HTML

<!DOCTYPE html>

<html lang="pt">

  <head>

    <meta charset="UTF-8" />

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

    <link rel="icon" href="assets/icons/icon-192.png" />

    <meta name="theme-color" content="#007BFF" />

    <title>Calculadora Financeira</title>

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

    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

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

    <!-- Enviar email via EmailJS-->

    <script>

      (function () {

        emailjs.init("1Ro3I9ZseyDAc6RTM"); // substitui pelo teu public key

      })();

    </script>

    <!--Firebase-->

    <script src="https://www.gstatic.com/firebasejs/8.10.0/firebase-app.js"></script>

    <script src="https://www.gstatic.com/firebasejs/8.10.0/firebase-firestore.js"></script>

  </head>

  <body>

    <!-- SCREEN 1 -->

    <div id="screen1" class="screen">

      <h1>INVESTMENT PORTEFOLIO</h1>

      <h3>made by antonioappleton</h3>

      <h2></h2>

      <h5>Top 10 Empresas com mais crescimento da semana</h5>

      <div id="top10Crescimento" class="top10-box">

        <ul id="listaTop10"></ul>

      </div>

      <button onclick="goToScreen2()">Começar</button>

    </div>

    <!-- SCREEN 2 -->

    <div id="screen2" class="screen hidden">

      <div

        class="explanation"

        style="

          margin-bottom: 4px;

          padding: 10px;

          border: 1px solid #090909;

          border-radius: 5px;

          background: #fffcfc;

        "

      >

        <h3 style="margin: 10px 0 2px 0">TP1 (Target Price 1):</h3>

        <h6 style="margin: 0 0 8px 0; font-weight: normal">

          Preço atual ou inicial da ação. É o ponto de partida para os cálculos

          de valorização e investimento.

        </h6>

        <h3 style="margin: 10px 0 2px 0">TP2 (Target Price 2):</h3>

        <h6 style="margin: 0 0 8px 0; font-weight: normal">

          Preço futuro da ação. É o ponto de chegada para os cálculos de

          valorização e investimento.

        </h6>

        <h3 style="margin: 10px 0 2px 0">Dividendos:</h3>

        <h6 style="margin: 0 0 8px 0; font-weight: normal">

          Valor que a Empresa paga por cada Stock que é detida, que será somado

          ao lucro total.

        </h6>

      </div>

      <h2>Escolhe uma opção:</h2>

      <div class="buttons-grid">

        <!--<button onclick="abrirSecao(1)">Lucro Esperado</button>

        <button onclick="abrirSecao(2)">Quanto tens de Investir</button>

        <button onclick="abrirSecao(3)"> Lucro % de crescimento</button>-->

        <button onclick="abrirSecao(4)">Quanto precisas Investir?</button>

        <button onclick="abrirSecao(5)">Simulador de Lucros</button>

        <button onclick="abrirSecao(6)">Consulta e Registo de Acções</button>

        <button onclick="abrirSecao(7)">Objetivos por Ativo</button>

      </div>

      <div class="buttons">

        <button onclick="goToScreen1()">Voltar</button>

      </div>

    </div>

    <!-- SECÇÃO 1 -->

    <div id="sec1Screen" class="screen hidden">

      <div class="section">

        <div class="section-header">

          <h4>1. Calcular Lucro com TP1, TP2, Investimento e Dividendos.</h4>

          <button class="info-toggle" onclick="toggleInfo('info1')">

            ℹ️ +info

          </button>

        </div>

        <div class="info-text" id="info1">

          <h5>

            Introduz o preço inicial da ação (TP1), o preço esperado no futuro

            (TP2), o valor que vais investir e os dividendos esperados. O

            resultado será o lucro total esperado.

          </h5>

        </div>

        <label

          >Preço Inicial (TP1): <input type="number" id="tp1\_1" step="0.01"

        /></label>

        <label

          >Preço Final (TP2): <input type="number" id="tp2\_1" step="0.01"

        /></label>

        <label

          >Valor Investido: <input type="number" id="invest\_1" step="0.01"

        /></label>

        <label

          >Dividendo por ação:

          <input type="number" id="dividendo\_1" step="0.01"

        /></label>

        <p>Lucro valorização: <span id="lucro\_valorizacao1">-</span></p>

        <p>

          Lucro total (incluindo dividendos): <span id="lucro\_total1">-</span>

        </p>

        <button onclick="calcularLucro1()">Calcular</button>

      </div>

      <div class="buttons">

        <button onclick="limparCampos()">Apagar Tudo</button>

        <button onclick="voltarMenu()">Voltar ao Menu</button>

      </div>

    </div>

    <!-- SECÇÃO 2 -->

    <div id="sec2Screen" class="screen hidden">

      <div class="section">

        <div class="section-header">

          <h4>2. Calcular Valor a Investir com TP1, TP2, Lucro Desejado.</h4>

          <button class="info-toggle" onclick="toggleInfo('info2')">

            ℹ️ +info

          </button>

        </div>

        <div class="info-text" id="info2">

          <h5>

            Introduz o preço inicial (TP1), o preço futuro esperado (TP2) e o

            lucro desejado. O resultado será o valor que precisas investir para

            atingir esse lucro.

