ARQUIVOS HTML

ADMIN

```
<!-- admin-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="#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="#9B59B6" href="./funcionarios.html"><span
class="dot"></span>Funcionários</a>
        <a class="nav-item" data-color="#F43F5E" href="./clientes.html"><span
class="dot"></span>Clientes</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>
    </nav>
</aside>
```

```
<!-- AGENDA -->

<!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>

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

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

  <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>

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

  <script defer src="../../js/layout-loader.js"></script>
  <script defer src="../../js/agenda.js"></script>
</head>
<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
```

```
    <div id="sidebar-placeholder"></div>

    <!-- 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>
```

```
<!-- CAIXA -->

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

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

    <script defer src="../../js/layout-loader.js"></script>
    <script defer src="../../js/caixa.js"></script>
</head>
<body class="admin-body">
    <div class="admin-shell">
        <div id="sidebar-placeholder"></div>

        <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>
```

```
<!-- CLIENTES -->

<!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>

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

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

  <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>

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

  <script defer src="../../js/layout-loader.js"></script>
  <script defer src="../../js/clientes.js"></script>
</head>

<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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 -->

<!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>

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

    <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-firestore.js"></script>

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

    <script defer src="../../js/layout-loader.js"></script>
```

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

<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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 -->


<!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>

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

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

  <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>

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

  <script defer src="../../js/layout-loader.js"></script>
  <script defer src="../../js/funcionarios.js"></script>
</head>
<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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>
      </select>
    </div>
  </div>

  <div class="svc-row">
    <div>
      <label for="fPhone">Telefone</label>
      <input id="fPhone" type="text" placeholder="(21) 99999-9999" />
    </div>
  </div>

  <div class="svc-row">
    <div>
        <label for="fLoginId">E-mail de Login *</label>
        <input id="fLoginId" type="email" placeholder="email.login@barbearia.com" required />
    </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>
```

```
<!-- LANCAMENTOS -->

<!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>

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

    <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-firestore.js"></script>

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

    <script defer src="../../js/layout-loader.js"></script>
    <script defer src="../../js/lancamentos.js"></script>
</head>

<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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 -->

<!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>

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

    <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-firestore.js"></script>

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

    <script defer src="../../js/layout-loader.js"></script>
    <script defer src="../../js/produtos.js"></script>
</head>
<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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>
```

```
<!-- SERVICOS -->

<!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>

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

    <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-firestore.js"></script>

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

    <script defer src="../../js/layout-loader.js"></script>
    <script defer src="../../js/service.js"></script>
</head>
<body class="admin-body">
  <div class="admin-shell">
    <!-- SIDEBAR -->
    <div id="sidebar-placeholder"></div>

    <!-- 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>
```

FUNCIONARIO

```
<!-- AGENDA -->

<!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>

    <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>
    <script defer src="../../js/firebase-config.js"></script>
    <script defer src="../../js/agenda-funcionario.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>
            <div id="employeeName" class="employee-badge">Carregando...</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>
</body>
</html>
```

```
<!-- caixa do funcionario -->

<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <!-- Define o tipo de documento HTML5 -->
  <meta charset="UTF-8" />
  <!-- Define que o site será responsivo e se ajustará à tela do dispositivo -->
  <meta name="viewport" content="width=device-width, initial-scale=1" />

  <!-- Título exibido na aba do navegador -->
  <title>Funcionário | Caixa - Barbearia Carreiro</title>

  <!-- Importa o arquivo CSS principal da área administrativa -->
  <link rel="stylesheet" href="../../css/admin.css?v=7" />
</head>

<body class="admin-body">
  <!-- Corpo principal da página administrativa -->
  <div class="admin-shell">

    <!-- MENU LATERAL (Sidebar) -->
    <aside class="admin-sidebar">

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

      <!-- Exibe o nome do funcionário logado (carregado dinamicamente via JS) -->
      <div id="employeeName" class="employee-badge">Carregando...</div>

      <!-- Navegação lateral (links principais do painel do funcionário) -->
      <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>

    <!-- CONTEÚDO PRINCIPAL -->
    <main class="admin-main">

      <!-- Cabeçalho da seção principal -->
      <header class="admin-topbar">
        <h1>Caixa / Registrar Atendimento</h1>
      </header>

      <!-- CONTAINER GERAL DO CAIXA -->
      <div class="caixa-container">

        <!-- SEÇÃO DE INFORMAÇÕES INICIAIS -->
```

```
        <section class="panel caixa-topo">
          <div>
            <!-- Campo de seleção de cliente -->
            <label for="caixaCliente">Cliente*</label>
            <select id="caixaCliente" required>
              <option value="">Selecione...</option>
            </select>
          </div>

          <div>
            <!-- Campo de seleção de profissional (inicialmente desabilitado até carregar via JS)
-->
            <label for="caixaProfissional">Profissional*</label>
            <select id="caixaProfissional" required disabled>
              <option value="">Carregando...</option>
            </select>
          </div>

          <div>
            <!-- Campo para selecionar data e hora do atendimento -->
            <label for="caixaData">Data/Hora</label>
            <input id="caixaData" type="datetime-local" />
          </div>
        </section>

        <!-- SEÇÃO DE SERVIÇOS REALIZADOS -->
        <section class="caixa-section">
          <h3>Serviços Realizados</h3>

          <!-- Lista onde os serviços adicionados serão exibidos dinamicamente -->
          <div id="servicosLista"></div>

          <!-- Botão para adicionar novo serviço -->
          <button id="addServico" class="btn btn-ghost btn-add" style="margin-top: 10px;">+
Adicionar Serviço</button>
        </section>

        <!-- SEÇÃO DE PRODUTOS CONSUMIDOS/VENDIDOS -->
        <section class="caixa-section">
          <h3>Produtos Consumidos/Vendidos</h3>

          <!-- Lista de produtos adicionados -->
          <div id="produtosLista"></div>

          <!-- Botão para adicionar novo produto -->
          <button id="addProduto" class="btn btn-ghost btn-add" style="margin-top: 10px;">+
Adicionar Produto</button>
        </section>

        <!-- SEÇÃO DE RESUMO E PAGAMENTO -->
        <section class="caixa-section">
          <h3>Resumo e Pagamento</h3>

          <div class="resumo-grid">
            <!-- Totais atualizados dinamicamente pelo JS -->
            <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>

            <!-- Seleção da forma de pagamento -->
            <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>

            <!-- Exibe taxa e total final (calculados automaticamente) -->
            <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>

        <!-- SEÇÃO DE BOTÕES DE AÇÃO -->
        <section class="caixa-acoes">
          <!-- Botão para cancelar o atendimento atual -->
          <button id="btnCancelar" class="btn btn-red">Cancelar</button>

          <!-- Botão para finalizar e registrar a venda no sistema -->
          <button id="btnFinalizar" class="btn btn-green">Finalizar Venda</button>
        </section>

      </div>
    </main>
  </div>

  <!-- ============================== -->
  <!-- IMPORTAÇÃO DE SCRIPTS (FIREBASE E LÓGICA DO CAIXA) -->
  <!-- ============================== -->

  <!-- Importa bibliotecas do Firebase (versão 8.10.1) -->
  <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>

  <!-- Arquivo com as configurações de conexão ao Firebase -->
  <script defer src="../../js/firebase-config.js"></script>

  <!-- Lógica principal de cálculo e controle do caixa -->
  <script defer src="../../js/caixa-logic.js"></script>

  <!-- Script com as funções específicas para o funcionário (ex: carregar nome, restrições, etc.)
-->
  <script defer src="../../js/caixa-funcionario.js"></script>

</body>
</html>
```

```
<!-- PERFIL DO 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>

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

            <div id="employeeName" class="employee-badge">Carregando...</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>
```

HTML PASTA RAIZ

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


<!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>
```

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

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

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


<!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>
```

```
<!-- historico.html -->

<!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>
```

```
<!-- HOME -->

<!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>
```

```
<!-- INDEX -->


<!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>

</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 */
```

```css
.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;
```

```css
}

.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;
```

```css
}

.caixa-acoes .btn.rascunho:hover {
  background: #eab308;
}

.caixa-acoes .btn.finalizar {
  background: #22c55e;
  color: #fff;
}

.caixa-acoes .btn.finalizar:hover {
  background: #16a34a;
}

/* Estilos para os status dos slots na agenda do admin */
.slot.disponivel {
    border-color: #2ea66b; /* Verde para disponível */
}
.slot.bloqueado {
    border-color: #333;
    background-color: #111;
    color: #555;
}
.slot.agendado {
    border-color: #3b82f6; /* Azul para agendado */
    background-color: #0f224a;
    color: #dbeafe;
    font-weight: 900;
}

/* Estilos para a lista de agendamentos na agenda do admin */
.ag-right .scheduled-list {
  list-style: none;
  padding: 0;
  margin: 0;
}
.ag-right .scheduled-list li {
  background-color: #0f224a;
  color: #dbeafe;
  padding: 12px;
  border-radius: 10px;
  border-left: 4px solid #3b82f6;
  margin-bottom: 8px;
  font-size: 14px;
}
.ag-right .scheduled-list li strong {
  font-weight: 900;
  margin-right: 8px;
}
.ag-right .scheduled-list li span {
  opacity: 0.8;
}
.ag-right .no-appointments {
  padding: 20px;
  text-align: center;
  font-style: italic;
```

```
}

.scheduled-list li {
    display: flex;
    justify-content: space-between;
    align-items: center;
}
.appointment-details {
    display: flex;
    flex-direction: column;
}
.appointment-details em {
    font-style: normal;
    font-size: 12px;
    opacity: 0.7;
    margin-top: 2px;
}
.appointment-actions .btn-sm {
    padding: 6px 10px;
    font-size: 12px;
}

/* ============================================== */
/* ESTILOS MOVIDOS DOS ARQUIVOS HTML PARA CÁ      */
/* ============================================== */

/* --- Estilos da página: Caixa do Admin (caixa.html) --- */
.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; width: 36px; }
.item-lista-row .del-btn:before { content: 'X'; }
.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; }


/* --- Estilos da página: Agenda do Funcionário (agenda-funcionario.html) --- */
.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; }
```

```
.appointment-actions { display: flex; flex-direction: column; gap: 8px; }


/* --- Estilos da página: Perfil do Funcionário (perfil-funcionario.html) --- */
.day-selector { display: flex; align-items: center; justify-content: center; gap: 10px; }
.day-strip-container { flex-grow: 1; overflow: hidden; }
.day-strip { transition: transform 0.3s ease-in-out; }
.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; }
```

```
/* ============================
   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 ---------- */
```

```css
.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
   ================================================== */
```

