.dataurl || (getById(fId.value || fGenId.value)?.photo || "")

    };

    const idx = existing.findIndex(e => e.id === payload.id);

    if (idx >= 0) existing[idx] = payload; else existing.push(payload);

    state = existing;

    writeLS(state);

    render();

    close(modal);

  });

  // ---------- Tabela: editar / excluir ----------

  document.getElementById("empTbody")?.addEventListener("click", (e) => {

    const btn = e.target.closest("[data-action]");

    if (!btn) return;

    const id = btn.dataset.id;

    const action = btn.dataset.action;

    if (action === "edit"){

      const emp = getById(id);

      if (!emp) return alert("Funcionário não encontrado.");

      // preencher form

      fId.value = emp.id;

      fGenId.value = emp.id;

      fName.value = emp.name || "";

      fRole.value = emp.role || "";

      fPhone.value = emp.phone || "";

      fAddress.value = emp.address || "";

      fContract.value = emp.contractType || "";

      fFixedSalary.value = emp.fixedSalary ?? "";

      fComType.value = emp.commissionType || "";

      fComValue.value = emp.commissionValue ?? "";

      fAccess.value = emp.accessType || "";

      fLoginId.value = emp.loginId || "";

      fPassword.value = PASSWORD\_DEFAULT; // ação recomendada: redefinir ao editar

      fMustChange.checked = true;

      // foto

      if (emp.photo){

        fPhotoPrev.textContent = "";

        fPhotoPrev.style.backgroundImage = `url('${emp.photo}')`;

        fPhotoPrev.style.backgroundSize = "cover";

        fPhotoPrev.dataset.dataurl = emp.photo;

      } else {

        fPhotoPrev.textContent = "FOTO";

        fPhotoPrev.style.backgroundImage = "";

        fPhotoPrev.dataset.dataurl = "";

      }

      document.getElementById("empModalTitle").textContent = "Editar funcionário";

      open(modal);

    }

    if (action === "delete"){

      const emp = getById(id);

      deletingId = id;

      document.getElementById("btnEmpConfirmDelete").dataset.id = id;

      document.getElementById("empDelNome").textContent = emp?.name || "este funcionário";

      open(modalDel);

    }

  });

  document.getElementById("btnEmpConfirmDelete")?.addEventListener("click", (e) => {

    const id = e.currentTarget.dataset.id;

    if (!id) return;

    state = readLS().filter(x => x.id !== id);

    writeLS(state);

    render();

    close(modalDel);

  });

  // ---------- Busca ----------

  document.getElementById("empQ")?.addEventListener("input", (e) => {

    q = e.target.value || "";

    render();

  });

  // ---------- Exportar / Importar / Zerar ----------

  document.getElementById("empExport")?.addEventListener("click", () => {

    const blob = new Blob([JSON.stringify(state, null, 2)], {type:"application/json"});

    const url = URL.createObjectURL(blob);

    const a = document.createElement("a");

    a.href = url;

    a.download = "funcionarios\_barbearia.json";

    a.click();

    URL.revokeObjectURL(url);

  });

  document.getElementById("empImportFile")?.addEventListener("change", async (e) => {

    const file = e.target.files?.[0];

    if (!file) return;

    try{

      const text = await file.text();

      const data = JSON.parse(text);

      if (!Array.isArray(data)) throw new Error("JSON inválido");

      state = data;

      writeLS(state);

      render();

      alert("Importado com sucesso.");

    }catch(err){

      alert("Falha ao importar JSON.");

    } finally {

      e.target.value = "";

    }

  });

  document.getElementById("empZerar")?.addEventListener("click", () => {

    if (!confirm("Tem certeza que deseja apagar TODOS os funcionários?")) return;

    state = [];

    writeLS(state);

    render();

  });

  // ---------- Utils ----------

  function getById(id){ return readLS().find(x => x.id === id); }

  // ---------- Boot ----------

  // cria 2 exemplos se estiver vazio (só para visualizar)

  if (!state.length){

    state = [

      {

        id: genId([]), name: "Igor", role: "Barbeiro",

        phone: "(21) 99999-0001", address: "Rua A, 100 - Centro",

        contractType: "PJ", commissionType: "percentual", commissionValue: 40,

        fixedSalary: null, accessType: "Lancamentos",

        loginId: "igor", password: PASSWORD\_DEFAULT, mustChangePassword: true,

        photo: ""

      },

      {

        id: genId([]), name: "Moisés", role: "Barbeiro",

        phone: "(21) 99999-0002", address: "Rua B, 200 - Bairro",

        contractType: "Comissionista", commissionType: "valor\_fixo", commissionValue: 5,

        fixedSalary: null, accessType: "Lancamentos",

        loginId: "moises", password: PASSWORD\_DEFAULT, mustChangePassword: true,

        photo: ""

      }

    ];

    writeLS(state);

  }

  render();

})();

// historico.js

// Protege a rota, busca agendamentos do usuário e desenha a lista com cores

document.addEventListener("DOMContentLoaded", () => {

  const auth = firebase.auth();

  const db   = firebase.firestore();

  const list = document.getElementById("historyList");

  const btnVoltar = document.getElementById("btnVoltar");

  // Voltar para Home

  btnVoltar.addEventListener("click", () => {

    window.location.href = "home.html";

  });

  // Guarda de autenticação

  auth.onAuthStateChanged(async (user) => {

    if (!user) {

      window.location.href = "index.html";

      return;

    }

    try {

      // Exemplo de estrutura: collection("agendamentos")

      // Campos mínimos: userId, dataHora (Timestamp), status (compareceu|faltou|cancelado|—)

      const snap = await db.collection("agendamentos")

        .where("userId", "==", user.uid)

        .orderBy("dataHora", "desc")

        .limit(100)

        .get();

      if (snap.empty) {

        list.innerHTML = `<li class="history-item"><span class="when">Nenhum agendamento encontrado.</span></li>`;

        return;

      }

      const now = new Date();

      const frag = document.createDocumentFragment();

      snap.forEach(doc => {

        const a = doc.data();

        // data/hora

        const dt = a.dataHora && a.dataHora.toDate ? a.dataHora.toDate() : null;

        const whenStr = dt ? formatDateTime(dt) : "(sem data)";

        // status → classe da caixinha

        const statusClass = getStatusClass(a.status, dt, now);

        const li = document.createElement("li");

        li.className = "history-item";

        const spanWhen = document.createElement("span");

        spanWhen.className = "when";

        spanWhen.textContent = whenStr;

        const box = document.createElement("span");

        box.className = `status-box ${statusClass}`;

        box.setAttribute("title", getStatusLabel(statusClass));

        li.appendChild(spanWhen);

        li.appendChild(box);

        frag.appendChild(li);

      });

      list.innerHTML = "";

      list.appendChild(frag);

    } catch (err) {

      console.error("Erro ao carregar histórico:", err);

      list.innerHTML = `<li class="history-item"><span class="when">Erro ao carregar histórico.</span></li>`;

    }

  });

  function formatDateTime(d){

    // dd/mm/yyyy HH:mm

    const dd   = String(d.getDate()).padStart(2,"0");

    const mm   = String(d.getMonth()+1).padStart(2,"0");

    const yyyy = d.getFullYear();

    const hh   = String(d.getHours()).padStart(2,"0");

    const mi   = String(d.getMinutes()).padStart(2,"0");

    return `${dd}/${mm}/${yyyy} ${hh}:${mi}`;

    // se quiser 12h, adaptamos

  }

  // Mapeia status → classe CSS

  function getStatusClass(status, dt, now){

    // Normaliza status (string opcional)

    const s = (status || "").toLowerCase();

    if (s === "compareceu") return "status--ok";

    if (s === "faltou")     return "status--no";

    if (s === "cancelado")  return "status--cancel";

    // Sem status explícito → decide pelo tempo:

    if (dt && dt > now) return "status--future"; // ainda não chegou o dia (branco)

    // Passado sem status: você pode assumir "compareceu" ou "faltou".

    // Aqui deixo sem cor especial (verde) só se quiser:

    return "status--ok";

  }

  function getStatusLabel(cls){

    switch(cls){

      case "status--ok":     return "Compareceu";

      case "status--no":     return "Faltou";

      case "status--cancel": return "Cancelado";

      case "status--future": return "Agendamento futuro";

      default:               return "Status";

    }

  }

});

// js/home.js

// Protege a página, busca o nome do cliente no Firestore e liga os botões

document.addEventListener("DOMContentLoaded", () => {

    // Inicializa os serviços do Firebase

    const auth = firebase.auth();

    const db = firebase.firestore();

    // Pega os elementos da página que vamos manipular

    const nomeSpan = document.getElementById("nomeCliente");

    const btnAgendar = document.getElementById("btnAgendar");

    const btnHistorico = document.getElementById("btnHistorico");

    // --- PROTEÇÃO DA ROTA E CARREGAMENTO DE DADOS ---

    // Esta função verifica se o usuário está logado. Ela é chamada sempre que o estado da autenticação muda.

    auth.onAuthStateChanged(async (user) => {

        if (!user) {

            // Se não houver usuário logado, redireciona imediatamente para a página de login

            window.location.href = "index.html";

            return;

        }

        // Se o usuário está logado, vamos buscar o nome dele para exibir na saudação

        try {

            // Busca o documento do cliente no Firestore usando o ID do usuário logado (user.uid)

            const doc = await db.collection("clientes").doc(user.uid).get();

            let nome = "";

            if (doc.exists && doc.data().nome) {

                // Se encontramos o documento e ele tem um nome, usamos esse nome

                nome = doc.data().nome;

            } else if (user.displayName) {

                // Se não, mas o usuário logou com Google, usamos o nome do Google

                nome = user.displayName;

            } else {

                // Como último recurso, usamos "Cliente"

                nome = "Cliente";

            }

            // Atualiza o texto na tela com o primeiro nome do cliente

            nomeSpan.textContent = nome.split(' ')[0];

        } catch (err) {

            console.error("Erro ao carregar dados do usuário:", err);

            nomeSpan.textContent = "Cliente";

        }

    });

    // --- AÇÕES DOS BOTÕES ---

    // Evento de clique para o botão "Agendar agora"

    if (btnAgendar) {

        btnAgendar.addEventListener("click", () => {

            // Redireciona para a primeira página do fluxo de agendamento

            // <<-- CORREÇÃO AQUI -->>

            window.location.href = "agendar-servico.html";

        });

    }

    // Evento de clique para o botão "Histórico"

    if (btnHistorico) {

        btnHistorico.addEventListener("click", () => {

            // Redireciona para a página de histórico de agendamentos

            window.location.href = "historico.html"; // Este já estava correto

        });

    }

});

// ============================================================

// Admin • Lançamentos – Barbearia

// Histórico + filtros + KPIs + modal de novo lançamento

// Persistência: localStorage (v1). Integrações de Estoque/Combos comentadas.