          </h5>

        </div>

        <label

          >Preço Inicial (TP1): <input type="number" id="tp1\_2" step="0.01"

        /></label>

        <!--teste <label>Preço Final (TP2): <input type="number" id="tp2\_2" step="0.01" /></label> -->

        <label

          >Preço Final (TP2):

          <input

            type="number"

            id="tp2\_2"

            step="0.01"

            oninput="atualizarTP2\_2()"

          />

          % Crescimento: <span id="percent\_2">-</span></label

        >

        <label

          >Lucro Desejado: <input type="number" id="lucro\_2" step="0.01"

        /></label>

        <p>Investimento necessário: <span id="invest2">-</span></p>

        <button onclick="calcularInvestimento2()">Calcular</button>

      </div>

      <div class="buttons">

        <button onclick="limparCampos()">Apagar Tudo</button>

        <button onclick="voltarMenu()">Voltar ao Menu</button>

      </div>

    </div>

    <!-- SECÇÃO 3 -->

    <div id="sec3Screen" class="screen hidden">

      <div class="section">

        <div class="section-header">

          <h4>

            3. Calcular Lucro com TP1, % Crescimento, Investimento e Dividendos.

          </h4>

          <button class="info-toggle" onclick="toggleInfo('info3')">

            ℹ️ +info

          </button>

        </div>

        <div class="info-text" id="info3">

          <h5>

            Introduz o preço inicial (TP1), a percentagem de crescimento

            esperada, o valor que vais investir e os dividendos esperados. O

            resultado será o lucro total esperado.

          </h5>

        </div>

        <label

          >Preço Inicial (TP1): <input type="number" id="tp1\_3" step="0.01"

        /></label>

        <label

          >% Crescimento: <input type="number" id="percent\_3" step="0.01"

        /></label>

        <label

          >Valor Investido: <input type="number" id="invest\_3" step="0.01"

        /></label>

        <label

          >Dividendo por ação:

          <input type="number" id="dividendo\_3" step="0.01"

        /></label>

        <p>Lucro valorização: <span id="lucro\_valorizacao3">-</span></p>

        <p>

          Lucro total (incluindo dividendos): <span id="lucro\_total3">-</span>

        </p>

        <button onclick="calcularLucro3()">Calcular</button>

      </div>

      <div class="buttons">

        <button onclick="limparCampos()">Apagar Tudo</button>

        <button onclick="voltarMenu()">Voltar ao Menu</button>

      </div>

    </div>

    <!-- SECÇÃO 4 -->

    <div id="sec4Screen" class="screen hidden">

      <div class="section">

        <div class="section-header">

          <h4>1. Quanto precisas Investir?</h4>

          <button class="info-toggle" onclick="toggleInfo('info4')">

            ℹ️ +info

          </button>

        </div>

        <div class="info-text" id="info4">

          <h5>

            Introduz o preço inicial (TP1), a percentagem de crescimento

            esperada e o lucro desejado. O resultado será o valor que precisas

            investir.

          </h5>

        </div>

        <label

          >Preço Inicial (TP1):

          <input

            type="number"

            id="tp1\_4"

            step="0.01"

            oninput="atualizarTP2\_4()"

        /></label>

        <label

          >% Crescimento:

          <input

            type="number"

            id="percent\_4"

            step="0.01"

            oninput="atualizarTP2\_4()"

          />

          TP2: <span id="tp2\_4">-</span></label

        >

        <label

          >Lucro Desejado: <input type="number" id="lucro\_4" step="0.01"

        /></label>

        <p>Investimento necessário: <span id="invest4">-</span></p>

        <button onclick="calcularInvestimento4()">Calcular</button>

      </div>

      <div class="buttons">

        <button onclick="limparCampos()">Apagar Tudo</button>

        <button onclick="voltarMenu()">Voltar ao Menu</button>

      </div>

    </div>

    <!-- SECÇÃO 5 - Simulação (secção 2 agora)-->

    <div id="sec5Screen" class="screen hidden">

      <div class="section">

        <div class="section-header">

          <h4>2. Simulador de Lucros</h4>

        </div>

        <button class="info-toggle" onclick="toggleInfo('info5')">

          ℹ️ +info

        </button>

        <div class="info-text" id="info5">

          <h5>

            Este ecrã permite simular lucros com ações, incluindo dividendos,

            com visualização em tabela e gráfico. Oferece ainda funções de

            simulação rápida, cálculo de reforços de investimento e envio por

            email.

          </h5>

        </div>

        <div

          class="buttons"

          style="

            display: flex;

            flex-direction: column;

            gap: 10px;

            max-width: 300px;

            margin: auto;

          "

        >

          <button onclick="abrirPopupSimulacao()">Simular com Gráfico</button>

          <button onclick="abrirPopupReforco()">Simular Reforço</button>

          <button onclick="abrirSimuladorTP2()">Calcular TP2</button>

          <button onclick="abrirSimulador()">🔎 TOP 10 para Investir</button>

        </div>

        <div class="buttons">

          <button onclick="voltarMenu()">Voltar ao Menu</button>

        </div>

        <!-- POPUP SIMULAÇÃO COM GRÁFICO -->

        <div id="popupSimulacao" class="popup hidden">

          <div class="popup-content">

            <h3>📈 Simulador com Gráfico</h3>

            <input type="text" id="nomeAcao" placeholder="Nome da Ação" />

            <input type="number" id="tp1" placeholder="TP1" />

            <input type="number" id="tp2" placeholder="TP2" />

            <input

              type="number"

              id="investimento"

              placeholder="Investido (€)"

            />

            <input

              type="number"

              id="dividendo"

              placeholder="Dividendo por ação (€)"

            />

            <div

              id="botoesSimulacaoRapida"

              style="text-align: center; margin: 10px 0"