```css
/* 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 JS

```
// ../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;
});
```

```
// js/admin.js
```

```
// Lógica do Dashboard, agora integrado com dados reais do Firebase Firestore

document.addEventListener("DOMContentLoaded", () => {
    // Inicializa o serviço do Firestore
    const db = firebase.firestore();

    // --- Helpers ---
    const toBRL = v => (v || 0).toLocaleString("pt-BR", { style: "currency", currency: "BRL" });
    const setText = (id, v) => {
        const el = document.getElementById(id);
        if (el) el.textContent = v;
    };

    // Função principal para carregar e processar os dados
    async function carregarDadosDoDashboard() {
        try {
            // Busca todos os lançamentos ordenados pela data mais recente
            const lancamentosSnap = await db.collection("lancamentos").orderBy("dataISO",
"desc").get();
            const lancamentos = lancamentosSnap.docs.map(doc => doc.data());

            if (lancamentos.length === 0) {
                // Se não houver dados, exibe uma mensagem
                document.getElementById('view-dashboard').innerHTML = '<p style="text-
align:center; padding: 20px;">Nenhum lançamento encontrado para gerar o dashboard.</p>';
                return;
            }

            // Processa os dados para o Dashboard
            processarErenderizarDashboard(lancamentos);

        } catch (error) {
            console.error("Erro ao carregar dados do dashboard:", error);
            alert("Falha ao carregar dados do dashboard.");
        }
    }

    function processarErenderizarDashboard(lancamentos) {
        // --- 1. CÁLCULO DOS KPIs PRINCIPAIS ---

        const receitaTotal = lancamentos.reduce((acc, l) => acc + (l.valores?.receitaBruta || 0),
0);
        // Despesas e Lucro (simplificado por enquanto, pois não temos um módulo de despesas)
        const despesasTotais = lancamentos.reduce((acc, l) => acc + (l.valores?.taxaPagamento ||
0) + (l.valores?.comissao || 0) + (l.valores?.custoTotal || 0), 0);
        const lucroLiquido = receitaTotal - despesasTotais;
        const totalAtendimentos = lancamentos.filter(l => l.servico && l.servico.length >
0).length;
        const ticketMedio = totalAtendimentos > 0 ? receitaTotal / totalAtendimentos : 0;

        // Renderiza os KPIs no HTML
        setText("kpiReceita", toBRL(receitaTotal));
        setText("kpiDespesas", toBRL(despesasTotais));
        setText("kpiLucro", toBRL(lucroLiquido));
        setText("kpiTicket", toBRL(ticketMedio));

        // --- 2. CÁLCULO DOS RANKINGS (TOP SERVIÇOS E PRODUTOS) ---
```

```
        const servicosContador = {};
        const produtosContador = {};

        lancamentos.forEach(l => {
            // Conta os serviços
            if (l.servico) {
                l.servico.forEach(s => {
                    if (!servicosContador[s.nome]) servicosContador[s.nome] = { nome: s.nome, qtd:
0, receita: 0 };
                    servicosContador[s.nome].qtd++;
                    servicosContador[s.nome].receita += s.precoFinal || 0;
                });
            }
            // Conta os produtos
            if (l.produtos) {
                l.produtos.forEach(p => {
                    if (!p.brinde) { // Apenas produtos pagos contam para o ranking de receita
                        if (!produtosContador[p.nome]) produtosContador[p.nome] = { nome: p.nome,
qtd: 0, receita: 0 };
                        produtosContador[p.nome].qtd += p.quantidade || 0;
                        produtosContador[p.nome].receita += (p.precoUnitarioAplicado || 0) *
(p.quantidade || 0);
                    }
                });
            }
        });

        // Converte os contadores em arrays e ordena por receita
        const topServicos = Object.values(servicosContador).sort((a, b) => b.receita - a.receita);
        const topProdutos = Object.values(produtosContador).sort((a, b) => b.receita - a.receita);

        // Renderiza os rankings
        renderRank("rankServicos", topServicos);
        renderRank("rankProdutos", topProdutos);

        // --- 3. TABELA DE ENTRADAS RECENTES ---
        const entradasRecentes = lancamentos.slice(0, 5); // Pega os 5 mais recentes
        renderTabelaEntradas("tabEntradas", entradasRecentes);
    }

    // Funções auxiliares de renderização
    function renderRank(ulId, list) {
        const ul = document.getElementById(ulId);
        if (!ul) return;
        ul.innerHTML = list.slice(0, 5).map(item => // Mostra o Top 5
            `<li><span>${item.nome} (${item.qtd})</span><span>${toBRL(item.receita)}</span></li>`
        ).join("");
    }

    function renderTabelaEntradas(tableId, lancamentos) {
        const tbody = document.querySelector(`#${tableId} tbody`);
        if (!tbody) return;
        tbody.innerHTML = lancamentos.map(l => {
            const desc = (l.servico && l.servico[0]?.nome) || (l.produtos && l.produtos[0]?.nome)
|| 'Venda';
            return `
```

```
                <tr>
                    <td>${new Date(l.dataISO).toLocaleDateString('pt-BR')}</td>
                    <td>${l.tipo || 'Venda'}</td>
                    <td>${desc}</td>
                    <td>${l.pagamento?.forma || '-'}</td>
                    <td>${toBRL(l.valores?.resultadoLiquido || 0)}</td>
                </tr>
            `;
        }).join("");
    }

    // --- INICIALIZAÇÃO ---
    carregarDadosDoDashboard();
});
```

```
// js/agenda-funcionario.js

// VERSÃO CORRIGIDA: Garante que a UI só é atualizada após o sucesso da operação no Firestore.
document.addEventListener('DOMContentLoaded', () => {
    const db = firebase.firestore();
    const auth = firebase.auth();
    const listaAgendaEl = document.getElementById('lista-agenda');
    const headerTitleEl = document.getElementById('header-title');
    const employeeNameBadge = document.getElementById('employeeName');
    let currentUserId = null; // Variável para guardar o ID do usuário logado

    auth.onAuthStateChanged(async user => {
        if (user) {
            currentUserId = user.uid; // Armazena o ID do usuário
            try {
                // Busca o nome do perfil para exibir na tela
                const userProfile = await db.collection("funcionarios").doc(user.uid).get();
                const userName = userProfile.exists && userProfile.data().nome ?
userProfile.data().nome : user.email;
                if (employeeNameBadge) employeeNameBadge.textContent = userName;

                // Carrega a agenda do usuário
                await carregarAgenda(user.uid);
            } catch (error) {
                console.error("Erro ao inicializar a página da agenda:", error);
                if (listaAgendaEl) listaAgendaEl.innerHTML = `<p>Ocorreu um erro ao carregar seus
dados.</p>`;
            }
        } else {
            window.location.href = "../../index.html";
        }
    });

    async function carregarAgenda(uid) {
        const hoje = new Date();
        if (headerTitleEl) headerTitleEl.textContent = 'Sua Agenda de Hoje';
        const inicioDoDia = new Date(hoje.setHours(0, 0, 0, 0)).toISOString();
        const fimDoDia = new Date(hoje.setHours(23, 59, 59, 999)).toISOString();
```

```
        try {
            const agendSnapshot = await db.collection("agendamentos")
                .where('profissionalId', '==', uid)
                .where('inicioISO', '>=', inicioDoDia)
                .where('inicioISO', '<=', fimDoDia)
                .orderBy('inicioISO')
                .get();

            if (agendSnapshot.empty) {
                listaAgendaEl.innerHTML = '<p class="muted" style="text-align: center; padding:
20px;">Nenhum agendamento para hoje.</p>';
                return;
            }

            const agendamentosDoDia = agendSnapshot.docs.map(doc => ({ id: doc.id, ...doc.data()
}));
            renderLista(agendamentosDoDia);

        } catch (error) {
            console.error("Erro ao carregar agenda:", error);
            if (error.code === 'failed-precondition') {
                listaAgendaEl.innerHTML = '<p style="color:red; text-align:center;">Erro: É
necessário criar um índice no Firestore. Abra o console do admin (F12) para encontrar o
link.</p>';
            } else {
                listaAgendaEl.innerHTML = '<p>Erro ao carregar a agenda.</p>';
            }
        }
    }

    function renderLista(agendamentos) {
        listaAgendaEl.innerHTML = agendamentos.map(ag => {
            const hora = new Date(ag.inicioISO).toLocaleTimeString('pt-BR', { hour: '2-digit',
minute: '2-digit' });
            const status = ag.status || 'Pendente';
            // Os botões só aparecem se o status ainda for 'Pendente'
            const acoesVisiveis = status === 'Pendente';

            return `
                <div class="agenda-item" data-id="${ag.id}">
                    <div class="agenda-details">
                        <h4>${hora} - ${ag.clienteNome}</h4>
                        <p>Serviço: ${ag.servicoNome}</p>
                        <p>Status: <strong class="appointment-status">${status}</strong></p>
                    </div>
                    <div class="appointment-actions">
                        ${acoesVisiveis ? `
                            <button class="btn btn-green btn-sm" data-action="marcar-compareceu"
title="Marcar Compareceu">✓ Compareceu</button>
                            <button class="btn btn-red btn-sm" data-action="marcar-nao-compareceu"
title="Marcar Falta">X Faltou</button>
                        ` : `<strong>Status Definido!</strong>`
                        }
                    </div>
                </div>`;
        }).join('');
    }
```

```
    listaAgendaEl.addEventListener('click', (e) => {
        const btn = e.target.closest('[data-action]');
        if (!btn) return;

        const agendamentoId = btn.closest('.agenda-item').dataset.id;
        const action = btn.dataset.action;

        if (action === "marcar-compareceu") {
            atualizarStatus(agendamentoId, "Compareceu");
        } else if (action === "marcar-nao-compareceu") {
            atualizarStatus(agendamentoId, "Não Compareceu");
        }
    });

    async function atualizarStatus(agendamentoId, novoStatus) {
        try {
            // Primeiro, tenta atualizar o banco de dados
            await db.collection("agendamentos").doc(agendamentoId).update({ status: novoStatus });

            // --- ALTERAÇÃO: Se a atualização der certo, recarrega a lista inteira ---
            // Isso garante que a UI sempre mostrará a informação real do banco de dados.
            alert(`Status do agendamento atualizado para "${novoStatus}".`);
            await carregarAgenda(currentUserId);

        } catch (error) {
            console.error("Erro ao atualizar status:", error);
            alert("Não foi possível atualizar o status. Tente novamente.");
        }
    }
});
```

```
// js/agenda.js
// VERSÃO FINAL: Exibe a lista de agendamentos e ativa a funcionalidade de edição de horários via
modal.

(function() {
    // Serviços do Firebase
    const db = firebase.firestore();
    const funcionariosCollection = db.collection("funcionarios");
    const agendamentosCollection = db.collection("agendamentos");
    const configCollection = db.collection("agenda_config");

    // --- Helpers de data ---
    const pad2 = n => String(n).padStart(2, '0');
    const toYMD = d => `${d.getFullYear()}-${pad2(d.getMonth() + 1)}-${pad2(d.getDate())}`;
    const toLocalDateInputValue = d => `${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; };
// Seg=1, Dom=7
    const DOW_PT = ['Dom', 'Seg', 'Ter', 'Qua', 'Qui', 'Sex', 'Sáb'];
```

```
// --- Referências do DOM ---
const agendaData = document.getElementById('agendaData');
const filtroProf = document.getElementById('filtroProf');
const grid = document.getElementById('agendaGrid');
const btnHoje = document.getElementById('btnHoje');
const btnPrev = document.getElementById('btnPrev');
const btnNext = document.getElementById('btnNext');

// <<-- ADICIONADO: Referências do Modal -->>
const modal = document.getElementById('modalConfig');
const configTitle = document.getElementById('configTitle');
const dayStrip = document.getElementById('dayStrip');
const weekMatrix = document.getElementById('weekMatrix');
const dayMonth = document.getElementById('dayMonth');
const saveConfigBtn = document.getElementById('saveConfig');

// --- Estado da Aplicação ---
let EMPs = [], AGENDs = [], CFG = {};
let currentDate = new Date();
// <<-- ADICIONADO: Estado do Modal -->>
let selectedEmp = null, workingCopy = null, editingDateYMD = null;

const TIMES = Array.from({ length: (20 - 10) * 2 }, (_, i) => {
    const hour = 10 + Math.floor(i / 2);
    const minute = (i % 2) * 30;
    return `${pad2(hour)}:${pad2(minute)}`;
});

// --- FUNÇÕES DE INICIALIZAÇÃO E CARREGAMENTO ---

async function init() {
    setupEventListeners();
    await carregarFuncionarios();
    await carregarDadosDaAgenda();
}

function setupEventListeners() {
    agendaData.addEventListener('change', () => {
        const [year, month, day] = agendaData.value.split('-').map(Number);
        currentDate = new Date(year, month - 1, day);
        carregarDadosDaAgenda();
    });
    filtroProf.addEventListener('change', renderGrid);
    btnHoje.addEventListener('click', () => {
        currentDate = new Date();
        carregarDadosDaAgenda();
    });
    btnPrev.addEventListener('click', () => {
        currentDate.setDate(currentDate.getDate() - 1);
        carregarDadosDaAgenda();
    });