// Âncoras: [CONFIG], [UTIL], [LS], [STATE], [FILTERS], [RENDER],

//          [KPIS], [MODAL], [CALC], [INTEGRACAO ESTOQUE], [INTEGRACAO COMBOS],

//          [EXPORT/IMPORT], [BOOT]

// ============================================================

(function(){

  // ---------------- [CONFIG: TAXAS + BASE COMISSAO] ----------------

  // Carrega config global (se existir) ou usa padrão

  const LS\_CONF = "barbearia\_config\_v1";

  const DEFAULT\_CONF = {

    politicaCusto: "CMP",

    alertaValidadeDias: 30,

    permitirEstoqueNegativo: false,

    taxas: { Dinheiro:0, Pix:0, Debito:1.5, CreditoAVista:3.0, CreditoParcelado:4.0 },

    baseComissao: "ANTES\_TAXA",            // ANTES\_TAXA | DEPOIS\_TAXA

    comissionarProdutos: false             // por padrão, só comissiona serviço

  };

  // ---------------- [UTIL: BRL + DATAS] ----------------

  const $ = (s, r=document)=>r.querySelector(s);

  const $$ = (s, r=document)=>Array.from(r.querySelectorAll(s));

  const fmtBRL = n => new Intl.NumberFormat("pt-BR",{style:"currency",currency:"BRL"}).format(n||0);

  const toISODateStart = (d)=> new Date(d.getFullYear(), d.getMonth(), d.getDate()).toISOString();

  const toISODateEnd   = (d)=> new Date(d.getFullYear(), d.getMonth(), d.getDate(), 23,59,59,999).toISOString();

  function rangeOf(type, baseDate=new Date()){

    const d = new Date(baseDate);

    if (type==="dia"){

      return { de: toISODateStart(d), ate: toISODateEnd(d) };

    }

    if (type==="semana"){

      const day = d.getDay(); // 0 dom … 6 sáb

      const diff = (day+6)%7; // segunda como início

      const ini = new Date(d); ini.setDate(d.getDate()-diff);

      const fim = new Date(ini); fim.setDate(ini.getDate()+6);

      return { de: toISODateStart(ini), ate: toISODateEnd(fim) };

    }

    if (type==="mes"){

      const ini = new Date(d.getFullYear(), d.getMonth(), 1);

      const fim = new Date(d.getFullYear(), d.getMonth()+1, 0);

      return { de: toISODateStart(ini), ate: toISODateEnd(fim) };

    }

    if (type==="ano"){

      const ini = new Date(d.getFullYear(), 0, 1);

      const fim = new Date(d.getFullYear(), 11, 31);

      return { de: toISODateStart(ini), ate: toISODateEnd(fim) };

    }

    return { de:null, ate:null };

  }

  // ---------------- [LS] ----------------

  const LS\_LANC   = "barbearia\_lancamentos\_v1";

  const LS\_PROD   = "barbearia\_produtos\_v1";       // leitura para lista/custos

  const LS\_KARDEX = "barbearia\_kardex\_v1";         // movimentações de estoque

  const LS\_SERV   = "barbearia\_servicos\_v2";       // leitura para serviço/custos

  const LS\_FUNC   = "barbearia\_funcionarios\_v1";   // leitura para comissão

  const LS\_CLI    = "barbearia\_clientes\_v1";       // leitura para select cliente

  const LS\_CRED   = "barbearia\_clientes\_creditos\_v1"; // combos por cliente

  const read = (k, fb)=>{ try{ const v=JSON.parse(localStorage.getItem(k)); return v??fb; }catch{ return fb; } };

  const write= (k, v)=> localStorage.setItem(k, JSON.stringify(v));

  // ---------------- [STATE] ----------------

  let conf   = Object.assign({}, DEFAULT\_CONF, read(LS\_CONF, {}));

  let lancs  = read(LS\_LANC, []);

  let prods  = read(LS\_PROD, []);

  let kardex = read(LS\_KARDEX, []);

  let servs  = read(LS\_SERV,  []);

  let funcs  = read(LS\_FUNC,  []);

  let clis   = read(LS\_CLI,   []);

  let creds  = read(LS\_CRED,  []);

  const tbody = $("#lanTbody");

  // Filtros atuais

  let f = {

    de: null, ate: null,

    prof: "", cli: "",

    forma: "", tipo: "",

    ord: "desc"

  };

  // ---------------- [FILTERS] UI ----------------

  // Preenche selects de profissional/cliente

  function fillSelect(sel, arr, labelKey="name"){

    sel.innerHTML = `<option value="">Todos</option>` + arr.map(x=>`<option value="${x.id}">${x[labelKey]||x.id}</option>`).join("");

  }

  fillSelect($("#fProf"), funcs);

  fillSelect($("#fCli"),  clis);

  // Range rápido

  $$(".btn-group [data-range]").forEach(btn=>{

    btn.addEventListener("click", ()=>{

      const r = rangeOf(btn.dataset.range, new Date());

      if (r.de){ $("#fDataDe").value = r.de.slice(0,10); }

      if (r.ate){ $("#fDataAte").value = r.ate.slice(0,10); }

    });

  });

  // Aplicar/Limpar

  $("#btnAplicar")?.addEventListener("click", ()=>{

    f.de    = $("#fDataDe").value ? new Date($("#fDataDe").value) : null;

    f.ate   = $("#fDataAte").value ? new Date($("#fDataAte").value) : null;

    f.prof  = $("#fProf").value;

    f.cli   = $("#fCli").value;

    f.forma = $("#fForma").value;

    f.tipo  = $("#fTipo").value;

    f.ord   = $("#fOrd").value || "desc";

    render();

  });

  $("#btnLimpar")?.addEventListener("click", ()=>{

    $("#fDataDe").value = ""; $("#fDataAte").value="";

    $("#fProf").value=""; $("#fCli").value=""; $("#fForma").value="";

    $("#fTipo").value=""; $("#fOrd").value="desc";

    f = {de:null, ate:null, prof:"", cli:"", forma:"", tipo:"", ord:"desc"};

    render();

  });

  // ---------------- [RENDER] Tabela ----------------

  function within(dtISO){

    if (!f.de && !f.ate) return true;

    const d = new Date(dtISO);

    if (f.de && d < new Date(toISODateStart(f.de))) return false;

    if (f.ate && d > new Date(toISODateEnd(f.ate))) return false;

    return true;

  }

  function rowProdutos(pList){

    if (!pList?.length) return "—";

    const parts = pList.map(p=>{

      const tag = p.brinde ? " (Brinde)" : "";

      return `${p.nome||p.produtoId}×${p.quantidade}${tag}`;

    });

    return parts.join(", ");

  }

  function render(){

    // filtra

    let list = lancs.filter(x=>{

      if (!within(x.dataISO)) return false;

      if (f.prof && x.profissionalId !== f.prof) return false;

      if (f.cli  && x.clienteId     !== f.cli)  return false;

      if (f.forma && x.pagamento?.forma !== f.forma) return false;

      if (f.tipo){

        const isProdOnly = (!x.servico || (x.servico && x.servico.precoFinal===0 && !x.produtos?.length)) ? false : false; // placeholder

        if (f.tipo==="servico"    && !(x.tipo?.includes("servico"))) return false;

        if (f.tipo==="combo\_uso"  && !(x.tipo?.includes("combo\_uso"))) return false;

        if (f.tipo==="produto"    && !(x.produtos && x.produtos.length && !x.tipo?.includes("servico") && !x.tipo?.includes("combo\_uso"))) return false;

      }

      return true;

    });

    // ordena

    list.sort((a,b)=> f.ord==="asc" ? (new Date(a.dataISO)-new Date(b.dataISO)) : (new Date(b.dataISO)-new Date(a.dataISO)));

    // render

    tbody.innerHTML = list.map(x=>{

      return `

        <tr>

          <td>${new Date(x.dataISO).toLocaleDateString("pt-BR")} ${new Date(x.dataISO).toLocaleTimeString("pt-BR",{hour:"2-digit",minute:"2-digit"})}</td>

          <td>${x.profissionalNome||x.profissionalId||"—"}</td>

          <td>${x.clienteNome||x.clienteId||"—"}</td>

          <td>${x.servico?.nome || (x.tipo?.includes("combo\_uso") ? "Uso de combo" : "—")}</td>

          <td>${rowProdutos(x.produtos)}</td>

          <td>${x.pagamento?.forma || "—"}</td>

          <td>${fmtBRL(x.valores?.receitaBruta||0)}</td>

          <td>${fmtBRL(x.valores?.taxaPagamento||0)}</td>

          <td>${fmtBRL(x.valores?.comissao||0)}</td>

          <td>${fmtBRL(x.valores?.custoTotal||0)}</td>

          <td>${fmtBRL(x.valores?.resultadoLiquido||0)}</td>

          <td>

            <div class="cell-actions">

              <!-- edição futura controlada por permissão -->

              <!-- <button class="btn btn-yellow" data-act="edit" data-id="${x.id}">✏️</button> -->

              <button class="btn btn-red" data-act="del" data-id="${x.id}">🗑️</button>

            </div>

          </td>

        </tr>

      `;

    }).join("");

    // KPIs do período

    renderKPIs(list);

  }

  // ---------------- [KPIS] ----------------

  function renderKPIs(list){

    const sum = (k)=> list.reduce((a,b)=> a + (b.valores?.[k]||0), 0);

    $("#kpiReceitaBruta").textContent = fmtBRL(sum("receitaBruta"));

    $("#kpiTaxas").textContent        = fmtBRL(sum("taxaPagamento"));

    $("#kpiComissoes").textContent    = fmtBRL(sum("comissao"));

    $("#kpiCustos").textContent       = fmtBRL(sum("custoTotal"));

    $("#kpiResultado").textContent    = fmtBRL(sum("resultadoLiquido"));

  }

  // ---------------- [MODAL] Novo Lançamento ----------------

  const modal = $("#lanModal");

  function open(el){ el.classList.add("show"); }

  function close(el){ el.classList.remove("show"); }

  $$(".modal [data-close]").forEach(b=> b.addEventListener("click", e=> close(e.target.closest(".modal")) ));

  $("#btnFabAddLan")?.addEventListener("click", ()=>{

    // prepara selects do form

    fillSelect($("#lProf"), funcs);

    fillSelect($("#lCli"),  clis);

    fillServicos();

    fillCombosDoCliente(null); // vazio inicialmente

    $("#lanForm").reset();

    // defaults

    $("#lForma").value="Dinheiro";

    $("#lParcelas").value=1;

    $("#lTipoSrv").value="servico";

    $("#comboSelectBox").hidden = true;

    $("#srvPrecoBox").style.display = "";

    open(modal);

    recalcLive(); // KPIs ao vivo zerados

  });

  // Switch serviço/combo

  $("#lTipoSrv")?.addEventListener("change", e=>{

    const v = e.target.value;

    const isCombo = (v==="combo\_uso");

    $("#comboSelectBox").hidden = !isCombo;

    $("#srvSelectBox").hidden   = isCombo;

    $("#srvPrecoBox").style.display = isCombo ? "none" : "";

    recalcLive();

  });

  // Carrega serviços unitários do catálogo (para preço/custo)

  function fillServicos(){

    const unit = servs.filter(s=> s.tipoServico==="unitario");

    $("#lSrv").innerHTML = unit.map(s=>`<option value="${s.id}" data-preco="${Number(s.preco||0)}" data-custo="${Number(s.custoTotal||0)}">${s.nome||s.id}</option>`).join("");

    // set preço praticado inicial

    const opt = $("#lSrv").selectedOptions[0];

    $("#lPreco").value = opt ? Number(opt.dataset.preco||0) : 0;

  }

  $("#lSrv")?.addEventListener("change", ()=>{

    const opt = $("#lSrv").selectedOptions[0];

    $("#lPreco").value = opt ? Number(opt.dataset.preco||0) : 0;

    recalcLive();

  });

  // Combos do cliente selecionado

  $("#lCli")?.addEventListener("change", e=>{

    fillCombosDoCliente(e.target.value);

  });

  function fillCombosDoCliente(cliId){

    const mine = creds.filter(c=> c.clienteId===cliId);