            >

              <button onclick="simularValorRapido(500)">💶 500</button>

              <button onclick="simularValorRapido(1000)">💶 1000</button>

              <button onclick="simularValorRapido(1500)">💶 1500</button>

              <button onclick="simularValorRapido(2000)">💶 2000</button>

            </div>

            <button onclick="simularEGUardar()">Simular com gráfico</button>

            <button onclick="somarLucros()">Somar Lucros</button>

            <table id="tabelaSimulacoes">

              <thead>

                <tr>

                  <th>Ação</th>

                  <th>TP1</th>

                  <th>TP2</th>

                  <th>Investido</th>

                  <th>Lucro</th>

                  <th>% Crescimento</th>

                </tr>

              </thead>

              <tbody></tbody>

            </table>

            <canvas id="graficoLucro" height="200"></canvas>

            <div class="buttons">

              <button onclick="limparCampos()">Limpar</button>

              <button onclick="limparGrafico()">🧹 Limpar Gráfico</button>

              <button onclick="abrirPopupEmail()">📧 Enviar Email</button>

              <button onclick="fecharPopupSimulacao()">❌ Fechar</button>

            </div>

          </div>

        </div>

        <!-- Popup Reforço -->

        <div id="popupReforco" class="popup hidden">

          <div class="popup-content">

            <h3>📈 Simular Reforço de Investimento</h3>

            <div class="form-grid">

              <input

                type="number"

                id="invest1"

                placeholder="Investimento Inicial (€)"

              />

              <input

                type="number"

                id="preco1"

                placeholder="Preço por Ação 1 (€)"

              />

              <input type="number" id="invest22" placeholder="Reforço (€)" />

              <input

                type="number"

                id="preco2"

                placeholder="Preço por Ação 2 (€)"

              />

            </div>

            <div id="resultadoReforco" class="resultado-popup"></div>

            <div class="buttons">

              <button onclick="calcularMediaPonderada()">Calcular</button>

              <button onclick="limparCampos()">Limpar</button>

              <button onclick="fecharPopupReforco()">Fechar</button>

            </div>

          </div>

        </div>

        <!-- Popup TP2 -->

        <div id="popupTP2" class="popup hidden">

          <div class="popup-content">

            <h3>🎯 Calcular TP2</h3>

            <input

              type="number"

              id="tp1Input"

              placeholder="TP1 (preço de compra €)"

            />

            <input

              type="number"

              id="investimentoInput"

              placeholder="Valor Investido (€)"

            />

            <input

              type="number"

              id="lucroDesejadoInput"

              placeholder="Lucro Desejado (€)"

            />

            <div class="buttons">

              <button onclick="calcularTP2()">Calcular TP2</button>

              <button onclick="limparCampos()">Limpar</button>

              <button onclick="fecharSimuladorTP2()">Fechar</button>

            </div>

            <div id="resultadoTP2" class="resultado-popup"></div>

          </div>

        </div>

        <!-- Popup TOP 10 -->

        <div id="simuladorModal" class="popup hidden">

          <div class="popup-content">

            <h3>🔎 Top 10 para Investir</h3>

            <label>Montante a investir (€)</label>

            <input

              type="number"

              id="inputInvestimento"

              placeholder="Ex: 1500"

            />

            <label>Lucro desejado (€)</label>

            <input type="number" id="inputLucro" placeholder="Ex: 100" />

            <label>Crescimento estimado (%) <small>(opcional)</small></label>

            <input type="number" id="inputCrescimento" placeholder="Ex: 5" />

            <div class="buttons">

              <button onclick="simular()">Simular</button>

              <button onclick="limparCampos()">Limpar</button>

              <button onclick="fecharSimulador()">Fechar</button>

            </div>

            <div id="resultadoSimulacao" class="resultado-container"></div>

          </div>

        </div>

        <!-- Popup Email -->

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

          <h3>Enviar Simulação</h3>

          <div id="dadosSimulacao"></div>

          <div class="email-form">

            <input

              type="email"

              id="emailDestino"

              placeholder="Email de destino"

              required

            />

          </div>

          <div class="botoes-popup">

            <button onclick="enviarEmail()">Enviar</button>

            <button onclick="limparCampos()">Limpar</button>

            <button onclick="fecharPopupEmail()">Cancelar</button>

          </div>

        </div>

      </div>

    </div>

    <!-- SECÇÃO 6: Registo e Filtro de Ações -->

    <div id="sec6Screen" class="screen hidden">

      <!-- REGISTO -->

      <div class="section">

        <h3>4.Registo e Consulta de Acções c/ Dividendos</h3>

        <div class="form-grid">

          <input type="search" id="nomeAcaoReg" placeholder="Nome da Ação" />

          <input type="search" id="tickerAcaoReg" placeholder="Ticker" />

          <select id="Setor" placeholder="Setor">

            <option value="">Selecione um Setor</option>

            <option value="ETF iTech">ETF iTech</option>

            <option value="ETF Finance">ETF Finance</option>

            <option value="ETF Energia">ETF Energia</option>

            <option value="ETF Materiais">ETF Materiais</option>

            <option value="Alimentção">Alimentação</option>

            <option value="Automovel">Automóvel</option>

            <option value="Bens de Consumo">Bens de Consumo</option>

            <option value="Consumo Cíclico">Consumo Cíclico</option>

            <option value="Consumo Defensivo">Consumo Defensivo</option>

            <option value="Defesa">Defesa</option>

            <option value="Energia">Energia</option>

            <option value="Finanças">Finanças</option>

            <option value="Imobiliário">Imobiliário</option>

            <option value="Indústria">Indústria</option>

            <option value="Infraestruturas / Energia">

              Infraestruturas / Energia

            </option>

            <option value="Materiais">Materiais</option>

            <option value="Restauração">Restauração</option>

            <option value="Saúde">Saúde</option>

            <option value="Tecnologia">Tecnologia</option>

            <option value="Telecomunicações">Telecomunicações</option>

          </select>

          <select id="Mercado" placeholder="Mercado">

            <option value="">Seleciona o Mercado</option>

            <option value="Portugal">Portugal</option>

            <option value="Europeu">Europeu</option>

            <option value="Americano">Americano</option>

            <option value="Americano SP500">Americano SP500</option>

          </select>

          <input

            type="number"

            id="valorStock"

            step="0.01"

            placeholder="Valor da Ação (€)"