    btnNext.addEventListener('click', () => {
        currentDate.setDate(currentDate.getDate() + 1);
        carregarDadosDaAgenda();
    });

    // <<-- ADICIONADO: Listeners (Ouvintes) de Eventos do Modal -->>
```

```
        document.getElementById('closeConfig')?.addEventListener('click', closeModal);
        document.getElementById('cancelConfig')?.addEventListener('click', closeModal);
        saveConfigBtn?.addEventListener('click', salvarConfiguracao);
        grid.addEventListener('click', (e) => {
            const gearButton = e.target.closest('.btn-gear');
            if (gearButton) {
                const empId = gearButton.dataset.empId;
                const emp = EMPs.find(e => e.id === empId);
                if (emp) openConfig(emp);
            }
        });
    }

    async function carregarFuncionarios() {
        try {
            const snapshot = await funcionariosCollection.orderBy("nome").get();
            EMPs = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            filtroProf.innerHTML = '<option value="*">Todos</option>';
            EMPs.forEach(e => {
                filtroProf.innerHTML += `<option value="${e.id}">${e.nome}</option>`;
            });
        } catch (error) {
            console.error("Erro ao carregar funcionários:", error);
        }
    }

    async function carregarDadosDaAgenda() {
        agendaData.value = toLocalDateInputValue(currentDate);
        try {
            const configSnapshot = await configCollection.get();
            CFG = {};
            configSnapshot.forEach(doc => { CFG[doc.id] = doc.data(); });

            const inicioDoDia = new Date(currentDate.getFullYear(), currentDate.getMonth(),
currentDate.getDate()).toISOString();
            const fimDoDia = new Date(currentDate.getFullYear(), currentDate.getMonth(),
currentDate.getDate(), 23, 59, 59, 999).toISOString();

            const agendSnapshot = await agendamentosCollection
                .where('inicioISO', '>=', inicioDoDia)
                .where('inicioISO', '<=', fimDoDia).get();

            AGENDs = agendSnapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            renderGrid();
        } catch (error) {
            console.error("Erro ao carregar dados da agenda:", error);
            grid.innerHTML = `<p style="padding:20px; text-align:center;">Erro ao carregar dados.
Verifique o console (F12) para criar um índice no Firestore.</p>`;
        }
    }

    // --- RENDERIZAÇÃO DA GRADE PRINCIPAL ---

    function renderGrid() {
        if (!grid) return;
        grid.innerHTML = '';
        const selectedProfId = filtroProf.value;
```

```
        const funcionariosParaRenderizar = EMPs.filter(e => selectedProfId === '*' || e.id ===
selectedProfId);

        if (funcionariosParaRenderizar.length === 0) {
            grid.innerHTML = "<p style='text-align:center; padding: 20px;'>Nenhum funcionário
encontrado.</p>";
            return;
        }

        funcionariosParaRenderizar.forEach(emp => {
            const row = document.createElement('div');
            row.className = 'ag-row';
            row.innerHTML = `
                <div class="ag-left">
                    <div class="ag-avatar">${emp.nome ? emp.nome.charAt(0) : '?'}</div>
                    <div class="ag-name">
                        ${emp.nome}
                        <button class="btn-gear" data-emp-id="${emp.id}" title="Configurar
agenda">⚙</button>
                    </div>
                </div>`;

            const rightDiv = document.createElement('div');
            rightDiv.className = 'ag-right';
            const agendamentosDoFuncionario = AGENDs.filter(a => a.profissionalId ===
emp.id).sort((a,b) => new Date(a.inicioISO) - new Date(b.inicioISO));

            if (agendamentosDoFuncionario.length > 0) {
                const list = document.createElement('ul');
                list.className = 'scheduled-list';
                agendamentosDoFuncionario.forEach(ag => {
                    const hora = new Date(ag.inicioISO).toLocaleTimeString('pt-BR', { hour: '2-
digit', minute: '2-digit' });
                    list.innerHTML += `<li><strong>${hora}</strong> - ${ag.clienteNome}
<span>(${ag.servicoNome})</span></li>`;
                });
                rightDiv.appendChild(list);
            } else {
                rightDiv.innerHTML = `<p class="muted no-appointments">Nenhum horário agendado
para hoje.</p>`;
            }
            row.appendChild(rightDiv);
            grid.appendChild(row);
        });
    }

    // --- ADICIONADO: LÓGICA DO MODAL DE CONFIGURAÇÃO ---

    function openConfig(emp) {
        selectedEmp = emp;
        if (configTitle) configTitle.textContent = `Configurar agenda de ${emp.nome}`;
        workingCopy = JSON.parse(JSON.stringify(CFG[emp.id] || { _overrides: {} }));
        buildDayStripInModal();
        selectDayForEdit(toYMD(currentDate));
        if (modal) modal.classList.add('show');
    }
```

```
    function closeModal() {
        if (modal) modal.classList.remove('show');
        selectedEmp = null;
        workingCopy = null;
    }

    async function salvarConfiguracao() {
        if (!selectedEmp || !workingCopy) return alert("Nenhum funcionário selecionado.");
        try {
            await configCollection.doc(selectedEmp.id).set(workingCopy);
            alert(`Agenda de ${selectedEmp.nome} salva com sucesso!`);
            closeModal();
            await carregarDadosDaAgenda();
        } catch (error) {
            console.error("Erro ao salvar configuração:", error);
            alert("Falha ao salvar a configuração.");
        }
    }

    function buildDayStripInModal() {
        if (!dayStrip) return;
        dayStrip.innerHTML = '';
        const base = new Date();
        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.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));
            dayStrip.appendChild(pill);
        }
    }

    function selectDayForEdit(ymd) {
        editingDateYMD = ymd;
        document.querySelectorAll('#modalConfig .day-pill').forEach(p =>
p.classList.toggle('active', p.dataset.ymd === ymd));
        const d = new Date(`${ymd}T12:00:00`);
        if (dayMonth) dayMonth.textContent = d.toLocaleString('pt-BR', { month: 'long', year:
'numeric' });
        buildDayTableForEdit();
    }

    function buildDayTableForEdit() {
        if (!weekMatrix) return;
        const d = new Date(`${editingDateYMD}T12:00:00`);
        const dow = dow1to7(d);
        const baseDayConfig = workingCopy[dow] || {};
        const overrideConfig = workingCopy._overrides?.[editingDateYMD] || {};
        let tableHTML = `<table class="week-table"><thead><tr><th class="week-
time"></th><th>${d.toLocaleDateString('pt-BR')}</th></tr></thead><tbody>`;
        for (const time of TIMES) {
```

```
            const isAvailable = overrideConfig.hasOwnProperty(time) ? overrideConfig[time] :
(baseDayConfig[time] === true);
            tableHTML += `<tr><th class="week-time">${time}</th><td class="cell ${isAvailable ?
'active' : ''}" data-time="${time}">${isAvailable ? 'Disponível' : 'Bloqueado'}</td></tr>`;
        }
        tableHTML += `</tbody></table>`;
        weekMatrix.innerHTML = tableHTML;
        weekMatrix.querySelectorAll('.cell').forEach(cell => cell.addEventListener('click',
toggleCellAvailability));
    }

    function toggleCellAvailability(e) {
        const td = e.target.closest('.cell');
        if (!td) return;
        const time = td.dataset.time;
        if (!workingCopy._overrides) workingCopy._overrides = {};
        if (!workingCopy._overrides[editingDateYMD]) workingCopy._overrides[editingDateYMD] = {};
        const newStatus = !td.classList.contains('active');
        workingCopy._overrides[editingDateYMD][time] = newStatus;
        td.classList.toggle('active', newStatus);
        td.textContent = newStatus ? 'Disponível' : 'Bloqueado';
    }

    // --- INICIALIZAÇÃO ---
    document.addEventListener('DOMContentLoaded', init);
})();
```

```
// js/agendamento.js

// Estado compartilhado via sessionStorage + utilitários das 4 etapas
const BOOKING_KEY = "booking";

// --- ALTERAÇÃO INICIADA: Remoção dos dados estáticos ---
// As listas de Serviços e Profissionais agora serão carregadas do Firebase.
// Deixei as variáveis aqui para que o resto do código continue funcionando
// após elas serem preenchidas com os dados do banco.
let Services = [];
let Pros = [];
// --- ALTERAÇÃO FINALIZADA ---

// --- NOVO: Adicionada referência ao Firestore para buscar os dados ---
const db = firebase.firestore();

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$ " + Number(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}/${yyyy}`;
}
function weekdayPT(d) {
    return ["Dom", "Seg", "Ter", "Qua", "Qui", "Sex", "Sáb"][d.getDay()];
}
function monthLabel(d) {
    const m1 = d.toLocaleString("pt-BR", { month: "long" });
    const next = new Date(d); next.setDate(d.getDate() + 4);
    const m2 = next.toLocaleString("pt-BR", { month: "long" });
    const y = next.getFullYear();
    return (m1 === m2 ? m1 : `${m1}/${m2}`) + " " + y;
}

/* ETAPA 1: Serviços ====== */
// --- ALTERAÇÃO: A função agora é 'async' para poder esperar os dados do Firebase ---
async function initStep1() {
    const ul = document.getElementById("serviceList");
    ul.innerHTML = '<li>Carregando serviços...</li>'; // Mensagem de loading

    try {
        // --- NOVO: Busca os serviços da coleção 'servicos' no Firestore ---
        const snapshot = await db.collection("servicos").orderBy("nome").get();
        Services = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));

        // monta lista
        ul.innerHTML = Services.map(s => `
            <li class="select-item">
                <div class="thumb">${s.foto ? `<img src="${s.foto}" alt="" />` : 'FOTO'}</div>
                <div class="item-main">
                    <div class="title">${s.nome} ${s.obs ? `<span class="meta">- ${s.obs}</span>`
: ''}</div>
                    <div class="price">${fmtMoney(s.preco)}</div>
                    <div class="meta">${s.duracao} min</div>
                </div>
                <label>
                    <input type="radio" name="service" class="pick" value="${s.id}">
                    <span class="check"></span>
                </label>
            </li>
        `).join("");

    } catch (error) {
        console.error("Erro ao carregar serviços:", error);
        ul.innerHTML = '<li>Ocorreu um erro ao carregar os serviços. Tente novamente.</li>';
    }


    // seleção única
    ul.addEventListener("change", (e) => {
        if (e.target && e.target.matches("input.pick")) {
            // --- ALTERAÇÃO: 'title' para 'nome', 'price' para 'preco', 'mins' para 'duracao'
para ser consistente com o Firestore ---
            const s = Services.find(x => x.id === e.target.value);
            saveBooking({ serviceId: s.id, serviceTitle: s.nome, servicePrice: s.preco,
serviceMins: s.duracao });
        }
```

```
    });

    // 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 ====== */
// --- ALTERAÇÃO: A função agora é 'async' para poder esperar os dados do Firebase ---
async function initStep2() {
    const ul = document.getElementById("proList");
    ul.innerHTML = '<li>Carregando profissionais...</li>'; // Mensagem de loading

    try {
        // --- NOVO: Busca os funcionários da coleção 'funcionarios' no Firestore ---
        const snapshot = await db.collection("funcionarios").orderBy("nome").get();
        Pros = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));

        ul.innerHTML = Pros.map(p => `
            <li class="select-item">
                <div class="avatar">${p.foto ? `<img src="${p.foto}" alt="" />` : 'FOTO'}</div>
                <div class="item-main"><div class="title">${p.nome}</div></div>
                <label>
                    <input type="radio" name="pro" class="pick" value="${p.id}">
                    <span class="check"></span>
                </label>
            </li>
        `).join("");

    } catch (error) {
        console.error("Erro ao carregar profissionais:", error);
        ul.innerHTML = '<li>Ocorreu um erro ao carregar os profissionais. Tente novamente.</li>';
    }


    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.nome });
        }
    });

    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() {
    // Esta função permanece a mesma, pois a lógica de horários é mais complexa e
    // já estava preparada para ser dinâmica (ela será conectada à agenda_config no futuro)
    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);
        morning[2].disabled = true; // Exemplo de desabilitado
        afternoon[1].disabled = true; // Exemplo de desabilitado

        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.parentElement.querySelectorAll('.slot')].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";

        const close = () => {
            modal.style.display = "none";
            clearBooking();
            window.location.href = "home.html";
```

```
        };
        modal.querySelector(".close").onclick = close;
    });
}

/* Expose */
window.BookingSteps = { initStep1, initStep2, initStep3, initStep4 };
```

```
// js/caixa-funcionario.js

// VERSÃO REFATORADA: Este arquivo agora é responsável apenas por carregar os dados
// específicos do funcionário e iniciar a lógica de caixa compartilhada.
document.addEventListener('DOMContentLoaded', () => {
    const db = firebase.firestore();
    const auth = firebase.auth();
    const employeeNameBadge = document.getElementById('employeeName');