    $("#lCombo").innerHTML = mine.length

      ? mine.map(c=>`<option value="${c.clienteCreditoId||c.comboServicoId}">${c.comboNome||"Combo"} — ${c.qtdDisponiveis}/${c.qtdTotal}${c.dataExpiraISO ? " • expira "+new Date(c.dataExpiraISO).toLocaleDateString("pt-BR") : ""}</option>`).join("")

      : `<option value="">(sem créditos)</option>`;

  }

  // Produtos no lançamento

  const prodListEl = $("#lanProdList");

  $("#btnAddProd")?.addEventListener("click", addProdRow);

  function addProdRow(){

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

    row.className = "svc-mat-row"; // reaproveita grade

    row.innerHTML = `

      <select class="lan-prod" style="width:100%">

        ${prods.map(p=>`<option value="${p.id}" data-preco="${Number(p.precoVenda||0)}" data-cmp="${Number(p.cmp||0)}">${p.nome||p.id}</option>`).join("")}

      </select>

      <input class="lan-qtd num" type="number" min="1" value="1" />

      <label style="display:flex;align-items:center;gap:6px"><input class="lan-brinde" type="checkbox" /> Brinde</label>

      <input class="lan-preco num" type="number" step="0.01" min="0" />

      <div class="del">Remover</div>

    `;

    prodListEl.appendChild(row);

    // set preço aplicado igual ao preçoVenda

    const sel = row.querySelector(".lan-prod");

    const preco = row.querySelector(".lan-preco");

    const opt = sel.selectedOptions[0];

    preco.value = Number(opt?.dataset?.preco||0);

    // handlers

    row.querySelector(".del").addEventListener("click", ()=> row.remove());

    row.addEventListener("input", recalcLive);

    sel.addEventListener("change", ()=>{

      const o = sel.selectedOptions[0];

      preco.value = Number(o?.dataset?.preco||0);

      recalcLive();

    });

  }

  // Live KPIs

  ["#lPreco","#lDescTipo","#lDescVal","#lForma","#lParcelas"].forEach(s=>{

    $(s)?.addEventListener("input", recalcLive);

    $(s)?.addEventListener("change", recalcLive);

  });

  function getFormSnapshot(){

    const tipoSrv = $("#lTipoSrv").value;

    const isCombo = (tipoSrv==="combo\_uso");

    // Serviço

    let srv = null;

    if (!isCombo){

      const sOpt = $("#lSrv").selectedOptions[0];

      const precoCat = Number(sOpt?.dataset?.preco||0);

      const custoMat = Number(sOpt?.dataset?.custo||0);

      const precoPrat= Number($("#lPreco").value||0);

      const dt = $("#lDescTipo").value;

      const dv = Number($("#lDescVal").value||0);

      let precoFinal = precoPrat;

      if (dt==="valor") precoFinal = Math.max(precoPrat - dv, 0);

      if (dt==="percentual") precoFinal = Math.max(precoPrat \* (1 - (dv/100)), 0);

      srv = {

        id: sOpt?.value, nome: sOpt?.textContent, precoCatalogo: precoCat,

        precoPraticado: precoPrat, descontoTipo: dt||null, descontoValor: dv||0,

        precoFinal, custoMateriais: custoMat,

        comboUso: { ativo:false, clienteCreditoId:null, comboServicoId:null }

      };

    }else{

      // combo uso: preço final=0, custo materiais opcional (puxa do base? aqui ficar 0)

      const cOpt = $("#lCombo").selectedOptions[0];

      srv = {

        id: null, nome: "Combo (uso)", precoCatalogo: 0, precoPraticado: 0,

        descontoTipo: null, descontoValor: 0, precoFinal: 0, custoMateriais: 0,

        comboUso: { ativo:true, clienteCreditoId: cOpt?.value||null, comboServicoId: cOpt?.value||null }

      };

    }

    // Produtos

    const itens = $$(".svc-mat-row", prodListEl).map(row=>{

      const sel  = row.querySelector(".lan-prod");

      const qtd  = Number(row.querySelector(".lan-qtd").value||0);

      const br   = row.querySelector(".lan-brinde").checked;

      const prc  = Number(row.querySelector(".lan-preco").value||0);

      const cmp  = Number(sel.selectedOptions[0]?.dataset?.cmp||0);

      return {

        produtoId: sel.value, nome: sel.selectedOptions[0]?.textContent,

        quantidade: qtd, brinde: br,

        precoUnitarioAplicado: br ? 0 : prc,

        custoUnitarioAplicado: cmp

      };

    });

    const forma = $("#lForma").value;

    const parcelas = Number($("#lParcelas").value||1);

    return { tipoSrv, isCombo, srv, itens, forma, parcelas };

  }

  // ---------------- [CALC] valores do lançamento ----------------

  function calcValores(snapshot, profId){

    const { isCombo, srv, itens, forma } = snapshot;

    // Receita

    const receitaServ = srv?.precoFinal || 0;               // 0 se combo

    const receitaProd = itens.reduce((a,b)=> a + (b.brinde?0:(b.precoUnitarioAplicado\*b.quantidade)), 0);

    const receitaBruta = receitaServ + receitaProd;

    // Taxa pagamento sobre total pago

    const perc = (conf.taxas?.[forma] ?? 0) / 100;

    const taxaPagamento = receitaBruta \* perc; // (parcelado poderia ter fixa extra — encaixe aqui se quiser)

    // Comissão (regra: por padrão SÓ serviço; base = antes da taxa)

    const func = funcs.find(x=>x.id===profId);

    let base = receitaServ;

    if (conf.baseComissao==="DEPOIS\_TAXA") base = Math.max(receitaServ - taxaPagamento, 0);

    let comissao = 0;

    if (func?.commissionType==="percentual") comissao = base \* (Number(func.commissionValue||0)/100);

    if (func?.commissionType==="valor\_fixo") comissao = Number(func.commissionValue||0);

    // (Se quiser comissionar produtos: some receitaProd em 'base' quando conf.comissionarProdutos=true)

    // Custos: materiais do serviço + custo CMP dos produtos

    const custoServ = Number(srv?.custoMateriais||0);

    const custoProd = itens.reduce((a,b)=> a + (b.custoUnitarioAplicado\*b.quantidade), 0);

    const custoTotal= custoServ + custoProd;

    const resultado = receitaBruta - taxaPagamento - comissao - custoTotal;

    return {

      receitaBruta, taxaPagamento, comissao, custoTotal, resultadoLiquido: resultado

    };

  }

  function recalcLive(){

    const snap = getFormSnapshot();

    const profId = $("#lProf").value||"";

    const vals = calcValores(snap, profId);

    $("#kpiLiveReceita").textContent  = fmtBRL(vals.receitaBruta);

    $("#kpiLiveTaxa").textContent     = fmtBRL(vals.taxaPagamento);

    $("#kpiLiveComissao").textContent = fmtBRL(vals.comissao);

    $("#kpiLiveCustos").textContent   = fmtBRL(vals.custoTotal);

    $("#kpiLiveResultado").textContent= fmtBRL(vals.resultadoLiquido);

  }

  // Submit do lançamento

  $("#lanForm")?.addEventListener("submit", (e)=>{

    e.preventDefault();

    const profSel = $("#lProf");

    const cliSel  = $("#lCli");

    if (!profSel.value) return alert("Selecione o profissional.");

    if (!cliSel.value)  return alert("Selecione o cliente.");

    const snap = getFormSnapshot();

    // validações rápidas

    if (!snap.isCombo && (!snap.srv || snap.srv.precoFinal<0)) return alert("Serviço inválido.");

    for (const it of snap.itens){

      if (it.quantidade<=0) return alert("Quantidade de produto inválida.");

      // estoque negativo?

      const p = prods.find(x=>x.id===it.produtoId);

      if (p && !conf.permitirEstoqueNegativo && (p.estoqueAtual||0) - it.quantidade < 0){

        return alert(`Estoque insuficiente para ${p.nome||p.id}.`);

      }

    }

    // calcula valores finais

    const vals = calcValores(snap, profSel.value);

    // monta objeto lançamento

    const id = "lan\_" + Math.random().toString(16).slice(2,6).toUpperCase();

    const lanc = {

      id,

      dataISO: new Date().toISOString(),

      profissionalId: profSel.value,

      profissionalNome: profSel.selectedOptions[0]?.textContent,

      clienteId: cliSel.value,

      clienteNome: cliSel.selectedOptions[0]?.textContent,

      tipo: `${snap.isCombo?"combo\_uso":"servico"}${snap.itens.length?"+produtos":""}`,

      servico: snap.srv,

      produtos: snap.itens,

      pagamento: { forma: snap.forma, parcelas: snap.parcelas||1 },

      valores: {

        receitaBruta: vals.receitaBruta,

        taxaPagamento: vals.taxaPagamento,

        comissao: vals.comissao,

        custoTotal: vals.custoTotal,

        resultadoLiquido: vals.resultadoLiquido

      },

      obs: $("#lObs").value||""

    };

    // ---------------- [INTEGRACAO ESTOQUE] ----------------

    // Para cada produto, cria movimentação de SAÍDA no kardex e baixa estoque.

    // Custo aplicado = CMP vigente (prods[].cmp). Se FIFO, aqui substitua pela lógica de lotes.

    snap.itens.forEach(it=>{

      // baixa estoque

      const p = prods.find(x=>x.id===it.produtoId);

      if (p){

        p.estoqueAtual = Number(p.estoqueAtual||0) - it.quantidade;

        // (se FIFO: remover do lote mais antigo; se CMP: apenas reduz)

        // grava kardex

        kardex.push({

          id: "mov\_" + Math.random().toString(16).slice(2,8),

          productId: p.id,

          tipo: "saida",

          motivo: it.brinde ? "brinde" : "venda",

          dataISO: new Date().toISOString(),

          quantidade: it.quantidade,

          custoUnitario: Number(p.cmp||it.custoUnitarioAplicado||0),

          precoUnitario: it.brinde ? 0 : it.precoUnitarioAplicado,

          lote: null, validadeISO: null,

          fornecedor: null,

          observacao: `Lançamento ${id}`

        });

      }

    });

    write(LS\_PROD, prods);

    write(LS\_KARDEX, kardex);

    // ---------------- [INTEGRACAO COMBOS] ----------------

    // Se uso de combo: debitar 1 crédito do cliente (se houver)

    if (snap.isCombo){

      const cliId = cliSel.value;

      const c = creds.find(x=>x.clienteId===cliId && (x.clienteCreditoId===snap.srv.comboUso.clienteCreditoId || x.comboServicoId===snap.srv.comboUso.comboServicoId));

      if (c){

        if ((c.qtdDisponiveis||0) <= 0){

          alert("Crédito indisponível/vencido para este combo."); return;

        }

        c.qtdConsumidos = Number(c.qtdConsumidos||0) + 1;

        c.qtdDisponiveis= Math.max(0, Number(c.qtdTotal||0) - Number(c.qtdConsumidos||0));

        write(LS\_CRED, creds);

      }

    }

    // salva lançamento

    lancs.push(lanc);

    write(LS\_LANC, lancs);

    // feedback + fecha modal + re-render

    close(modal);

    render();

  });

  // Excluir lançamento (simples — recomendável usar estorno no futuro)

  let delId = null;

  tbody?.addEventListener("click", (e)=>{

    const btn = e.target.closest("[data-act]");

    if (!btn) return;

    const id = btn.dataset.id;

    if (btn.dataset.act==="del"){

      delId = id; open($("#lanDelModal"));

    }

  });

  $("#lanConfirmDel")?.addEventListener("click", ()=>{

    if (!delId) return;

    lancs = lancs.filter(x=>x.id!==delId);

    write(LS\_LANC, lancs);

    delId = null;

    close($("#lanDelModal"));

    render();