          />

          <input

            type="number"

            id="valorDividendoReg"

            step="0.01"

            placeholder="Valor do Dividendo (€)"

          />

          <select id="mesDividendoReg">

            <option value="">Seleciona o mês</option>

            <option value="n/A">n/A</option>

            <option value="Janeiro">Janeiro</option>

            <option value="Fevereiro">Fevereiro</option>

            <option value="Março">Março</option>

            <option value="Abril">Abril</option>

            <option value="Maio">Maio</option>

            <option value="Junho">Junho</option>

            <option value="Julho">Julho</option>

            <option value="Agosto">Agosto</option>

            <option value="Setembro">Setembro</option>

            <option value="Outubro">Outubro</option>

            <option value="Novembro">Novembro</option>

            <option value="Dezembro">Dezembro</option>

          </select>

          <select id="periodicidade" placeholder="Periodicidade">

            <option value="">Seleciona a Frequência</option>

            <option value="n/A">n/A</option>

            <option value="Anual">Anual</option>

            <option value="Semestral">Semestral</option>

            <option value="Trimestral">Trimestral</option>

            <option value="Mensal">Mensal</option>

          </select>

        </div>

        <div class="buttons">

          <button onclick="guardarOuAtualizarAcaoFirebase()">

            💾 Guardar / Atualizar

          </button>

        </div>

      </div>

      <!-- BOTÃO PARA ABRIR POPUP PESQUISA DE ACÇÕES -->

      <div class="section">

        <div class="section-header">

          <h4>🔍 Pesquisar por Ações</h4>

          <button onclick="abrirPopupFiltro()">🔍 Abrir Pesquisa</button>

        </div>

      </div>

      <div class="buttons">

        <button onclick="limparCampos()">Apagar Tudo</button>

        <button onclick="voltarMenu()">Voltar ao Menu</button>

      </div>

    </div>

        <!--SECÇÃO 7 REGISTO DE COMPRAS -->

        <div id="sec7Screen" class="screen hidden">

          <h2>🎯 Definir Objetivo por Ativo</h2>

          <input id="nomeAtivo" placeholder="Nome do ativo" />

          <input id="tickerAtivo" placeholder="Ticker" />

          <select id="tipoAtivo">

            <option value="ação">Ação</option>

            <option value="ETF">ETF</option>

          </select>

          <input

            id="precoCompra"

            type="number"

            step="0.01"

            placeholder="Preço de compra (€)"

          />

          <input

            id="quantidade"

            type="number"

            step="0.01"

            placeholder="Quantidade comprada"

          />

          <input id="setorCompra" placeholder="Setor" />

          <input id="mercadoCompra" placeholder="Mercado" />

          <label for="tipoObjetivo">Tipo de Objetivo:</label>

          <select id="tipoObjetivo">

            <option value="valorFinal">Valor Final Alvo (€)</option>

            <option value="lucro">Lucro Alvo (€)</option>

            <option value="quantidade">Quantidade Alvo (unidades)</option>

          </select>

          <input

            id="objetivoFinanceiro"

            type="number"

            step="0.01"

            placeholder="Valor do objetivo"

          />

          <button onclick="guardarCompraAcaoOuEtf()">💾 Guardar Compra e Objetivo</button>

          <button onclick="abrirPopupProgresso()">📈 Ver Progresso dos Ativos</button>

          <button onclick="voltarMenu()">Voltar ao Menu</button>

        </div>

        <!-- POPUP -->

          <!-- Popup -->

              <div id="popupProgresso" class="modal hidden">

                <div class="modal-content">

                  <span class="close" onclick="fecharPopupProgresso()">&times;</span>

                  <h3>📊 Progresso dos Ativos</h3>

                  <div id="listaProgressoAtivos" class="tabela-wrapper">

                    <table class="tabela-progresso">

                      <thead>

                        <tr>

                          <th>Nome</th>

                          <th>Ticker</th>

                          <th>Preço Compra (€)</th>

                          <th>Qtd</th>

                          <th>Objetivo</th>

                          <th>TP2 (€)</th>

                          <th>Atual (€)</th>

                          <th>Progresso</th>

                        </tr>

                      </thead>

                      <tbody>

                        <!-- Dados dinâmicos serão inseridos aqui via JS -->

                      </tbody>

                    </table>

                  </div>

                </div>

              </div>

              <!-- POPUP FILTRO AÇÕES -->

              <div id="popupFiltro" class="popup hidden">

                <div class="popup-content">

                  <h4>🔍 Filtros</h4>

                  <input

                    type="search"

                    id="filtroNome"

                    placeholder="🔠 Filtrar por Nome da Ação"

                    oninput="filtrarAcoes()"

                    style="

                      margin-bottom: 10px;

                      width: 100%;

                      padding: 8px;

                      border-radius: 5px;

                    "