    const selProfissional = document.getElementById("caixaProfissional");

    auth.onAuthStateChanged(user => {
        if (user) {
            carregarDadosIniciais(user);
        } else {
            window.location.href = '../../index.html';
        }
    });

    async function carregarDadosIniciais(user) {
        try {
            // Carrega os dados de apoio (clientes, serviços, produtos)
            const [clientesSnap, servsSnap, prodsSnap, funcDoc] = await Promise.all([
                db.collection("clientes").orderBy("nome").get(),
                db.collection("servicos").where("tipoServico", "==",
"unitario").orderBy("nome").get(),
                db.collection("produtos").orderBy("nome").get(),
                db.collection("funcionarios").doc(user.uid).get()
            ]);

            const clientes = clientesSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            const servicos = servsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            const produtos = prodsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));

            if (!funcDoc.exists) throw new Error("Perfil de funcionário não encontrado.");
            const funcionarioLogado = { id: user.uid, ...funcDoc.data() };

            if (employeeNameBadge) employeeNameBadge.textContent = funcionarioLogado.nome;

            // Preenche e desabilita o seletor de profissional
            selProfissional.innerHTML = `<option
value="${funcionarioLogado.id}">${funcionarioLogado.nome}</option>`;
            selProfissional.disabled = true;

            // --- NOVO: Inicializa a lógica de caixa compartilhada ---
            // O objeto 'CaixaLogic' virá do novo arquivo 'caixa-logic.js'
            // Passamos para ele os dados carregados e o funcionário logado
```

```
            CaixaLogic.init({
                clientes: clientes,
                servicos: servicos,
                produtos: produtos,
                funcionarios: [funcionarioLogado], // O caixa só precisa conhecer o funcionário
logado
                funcionarioFixo: funcionarioLogado // Passa o funcionário para já deixá-lo
selecionado
            });

        } catch (error) {
            console.error("Erro ao carregar dados do caixa:", error);
            alert("Não foi possível carregar dados para o Caixa. Verifique o console (F12).");
        }
    }
});
```

```
// js/caixa.js

// VERSÃO REFATORADA: Este arquivo agora é responsável apenas por carregar os dados
// para o painel de Admin e iniciar a lógica de caixa compartilhada.
(function() {
    const db = firebase.firestore();

    async function carregarDadosIniciais() {
        try {
            // Carrega todos os dados necessários para o Admin
            const [clientesSnap, funcsSnap, servsSnap, prodsSnap] = await Promise.all([
                db.collection("clientes").orderBy("nome").get(),
                db.collection("funcionarios").orderBy("nome").get(),
                db.collection("servicos").where("tipoServico", "==",
"unitario").orderBy("nome").get(),
                db.collection("produtos").orderBy("nome").get()
            ]);

            const clientes = clientesSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            const funcionarios = funcsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            const servicos = servsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            const produtos = prodsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));

            // --- NOVO: Inicializa a lógica de caixa compartilhada ---
            // Passamos todos os dados para que o Admin possa selecionar qualquer profissional
            CaixaLogic.init({
                clientes: clientes,
                servicos: servicos,
                produtos: produtos,
                funcionarios: funcionarios
            });

        } catch (error) {
            console.error("Erro fatal ao carregar dados iniciais:", error);
            alert("Não foi possível carregar os dados. Verifique o console (F12). Pode ser
necessário criar um indice no Firestore.");
```

```
        }
    }

    // Inicia o carregamento dos dados quando o DOM estiver pronto
    document.addEventListener("DOMContentLoaded", carregarDadosIniciais);

})();
```

```
// js/clientes.js
// Módulo de clientes do Admin, AGORA integrado com o Firebase Firestore

(function() {
    // Inicializa os serviços do Firebase que vamos usar na página
    const db = firebase.firestore();
    const auth = firebase.auth(); // Adicionado para a re-autenticação

    // Coleções que vamos usar
    const clientesCollection = db.collection("clientes");
    const funcionariosCollection = db.collection("funcionarios");

    // Estado da aplicação (dados que vêm do Firebase)
    let state = [];
    let query = ""; // Para a barra de busca

    // Referências do DOM
    const tbody = document.getElementById("cliTbody");

    // --- LÓGICA DE DADOS COM FIRESTORE ---

    // Função principal para carregar os clientes do Firestore e renderizar a tabela
    async function carregarErenderizarClientes() {
        try {
            const snapshot = await clientesCollection.get();
            state = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            render();
        } catch (error) {
            console.error("Erro ao carregar clientes do Firestore:", error);
            tbody.innerHTML = `<tr><td colspan="7">Erro ao carregar dados. Verifique o console
(F12).</td></tr>`;
        }
    }

    // Função que "promove" um cliente a funcionário, com verificação de senha
    async function promoverParaFuncionario(clienteId) {
        const adminUser = auth.currentUser;
        if (!adminUser) {
            return alert("Erro: Admin não está logado. Faça login novamente.");
        }

        const senha = prompt("Para confirmar esta operação, por favor, digite sua senha de
administrador:");
        if (!senha) {
            return alert("Operação cancelada.");
```

```
        }

        try {
            const credencial = firebase.auth.EmailAuthProvider.credential(adminUser.email, senha);
            await adminUser.reauthenticateWithCredential(credencial);

            if (!confirm("Senha correta! Tem certeza que deseja promover este cliente a
funcionário? O perfil de cliente dele será removido.")) {
                return;
            }

            const cliente = state.find(c => c.id === clienteId);
            if (!cliente) throw new Error("Cliente não encontrado.");

            const novoFuncionario = {
                nome: cliente.nome,
                email: cliente.email,
                phone: cliente.telefone || null,
                role: "Barbeiro", // Cargo padrão
                accessType: "Funcionario"
            };

            await funcionariosCollection.doc(cliente.id).set(novoFuncionario);
            await clientesCollection.doc(cliente.id).delete();

            alert(`${cliente.nome} foi promovido a funcionário com sucesso!`);
            carregarErenderizarClientes(); // Atualiza a lista de clientes

        } catch (error) {
            console.error("Erro na operação de promoção:", error);
            if (error.code === 'auth/wrong-password') {
                alert("Senha incorreta. Operação cancelada.");
            } else {
                alert(`Ocorreu um erro: ${error.message}`);
            }
        }
    }

    // --- FUNÇÕES DE RENDERIZAÇÃO E UI ---

    // Função que desenha a tabela de clientes
    function render() {
        if (!tbody) return;

        const term = query.trim().toLowerCase();
        const filtered = state.filter(cliente => (cliente.nome ||
"").toLowerCase().includes(term));

        if (filtered.length === 0) {
            tbody.innerHTML = '<tr><td colspan="7" style="text-align:center; padding:
20px;">Nenhum cliente encontrado no banco de dados.</td></tr>';
            return;
        }

        tbody.innerHTML = filtered.map(cliente => {
            const dataNasc = cliente.nascimento ? new
Date(cliente.nascimento).toLocaleDateString('pt-BR') : "—";
```

```
            return `
                <tr data-id="${cliente.id}">
                    <td>${cliente.foto ? `<img src="${cliente.foto}" alt="${cliente.nome}"
style="width:40px;height:40px;border-radius:50%;">` : 'Sem Foto'}</td>
                    <td>${cliente.nome || "—"}</td>
                    <td>${cliente.telefone || "—"}</td>
                    <td class="hide-sm">${cliente.email || "—"}</td>
                    <td class="hide-sm">${dataNasc}</td>
                    <td>—</td> <td>
                        <div class="cell-actions">
                            <button class="btn btn-green" data-action="promote" title="Promover a
Funcionário">Promover</button>
                            <button class="btn btn-yellow" data-action="edit">Editar</button>
                            <button class="btn btn-red" data-action="delete">Excluir</button>
                        </div>
                    </td>
                </tr>
            `;
        }).join("");
    }

    // Delegação de eventos na tabela para os botões de ação
    tbody?.addEventListener("click", (e) => {
        const btn = e.target.closest("[data-action]");
        if (!btn) return;

        const id = btn.closest("tr").dataset.id;
        const action = btn.dataset.action;

        if (action === "promote") {
            promoverParaFuncionario(id);
        } else if (action === "edit") {
            alert(`Funcionalidade "Editar Cliente" (ID: ${id}) a ser implementada.`);
        } else if (action === "delete") {
            alert(`Funcionalidade "Excluir Cliente" (ID: ${id}) a ser implementada.`);
        }
    });

    // Lógica da barra de busca
    document.getElementById("cliQ")?.addEventListener("input", (e) => {
        query = e.target.value || "";
        render();
    });

    // --- INICIALIZAÇÃO ---
    // A primeira coisa que a página faz é carregar os clientes do Firebase
    carregarErenderizarClientes();
})();
```

```
// js/firebase-config.js
// Este arquivo contém as "chaves" de acesso para o seu projeto Firebase e inicializa a conexão.

// 1. Objeto de configuração (copiado do painel do Firebase)
const firebaseConfig = {
```

```
  apiKey: "AIzaSyDnonB0uJT7281aF1Gz4tK2pfmmF_Hnz5M",
  authDomain: "barbearia-srcarreiro.firebaseapp.com",
  projectId: "barbearia-srcarreiro",
  storageBucket: "barbearia-srcarreiro.appspot.com",
  messagingSenderId: "572916176261",
  appId: "1:572916176261:web:1897acc19ad63ed80860ec"
};

// 2. Inicializa o Firebase (este é o comando para a versão 8 do SDK)
firebase.initializeApp(firebaseConfig);
```

```
// js/funcionarios.js
// Módulo de funcionários com cadastro AUTOMATIZADO de usuários (Firebase Auth + Firestore) -
VERSÃO COM TABELA CORRIGIDA

(function() {
    // Inicializa o serviço do Firestore
    const db = firebase.firestore();

    // Estado da aplicação
    let state = [];
    let query = "";
    let editingId = null;

    // --- Referências do DOM ---
    const tbody = document.getElementById("empTbody");
    const modal = document.getElementById("empModal");
    const modalDel = document.getElementById("empModalDelete");
    const form = document.getElementById("empForm");
    const fId = document.getElementById("fId");
    const fLoginId = document.getElementById("fLoginId");

    // --- LÓGICA DE DADOS COM FIRESTORE ---

    async function carregarErenderizarFuncionarios() {
        try {
            const snapshot = await db.collection("funcionarios").get();
            state = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            render();
        } catch (error) {
            console.error("Erro ao carregar funcionários:", error);
            alert("Não foi possível carregar os dados dos funcionários.");
        }
    }

    // --- LÓGICA DO FORMULÁRIO (SALVA NO FIRESTORE) ---
    form?.addEventListener("submit", async (e) => {
        e.preventDefault();
        const isEditing = !!editingId;
        const email = fLoginId.value.trim();
        const nome = document.getElementById("fName").value.trim();
        const role = document.getElementById("fRole").value;
```

```
        if (!nome || !role || !email) {
            return alert("Preencha Nome, Função e E-mail de Login.");
        }

        let uid_final = editingId;

        if (!isEditing) {
            uid_final = prompt(`Cadastro de Novo Funcionário para o e-mail: ${email}\n\n1. Crie este usuário na aba 'Authentication' do Firebase.\n2. Copie o UID gerado.\n3. Cole o UID aqui:`);
            if (!uid_final) {
                alert("Cadastro cancelado. O UID é necessário.");
                return;
            }
        }

        const payload = {
            nome: nome,
            role: role,
            email: email,
            accessType: (role === 'Gerente') ? 'Admin' : 'Funcionario',
            phone: document.getElementById("fPhone").value.trim() || null,
        };

        try {
            await db.collection("funcionarios").doc(uid_final).set(payload, { merge: true });
            alert("Funcionário salvo com sucesso!");
            close(modal);
            carregarErenderizarFuncionarios();
        } catch (error) {
            console.error("Erro ao salvar funcionário:", error);
            alert(`Falha ao salvar funcionário: ${error.message}`);
        }
    });

    // --- LÓGICA DE EXCLUSÃO (DELETA DO FIRESTORE) ---
    document.getElementById("btnEmpConfirmDelete")?.addEventListener("click", async (e) => {
        const idParaDeletar = e.currentTarget.dataset.id;