  });

  // ---------------- [EXPORT/IMPORT] ----------------

  $("#lanExport")?.addEventListener("click", ()=>{

    // exporta somente os registros visíveis pelo filtro atual

    const rows = Array.from(tbody.querySelectorAll("tr")).map(tr=>{

      const idCell = tr.querySelector("[data-id]"); // (não usamos aqui)

      return null;

    });

    const blob = new Blob([JSON.stringify({ lancamentos: lancs }, null, 2)], {type:"application/json"});

    const url  = URL.createObjectURL(blob);

    const a = document.createElement("a");

    a.href = url; a.download = "lancamentos\_barbearia.json"; a.click();

    URL.revokeObjectURL(url);

  });

  $("#lanImport")?.addEventListener("change", async e=>{

    const f = e.target.files?.[0]; if (!f) return;

    try{

      const obj = JSON.parse(await f.text());

      if (!obj || !Array.isArray(obj.lancamentos)) throw 0;

      // opção simples: mesclar (concat); se preferir substituir, faça lancs = obj.lancamentos

      lancs = lancs.concat(obj.lancamentos);

      write(LS\_LANC, lancs);

      render();

    }catch{

      alert("JSON inválido.");

    }

    e.target.value = "";

  });

  // ---------------- [BOOT] ----------------

  // Caso não haja nada, cria alguns seeds para visualização

  if (!funcs.length){

    funcs = [{ id:"EMP-77BC", name:"Carlos", commissionType:"percentual", commissionValue:40 }];

    write(LS\_FUNC, funcs);

  }

  if (!clis.length){

    clis = [{ id:"C-8A3F", name:"Rafael Brandão" }];

    write(LS\_CLI, clis);

  }

  if (!servs.length){

    servs = [{ id:"svc\_corte", tipoServico:"unitario", nome:"Corte", preco:45, custoTotal:2.5 }];

    write(LS\_SERV, servs);

  }

  if (!prods.length){

    prods = [{ id:"P-8A3F", nome:"Pomada 150g", precoVenda:39.9, cmp:21.3, estoqueAtual:18 }];

    write(LS\_PROD, prods);

  }

  if (!lancs.length){

    const seed = {

      id:"lan\_SEED", dataISO:new Date().toISOString(),

      profissionalId:"EMP-77BC", profissionalNome:"Carlos",

      clienteId:"C-8A3F", clienteNome:"Rafael",

      tipo:"servico+produtos",

      servico:{ id:"svc\_corte", nome:"Corte", precoCatalogo:45, precoPraticado:45, descontoTipo:null, descontoValor:0, precoFinal:45, custoMateriais:2.5, comboUso:{ativo:false}},

      produtos:[{ produtoId:"P-8A3F", nome:"Pomada 150g", quantidade:1, brinde:false, precoUnitarioAplicado:39.9, custoUnitarioAplicado:21.3 }],

      pagamento:{ forma:"Pix", parcelas:1 },

      valores:{ receitaBruta:84.9, taxaPagamento:0, comissao:18, custoTotal:23.8, resultadoLiquido:43.1 },

      obs:"seed"

    };

    lancs = [seed]; write(LS\_LANC, lancs);

  }

  // Seta filtro padrão: Hoje + Todos

  const rHoje = rangeOf("dia", new Date());

  $("#fDataDe").value = rHoje.de.slice(0,10);

  $("#fDataAte").value= rHoje.ate.slice(0,10);

  render();

})();

// js/login.js

// Responsável por login, cadastro e autenticação via Google

document.addEventListener("DOMContentLoaded", () => {

    // Inicializa os serviços do Firebase que vamos usar

    const auth = firebase.auth();

    const firestore = firebase.firestore();

    // --- LÓGICA DE LOGIN COM EMAIL E SENHA ---

    const formLogin = document.getElementById("formLogin");

    if (formLogin) {

        formLogin.addEventListener("submit", (e) => {

            e.preventDefault(); // Impede o recarregamento da página

            const email = document.getElementById("loginEmail").value;

            const senha = document.getElementById("loginSenha").value;

            const erroLogin = document.getElementById("loginErro");

            auth.signInWithEmailAndPassword(email, senha)

                .then(() => {

                    // Se o login for bem-sucedido, redireciona para a página principal do cliente

                    // <<-- CORREÇÃO AQUI -->>

                    window.location.href = "home.html";

                })

                .catch(err => {

                    // Se der erro, mostra a mensagem na tela

                    erroLogin.innerText = "Erro: " + err.message;

                });

        });

    }

    // --- LÓGICA DE CADASTRO COM EMAIL E SENHA ---

    const btnCadastro = document.getElementById("btnCadastro");

    if (btnCadastro) {

        btnCadastro.addEventListener("click", () => {

            const nome = document.getElementById("cadNome").value;

            const telefone = document.getElementById("cadTelefone").value;

            const email = document.getElementById("cadEmail").value;

            const senha = document.getElementById("cadSenha").value;

            const nascimento = document.getElementById("cadNascimento").value;

            const erroCadastro = document.getElementById("cadastroErro");

            // Validação simples para campos obrigatórios

            if (!nome || !telefone || !email || !senha) {

                erroCadastro.innerText = "Preencha todos os campos obrigatórios.";

                return;

            }

            // Cria o usuário no Firebase Authentication

            auth.createUserWithEmailAndPassword(email, senha)

                .then((userCred) => {

                    // Se o usuário for criado com sucesso, salva os dados adicionais no Firestore

                    const user = userCred.user;

                    return firestore.collection("clientes").doc(user.uid).set({

                        nome: nome,

                        telefone: telefone,

                        email: email,

                        nascimento: nascimento || null,

                        criadoEm: new Date()

                    });

                })

                .then(() => {

                    // Após salvar no Firestore, redireciona para a página principal do cliente

                    // <<-- CORREÇÃO AQUI -->>

                    window.location.href = "home.html";

                })

                .catch(err => {

                    // Se der erro, mostra a mensagem na tela

                    erroCadastro.innerText = "Erro: " + err.message;

                });

        });

    }

    // --- LÓGICA DE LOGIN COM GOOGLE ---

    const btnGoogle = document.getElementById("btnGoogle");

    if (btnGoogle) {

        btnGoogle.addEventListener("click", () => {

            const provider = new firebase.auth.GoogleAuthProvider();

            auth.signInWithPopup(provider)

                .then(async (result) => {

                    const user = result.user;

                    // Verifica se o cliente já existe no nosso banco de dados (Firestore)

                    const docRef = firestore.collection("clientes").doc(user.uid);

                    const docSnap = await docRef.get();

                    if (!docSnap.exists) {

                        // Se não existir, cria um novo registro para ele

                        await docRef.set({

                            nome: user.displayName,

                            email: user.email,

                            telefone: user.phoneNumber || "Não informado",

                            nascimento: null,

                            criadoEm: new Date()

                        });

                    }

                    // Redireciona para a página principal do cliente

                    // <<-- CORREÇÃO AQUI -->>

                    window.location.href = "home.html";

                })

                .catch(err => {

                    alert("Erro no login com Google: " + err.message);

                });

        });

    }

    // --- LÓGICA PARA ALTERNAR ENTRE FORMULÁRIOS DE LOGIN E CADASTRO ---

    const boxLogin = document.getElementById("boxLogin");

    const boxCadastro = document.getElementById("boxCadastro");

    const btnMostrarCadastro = document.getElementById("mostrarCadastro");

    const btnMostrarLogin = document.getElementById("mostrarLogin");

    // "Criar conta" -> esconde o login e mostra o cadastro

    if (btnMostrarCadastro) {

        btnMostrarCadastro.addEventListener("click", (e) => {

            e.preventDefault();

            boxLogin.style.display = "none";

            boxCadastro.style.display = "block";

        });

    }

    // "Já tenho conta" -> esconde o cadastro e mostra o login

    if (btnMostrarLogin) {

        btnMostrarLogin.addEventListener("click", (e) => {

            e.preventDefault();

            boxCadastro.style.display = "none";

            boxLogin.style.display = "block";

        });

    }

});

// js/perfil-funcionario.js - VERSÃO FINAL (com label de mês dinâmico)

(function() {

    // --- SIMULAÇÃO DE LOGIN ---

    const mockFuncionarioLogado = { id: "EMP-IGOR", nome: "Igor" };

    // --- Chaves e Helpers ---

    const LS\_CFG = 'barbearia\_agenda\_config\_v1';

    const START\_H = 10, END\_H = 20;

    const pad2 = n => String(n).padStart(2, '0');

    const toYMD = d => `${d.getFullYear()}-${pad2(d.getMonth() + 1)}-${pad2(d.getDate())}`;

    const addDays = (d, n) => { const x = new Date(d); x.setDate(x.getDate() + n); return x; };

    const dow1to7 = date => { const js = (new Date(date)).getDay(); return js === 0 ? 7 : js; };

    const DOW\_PT = ['Dom', 'Seg', 'Ter', 'Qua', 'Qui', 'Sex', 'Sáb'];

    const loadCfg = () => { try { return JSON.parse(localStorage.getItem(LS\_CFG) || '{}'); } catch { return {}; } };

    const saveCfg = cfg => localStorage.setItem(LS\_CFG, JSON.stringify(cfg));

    // --- Referências do DOM ---

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

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

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

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

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

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

    // --- Estado ---

    const CFG = loadCfg();

    let workingCopy, editingDateYMD;

    const currentDate = new Date();

    const TOTAL\_DAYS\_TO\_SHOW = 30;

    const VISIBLE\_DAYS = 5;

    let dayStripStartIndex = 0;

    let allDayPills = [];

    function init() {

        if (!mockFuncionarioLogado?.id) {

            document.querySelector('.admin-main').innerHTML = "<h1>Erro: Funcionário não identificado.</h1>";

            return;

        }

        if (!CFG[mockFuncionarioLogado.id]) CFG[mockFuncionarioLogado.id] = { \_overrides: {} };

        workingCopy = JSON.parse(JSON.stringify(CFG[mockFuncionarioLogado.id]));

        generateAllDayPills();

        updateDayStripView();

        updateMonthHeader();

        selectDayForEdit(toYMD(currentDate));

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

            dayStripStartIndex = Math.max(0, dayStripStartIndex - 1);

            updateDayStripView();

            updateMonthHeader();

        });

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

            dayStripStartIndex = Math.min(TOTAL\_DAYS\_TO\_SHOW - VISIBLE\_DAYS, dayStripStartIndex + 1);

            updateDayStripView();

            updateMonthHeader();

        });

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

            if (!workingCopy) return;

            CFG[mockFuncionarioLogado.id] = workingCopy;

            saveCfg(CFG);

            alert(`Disponibilidade de ${mockFuncionarioLogado.nome} salva com sucesso!`);

        });

    }

    // --- FUNÇÃO ATUALIZADA ---

    // Agora verifica o primeiro e o último dia visível para criar o label.

    function updateMonthHeader() {

        if (!dayMonth || allDayPills.length === 0) return;

        const firstVisibleDayYMD = allDayPills[dayStripStartIndex].dataset.ymd;

        const lastVisibleIndex = Math.min(dayStripStartIndex + VISIBLE\_DAYS - 1, allDayPills.length - 1);

        const lastVisibleDayYMD = allDayPills[lastVisibleIndex].dataset.ymd;

        const startDate = new Date(`${firstVisibleDayYMD}T12:00:00`);

        const endDate = new Date(`${lastVisibleDayYMD}T12:00:00`);

        const m1 = startDate.toLocaleString('pt-BR', { month: 'long' });

        const y1 = startDate.getFullYear();

        const m2 = endDate.toLocaleString('pt-BR', { month: 'long' });

        const y2 = endDate.getFullYear();

        const cap = str => str.charAt(0).toUpperCase() + str.slice(1);

        let label = "";

        if (y1 === y2) {

            if (m1 === m2) {

                // Mesmo mês e ano: "Setembro de 2025"

                label = `${cap(m1)} de ${y1}`;