                  />

                  <input

                    type="search"

                    id="filtroTicker"

                    placeholder="🔠 Filtrar por Ticker"

                    oninput="filtrarAcoes()"

                    style="

                      margin-bottom: 15px;

                      width: 100%;

                      padding: 8px;

                      border-radius: 5px;

                    "

                  />

                  <select id="filtroSetor">

                    <option value="">Selecione um Setor</option>

                    <option value="ETF iTech">ETF iTech</option>

                    <option value="ETF Finance">ETF Finance</option>

                    <option value="ETF Energia">ETF Energia</option>

                    <option value="ETF Materiais">ETF Materiais</option>

                    <option value="Alimentção">Alimentação</option>

                    <option value="Automovel">Automóvel</option>

                    <option value="Bens de Consumo">Bens de Consumo</option>

                    <option value="Consumo Cíclico">Consumo Cíclico</option>

                    <option value="Consumo Defensivo">Consumo Defensivo</option>

                    <option value="Defesa">Defesa</option>

                    <option value="Energia">Energia</option>

                    <option value="Finanças">Finanças</option>

                    <option value="Imobiliário">Imobiliário</option>

                    <option value="Indústria">Indústria</option>

                    <option value="Infraestruturas / Energia">

                      Infraestruturas / Energia

                    </option>

                    <option value="Materiais">Materiais</option>

                    <option value="Restauração">Restauração</option>

                    <option value="Saúde">Saúde</option>

                    <option value="Tecnologia">Tecnologia</option>

                    <option value="Telecomunicações">Telecomunicações</option>

                  </select>

                  <select id="filtroPeriodicidade">

                    <option value="">Selecione a Periodicidade</option>

                    <option value="n/A">n/A</option>

                    <option value="Mensal">Mensal</option>

                    <option value="Trimestral">Trimestral</option>

                    <option value="Semestral">Semestral</option>

                    <option value="Anual">Anual</option>

                  </select>

                  <select id="filtroMercado">

                    <option value="">Seleciona o Mercado</option>

                    <option value="Portugal">Portugal</option>

                    <option value="Europeu">Europeu</option>

                    <option value="Americano">Americano</option>

                    <option value="Americano SP500">Americano SP500</option>

                  </select>

                  <select id="filtroMes">

                    <option value="">Selecione um Mês</option>

                    <option value="n/A">n/A</option>

                    <option value="Janeiro">Janeiro</option>

                    <option value="Fevereiro">Fevereiro</option>

                    <option value="Março">Março</option>

                    <option value="Abril">Abril</option>

                    <option value="Maio">Maio</option>

                    <option value="Junho">Junho</option>

                    <option value="Julho">Julho</option>

                    <option value="Agosto">Agosto</option>

                    <option value="Setembro">Setembro</option>

                    <option value="Outubro">Outubro</option>

                    <option value="Novembro">Novembro</option>

                    <option value="Dezembro">Dezembro</option>

                  </select>

                  <div class="buttons">

                    <button onclick="filtrarAcoes()">🔎 Aplicar Filtros</button>

                    <button onclick="fecharPopupFiltro()">❌ Fechar</button>

                    <button onclick="prepararSimulacaoBloco()">

                      Simular Ações Selecionadas

                    </button>

                  </div>

                  <!-- Checkbox Selecionar Tudo -->

                  <label>

                    <input

                      type="checkbox"

                      id="checkboxSelecionarTudo"

                      onchange="selecionarTudoCheckboxes(this)"

                    />

                    Selecionar Todos

                  </label>

                  <!--prepararSimulacaoBloco-->

                  <div id="resultadoFiltroMes" class="resultado-popup"></div>

                </div>

              </div>

    <!-- SIMULAÇÃO EM BLOCO COMO POPUP -->

    <div id="popupSimulacaoBloco" class="popup hidden">

      <div class="popup-content">

        <h3>📊 Simulação em Bloco</h3>

        <label for="periodoCrescimento">Período de Crescimento:</label>

        <select id="periodoCrescimento">

          <option value="taxaCrescimento\_1s">1 Semana</option>

          <option value="taxaCrescimento\_1m">1 Mês</option>

          <option value="taxaCrescimento\_1a">1 Ano</option>

        </select>

        <div style="margin-bottom: 1em">

          <label for="investimentoTotal"

            ><strong>💰 Investimento Total (€):</strong></label

          >

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

          <div

            style="

              margin-top: 0.5em;

              display: flex;

              gap: 1em;

              align-items: center;

            "

          >

            <label>

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

              Usar todo o investimento

            </label>

            <label>

              <input type="checkbox" id="acoesCompletas" />

              Apenas ações completas

            </label>

          </div>

        </div>

        <div id="resultadoDistribuicao" class="tabela-wrapper"></div>

        <div id="tabelaAcoesSelecionadas" class="tabela-wrapper">

          <table class="tabela-bloco">

            <thead>

              <tr>

                <th>Ação</th>

                <th>Ticker</th>

                <th>Preço Atual (€)</th>

                <th>Dividendo (€)</th>

              </tr>

            </thead>

            <tbody></tbody>

          </table>

        </div>

        <div class="buttons">

          <!--<button onclick="calcularDistribuicao()">

            📈 Calcular Lucro Máximo

          </button>-->

          <button onclick="fecharPopupSimulacaoBloco()">❌ Fechar</button>

        </div>

      </div>

    </div>

    <script src="script.js"></script>

    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

  </body>

</html>

CSS

  .buttons-grid {

    display: flex;

    gap: 10px;

    flex-wrap: wrap;

    justify-content: space-between;

    margin-bottom: 20px;

    text-align: center;

    width: 100%;