        if (!idParaDeletar) return;
        try {
            await db.collection("funcionarios").doc(idParaDeletar).delete();
            alert("Perfil do funcionário excluído do banco de dados.\n\nATENÇÃO: O login deste usuário ainda precisa ser removido manualmente na aba 'Authentication' do Firebase.");
            close(modalDel);
            carregarErenderizarFuncionarios();
        } catch (error) {
            console.error("Erro ao excluir funcionário:", error);
            alert("Falha ao excluir funcionário.");
        }
    });

    // --- FUNÇÃO DE RENDERIZAÇÃO (CORRIGIDA) ---
    function render() {
        if (!tbody) return;

        const term = query.trim().toLowerCase();
        const filtered = state.filter(emp => (emp.nome || "").toLowerCase().includes(term));
```

```
        if (filtered.length === 0) {
            tbody.innerHTML = '<tr><td colspan="5" style="text-align:center; padding:
20px;">Nenhum funcionário encontrado.</td></tr>';
            return;
        }

        // <<-- CÓDIGO CORRIGIDO AQUI -->>
        // A tabela agora é construída com todas as colunas (<td>) corretamente.
        tbody.innerHTML = filtered.map(emp => `
            <tr data-id="${emp.id}">
                <td>${emp.foto ? `<img src="${emp.foto}" alt="${emp.nome}" class="avatar-sm">` :
'Sem Foto'}</td>
                <td>${emp.nome || "Sem nome"}</td>
                <td>${emp.role || "Não definido"}</td>
                <td>${emp.phone || "Não informado"}</td>
                <td class="hide-sm">${emp.address || "—"}</td>
                <td>${emp.contractType || "—"}</td>
                <td>...</td> <td class="hide-sm">...</td> <td>${emp.accessType || "—"}</td>
                <td>${emp.id || "—"}</td>
                <td>...</td> <td>
                    <div class="cell-actions">
                        <button class="btn btn-yellow" data-action="edit">Editar</button>
                        <button class="btn btn-red" data-action="delete">Excluir</button>
                    </div>
                </td>
            </tr>
        `).join("");
    }

    // --- FUNÇÕES DE UI (CONTINUAM AS MESMAS) ---
    tbody?.addEventListener("click", (e) => {
        const btn = e.target.closest("[data-action]");
        if (!btn) return;
        const id = btn.closest("tr").dataset.id;
        const emp = state.find(e => e.id === id);
        if (!emp) return;

        if (btn.dataset.action === "edit") {
            editingId = emp.id;
            document.getElementById("empModalTitle").textContent = "Editar funcionário";
            fId.value = emp.id;

            fLoginId.value = emp.email || "";
            fLoginId.readOnly = true;
            fLoginId.style.backgroundColor = '#333';

            document.getElementById("fName").value = emp.nome || "";
            document.getElementById("fRole").value = emp.role || "";
            document.getElementById("fPhone").value = emp.phone || "";
            open(modal);
        } else if (btn.dataset.action === "delete") {
            document.getElementById("empDelNome").textContent = emp.nome || "este funcionário";
            document.getElementById("btnEmpConfirmDelete").dataset.id = id;
            open(modalDel);
        }
    });
```

```
    document.getElementById("btnFabAddEmp")?.addEventListener("click", () => {
        editingId = null;
        form.reset();
        document.getElementById("empModalTitle").textContent = "Novo funcionário";

        fLoginId.readOnly = false;
        fLoginId.style.backgroundColor = '';

        open(modal);
    });

    const open = (el) => el.classList.add("show");
    const close = (el) => el.classList.remove("show");
    document.querySelectorAll("[data-close]").forEach(b => b.addEventListener("click", (e) =>
close(e.target.closest(".modal"))));
    document.getElementById("empQ")?.addEventListener("input", (e) => {
        query = e.target.value || "";
        render();
    });

    // --- INICIALIZAÇÃO ---
    carregarErenderizarFuncionarios();
})();
```

```
// 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
        });
    }
});
```

```
// js/lancamentos.js
// Módulo de Lançamentos, integrado com o Firebase Firestore

(function() {
    // Inicializa o serviço do Firestore
    const db = firebase.firestore();

    // --- Helpers ---
    const fmtBRL = n => new Intl.NumberFormat("pt-BR", { style: "currency", currency: "BRL"
}).format(n || 0);

    // --- Estado da Aplicação (dados que virão do Firebase) ---
```

```
    let todosLancs = []; // Todos os lançamentos carregados
    let funcs = [], clis = [], servs = [], prods = []; // Dados de outras coleções

    // --- Referências do DOM ---
    const tbody = document.getElementById("lanTbody");
    // ... (outras referências de filtros e modais)

    // --- LÓGICA DE CARREGAMENTO DE DADOS ---

    // Função principal que carrega todos os dados necessários
    async function carregarTodosOsDados() {
        try {
            // Carrega em paralelo os dados de apoio (funcionários, clientes, etc.)
            const [funcsSnap, clisSnap, servsSnap, prodsSnap] = await Promise.all([
                db.collection("funcionarios").get(),
                db.collection("clientes").get(),
                db.collection("servicos").get(),
                db.collection("produtos").get()
            ]);

            funcs = funcsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            clis = clisSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            servs = servsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            prods = prodsSnap.docs.map(doc => ({ id: doc.id, ...doc.data() }));

            // Preenche os filtros da página
            preencherFiltros();

            // Carrega os lançamentos e renderiza a tabela
            await carregarErenderizarLancamentos();

        } catch (error) {
            console.error("Erro ao carregar dados de apoio:", error);
            alert("Falha ao carregar dados essenciais para a página de lançamentos.");
        }
    }

    async function carregarErenderizarLancamentos() {
        try {
            // Por performance, em um app real, faríamos a filtragem no backend.
            // Aqui, vamos carregar todos e filtrar no cliente para manter a simplicidade.
            const snapshot = await db.collection("lancamentos").orderBy("dataISO", "desc").get();
            todosLancs = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            render(); // Renderiza a tabela com os dados carregados
        } catch (error) {
            console.error("Erro ao carregar lançamentos:", error);
            tbody.innerHTML = `<tr><td colspan="13">Erro ao carregar lançamentos.</td></tr>`;
        }
    }

    function preencherFiltros() {
        const fProf = document.getElementById('fProf');
        const fCli = document.getElementById('fCli');

        if (fProf) {
            fProf.innerHTML = '<option value="">Todos</option>' + funcs.map(f => `<option
value="${f.id}">${f.nome}</option>`).join('');
```

```
        }
        if (fCli) {
            fCli.innerHTML = '<option value="">Todos</option>' + clis.map(c => `<option
value="${c.id}">${c.nome}</option>`).join('');
        }
    }

    // --- RENDERIZAÇÃO E FILTROS ---

    function render() {
        if (!tbody) return;

        // Aplica os filtros selecionados na tela (lógica simplificada)
        const filteredLancs = aplicarFiltros(todosLancs);

        // Renderiza KPIs (indicadores)
        renderKPIs(filteredLancs);

        if (filteredLancs.length === 0) {
            tbody.innerHTML = `<tr><td colspan="13" style="text-align:center; padding:
20px;">Nenhum lançamento encontrado para os filtros selecionados.</td></tr>`;
            return;
        }

        tbody.innerHTML = filteredLancs.map(lanc => {
            const data = new Date(lanc.dataISO).toLocaleString('pt-BR', { dateStyle: 'short',
timeStyle: 'short' });
            return `
                <tr>
                    <td>${data}</td>
                    <td>${lanc.profissionalNome || "—"}</td>
                    <td>${lanc.clienteNome || "—"}</td>
                    <td>${(lanc.servico && lanc.servico.length > 0) ? lanc.servico.map(s =>
s.nome).join(', ') : "—"}</td>
                    <td>${(lanc.produtos && lanc.produtos.length > 0) ? lanc.produtos.map(p =>
`${p.nome} x${p.quantidade}`).join(', ') : "—"}</td>
                    <td>${lanc.pagamento?.forma || "—"}</td>
                    <td>${fmtBRL(lanc.valores?.receitaBruta)}</td>
                    <td>${fmtBRL(lanc.valores?.taxaPagamento)}</td>
                    <td>${fmtBRL(lanc.valores?.comissao)}</td>
                    <td>${fmtBRL(lanc.valores?.custoTotal)}</td>
                    <td>${fmtBRL(lanc.valores?.resultadoLiquido)}</td>
                    <td>
                        <div class="cell-actions">
                            <button class="btn btn-red" data-action="delete" data-
id="${lanc.id}">🗑</button>
                        </div>
                    </td>
                </tr>
            `;
        }).join("");
    }

    function aplicarFiltros(lancamentos) {
        // Esta função pode ser expandida com a lógica de todos os filtros da tela
        // Por enquanto, ela retorna a lista completa
        return lancamentos;
```

```
    }

    function renderKPIs(lancamentosFiltrados) {
        const sum = (key) => lancamentosFiltrados.reduce((acc, lanc) => acc + (lanc.valores?.[key]
|| 0), 0);

        document.getElementById("kpiReceitaBruta").textContent = fmtBRL(sum("receitaBruta"));
        document.getElementById("kpiTaxas").textContent = fmtBRL(sum("taxaPagamento"));
        document.getElementById("kpiComissoes").textContent = fmtBRL(sum("comissao"));
        document.getElementById("kpiCustos").textContent = fmtBRL(sum("custoTotal"));
        document.getElementById("kpiResultado").textContent = fmtBRL(sum("resultadoLiquido"));
    }

    // --- AÇÕES (Exclusão, etc.) ---
    tbody?.addEventListener("click", (e) => {
        const btn = e.target.closest("[data-action='delete']");
        if (!btn) return;

        const id = btn.dataset.id;
        const lanc = todosLancs.find(l => l.id === id);
        if (!lanc) return;

        if (confirm(`Tem certeza que deseja excluir o lançamento de ${lanc.clienteNome} no valor
de ${fmtBRL(lanc.valores?.receitaBruta)}?`)) {
            db.collection("lancamentos").doc(id).delete()
                .then(() => {
                    alert("Lançamento excluído.");
                    carregarErenderizarLancamentos(); // Recarrega a lista
                })
                .catch(err => {
                    console.error("Erro ao excluir lançamento:", err);
                    alert("Falha ao excluir lançamento.");
                });
        }
    });

    // --- INICIALIZAÇÃO ---
    // A primeira coisa que a página faz é carregar todos os dados
    carregarTodosOsDados();
})();
```

```
// js/layout-loader.js
// Responsável por carregar o menu (sidebar) e destacar o link ativo.

(function () {
    const sidebarPlaceholder = document.getElementById('sidebar-placeholder');

    if (sidebarPlaceholder) {
        // Busca o arquivo _menu.html a partir da mesma pasta da página atual (ex: /admin/)
        fetch('./_menu.html')
            .then(response => {
                if (!response.ok) {
                    throw new Error(`Arquivo _menu.html não encontrado (Status:
${response.status})`);
```

```
                }
                return response.text();
            })
            .then(menuHtml => {
                sidebarPlaceholder.innerHTML = menuHtml;
                highlightActiveNav(); // Chama a função para destacar o link
            })
            .catch(error => {
                console.error('Erro ao carregar o layout:', error);
                sidebarPlaceholder.innerHTML = `<div style="padding: 20px; color:
red;">${error.message}</div>`;
            });
    }

    function highlightActiveNav() {
        const links = Array.from(document.querySelectorAll(".admin-nav .nav-item"));
        if (!links.length) return;

        const currentPage = window.location.pathname.split("/").pop();
        const activeLink = links.find(a => (a.getAttribute("href") || "").includes(currentPage));

        if (activeLink) {
            links.forEach(a => a.classList.remove("active"));
            activeLink.classList.add("active");

            const color = activeLink.dataset.color || "#fff";
            activeLink.style.borderColor = color;
            activeLink.style.outlineColor = color;
            const dot = activeLink.querySelector('.dot');
            if(dot) dot.style.backgroundColor = color;
        }
    }
})();
```

```
// js/login.js
// Responsável por login, cadastro, autenticação com Google e REDIRECIONAMENTO POR NÍVEL DE ACESSO

document.addEventListener("DOMContentLoaded", () => {
    const auth = firebase.auth();
    const firestore = firebase.firestore();

    // Função que decide para onde redirecionar o usuário após o login
    function redirecionarPorNivelDeAcesso(uid) {
        const funcionarioRef = firestore.collection("funcionarios").doc(uid);

        funcionarioRef.get().then((doc) => {
            if (doc.exists) {
                const accessType = doc.data().accessType;
                if (accessType === 'Admin') {
                    // <<-- CAMINHO CORRIGIDO -->>
                    window.location.href = "admin/agenda.html";
                } else {
                    // <<-- CAMINHO CORRIGIDO -->>