            } else {

                // Meses diferentes, mesmo ano: "Setembro / Outubro de 2025"

                label = `${cap(m1)} / ${cap(m2)} de ${y1}`;

            }

        } else {

            // Anos diferentes: "Dezembro de 2025 / Janeiro de 2026"

            label = `${cap(m1)} de ${y1} / ${cap(m2)} de ${y2}`;

        }

        dayMonth.textContent = label;

    }

    function generateAllDayPills() {

        allDayPills = [];

        const base = new Date();

        for (let i = 0; i < TOTAL\_DAYS\_TO\_SHOW; i++) {

            const d = addDays(base, i);

            const ymd = toYMD(d);

            const pill = document.createElement('button');

            pill.type = 'button';

            pill.className = 'day-pill';

            pill.dataset.ymd = ymd;

            pill.innerHTML = `<span class="dow">${DOW\_PT[d.getDay()]}</span><span class="d">${d.getDate()}</span>`;

            if (toYMD(new Date()) === ymd) pill.classList.add('today');

            pill.addEventListener('click', () => selectDayForEdit(ymd));

            allDayPills.push(pill);

        }

    }

    function updateDayStripView() {

        dayStrip.innerHTML = '';

        const visiblePills = allDayPills.slice(dayStripStartIndex, dayStripStartIndex + VISIBLE\_DAYS);

        visiblePills.forEach(pill => dayStrip.appendChild(pill));

        prevDaysBtn.disabled = dayStripStartIndex === 0;

        nextDaysBtn.disabled = dayStripStartIndex >= TOTAL\_DAYS\_TO\_SHOW - VISIBLE\_DAYS;

    }

    function selectDayForEdit(ymd) {

        editingDateYMD = ymd;

        allDayPills.forEach(p => p.classList.toggle('active', p.dataset.ymd === ymd));

        buildDayTableForEdit();

    }

    function buildDayTableForEdit() {

        const TIMES = Array.from({ length: (END\_H - START\_H) \* 2 }, (\_, i) => `${pad2(START\_H + Math.floor(i / 2))}:${pad2((i % 2) \* 30)}`);

        const d = new Date(`${editingDateYMD}T12:00:00`);

        const dow = dow1to7(d);

        const baseDay = (workingCopy[dow] || {});

        const ov = (workingCopy.\_overrides && workingCopy.\_overrides[editingDateYMD]) || {};

        weekMatrix.innerHTML = '';

        const table = document.createElement('table');

        table.className = 'week-table';

        const thead = table.createTHead();

        thead.insertRow().innerHTML = `<th class="week-time"></th><th>${d.toLocaleDateString('pt-BR')}</th>`;

        const tbody = table.createTBody();

        for (const time of TIMES) {

            const isAvailable = ov.hasOwnProperty(time) ? !!ov[time] : !!baseDay[time];

            const row = tbody.insertRow();

            row.insertCell().outerHTML = `<th class="week-time">${time}</th>`;

            const cell = row.insertCell();

            cell.className = 'cell';

            cell.classList.toggle('active', isAvailable);

            cell.dataset.time = time;

            cell.textContent = isAvailable ? 'Disponível' : 'Bloqueado';

            cell.addEventListener('click', toggleCell);

        }

        weekMatrix.appendChild(table);

    }

    function toggleCell(e) {

        const td = e.currentTarget;

        const time = td.dataset.time;

        if (!workingCopy.\_overrides) workingCopy.\_overrides = {};

        if (!workingCopy.\_overrides[editingDateYMD]) workingCopy.\_overrides[editingDateYMD] = {};

        const newOverrideState = !td.classList.contains('active');

        workingCopy.\_overrides[editingDateYMD][time] = newOverrideState;

        td.classList.toggle('active', newOverrideState);

        td.textContent = newOverrideState ? 'Disponível' : 'Bloqueado';

    }

    if (document.readyState === 'loading') {

        document.addEventListener('DOMContentLoaded', init);

    } else {

        init();

    }

})();

/\* Produtos + Estoque + Kardex (localStorage)

   Chaves:

   - barbearia\_produtos\_v1 (array de produtos)

   - barbearia\_kardex\_v1   (array de movimentações)

   - barbearia\_config\_v1   (config global simples)

   Política de custo padrão: CMP (Custo Médio Ponderado).

   Comentários explicam onde alternar para FIFO no futuro.

\*/

(function(){

  // ===== [CONFIG ESTOQUE] =====

  const LS\_PROD  = "barbearia\_produtos\_v1";

  const LS\_KDX   = "barbearia\_kardex\_v1";

  const LS\_CFG   = "barbearia\_config\_v1";

  const DEFAULT\_CFG = { politicaCusto: "CMP", alertaValidadeDias: 30, permitirEstoqueNegativo: false };

  // ===== [UTIL BRL] =====

  const fmtBRL = (v) => new Intl.NumberFormat("pt-BR",{style:"currency",currency:"BRL"}).format(Number(v||0));

  const badge = (label, color) => `<span style="display:inline-block;padding:2px 8px;border-radius:999px;font-size:12px;background:${color};color:#000;font-weight:800">${label}</span>`;

  // ===== [STATE + LS] =====

  let produtos = read(LS\_PROD, []);

  let kardex   = read(LS\_KDX, []);

  let config   = read(LS\_CFG, DEFAULT\_CFG);

  if (!config.politicaCusto) config = {...DEFAULT\_CFG, ...config}, write(LS\_CFG, config);

  function read(key, fallback){ try{ const r = JSON.parse(localStorage.getItem(key)||"null"); return r ?? fallback; }catch{ return fallback; } }

  function write(key, val){ localStorage.setItem(key, JSON.stringify(val)); }

  // ===== IDs =====

  const genPid = () => {

    const set = new Set(produtos.map(p => p.id));

    let id;

    do { id = "P-" + Math.random().toString(16).slice(2,6).toUpperCase(); } while(set.has(id));

    return id;

  };

  // ===== DOM refs =====

  const $ = (q) => document.querySelector(q);

  const $$ = (q) => document.querySelectorAll(q);

  const tbody = $("#prdTbody");

  const qInput = $("#prdQ");

  const btnFab = $("#btnFabAddPrd");

  const modal = $("#prdModal");

  const delModal = $("#prdModalDelete");

  const movModal = $("#movModal");

  // form produto

  const pId = $("#pId"), pNome=$("#pNome"), pSKU=$("#pSKU"), pCategoria=$("#pCategoria"),

        pUn=$("#pUn"), pPreco=$("#pPreco"), pFornecedor=$("#pFornecedor"),

        pEstMin=$("#pEstMin"), pPolitica=$("#pPolitica"), pObs=$("#pObs");

  // form mov

  const mProductId=$("#mProductId"), mTipo=$("#mTipo"), mQtd=$("#mQtd"),

        mCusto=$("#mCusto"), mFornecedor=$("#mFornecedor"), mLote=$("#mLote"),

        mValidade=$("#mValidade"), mMotivo=$("#mMotivo"), mPreco=$("#mPreco"),

        mJust=$("#mJust"), mObs=$("#mObs"), boxEntrada=$("#boxEntrada"),

        boxSaida=$("#boxSaida"), boxAjuste=$("#boxAjuste");

  // ===== Render =====

  let query = "";

  function render(){

    const term = query.trim().toLowerCase();

    const data = produtos.filter(p=>{

      if(!term) return true;

      return [p.nome,p.sku,p.categoria,p.fornecedor].join(" ").toLowerCase().includes(term);

    });

    tbody.innerHTML = data.map(p=>{

      const lucro = Number(p.precoVenda||0) - Number(p.cmp||0);

      const margem = p.precoVenda ? (lucro / Number(p.precoVenda))\*100 : 0;

      const st = statusProd(p);

      const stChip =

        st==="ZERADO"  ? badge("ZERADO","#ff7878") :

        st==="BAIXO"   ? badge("BAIXO","#ffd166") :

        st==="VENCENDO"? badge("VENCENDO","#7ee787") :

                         badge("OK","#a0e7a0");

      return `

        <tr>

          <td>${p.nome||""}</td>

          <td>${p.sku||p.id}</td>

          <td>${p.categoria||"—"}</td>

          <td>${p.unidade||""}</td>

          <td>${p.estoqueAtual ?? 0}</td>

          <td>${p.estoqueMinimo ?? 0}</td>

          <td>${fmtBRL(p.precoVenda||0)}</td>

          <td>${fmtBRL(p.cmp||0)}</td>

          <td>${fmtBRL(lucro)} <span class="muted">(${margem.toFixed(0)}%)</span></td>

          <td>${stChip}</td>

          <td class="hide-sm">${p.fornecedor||"—"}</td>

          <td>

            <div class="cell-actions">

              <button class="btn btn-yellow" data-act="edit" data-id="${p.id}" title="Editar">✏️</button>

              <button class="btn btn-blue"   data-act="move" data-id="${p.id}" title="Movimentar">🔁</button>

              <button class="btn btn-red"    data-act="del"  data-id="${p.id}" title="Excluir">🗑️</button>

            </div>

          </td>

        </tr>

      `;

    }).join("");

  }

  function statusProd(p){

    if ((p.estoqueAtual ?? 0) <= 0) return "ZERADO";

    if ((p.estoqueAtual ?? 0) <= (p.estoqueMinimo ?? 0)) return "BAIXO";

    // “VENCENDO” simples: se houver lotes e algum com validade <= X dias

    const dias = (d1,d2)=> Math.ceil((d2-d1)/86400000);

    const lim = Number(config.alertaValidadeDias||30);

    const now = new Date();

    if (Array.isArray(p.lotes) && p.lotes.some(l=> l.validadeISO && dias(now,new Date(l.validadeISO)) <= lim)) return "VENCENDO";

    return "OK";

  }

  // ===== Modais util =====

  const open = (el)=>{ el.classList.add("show"); el.setAttribute("aria-hidden","false"); };

  const close = (el)=>{ el.classList.remove("show"); el.setAttribute("aria-hidden","true"); };

  $$("[data-close]").forEach(b=> b.addEventListener("click",()=>{ close(modal); close(delModal); close(movModal);} ));

  // ===== Novo produto =====

  btnFab?.addEventListener("click", ()=>{

    resetFormProduto();

    $("#prdModalTitle").textContent = "Novo produto";

    open(modal);

  });

  function resetFormProduto(p=null){

    pId.value = p?.id || "";

    pNome.value = p?.nome || "";

    pSKU.value  = p?.sku  || (genPid());

    pCategoria.value = p?.categoria || "";

    pUn.value   = p?.unidade || "un";

    pPreco.value= p?.precoVenda ?? 0;

    pFornecedor.value = p?.fornecedor || "";

    pEstMin.value = p?.estoqueMinimo ?? 0;

    pPolitica.value = (config.politicaCusto||"CMP").toUpperCase()+" (padrão)";

    pObs.value = p?.obs || "";

  }

  $("#prdForm")?.addEventListener("submit",(e)=>{

    e.preventDefault();

    if (!pNome.value.trim() || !pSKU.value.trim() || !pUn.value.trim()) return alert("Preencha Nome, SKU e Unidade.");

    if (Number(pPreco.value) < 0 || Number(pEstMin.value) < 0) return alert("Preço/Estoque mínimo inválidos.");