    /\* <- antes estava 250px, estava a limitar os botões \*/

  }

  /\*botões top\*/

  .botoes {

  display: flex;

  gap: 4px;

}

.botoes button {

  width: 80px;

  height: 40px;

  padding: 0;

  border: none;

  background-color: #272828;

  color: white;

  font-size: 1em;

  font-family: sans-serif;

  border-radius: 4px;

  display: flex;

  align-items: center;

  justify-content: center;

  text-align: center;

  overflow: hidden;

  white-space: nowrap;

  text-overflow: ellipsis;

  transition: background-color 0.3s;

}

.botoes button:hover {

  background-color: #38393a;

}

.botoes button span {

  font-size: calc(6px + 0.3vw);

}

  /\*Top 10 empresas em crescimento\*/

  .top10-box {

  background-color: #f5f8fa;

  border: 1px solid #ddd;

  border-radius: 10px;

  padding: 15px;

  margin: 20px 0;

  max-height: 300px;

  overflow-y: auto;

}

.top10-box h3 {

  margin-bottom: 10px;

  font-size: 1.2rem;

}

#listaTop10 li {

  list-style: none;

  padding: 6px 0;

  border-bottom: 1px dashed #ccc;

  font-size: 0.95rem;

  color: #2c3e50;

}

#topAcoesContainer {

  background: #f9f9f9;

  border-radius: 8px;

  padding: 15px;

  margin-top: 20px;

}

#topAcoesContainer ul {

  list-style: none;

  padding-left: 0;

}

#topAcoesContainer li {

  margin: 5px 0;

  padding: 6px 10px;

  background: #fff;

  border: 1px solid #ddd;

  border-radius: 5px;

}

  .buttons-grid button {

    flex: 1 1 100%;

    max-width: 100%;

    padding: 15px;

    font-size: 1rem;

    border-radius: 60px;

    background: #626466;

    border: none;

    color: white;

    cursor: pointer;

    text-align: center;

    white-space: normal;

    /\* permite quebra de linha se o texto for longo \*/

  }

  .buttons-grid button:hover {

    background: #323234;

  }

  body {

    font-family: Arial, sans-serif;

    background: #ffffff;

    color: #fff;

    margin: 0;

    padding: 20px;

    display: flex;

    justify-content: center;

    align-items: flex-start;

    min-height: 100vh;

  }

  .screen {

    background: #fcfcfc;

    border-radius: 10px;

    box-sizing: border-box;

    box-shadow: 0 0 12px rgba(0, 0, 0, 0.3);

    width: 100%;

    max-width: 600px;/\* ou 600px para tablets \*/

    margin: 0 auto;

    padding: 20px;

  }

  /\* Estilo para ecrãs pequenos (telemóveis) \*/

  /\*@media (max-width: 600px) {

    .screen {

      padding: 15px;

      max-width: 100%;

    }

    .form-grid input,

    .form-grid select {

      font-size: 1rem;

    }

    button {

      font-size: 0.95rem;

      padding: 10px 12px;

    }

  }\*/

  .hidden {

    display: none;

  }

  h1,

  h2 {

    color: #141515;

    margin-bottom: 4px;

  }

  h3,

  h5,

  h6,

  h4 {

    color: #0e0f0f;

    margin-bottom: 0px;

  }

  label {

    display: block;

    margin-bottom: 8px;

    color: #131313;

  }

  input[type="number"] {

    width: 100%;

    padding: 6px 8px;

    border-radius: 6px;

    border: none;

    margin-top: 3px;

    margin-bottom: 12px;

    box-sizing: border-box;

  }

  button {

    background-color: #1c1c1d;

    border: none;

    color: white;

    padding: 10px 20px;

    margin: 8px 6px 0 0;

    border-radius: 60px;

    cursor: pointer;

    font-size: 16px;

    transition: background-color 0.25s ease;

    width: 250px;

  }

  button:hover {

    background-color: #383a3b;

  }

  p {

    margin: 6px 0 10px;

    font-weight: bold;

    color: #313334;

  }

  .buttons {

    margin-top: 15px;

    width: 100%;

    display: flex;

    justify-content: center;/\* ou 'flex-end' se quiseres à direita \*/

  }

  .buttons button {

    width: 250px;

    padding: 20px;

    font-size: 1rem;

    border-radius: 10px;

    background: #212122;

    border: none;

    color: white;

    cursor: pointer;

    text-align: center;

  }

  .buttons button:hover {

    background: #505152;

  }

  label input {

    margin-right: 10px;

    color: #121212;

  }

  #tp2\_4 {

    font-weight: bold;

    margin-left: 5px;

  }

  .info-text {

    display: none;

    background-color: #f3f4f5;

    padding: 0.1rem;

    margin: 0.5rem 0;

    border-left: 4px solid rgb(34, 34, 34);

    border-radius: 4px;

    font-size: 0.9rem;

    color: #343636;

  }

  .section-header {

    display: fixed;

    justify-content: space-between;

    align-items: center;

  }

  .section-header h3 {

    margin: 0;

    font-size: 1.1rem;

    font-weight: bold;

    color: rgb(25, 24, 24);

  }

  .info-toggle {

    background: none;

    border: none;

    color: #3e4447;

    font-size: 0.9rem;

    cursor: pointer;

    display: flex;

    align-items: center;

    gap: 4px;

  }

  .section {

    background-color: #ffffff;

    padding: 20px;

    border-radius: 10px;

    margin-bottom: 30px;/\* 👈 Espaçamento entre as secções \*/

  }

  .tabela-scroll-wrapper {

    overflow-x: auto;

    width: 100%;

    margin-top: 1rem;

    border-radius: 8px;

    background: white;