                    window.location.href = "funcionario/agenda.html";
```

```
                }
            } else {
                // Se não é funcionário, é cliente
                window.location.href = "home.html";
            }
        }).catch(err => {
            console.error("Erro ao verificar nível de acesso:", err);
            // Em caso de erro, manda para a home de cliente como padrão seguro
            window.location.href = "home.html";
        });
    }

    // --- LÓGICA DE LOGIN COM EMAIL E SENHA ---
    const formLogin = document.getElementById("formLogin");
    if (formLogin) {
        formLogin.addEventListener("submit", (e) => {
            e.preventDefault();
            const email = document.getElementById("loginEmail").value;
            const senha = document.getElementById("loginSenha").value;
            const erroLogin = document.getElementById("loginErro");

            auth.signInWithEmailAndPassword(email, senha)
                .then((userCredential) => {
                    // Chama a função central de redirecionamento
                    redirecionarPorNivelDeAcesso(userCredential.user.uid);
                })
                .catch(err => {
                    erroLogin.innerText = "Erro: " + err.message;
                });
        });
    }

    // --- LÓGICA DE LOGIN COM GOOGLE (AGORA INTELIGENTE) ---
    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;
                    const uid = user.uid;

                    // <<-- LÓGICA ATUALIZADA -->>
                    // Procura primeiro em 'funcionarios'
                    const funcionarioDoc = await
firestore.collection("funcionarios").doc(uid).get();

                    if (funcionarioDoc.exists) {
                        // Se encontrou, redireciona como funcionário/admin
                        redirecionarPorNivelDeAcesso(uid);
                    } else {
                        // Se não, trata como cliente (cria se não existir)
                        const clienteDoc = await firestore.collection("clientes").doc(uid).get();
                        if (!clienteDoc.exists) {
                            await firestore.collection("clientes").doc(uid).set({
                                nome: user.displayName,
                                email: user.email,
```

```
                            telefone: user.phoneNumber || "Não informado",
                            criadoEm: new Date()
                        });
                    }
                    // E redireciona para a home de cliente
                    window.location.href = "home.html";
                }
            })
            .catch(err => {
                alert("Erro no login com Google: " + err.message);
            });
    });
}

// --- LÓGICA DE CADASTRO (PARA CLIENTES) ---
const btnCadastro = document.getElementById("btnCadastro");
if (btnCadastro) {
    btnCadastro.addEventListener("click", () => {
        // ... (o restante do código de cadastro continua igual, pois ele já está correto)
        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 erroCadastro = document.getElementById("cadastroErro");

        if (!nome || !telefone || !email || !senha) {
            erroCadastro.innerText = "Preencha todos os campos obrigatórios.";
            return;
        }

        auth.createUserWithEmailAndPassword(email, senha)
            .then((userCred) => {
                const user = userCred.user;
                return firestore.collection("clientes").doc(user.uid).set({
                    nome: nome,
                    telefone: telefone,
                    email: email,
                    criadoEm: new Date()
                });
            })
            .then(() => {
                window.location.href = "home.html";
            })
            .catch(err => {
                erroCadastro.innerText = "Erro: " + err.message;
            });
    });
}

// --- LÓGICA PARA ALTERNAR ENTRE FORMULÁRIOS E VER SENHA ---
// ... (o restante do código para alternar formulários e ver senha continua o mesmo)
const boxLogin = document.getElementById("boxLogin");
const boxCadastro = document.getElementById("boxCadastro");
const btnMostrarCadastro = document.getElementById("mostrarCadastro");
const btnMostrarLogin = document.getElementById("mostrarLogin");

if (btnMostrarCadastro) {
```

```
        btnMostrarCadastro.addEventListener("click", (e) => {
            e.preventDefault();
            boxLogin.style.display = "none";
            boxCadastro.style.display = "block";
        });
    }

    if (btnMostrarLogin) {
        btnMostrarLogin.addEventListener("click", (e) => {
            e.preventDefault();
            boxCadastro.style.display = "none";
            boxLogin.style.display = "block";
        });
    }

    const toggleSenha = document.getElementById("toggleSenha");
    const campoSenha = document.getElementById("loginSenha");

    if (toggleSenha && campoSenha) {
        toggleSenha.addEventListener("click", () => {
            const type = campoSenha.getAttribute("type") === "password" ? "text" : "password";
            campoSenha.setAttribute("type", type);
            toggleSenha.textContent = type === "password" ? "👁□" : "🙈";
        });
    }
});
```

```
// js/perfil-funcionario.js
// VERSÃO FINAL: Usa 100% o usuário logado e estruturado corretamente.

document.addEventListener('DOMContentLoaded', () => {
    const db = firebase.firestore();
    const auth = firebase.auth();
    const configCollection = db.collection("agenda_config");

    // --- Helpers de data ---
    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'];

    // --- Referências do DOM ---
    const employeeNameBadge = document.getElementById('employeeName');
    const weekMatrix = document.getElementById('weekMatrix');
    const dayStrip = document.getElementById('dayStrip');
    const dayMonth = document.getElementById('dayMonth');
    const saveConfigBtn = document.getElementById('saveConfig');
    const prevDaysBtn = document.getElementById('prevDaysBtn');
    const nextDaysBtn = document.getElementById('nextDaysBtn');

    // --- Estado da Aplicação ---
    let currentUser = null;
    let workingCopy = null;
```

```
    let editingDateYMD = null;
    let allDayPills = [];
    let dayStripStartIndex = 0;
    const VISIBLE_DAYS = 5;
    const TOTAL_DAYS_TO_SHOW = 30;

    const TIMES = Array.from({ length: (20 - 10) * 2 }, (_, i) => {
        const hour = 10 + Math.floor(i / 2);
        const minute = (i % 2) * 30;
        return `${pad2(hour)}:${pad2(minute)}`;
    });

    // --- LÓGICA PRINCIPAL ---
    auth.onAuthStateChanged(async user => {
        if (user) {
            currentUser = user;
            await initPage();
        } else {
            window.location.href = "../../index.html";
        }
    });

    async function initPage() {
        if (!currentUser) return;
        try {
            const userProfile = await db.collection("funcionarios").doc(currentUser.uid).get();
            const userName = userProfile.exists && userProfile.data().nome ?
userProfile.data().nome : currentUser.email;

            if (employeeNameBadge) {
                employeeNameBadge.textContent = userName;
            }

            const configDoc = await configCollection.doc(currentUser.uid).get();
            workingCopy = configDoc.exists ? configDoc.data() : { _overrides: {} };

            setupEventListeners();
            generateAllDayPills();
            updateDayStripView();
            updateMonthHeader();
            selectDayForEdit(toYMD(new Date()));
        } catch (error) {
            console.error("Erro ao iniciar a página:", error);
            alert("Não foi possível carregar suas configurações de agenda.");
        }
    }

    function setupEventListeners() {
        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();
        });
        saveConfigBtn.addEventListener('click', salvarConfiguracao);
    }

    async function salvarConfiguracao() {
        if (!currentUser || !workingCopy) {
            return alert("Erro: Usuário não identificado.");
        }
        try {
            await configCollection.doc(currentUser.uid).set(workingCopy);

            const userProfile = await db.collection("funcionarios").doc(currentUser.uid).get();
            const userName = userProfile.exists && userProfile.data().nome ?
userProfile.data().nome : currentUser.email;
            alert(`Disponibilidade de ${userName} salva com sucesso!`);
        } catch (error) {
            console.error("Erro ao salvar configuração:", error);
            alert("Falha ao salvar a configuração.");
        }
    }

    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.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() {
        if(!dayStrip || !prevDaysBtn || !nextDaysBtn) return;
        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 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) { label = `${cap(m1)} de ${y1}`; }
            else { label = `${cap(m1)} / ${cap(m2)} de ${y1}`; }
        } else {
            label = `${cap(m1)} de ${y1} / ${cap(m2)} de ${y2}`;
        }
        dayMonth.textContent = label;
    }

    function selectDayForEdit(ymd) {
        editingDateYMD = ymd;
        allDayPills.forEach(p => p.classList.toggle('active', p.dataset.ymd === ymd));
        buildDayTableForEdit();
    }

    function buildDayTableForEdit() {
        if (!weekMatrix) return;
        const d = new Date(`${editingDateYMD}T12:00:00`);
        const dow = dow1to7(d);
        const baseDayConfig = workingCopy[dow] || {};
        const overrideConfig = workingCopy._overrides?.[editingDateYMD] || {};
        let tableHTML = `<table class="week-table"><thead><tr><th class="week-
time"></th><th>${d.toLocaleDateString('pt-BR')}</th></tr></thead><tbody>`;
        for (const time of TIMES) {
            const isAvailable = overrideConfig.hasOwnProperty(time) ? overrideConfig[time] :
(baseDayConfig[time] === true);
            tableHTML += `<tr><th class="week-time">${time}</th><td class="cell ${isAvailable ?
'active' : ''}" data-time="${time}">${isAvailable ? 'Disponível' : 'Bloqueado'}</td></tr>`;
        }
        tableHTML += `</tbody></table>`;
        weekMatrix.innerHTML = tableHTML;
        weekMatrix.querySelectorAll('.cell').forEach(cell => cell.addEventListener('click',
toggleCellAvailability));
    }

    function toggleCellAvailability(e) {
        const td = e.target.closest('.cell');
        if (!td) return;
        const time = td.dataset.time;
        if (!workingCopy._overrides) workingCopy._overrides = {};
        if (!workingCopy._overrides[editingDateYMD]) workingCopy._overrides[editingDateYMD] = {};
        const newStatus = !td.classList.contains('active');
        workingCopy._overrides[editingDateYMD][time] = newStatus;
        td.classList.toggle('active', newStatus);
        td.textContent = newStatus ? 'Disponível' : 'Bloqueado';
    }
});
```

```
// js/produtos.js
// Módulo de Produtos, Estoque e Kardex, integrado com o Firebase Firestore

(function() {
    // Inicializa o serviço do Firestore
    const db = firebase.firestore();
    const produtosCollection = db.collection("produtos");
    const kardexCollection = db.collection("kardex");

    // --- Helpers ---
    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>`;

    // --- Estado e Referências do DOM ---
    let produtos = []; // Guardará os produtos vindos do Firestore
    let query = "";
    const tbody = document.getElementById("prdTbody");
    const modal = document.getElementById("prdModal");
    const movModal = document.getElementById("movModal");
    const delModal = document.getElementById("prdModalDelete");
    const form = document.getElementById("prdForm");
    const movForm = document.getElementById("movForm");

    // --- LÓGICA DE DADOS COM FIRESTORE ---

    async function carregarErenderizarProdutos() {
        try {
            const snapshot = await produtosCollection.orderBy("nome").get();
            produtos = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
            render();
        } catch (error) {
            console.error("Erro ao carregar produtos:", error);
            tbody.innerHTML = `<tr><td colspan="12">Erro ao carregar produtos.</td></tr>`;
        }
    }

    // --- RENDERIZAÇÃO DA TABELA ---

    function render() {
        if (!tbody) return;
        const term = query.trim().toLowerCase();
        const data = produtos.filter(p => (p.nome || "").toLowerCase().includes(term));

        tbody.innerHTML = data.map(p => {
            const lucro = Number(p.precoVenda || 0) - Number(p.cmp || 0);
            const margem = p.precoVenda > 0 ? (lucro / Number(p.precoVenda)) * 100 : 0;
            const st = statusProd(p);
            const stChip = st === "ZERADO" ? badge("ZERADO", "#ff7878") : st === "BAIXO" ?
badge("BAIXO", "#ffd166") : badge("OK", "#a0e7a0");

            return `
                <tr data-id="${p.id}">
                    <td>${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 class="cell-actions">
                        <button class="btn btn-yellow" data-act="edit"
title="Editar">✏️</button>
                        <button class="btn btn-blue" data-act="move"
title="Movimentar">🔁</button>
                        <button class="btn btn-red" data-act="del" title="Excluir">🗑️</button>
                    </td>
                </tr>`;
        }).join("");
    }

    function statusProd(p) {
        if ((p.estoqueAtual ?? 0) <= 0) return "ZERADO";
        if ((p.estoqueAtual ?? 0) <= (p.estoqueMinimo ?? 0)) return "BAIXO";