    // SKU único

    const sku = pSKU.value.trim();

    const id  = pId.value || genPid();

    const dup = produtos.find(x=> (x.sku||x.id) === sku && x.id !== id);

    if (dup) return alert("SKU já existe.");

    const payload = {

      id, nome: pNome.value.trim(), sku,

      categoria: pCategoria.value.trim() || null,

      unidade: pUn.value.trim(),

      precoVenda: Number(pPreco.value||0),

      fornecedor: pFornecedor.value.trim() || null,

      estoqueMinimo: Number(pEstMin.value||0),

      estoqueAtual: produtos.find(x=>x.id===id)?.estoqueAtual ?? 0,

      politicaCusto: config.politicaCusto || "CMP",

      cmp: produtos.find(x=>x.id===id)?.cmp ?? 0,

      lotes: produtos.find(x=>x.id===id)?.lotes ?? [], // usado no FIFO/validade

      foto: produtos.find(x=>x.id===id)?.foto || "",

      obs: pObs.value.trim() || ""

    };

    const i = produtos.findIndex(x=>x.id===id);

    if (i>=0) produtos[i]=payload; else produtos.push(payload);

    write(LS\_PROD, produtos);

    render();

    close(modal);

  });

  // ===== Tabela: editar / deletar / movimentar =====

  tbody?.addEventListener("click",(e)=>{

    const btn = e.target.closest("[data-act]");

    if (!btn) return;

    const id = btn.dataset.id;

    const p = produtos.find(x=>x.id===id);

    if (!p) return;

    if (btn.dataset.act==="edit"){

      resetFormProduto(p);

      $("#prdModalTitle").textContent = "Editar produto";

      open(modal);

    }

    if (btn.dataset.act==="del"){

      $("#prdDelNome").textContent = p.nome || "este produto";

      $("#btnPrdConfirmDelete").dataset.id = id;

      open(delModal);

    }

    if (btn.dataset.act==="move"){

      resetMovForm(p);

      open(movModal);

    }

  });

  $("#btnPrdConfirmDelete")?.addEventListener("click",(e)=>{

    const id = e.currentTarget.dataset.id;

    produtos = produtos.filter(x=>x.id!==id);

    write(LS\_PROD, produtos);

    render();

    close(delModal);

  });

  // ===== Busca =====

  qInput?.addEventListener("input",(e)=>{ query = e.target.value || ""; render(); });

  // ===== Exportar / Importar / Zerar =====

  $("#prdExport")?.addEventListener("click", ()=>{

    const payload = { produtos, kardex, config };

    const blob = new Blob([JSON.stringify(payload,null,2)], {type:"application/json"});

    const url = URL.createObjectURL(blob);

    const a = document.createElement("a");

    a.href = url; a.download = "produtos\_kardex\_barbearia.json"; a.click();

    URL.revokeObjectURL(url);

  });

  $("#prdImportFile")?.addEventListener("change", async (e)=>{

    const file = e.target.files?.[0]; if(!file) return;

    try{

      const text = await file.text();

      const data = JSON.parse(text);

      if (!data || !Array.isArray(data.produtos) || !Array.isArray(data.kardex)) throw 0;

      produtos = data.produtos; kardex = data.kardex; config = {...DEFAULT\_CFG, ...(data.config||{})};

      write(LS\_PROD, produtos); write(LS\_KDX, kardex); write(LS\_CFG, config);

      render(); alert("Importado com sucesso.");

    }catch{ alert("JSON inválido."); }

    e.target.value = "";

  });

  $("#prdZerar")?.addEventListener("click", ()=>{

    if (!confirm("Apagar TODOS os produtos e o Kardex?")) return;

    produtos = []; kardex = [];

    write(LS\_PROD, produtos); write(LS\_KDX, kardex);

    render();

  });

  // ===== [MOVIMENTACAO: ENTRADA/SAIDA/AJUSTE] =====

  function resetMovForm(p){

    $("#movTitle").textContent = `Movimentar: ${p.nome}`;

    mProductId.value = p.id;

    mTipo.value = "entrada";

    mQtd.value = "";

    mCusto.value = "";

    mFornecedor.value = p.fornecedor || "";

    mLote.value = "";

    mValidade.value = "";

    mMotivo.value = "venda";

    mPreco.value = "";

    mJust.value = "";

    mObs.value = "";

    switchBoxes();

  }

  mTipo?.addEventListener("change", switchBoxes);

  function switchBoxes(){

    const t = mTipo.value;

    boxEntrada.hidden = (t!=="entrada");

    boxSaida.hidden   = (t!=="saida");

    boxAjuste.hidden  = (t!=="ajuste");

  }

  $("#movForm")?.addEventListener("submit",(e)=>{

    e.preventDefault();

    const id = mProductId.value;

    const p = produtos.find(x=>x.id===id); if(!p) return alert("Produto não encontrado.");

    const tipo = mTipo.value;

    const qtd = Number(mQtd.value || 0);

    const nowISO = new Date().toISOString();

    if (tipo==="entrada"){

      const custo = Number(mCusto.value||0);

      if (qtd<=0 || custo<=0) return alert("Informe quantidade e custo válidos.");

      // CMP: (EstoqueAnterior\*CMP + qtd\*custo) / (EstoqueAnterior+qtd)

      const estAnt = Number(p.estoqueAtual||0);

      const cmpAnt = Number(p.cmp||0);

      const cmpNovo = (estAnt\*cmpAnt + qtd\*custo) / (estAnt + qtd);

      p.cmp = isFinite(cmpNovo) ? cmpNovo : custo;

      p.estoqueAtual = estAnt + qtd;

      // Se quiser FIFO no futuro: empilhar lote {qtd,custo,lote,validadeISO}

      if (!Array.isArray(p.lotes)) p.lotes = [];

      if (mLote.value || mValidade.value) {

        p.lotes.push({ lote: mLote.value||null, qtd, custo, validadeISO: mValidade.value||null });

      }

      kardex.push({

        id: "mov\_"+cryptoRand(), productId: p.id, tipo:"entrada", motivo:"compra",

        dataISO: nowISO, quantidade: qtd, custoUnitario: custo, precoUnitario: 0,

        lote: mLote.value||null, validadeISO: mValidade.value||null,

        fornecedor: mFornecedor.value||p.fornecedor||null, observacao: mObs.value||""

      });

    }

    if (tipo==="saida"){

      if (qtd<=0) return alert("Quantidade deve ser > 0.");

      if (!config.permitirEstoqueNegativo && (p.estoqueAtual - qtd) < 0){

        return alert("Saída deixaria estoque negativo. Ajuste a quantidade ou permita negativo nas configurações.");

      }

      const custoAplicado = Number(p.cmp||0); // CMP vigente

      p.estoqueAtual = Number(p.estoqueAtual||0) - qtd;

      // (FIFO futuro: consumir p.lotes do mais antigo)

      const motivo = mMotivo.value;

      const preco = Number(mPreco.value||0);

      kardex.push({

        id: "mov\_"+cryptoRand(), productId: p.id, tipo:"saida", motivo,

        dataISO: nowISO, quantidade: qtd, custoUnitario: custoAplicado, precoUnitario: preco,

        lote: null, validadeISO: null, fornecedor: null, observacao: mObs.value||""

      });

    }

    if (tipo==="ajuste"){

      if (!qtd || qtd===0) return alert("Informe a quantidade (use + ou −).");

      const novoEst = Number(p.estoqueAtual||0)+qtd;

      if (!config.permitirEstoqueNegativo && novoEst < 0) return alert("Ajuste deixaria estoque negativo.");

      p.estoqueAtual = novoEst;

      // custo unitário do ajuste: CMP vigente (simples)

      kardex.push({

        id: "mov\_"+cryptoRand(), productId: p.id, tipo:"ajuste", motivo:"ajuste",

        dataISO: nowISO, quantidade: qtd, custoUnitario: Number(p.cmp||0), precoUnitario: 0,

        lote: null, validadeISO: null, fornecedor: null, observacao: mJust.value || mObs.value || ""

      });

    }

    write(LS\_PROD, produtos); write(LS\_KDX, kardex);

    render(); close(movModal);

  });

  function cryptoRand(){ return Math.random().toString(16).slice(2,8); }

  // ===== Boot (se vazio, cria 2 exemplos) =====

  if (!produtos.length){

    produtos = [

      { id: genPid(), nome:"Pomada Modeladora 150g", sku:"PM-150", categoria:"Pomadas", unidade:"un",

        precoVenda:39.9, fornecedor:"ACME", estoqueMinimo:5, estoqueAtual:0, politicaCusto:config.politicaCusto, cmp:0, lotes:[], foto:"", obs:"" },

      { id: genPid(), nome:"Shampoo 250ml", sku:"SH-250", categoria:"Shampoos", unidade:"un",

        precoVenda:29.9, fornecedor:"ACME", estoqueMinimo:4, estoqueAtual:0, politicaCusto:config.politicaCusto, cmp:0, lotes:[], foto:"", obs:"" }

    ];

    write(LS\_PROD, produtos);

  }

  render();

})();

document.addEventListener("DOMContentLoaded", () => {

  // refs

  const fab = document.getElementById("btnFabAddService");

  const modal = document.getElementById("svcModal");

  const modalDel = document.getElementById("svcModalDelete");

  const form = document.getElementById("svcForm");

  const inputId = document.getElementById("svcId");

  const delNome = document.getElementById("delNome");

  // util modal

  function openModal(el){ el.classList.add("show"); el.setAttribute("aria-hidden","false"); }

  function closeModal(el){ el.classList.remove("show"); el.setAttribute("aria-hidden","true"); }

  // abrir: novo

  fab?.addEventListener("click", () => {

    form.reset();

    inputId.value = "";

    document.getElementById("svcModalTitle").textContent = "Novo serviço";

    openModal(modal);

  });

  // fechar (qualquer [data-close])

  document.querySelectorAll("[data-close]").forEach(btn => {

    btn.addEventListener("click", () => {

      closeModal(modal);

      closeModal(modalDel);

    });

  });

  // submit salvar (novo/editar)

  form?.addEventListener("submit", (e) => {

    e.preventDefault();

    const data = {

      id: inputId.value || crypto.randomUUID(),

      nome: document.getElementById("mNome").value.trim(),

      duracao: Number(document.getElementById("mDuracao").value || 0),

      preco: Number(document.getElementById("mPreco").value || 0),

      custo: Number(document.getElementById("mCusto").value || 0),

      obs: document.getElementById("mObs").value.trim(),

      // foto: você pode ler com FileReader se quiser salvar base64

    };

    if (!data.nome || data.duracao <= 0 || data.preco <= 0){

      alert("Preencha Nome, Duração e Preço corretamente.");

      return;

    }

    // TODO: persistir (criar/atualizar) e re-renderizar tabela

    // saveService(data);   // implemente na sua lógica

    // renderServices();    // re-render da lista

    closeModal(modal);

  });

  // delegação de eventos na tabela (editar/excluir)

  const tbody = document.getElementById("svcTbody");

  tbody?.addEventListener("click", (e) => {

    const btn = e.target.closest("[data-action]");

    if (!btn) return;

    const id = btn.dataset.id;

    const action = btn.dataset.action;

    if (action === "edit"){

      // TODO: buscar serviço pelo id

      // const svc = getServiceById(id);

      const svc = window.\_\_mockGetService?.(id); // opcional para teste

      if (!svc){ alert("Serviço não encontrado."); return; }

      inputId.value = svc.id;

      document.getElementById("mNome").value = svc.nome || "";

      document.getElementById("mDuracao").value = svc.duracao || 0;

      document.getElementById("mPreco").value = svc.preco || 0;

      document.getElementById("mCusto").value = svc.custo || 0;

      document.getElementById("mObs").value = svc.obs || "";

      document.getElementById("svcModalTitle").textContent = "Editar serviço";

      openModal(modal);