  }

  #tabelaSimulacoes {

    border-collapse: collapse;

    background-color: #ecf0f1;

    font-size: 0.95em;

    min-width: 700px; /\* força scroll horizontal se necessário \*/

    width: 100%;

  }

  #tabelaSimulacoes th,

  #tabelaSimulacoes td {

    padding: 12px 16px;

    border-bottom: 1px solid #bdc3c7;

    white-space: nowrap;

  }

  #tabelaSimulacoes thead {

    background-color: #dadee1;

    color: #121212;

  }

  #tabelaSimulacoes tbody tr {

    background-color: #ffffff;

    color: #2c3e50;

  }

  #tabelaSimulacoes tbody tr:hover {

    background-color: #d6eaf8;

  }

  /\*Botão Limpar Gráfico\*/

  .grafico-wrapper {

    display: flex;

    justify-content: center; /\* Centraliza os botões horizontalmente \*/

    flex-wrap: wrap;         /\* Permite que os botões se ajustem em telas pequenas \*/

    gap: 10px;               /\* Espaço entre os botões \*/

    margin: 20px auto;

    max-width: 100%;

  }

  .grafico-wrapper button {

    padding: 10px 15px;

    font-size: 14px;

    background-color: black;

    color: white;

    border: none;

    border-radius: 5px;

    cursor: pointer;

  }

  #limparGraficoBtn {

    position: absolute;

    bottom: 10px;

    right: 10px;

    background: none;

    border: none;

    font-size: 20px;

    cursor: pointer;

    color: #ffffff;

    /\* Branco para sobressair \*/

  }

  #limparGraficoBtn:hover {

    color: #ff5555;

    /\* Fica vermelho ao passar o rato \*/

  }

  /\* Estilo geral do popup \*/

  #popupEmail {

    position: fixed;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    background-color: #ffffff;

    border: 2px solid #424243;

    border-radius: 10px;

    padding: 20px;

    width: 90%;

    max-width: 400px;

    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);

    z-index: 1000;

  }

  /\*ENVIAR EMAIL\*/

  /\* Estilo geral do popup \*/

  #popupEmail {

    position: fixed;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    background-color: #ffffff;

    border: 2px solid #242424;

    border-radius: 10px;

    padding: 20px;

    width: 90%;

    max-width: 400px;

    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);

    z-index: 1000;

  }

  /\* Título do popup \*/

  #popupEmail h3 {

    margin-top: 0;

    font-size: 1.4em;

    color: #151515;

    text-align: center;

  }

  /\* Campo de email \*/

  #popupEmail input[type="email"] {

    width: 100%;

    padding: 10px;

    margin-top: 10px;

    margin-bottom: 15px;

    border: 1px solid #3b3939;

    border-radius: 5px;

    font-size: 1em;

  }

  /\* Botões \*/

  #popupEmail button {

    padding: 10px 15px;

    margin: 5px;

    border: none;

    border-radius: 5px;

    font-size: 1em;

    cursor: pointer;

  }

  #popupEmail button:first-of-type {

    background-color: #2d2e30;

    color: white;

  }

  #popupEmail button:last-of-type {

    background-color: #ccc;

    color: #333;

  }

  /\* Conteúdo das simulações \*/

  #dadosSimulacao {

    background-color: #f0f5f6;

    color: #383636;

    padding: 10px;

    border-radius: 5px;

    max-height: 200px;

    overflow-y: auto;

    font-size: 0.95em;

    line-height: 1.5em;

    border: 1px solid #ddd;

    margin-bottom: 10px;

  }

  #dadosSimulacao p {

    margin: 6px 0;

  }

  #dadosSimulacao p strong {

    color: #007BFF;

  }

  #dadosSimulacao hr {

    border: none;

    border-top: 1px solid #ccc;

    margin: 10px 0;

  }

  /\* Ocultar popup \*/

  .hidden {

    display: none;

  }

  /\* Caixa de entrada de email \*/

  #popupEmail .email-form {

    margin-bottom: 15px;

    text-align: center;

  }

  #popupEmail input[type="email"] {

    width: 95%;

    max-width: 100%;

    padding: 10px;

    border: 1px solid #ccc;

    border-radius: 5px;

    font-size: 1em;

  }

  /\* Alinhamento central dos botões \*/

  #popupEmail .botoes-popup {

    display: flex;

    justify-content: center;

    gap: 10px;

  }

  /\* Secção 6 - Registo de Ações \*/

  #sec6Screen .section {

    display: flex;

    flex-direction: column;

    gap: 12px;

  }

  #sec6Screen label {

    display: flex;

    flex-direction: column;

    font-weight: bold;

    color: #ecf0f1;

  }

  #sec6Screen input,

  #sec6Screen select,

  #sec6Screen textarea {

    padding: 8px;

    border-radius: 6px;

    border: none;

    font-size: 1em;

    margin-top: 4px;

    box-sizing: border-box;

  }

  #sec6Screen textarea {

    resize: vertical;

  }

  /\* Botão para mostrar filtros \*/

  #btnMostrarFiltros {

    background-color: #2c2e2e;

    color: white;

    padding: 10px;

    border: none;

    border-radius: 6px;

    cursor: pointer;

    margin-top: 10px;

    transition: background-color 0.3s ease;

  }

  #btnMostrarFiltros:hover {

    background-color: #3d3f3f;

  }

  /\* Container dos botões de mês \*/

  #filtrosMesContainer {

    display: flex;

    flex-wrap: wrap;

    gap: 8px;

    margin-top: 15px;

    justify-content: center;

    animation: fadeIn 0.4s ease-in-out;