        return "OK";
    }

    // --- LÓGICA DO MODAL DE PRODUTO (CRIAR/EDITAR) ---

    form?.addEventListener("submit", async (e) => {
        e.preventDefault();
        const id = document.getElementById("pId").value;
        const nome = document.getElementById("pNome").value.trim();
        const sku = document.getElementById("pSKU").value.trim();
        if (!nome || !sku) return alert("Preencha Nome e SKU.");

        const isEditing = !!id;
        const docId = isEditing ? id : produtosCollection.doc().id;

        const existingProduct = isEditing ? produtos.find(p => p.id === id) : {};

        const payload = {
            nome: nome,
            sku: sku,
            categoria: document.getElementById("pCategoria").value.trim() || null,
            unidade: document.getElementById("pUn").value.trim() || "un",
            precoVenda: Number(document.getElementById("pPreco").value || 0),
            fornecedor: document.getElementById("pFornecedor").value.trim() || null,
            estoqueMinimo: Number(document.getElementById("pEstMin").value || 0),
            obs: document.getElementById("pObs").value.trim() || "",
            // Mantém os valores de estoque e custo se estiver editando
            estoqueAtual: existingProduct.estoqueAtual ?? 0,
            cmp: existingProduct.cmp ?? 0,
        };

        try {
            await produtosCollection.doc(docId).set(payload, { merge: true });
```

```
            toast("Produto salvo com sucesso!");
            close(modal);
            carregarErenderizarProdutos();
        } catch (error) {
            console.error("Erro ao salvar produto:", error);
            toast("Falha ao salvar produto.");
        }
    });

    // --- LÓGICA DO MODAL DE MOVIMENTAÇÃO DE ESTOQUE ---

    movForm?.addEventListener("submit", async (e) => {
        e.preventDefault();
        const id = document.getElementById("mProductId").value;
        const p = produtos.find(x => x.id === id);
        if (!p) return alert("Produto não encontrado.");

        const tipo = document.getElementById("mTipo").value;
        const qtd = Number(document.getElementById("mQtd").value || 0);
        const nowISO = new Date().toISOString();

        let novoEstoque = Number(p.estoqueAtual || 0);
        let novoCmp = Number(p.cmp || 0);
        const kardexEntry = { productId: p.id, dataISO: nowISO, quantidade: qtd };

        if (tipo === "entrada") {
            const custo = Number(document.getElementById("mCusto").value || 0);
            if (qtd <= 0 || custo < 0) return alert("Informe quantidade e custo válidos.");

            const estAnt = Number(p.estoqueAtual || 0);
            const cmpAnt = Number(p.cmp || 0);
            const novoEstoqueTotal = estAnt + qtd;
            novoCmp = novoEstoqueTotal > 0 ? (estAnt * cmpAnt + qtd * custo) / novoEstoqueTotal :
custo;
            novoEstoque = novoEstoqueTotal;

            Object.assign(kardexEntry, { tipo: "entrada", motivo: "compra", custoUnitario: custo
});

        } else if (tipo === "saida") {
            if (qtd <= 0) return alert("Quantidade deve ser > 0.");
            novoEstoque -= qtd;
            Object.assign(kardexEntry, { tipo: "saida", motivo:
document.getElementById("mMotivo").value, custoUnitario: novoCmp });

        } else if (tipo === "ajuste") {
            novoEstoque += qtd; // qtd pode ser negativa para ajuste de baixa
            Object.assign(kardexEntry, { tipo: "ajuste", motivo:
document.getElementById("mJust").value, custoUnitario: novoCmp });
        }

        try {
            // Usa uma transação para garantir que o produto e o kardex sejam atualizados juntos
            await db.runTransaction(async (transaction) => {
                const productRef = produtosCollection.doc(p.id);
                transaction.update(productRef, { estoqueAtual: novoEstoque, cmp: novoCmp });
```

```
            const kardexRef = kardexCollection.doc(); // Cria um novo documento com ID
automático
            transaction.set(kardexRef, kardexEntry);
        });

        toast("Movimentação de estoque salva com sucesso!");
        close(movModal);
        carregarErenderizarProdutos();
    } catch (error) {
        console.error("Erro ao movimentar estoque:", error);
        toast("Falha ao salvar movimentação.");
    }
});

// --- AÇÕES DA TABELA E ABERTURA DE MODAIS ---

tbody?.addEventListener("click", (e) => {
    const btn = e.target.closest("[data-act]");
    if (!btn) return;

    const id = btn.closest("tr").dataset.id;
    const p = produtos.find(x => x.id === id);
    if (!p) return;

    if (btn.dataset.act === "edit") {
        document.getElementById("prdModalTitle").textContent = "Editar produto";
        document.getElementById("pId").value = p.id;
        // Preencher o resto do formulário
        document.getElementById("pNome").value = p.nome || "";
        document.getElementById("pSKU").value = p.sku || "";
        // ... (outros campos)
        open(modal);

    } else if (btn.dataset.act === "del") {
        document.getElementById("prdDelNome").textContent = p.nome || "este produto";
        document.getElementById("btnPrdConfirmDelete").dataset.id = id;
        open(delModal);

    } else if (btn.dataset.act === "move") {
        document.getElementById("movTitle").textContent = `Movimentar: ${p.nome}`;
        document.getElementById("mProductId").value = p.id;
        open(movModal);
    }
});

document.getElementById("btnPrdConfirmDelete")?.addEventListener("click", async (e) => {
    const id = e.currentTarget.dataset.id;
    try {
        await produtosCollection.doc(id).delete();
        toast("Produto excluído.");
        close(delModal);
        carregarErenderizarProdutos();
    } catch (error) {
        console.error("Erro ao excluir produto:", error);
        toast("Falha ao excluir produto.");
    }
});
```

```
    // Abertura do modal de novo produto
    document.getElementById("btnFabAddPrd")?.addEventListener("click", () => {
        form.reset();
        document.getElementById("pId").value = "";
        document.getElementById("prdModalTitle").textContent = "Novo produto";
        open(modal);
    });

    // Funções de UI (abrir/fechar modais)
    const open = (el) => { if(el) el.classList.add("show"); };
    const close = (el) => { if(el) el.classList.remove("show"); };
    document.querySelectorAll("[data-close]").forEach(b => {
        b.addEventListener("click", () => b.closest(".modal") && close(b.closest(".modal")));
    });
    document.getElementById("prdQ")?.addEventListener("input", (e) => {
        query = e.target.value || "";
        render();
    });

    const toast = msg => alert(msg); // Placeholder para notificações

    // --- INICIALIZAÇÃO ---
    carregarErenderizarProdutos();
})();
```

```
// js/service.js
// Módulo de Serviços do Admin, integrado com o Firebase Firestore

(() => {
    // Inicializa o serviço do Firestore
    const db = firebase.firestore();
    const servicosCollection = db.collection("servicos");

    // --- Helpers ---
    const BRL = v => (isNaN(v) ? 0 : v).toLocaleString("pt-BR", { style: "currency", currency:
"BRL" });
    const pct = v => isFinite(v) ? `${v.toFixed(1)}%` : "—";
    const num = v => (typeof v === "number" ? v : parseFloat(String(v).replace(",", "."))) || 0;
    const toast = msg => alert(msg); // Simples alerta por enquanto

    // --- Estado e Referências do DOM ---
    let state = []; // Guardará os serviços vindos do Firestore
    let editId = null; // Guarda o ID do serviço em edição
    const tbody = document.getElementById("svcTbody");
    const modal = document.getElementById("svcModal");
    const form = document.getElementById("svcForm");

    // --- LÓGICA DE DADOS COM FIRESTORE ---

    async function carregarESalvarServicos() {
        try {
            const snapshot = await servicosCollection.get();
            state = snapshot.docs.map(doc => ({ id: doc.id, ...doc.data() }));
```

```
            renderTable();
        } catch (error) {
            console.error("Erro ao carregar serviços:", error);
            tbody.innerHTML = `<tr><td colspan="10">Erro ao carregar serviços.</td></tr>`;
        }
    }

    // --- RENDERIZAÇÃO DA TABELA ---

    function renderTable() {
        if (!tbody) return;

        const q = (document.getElementById("svcQ")?.value || "").trim().toLowerCase();
        const filtered = state.filter(s => {
            if (!q) return true;
            return (s.nome || "").toLowerCase().includes(q) || (s.obs ||
"").toLowerCase().includes(q);
        });

        if (filtered.length === 0) {
            tbody.innerHTML = `<tr><td colspan="10" style="text-align:center; padding:
20px;">Nenhum serviço cadastrado.</td></tr>`;
            return;
        }

        tbody.innerHTML = filtered.map(s => {
            const lucro = (s.preco || 0) - (s.custoTotal || 0);
            const margem = (s.preco > 0) ? (lucro / s.preco * 100) : 0;
            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>${s.duracao || "—"} min</td>
                    <td>${BRL(s.preco)}</td>
                    <td>${BRL(s.custoTotal)}</td>
                    <td>${BRL(lucro)}</td>
                    <td>${pct(margem)}</td>
                    <td class="hide-sm">${s.obs || "—"}</td>
                    <td class="hide-sm">${(s.materiais || []).map(m => m.nome).join(', ') || "—
"}</td>
                    <td class="cell-actions">
                        <button class="btn btn-yellow" data-action="edit">✏️</button>
                        <button class="btn btn-red" data-action="delete">🗑️</button>
                    </td>
                </tr>
            `;
        }).join("");
    }

    // --- LÓGICA DO MODAL (CRIAR/EDITAR) ---

    function resetForm() {
        editId = null;
        form.reset();
        document.getElementById("svcModalTitle").textContent = "Novo serviço";
        // Limpar pré-visualização da foto se houver
```

```
        const fotoPreview = document.getElementById("svcFotoPreview");
        if (fotoPreview) {
            fotoPreview.innerHTML = 'FOTO';
            delete fotoPreview.dataset.b64;
        }
    }

    form?.addEventListener("submit", async (e) => {
        e.preventDefault();

        const nome = document.getElementById("mNome").value.trim();
        const duracao = num(document.getElementById("mDuracao").value);
        const preco = num(document.getElementById("mPreco").value);

        if (!nome || duracao <= 0 || preco <= 0) {
            return toast("Preencha Nome, Duração e Preço corretamente.");
        }

        const payload = {
            nome: nome,
            duracao: duracao,
            preco: preco,
            custoTotal: num(document.getElementById("mCusto").value), // Custo simplificado por
enquanto
            obs: document.getElementById("mObs").value.trim(),
            // Adicionar lógica de foto e materiais se necessário
            materiais: [],
            foto: null,
            tipoServico: "unitario" // Simplificado como 'unitario' por enquanto
        };

        try {
            const docId = editId || servicosCollection.doc().id;
            await servicosCollection.doc(docId).set(payload, { merge: true });

            toast(`Serviço "${nome}" salvo com sucesso!`);
            closeModal(modal);
            carregarESalvarServicos();
        } catch (error) {
            console.error("Erro ao salvar serviço:", error);
            toast("Falha ao salvar o serviço.");
        }
    });

    // --- AÇÕES DA TABELA E MODAIS ---

    document.getElementById("svcTbody")?.addEventListener("click", (e) => {
        const btn = e.target.closest("[data-action]");
        if (!btn) return;

        const id = btn.closest("tr").dataset.id;
        const action = btn.dataset.action;
        const s = state.find(x => x.id === id);
        if (!s) return;

        if (action === "edit") {
            editId = s.id;
```

```
            document.getElementById("svcModalTitle").textContent = "Editar serviço";
            document.getElementById("mNome").value = s.nome || "";
            document.getElementById("mDuracao").value = s.duracao || "";
            document.getElementById("mPreco").value = s.preco || "";
            document.getElementById("mCusto").value = s.custoTotal || "";
            document.getElementById("mObs").value = s.obs || "";
            openModal(modal);
        } else if (action === "delete") {
            if (confirm(`Tem certeza que deseja excluir o serviço "${s.nome}"?`)) {
                servicosCollection.doc(id).delete()
                    .then(() => {
                        toast("Serviço excluído.");
                        carregarESalvarServicos();
                    })
                    .catch(err => {
                        console.error("Erro ao excluir:", err);
                        toast("Falha ao excluir serviço.");
                    });
            }
        }
    });

    // Funções de UI (abrir/fechar modais)
    const openModal = (el) => { if(el) el.classList.add("show"); };
    const closeModal = (el) => { if(el) el.classList.remove("show"); };
    document.getElementById("btnFabAddService")?.addEventListener("click", () => {
        resetForm();
        openModal(modal);
    });
    document.querySelectorAll("[data-close]").forEach(btn => {
        btn.addEventListener("click", () => closeModal(btn.closest(".modal")));
    });
    document.getElementById("svcQ")?.addEventListener("input", renderTable);

    // --- INICIALIZAÇÃO ---
    carregarESalvarServicos();
})();
```