    }

    if (action === "delete"){

      // TODO: buscar serviço para mostrar nome

      // const svc = getServiceById(id);

      const svc = window.\_\_mockGetService?.(id) || { nome: "este serviço" };

      delNome.textContent = svc.nome || "este serviço";

      document.getElementById("btnConfirmDelete").dataset.id = id;

      openModal(modalDel);

    }

  });

  // confirmar exclusão

  document.getElementById("btnConfirmDelete")?.addEventListener("click", (e) => {

    const id = e.currentTarget.dataset.id;

    if (!id) return;

    // TODO: excluir e re-renderizar

    // deleteService(id);

    // renderServices();

    closeModal(modalDel);

  });

});

// services.js – Catálogo de Serviços (Admin)

// v2 no localStorage + migração v1, unitário/combos, export/import

(() => {

  const LS\_V2 = "barbearia\_servicos\_v2";

  const LS\_V1 = "barbearia\_servicos\_v1";

  // helpers

  const $  = sel => document.querySelector(sel);

  const $$ = sel => Array.from(document.querySelectorAll(sel));

  const BRL = v => (isNaN(v)?0:v).toLocaleString("pt-BR",{style:"currency",currency:"BRL"});

  const pct = v => isFinite(v)? `${v.toFixed(1)}%` : "—";

  const uid = (p="id") => `${p}\_${Math.random().toString(36).slice(2,9)}`;

  const num = v => (typeof v==="number"? v : parseFloat(String(v).replace(",", "."))) || 0;

  const toast = msg => alert(msg);

  let editId = null;

  // migração v1 → v2 (se não houver v2)

  (function migrate(){

    if (localStorage.getItem(LS\_V2)) return;

    const v1 = localStorage.getItem(LS\_V1);

    if (!v1) return;

    try{

      const old = JSON.parse(v1);

      const migrated = old.map(s=>({

        id: uid("svc"),

        tipoServico: "unitario",

        nome: s.nome ?? "Serviço",

        duracao: s.duracao ?? null,

        preco: s.preco ?? 0,

        obs: s.obs ?? "",

        foto: s.foto ?? null,

        materiais: s.materiais ?? [],

        custoTotal: s.custoTotal ?? 0,

        lucro: s.lucro ?? 0,

        margem: s.margem ?? 0,

        comboTipo:null, qtdCreditos:null, precoPacote:null, validadeDias:null,

        servicoBaseId:null, servicosElegiveisIds:[], transferivel:false, reembolsavel:false

      }));

      localStorage.setItem(LS\_V2, JSON.stringify(migrated));

      console.log("Migrado v1 → v2:", migrated.length);

    }catch(e){ console.warn("Falha migração v1→v2", e); }

  })();

  const loadAll = () => { try{ return JSON.parse(localStorage.getItem(LS\_V2)||"[]"); }catch{ return []; } };

  const saveAll = list => localStorage.setItem(LS\_V2, JSON.stringify(list));

  // foto preview

  $("#svcFoto")?.addEventListener("change", async ev=>{

    const f = ev.target.files?.[0]; if(!f) return;

    const b64 = await fileToB64(f);

    const pv = $("#svcFotoPreview"); pv.innerHTML = `<img alt="" src="${b64}">`; pv.dataset.b64 = b64;

  });

  function fileToB64(file){ return new Promise((res,rej)=>{ const r=new FileReader(); r.onload=()=>res(r.result); r.onerror=()=>rej(r.error); r.readAsDataURL(file); }); }

  // materiais

  $("#svcAddMat")?.addEventListener("click", ()=> addMatRow());

  function addMatRow(mat){

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

    row.className = "svc-mat-row"; row.dataset.id = mat?.id || uid("mat");

    row.innerHTML = `

      <input class="m-nome" type="text" placeholder="Ex.: Lâmina" value="${mat?.nome||""}">

      <input class="m-custo num" type="number" step="0.01" min="0" value="${mat? num(mat.custo):""}">

      <input class="m-un" type="text" placeholder="un/ml/g" value="${mat?.unidade||""}">

      <input class="m-qt num" type="number" step="0.01" min="0" value="${mat? num(mat.quant):""}">

      <input class="m-sub num" type="number" step="0.01" min="0" value="${mat? num(mat.subtotal):0}" disabled>

      <div class="del" title="Remover">🗑</div>

    `;

    row.addEventListener("input", recomputeUnit);

    row.querySelector(".del").addEventListener("click", ()=>{ row.remove(); recomputeUnit(); });

    $("#svcMatList").appendChild(row);

  }

  // toggle combo

  $("#svcIsCombo")?.addEventListener("change", syncMode);

  $("#svcComboTipo")?.addEventListener("change", syncComboType);

  function syncMode(){

    const isCombo = $("#svcIsCombo").checked;

    $("#svcBlockUnit").hidden = isCombo;

    $("#svcBlockCombo").hidden = !isCombo;

  }

  function syncComboType(){

    const t = $("#svcComboTipo").value;

    $("#svcComboFixoBox").hidden = (t!=="fixo");

    $("#svcComboFlexBox").hidden = (t!=="flexivel");

  }

  // KPIs unitário

  ["#svcPreco","#svcDuracao"].forEach(sel => $(sel)?.addEventListener("input", recomputeUnit));

  function recomputeUnit(){

    let custo = 0;

    $$("#svcMatList .svc-mat-row").forEach(r=>{

      const c = num(r.querySelector(".m-custo").value);

      const q = num(r.querySelector(".m-qt").value);

      const sub = c\*q;

      r.querySelector(".m-sub").value = sub.toFixed(2);

      custo += sub;

    });

    const preco = num($("#svcPreco").value);

    const lucro = preco - custo;

    const margem = preco>0 ? (lucro/preco\*100) : 0;

    $("#svcKpiCusto").textContent  = BRL(custo);

    $("#svcKpiPreco").textContent  = BRL(preco);

    $("#svcKpiLucro").textContent  = BRL(lucro);

    $("#svcKpiMargem").textContent = pct(margem);

  }

  // KPIs combo

  ["#svcQtdCreditos","#svcPrecoPacote","#svcComboTipo","#svcBase","#svcElegiveis"].forEach(sel=>{

    const el=$(sel); if(!el) return; el.addEventListener("input", recomputeCombo);

  });

  function recomputeCombo(){

    const qtd = Math.max(0, num($("#svcQtdCreditos").value));

    const pkg = num($("#svcPrecoPacote").value);

    const porCred = qtd>0? (pkg/qtd) : 0;

    $("#svcKpiPrecoCred").textContent = BRL(porCred);

    const tipo = $("#svcComboTipo").value;

    const all = loadAll();

    if (tipo==="fixo"){

      const base = all.find(s=>s.id===$("#svcBase").value && s.tipoServico==="unitario");

      if (base){

        const m = porCred>0 ? ((porCred-(base.custoTotal||0))/porCred\*100) : 0;

        $("#svcKpiMargemCombo").textContent = `${pct(m)} (base: ${base.nome})`;

      } else {

        $("#svcKpiMargemCombo").textContent = "— selecione o serviço base";

      }

    } else {

      const ids = Array.from($("#svcElegiveis").selectedOptions).map(o=>o.value);

      const eles = all.filter(s=>ids.includes(s.id) && s.tipoServico==="unitario");

      if (eles.length){

        const custos = eles.map(s=>s.custoTotal||0);

        const mMin = porCred>0 ? ((porCred - Math.max(...custos))/porCred\*100) : 0;

        const mMax = porCred>0 ? ((porCred - Math.min(...custos))/porCred\*100) : 0;

        $("#svcKpiMargemCombo").textContent = `${pct(mMin)} a ${pct(mMax)} (elegíveis)`;

      } else {

        $("#svcKpiMargemCombo").textContent = "— selecione serviços elegíveis";

      }

    }

  }

  // selects

  function refreshSelects(){

    const unit = loadAll().filter(s=>s.tipoServico==="unitario");

    $("#svcBase").innerHTML = `<option value="">— selecione —</option>` + unit.map(s=>`<option value="${s.id}">${s.nome}</option>`).join("");

    $("#svcElegiveis").innerHTML = unit.map(s=>`<option value="${s.id}">${s.nome}</option>`).join("");

  }

  // validação

  function validate(){

    const isCombo = $("#svcIsCombo").checked;

    const nome = $("#svcNome").value.trim();

    if (!nome){ toast("Informe o nome do serviço."); return false; }

    if (!isCombo){

      if (num($("#svcDuracao").value) <= 0){ toast("Duração deve ser maior que zero."); return false; }

      if (num($("#svcPreco").value)   <= 0){ toast("Preço deve ser maior que zero."); return false; }

    } else {

      if (num($("#svcQtdCreditos").value) <= 0){ toast("Qtd. de créditos deve ser maior que zero."); return false; }

      if (num($("#svcPrecoPacote").value) <= 0){ toast("Preço do pacote deve ser maior que zero."); return false; }

      if ($("#svcComboTipo").value === "fixo" && !$("#svcBase").value){ toast("Selecione o serviço base."); return false; }

      if ($("#svcComboTipo").value === "flexivel" && Array.from($("#svcElegiveis").selectedOptions).length===0){ toast("Selecione serviços elegíveis."); return false; }

    }

    return true;

  }

  // coletar

  function collect(){

    const isCombo = $("#svcIsCombo").checked;

    const foto = $("#svcFotoPreview").dataset.b64 || null;

    if (!isCombo){

      const mats = $$("#svcMatList .svc-mat-row").map(r=>({

        id:r.dataset.id,

        nome:r.querySelector(".m-nome").value.trim(),

        custo:num(r.querySelector(".m-custo").value),

        unidade:r.querySelector(".m-un").value.trim(),

        quant:num(r.querySelector(".m-qt").value),

        subtotal:num(r.querySelector(".m-sub").value),

      }));

      const custo = mats.reduce((a,m)=>a+num(m.subtotal),0);

      const preco = num($("#svcPreco").value);

      const lucro = preco - custo;

      const margem= preco>0 ? (lucro/preco\*100) : 0;

      return {

        id: editId || uid("svc"),

        tipoServico:"unitario",

        nome: $("#svcNome").value.trim(),

        duracao: num($("#svcDuracao").value),

        preco,

        obs: $("#svcObs").value.trim(),

        foto, materiais:mats,

        custoTotal:custo, lucro, margem,

        comboTipo:null, qtdCreditos:null, precoPacote:null, validadeDias:null,

        servicoBaseId:null, servicosElegiveisIds:[],

        transferivel:false, reembolsavel:false

      };

    } else {

      const tipo = $("#svcComboTipo").value;

      const ids  = Array.from($("#svcElegiveis").selectedOptions).map(o=>o.value);

      return {

        id: editId || uid("svc"),

        tipoServico:"combo",

        nome: $("#svcNome").value.trim(),

        duracao:null, preco:null, obs: $("#svcObs").value.trim(),

        foto, materiais:[], custoTotal:null, lucro:null, margem:null,

        comboTipo:tipo,

        qtdCreditos:num($("#svcQtdCreditos").value),

        precoPacote:num($("#svcPrecoPacote").value),

        validadeDias:num($("#svcValidade").value),

        servicoBaseId:(tipo==="fixo") ? $("#svcBase").value : null,

        servicosElegiveisIds:(tipo==="flexivel") ? ids : [],

        transferivel: $("#svcTransferivel").checked,

        reembolsavel: $("#svcReembolsavel").checked

      };