AQUIVOS JSON NA RAIZ

```
{
  "hosting": {
    "public": ".",
    "ignore": [
      "firebase.json",
      "**/.*",
      "**/node_modules/**"
    ],
    "rewrites": [
      {
        "source": "/_menu.html",
        "destination": "/html/admin/_menu.html"
```

```
      },
      {
        "source": "/**",
        "destination": "/html/index.html"
      }
    ]
  }
}
```

```
{
  "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"
    }
  ]
}
```

AQUIVOS JS NA RAIZ

```
// 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);
      }
    )
  );
});
```

ARQUIVO HTML NA RAIZ

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Welcome to Firebase Hosting</title>

    <!-- update the version number as needed -->
    <script defer src="/__/firebase/12.3.0/firebase-app-compat.js"></script>
    <!-- include only the Firebase features as you need -->
    <script defer src="/__/firebase/12.3.0/firebase-auth-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-database-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-firestore-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-functions-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-messaging-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-storage-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-analytics-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-remote-config-compat.js"></script>
    <script defer src="/__/firebase/12.3.0/firebase-performance-compat.js"></script>
    <!--
      initialize the SDK after all desired features are loaded, set useEmulator to false
      to avoid connecting the SDK to running emulators.
    -->
    <script defer src="/__/firebase/init.js?useEmulator=true"></script>

    <style media="screen">
```

```
      body { background: #ECEFF1; color: rgba(0,0,0,0.87); font-family: Roboto, Helvetica, Arial,
sans-serif; margin: 0; padding: 0; }
      #message { background: white; max-width: 360px; margin: 100px auto 16px; padding: 32px 24px;
border-radius: 3px; }
      #message h2 { color: #ffa100; font-weight: bold; font-size: 16px; margin: 0 0 8px; }
      #message h1 { font-size: 22px; font-weight: 300; color: rgba(0,0,0,0.6); margin: 0 0 16px;}
      #message p { line-height: 140%; margin: 16px 0 24px; font-size: 14px; }
      #message a { display: block; text-align: center; background: #039be5; text-transform:
uppercase; text-decoration: none; color: white; padding: 16px; border-radius: 4px; }
      #message, #message a { box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px 2px rgba(0,0,0,0.24); }
      #load { color: rgba(0,0,0,0.4); text-align: center; font-size: 13px; }
      @media (max-width: 600px) {
        body, #message { margin-top: 0; background: white; box-shadow: none; }
        body { border-top: 16px solid #ffa100; }
      }
    </style>
  </head>
  <body>
    <div id="message">
      <h2>Welcome</h2>
      <h1>Firebase Hosting Setup Complete</h1>
      <p>You're seeing this because you've successfully setup Firebase Hosting. Now it's time to
go build something extraordinary!</p>
      <a target="_blank" href="https://firebase.google.com/docs/hosting/">Open Hosting
Documentation</a>
    </div>
    <p id="load">Firebase SDK Loading&hellip;</p>

    <script>
      document.addEventListener('DOMContentLoaded', function() {
        const loadEl = document.querySelector('#load');
        // // 🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥
        // // The Firebase SDK is initialized and available here!
        //
        // firebase.auth().onAuthStateChanged(user => { });
        // firebase.database().ref('/path/to/ref').on('value', snapshot => { });
        // firebase.firestore().doc('/foo/bar').get().then(() => { });
        // firebase.functions().httpsCallable('yourFunction')().then(() => { });
        // firebase.messaging().requestPermission().then(() => { });
        // firebase.storage().ref('/path/to/ref').getDownloadURL().then(() => { });
        // firebase.analytics(); // call to activate
        // firebase.analytics().logEvent('tutorial_completed');
        // firebase.performance(); // call to activate
        //
        // // 🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥🔥

        try {
          let app = firebase.app();
          let features = [
            'auth',
            'database',
            'firestore',
            'functions',
            'messaging',
            'storage',
            'analytics',
            'remoteConfig',
```

```
            'performance',
          ].filter(feature => typeof app[feature] === 'function');
          loadEl.textContent = `Firebase SDK loaded with ${features.join(', ')}`;
        } catch (e) {
          console.error(e);
          loadEl.textContent = 'Error loading the Firebase SDK, check the console.';
        }
      });
    </script>
  </body>
</html>
```