    }

  }

  // reset

  function resetForm(){

    editId=null; $("#svcEditBadge").hidden=true;

    $("#svcIsCombo").checked=false; syncMode();

    $("#svcNome").value=""; $("#svcObs").value="";

    $("#svcFoto").value=""; $("#svcFotoPreview").dataset.b64=""; $("#svcFotoPreview").textContent="FOTO";

    $("#svcDuracao").value=""; $("#svcPreco").value="";

    $("#svcMatList").innerHTML=""; addMatRow(); recomputeUnit();

    $("#svcComboTipo").value="fixo"; syncComboType();

    $("#svcQtdCreditos").value=""; $("#svcPrecoPacote").value=""; $("#svcValidade").value=0;

    $("#svcBase").value=""; $("#svcElegiveis").selectedIndex=-1;

    $("#svcTransferivel").checked=false; $("#svcReembolsavel").checked=false;

  }

  // preencher para edição

  function fillForm(s){

    editId = s.id; $("#svcEditBadge").hidden=false;

    if (s.tipoServico==="unitario"){

      $("#svcIsCombo").checked=false; syncMode();

      $("#svcNome").value=s.nome||""; $("#svcObs").value=s.obs||"";

      if (s.foto){ $("#svcFotoPreview").innerHTML=`<img alt="" src="${s.foto}">`; $("#svcFotoPreview").dataset.b64=s.foto; }

      else { $("#svcFotoPreview").dataset.b64=""; $("#svcFotoPreview").textContent="FOTO"; }

      $("#svcDuracao").value=s.duracao||""; $("#svcPreco").value=s.preco||"";

      $("#svcMatList").innerHTML=""; (s.materiais||[]).forEach(addMatRow); recomputeUnit();

    } else {

      $("#svcIsCombo").checked=true; syncMode();

      $("#svcNome").value=s.nome||""; $("#svcObs").value=s.obs||"";

      if (s.foto){ $("#svcFotoPreview").innerHTML=`<img alt="" src="${s.foto}">`; $("#svcFotoPreview").dataset.b64=s.foto; }

      else { $("#svcFotoPreview").dataset.b64=""; $("#svcFotoPreview").textContent="FOTO"; }

      $("#svcComboTipo").value=s.comboTipo||"fixo"; syncComboType();

      $("#svcQtdCreditos").value=s.qtdCreditos||""; $("#svcPrecoPacote").value=s.precoPacote||"";

      $("#svcValidade").value= s.validadeDias ?? 0;

      $("#svcBase").value=s.servicoBaseId||"";

      // elegíveis

      const opts=$$("#svcElegiveis option");

      opts.forEach(o=>o.selected=(s.servicosElegiveisIds||[]).includes(o.value));

      $("#svcTransferivel").checked=!!s.transferivel; $("#svcReembolsavel").checked=!!s.reembolsavel;

      recomputeCombo();

    }

    window.scrollTo({top:0,behavior:"smooth"});

  }

  // salvar/cancelar

  $("#svcSalvar")?.addEventListener("click", ()=>{

    if(!validate()) return;

    const novo = collect();

    const all = loadAll();

    if (!editId) all.push(novo);

    else {

      const ix = all.findIndex(x=>x.id===editId);

      if (ix>=0) all[ix]=novo;

    }

    saveAll(all); refreshSelects(); renderTable(); resetForm();

    toast("Serviço salvo.");

  });

  $("#svcCancelar")?.addEventListener("click", resetForm);

  // busca

  $("#svcQ")?.addEventListener("input", renderTable);

  // render tabela

  function renderTable(){

    refreshSelects();

    const q = ($("#svcQ")?.value||"").trim().toLowerCase();

    const all = loadAll();

    const filtered = all.filter(s => {

      if(!q) return true;

      const txt = [

        s.nome||"", s.obs||"", s.tipoServico||"", (s.servicosElegiveisIds||[]).join(",")

      ].join(" ").toLowerCase();

      return txt.includes(q);

    });

    $("#svcTbody").innerHTML = filtered.map(s=>{

      const isU = s.tipoServico==="unitario";

      const tipoTxt = isU ? "Unitário" : (s.comboTipo==="fixo" ? "Combo fixo" : "Combo flexível");

      const precoCred = (!isU && s.qtdCreditos>0) ? (s.precoPacote/s.qtdCreditos) : null;

      const baseOrList = isU ? "—" :

        (s.comboTipo==="fixo"

          ? (loadAll().find(x=>x.id===s.servicoBaseId)?.nome || "—")

          : (s.servicosElegiveisIds||[]).map(id=>loadAll().find(x=>x.id===id)?.nome||"").filter(Boolean).slice(0,3).join(", ") + ((s.servicosElegiveisIds||[]).length>3?"…":""));

      return `

        <tr data-id="${s.id}">

          <td>${s.foto? `<img class="mini" alt="" src="${s.foto}">` : `<div class="mini"></div>`}</td>

          <td>${s.nome||""}</td>

          <td>${tipoTxt}</td>

          <td>${isU? (s.duracao||"—") : "—"}</td>

          <td>${isU? BRL(s.preco||0) : "—"}</td>

          <td>${isU? BRL(s.custoTotal||0) : "—"}</td>

          <td>${isU? BRL((s.lucro??0)) : "—"}</td>

          <td>${isU? pct((s.margem??0)) : "—"}</td>

          <td>${!isU? (s.qtdCreditos||"—") : "—"}</td>

          <td class="hide-sm">${!isU? ((s.validadeDias||0)+" d") : "—"}</td>

          <td class="hide-sm">${!isU? (precoCred? BRL(precoCred):"—") : "—"}</td>

          <td class="hide-sm">${baseOrList||"—"}</td>

          <td class="hide-sm">${s.obs? s.obs:"—"}</td>

          <td class="cell-actions">

            <button class="btn btn-yellow svc-edit">✏️</button>

            <button class="btn btn-red svc-del">🗑</button>

          </td>

        </tr>

      `;

    }).join("");

    // ações

    $$("#svcTbody .svc-edit").forEach(b=>{

      b.addEventListener("click", ()=>{

        const id = b.closest("tr").dataset.id;

        const s = loadAll().find(x=>x.id===id);

        if (s) fillForm(s);

      });

    });

    $$("#svcTbody .svc-del").forEach(b=>{

      b.addEventListener("click", ()=>{

        const id = b.closest("tr").dataset.id;

        if (!confirm("Excluir este serviço?")) return;

        saveAll(loadAll().filter(x=>x.id!==id));

        refreshSelects(); renderTable();

      });

    });

  }

  // export/import/zerar

  $("#svcExport")?.addEventListener("click", ()=>{

    const blob = new Blob([JSON.stringify(loadAll(), null, 2)], {type:"application/json"});

    const a = document.createElement("a");

    a.href = URL.createObjectURL(blob);

    a.download = "servicos\_barbearia\_v2.json";

    a.click(); URL.revokeObjectURL(a.href);

  });

  $("#svcImportFile")?.addEventListener("change", async e=>{

    const f = e.target.files?.[0]; if(!f) return;

    try{

      const txt = await f.text();

      const arr = JSON.parse(txt);

      if (!Array.isArray(arr)) throw new Error("JSON inválido");

      const norm = arr.map(s=>{

        const out = { ...s };

        if (!out.id) out.id = uid("svc");

        if (!out.tipoServico) out.tipoServico = "unitario";

        if (out.tipoServico==="unitario"){

          out.materiais = Array.isArray(out.materiais)? out.materiais.map(m=>({...m, id: m.id||uid("mat")})) : [];

        } else { out.materiais = []; }

        return out;

      });

      saveAll(norm); refreshSelects(); renderTable(); resetForm(); toast("Importação concluída!");

    }catch(err){ console.error(err); toast("Falha ao importar JSON."); }

    finally{ e.target.value=""; }

  });

  $("#svcZerar")?.addEventListener("click", ()=>{

    if (!confirm("Tem certeza? Isso apagará TODO o catálogo.")) return;

    localStorage.removeItem(LS\_V2); refreshSelects(); renderTable(); resetForm();

  });

  // init

  function init(){

    if (!$("#view-servicos")) return; // só roda na view de serviços

    addMatRow(); recomputeUnit(); refreshSelects(); renderTable(); syncMode(); syncComboType();

  }

  document.addEventListener("DOMContentLoaded", init);

})();

// sw.js - Service Worker Básico

const CACHE\_NAME = 'barbearia-funcionario-v1';

// Lista de arquivos essenciais para o "app" funcionar offline

const urlsToCache = [

  '/',

  '/pages/funcionario/agenda.html',

  '/pages/funcionario/caixa.html',

  '/pages/funcionario/perfil.html',

  '/css/admin.css',

  '/js/agenda-funcionario.js',

  '/js/caixa-funcionario.js',

  '/js/perfil-funcionario.js',

  '/assets/icones-logo/sr.carreiro(4).png'

];

// Evento de Instalação: Salva os arquivos no cache

self.addEventListener('install', event => {

  event.waitUntil(

    caches.open(CACHE\_NAME)

      .then(cache => {

        console.log('Cache aberto');

        return cache.addAll(urlsToCache);

      })

  );

});

// Evento de Fetch: Intercepta as requisições

self.addEventListener('fetch', event => {

  event.respondWith(

    caches.match(event.request)

      .then(response => {

        // Se o arquivo estiver no cache, retorna ele.

        if (response) {

          return response;

        }

        // Se não, busca na rede.

        return fetch(event.request);

      }

    )

  );

});

. ARQUIVOS JSON .

{

  "name": "Barbearia Carreiro - Funcionário",

  "short\_name": "BC Funcionário",

  "start\_url": "/pages/funcionario/agenda.html",

  "display": "standalone",

  "background\_color": "#111111",

  "theme\_color": "#e0b25b",

  "orientation": "portrait-primary",

  "icons": [

    {

      "src": "/assets/icones-logo/sr.carreiro(4).png",

      "type": "image/png",

      "sizes": "192x192"

    },

    {

      "src": "/assets/icones-logo/sr.carreiro(4).png",

      "type": "image/png",

      "sizes": "512x512"

    }

  ]

}

. ESTRUTURA DO SITE.

barbearia-carreiro(pasta raiz)

|--------manifest.json

|--------sw.js

Assets (pasta)

|--------funcionários (pasta)

|--------icones-logo(pasta)

|--------sr.carreiro(4).png

|--------produtos(pasta)

CSS (pasta)

|--------admin.css

|--------styles.css

HTML (pasta)

|--------agendar-confirmar.html

|--------agendar-servicos.html

|--------agendar-horario.html

|--------agendar-profissional.html

|--------agendar-servico.html

|--------historico.html

|--------home.html

|--------index.html

|--------admin (pasta)

|--------agenda.html

|--------caixa.html

|--------clientes.html

|--------dashboard.html

|--------funcionarios.html

|--------lancamentos.html

|--------produtos.html

|--------servicos.html

|--------funcionarios (pasta)

|--------agenda.html

|--------caixa.html

|--------perfil.html

JS (pasta)

|--------admin-nav.js

|--------agenda-funcionario.js

|--------perfil-funcionario.js

|--------caixa-funcionario.js

|--------admin.js

|--------agenda.js

|--------agendamento.js

|--------caixa.js

|--------clientes.js

|--------firebase-config.js

|--------funcionarios.js

|--------historico.js

|--------home.js

|--------lancamentos.js

|--------login.js

|--------produtos.js

|--------service.js