  }

  #filtrosMesContainer.hidden {

    display: none;

  }

  #filtrosMesContainer button {

    flex: 1 1 30%;

    padding: 10px;

    background-color: #2c3e50;

    color: white;

    border: none;

    border-radius: 6px;

    cursor: pointer;

    font-size: 0.9em;

    transition: background-color 0.3s ease;

  }

  #filtrosMesContainer button:hover {

    background-color: #1a252f;

  }

  /\* Resultado do filtro \*/

  #resultadoFiltroMes {

    margin-top: 20px;

    background-color: #f3f0f0;

    color: #101212;

    padding: 12px;

    border-radius: 8px;

    max-height: 400px;

    overflow-y: auto;

    font-size: 0.95em;

    line-height: 1.5em;

  }

  /\* Animação suave \*/

  /\* Secção 6 - Registo e Filtro \*/

  #sec6Screen .form-grid {

    display: grid;

    grid-template-columns: 1fr;

    gap: 12px;

  }

  #sec6Screen .filtros-container {

    display: flex;

    flex-direction: column;

    gap: 10px;

    margin-bottom: 10px;

  }

  #sec6Screen label {

    display: flex;

    flex-direction: column;

    font-weight: bold;

    color: #ecf0f1;

  }

  #sec6Screen input,

  #sec6Screen select {

    padding: 8px;

    border-radius: 6px;

    border: none;

    font-size: 1em;

    margin-top: 4px;

    box-sizing: border-box;

  }

  #resultadoFiltroMes.resultado-acoes {

    margin-top: 15px;

    background-color: #f6f6f6;

    color: #141414;

    padding: 12px;

    border-radius: 8px;

    max-height: 300px;

    overflow-y: auto;

    font-size: 0.95em;

    line-height: 1.5em;

  }

  #resultadoFiltroMes ul {

    padding-left: 20px;

  }

  #resultadoFiltroMes li {

    margin-bottom: 10px;

    border-bottom: 1px solid #000000;

    padding-bottom: 6px;

  }

  @keyframes fadeIn {

    from {

      opacity: 0;

      transform: translateY(-10px);

    }

    to {

      opacity: 1;

      transform: translateY(0);

    }

  }

  .form-grid {

    display: grid;

    grid-template-columns: 1fr 1fr;

    gap: 10px;

  }

  .section-header {

    display: flex;

    justify-content: space-between;

    align-items: center;

  }

  \* {

    box-sizing: border-box;

  }

  /\*adicionado periência\*/

  /\*Tornar tudo responsivo e centrado na largura total\*/

  body, html {

    margin: 0;

    padding: 0;

    font-family: 'Segoe UI', sans-serif;

    background-color: #f7f9fa;

    color: #212121;

    width: 100%;

    min-height: 100vh;

    box-sizing: border-box;

  }

  /\* Garante que todas as "screens" ocupam a largura total \*/

  .screen {

    padding: 16px;

    width: 100%;

    max-width: 768px;

    margin: 0 auto;

  }

  /\*Melhorar o espaçamento e separadores entre secções e títulos\*/

  .section {

    margin-bottom: 32px;

    padding: 16px;

    background-color: #ffffff;

    border-radius: 12px;

    box-shadow: 0 2px 6px rgba(0,0,0,0.05);

  }

  /\* Títulos \*/

  .section h3, .section h4 {

    margin-bottom: 16px;

    font-size: 1.25rem;

    color: #2c3e50;

    border-bottom: 1px solid #dcdcdc;

    padding-bottom: 6px;

  }

  /\*Melhorar campos de input e etiquetas\*/

  .section label {

    font-weight: 600;

    font-size: 0.95rem;

    color: #333;

    margin-bottom: 6px;

  }

  .section input[type="text"],

  .section input[type="number"],

  .section select {

    width: 100%;

    padding: 10px;

    margin-bottom: 12px;

    border: 1px solid #ccc;

    border-radius: 6px;

    font-size: 1rem;

  }

  /\*Organizar botões e formulários responsivamente\*/

  button {

    padding: 10px 16px;

    font-size: 0.95rem;

    border: none;

    border-radius: 6px;

    margin: 6px 4px 0 0;

    cursor: pointer;

    transition: background-color 0.3s;

  }

  button:hover {

    opacity: 0.85;

  }

  .buttons {

    display: flex;

    flex-wrap: wrap;

    gap: 10px;

  }

  .buttons button {

    flex: 1 1 45%;

  }

  .form-grid {

    display: grid;

    grid-template-columns: 1fr;

    gap: 12px;

  }

  /\*Melhorar visual de listagens e resultados\*/

  .resultado-acoes {

    background-color: #f0f4f8;

    padding: 12px;

    border-radius: 8px;

    margin-top: 12px;

    font-size: 0.9rem;

    color: #1c2833;

  }

  .hidden {

    display: none;

  }

  .resultado-acoes ul {

    padding-left: 20px;

  }

  .resultado-acoes li {

    margin-bottom: 12px;

    padding: 10px;

    background: #ffffff;

    border: 1px solid #e9e9e9;

    border-radius: 6px;

    box-shadow: 0 1px 2px rgba(0,0,0,0.03);

  }

  /\*Mobile-first ajustes e boas práticas\*/

  @media (min-width: 600px) {

    .form-grid {

      grid-template-columns: repeat(2, 1fr);

    }

    .buttons {

      justify-content: space-between;

    }

  }

  /\* 💡 Cor do texto das labels na secção 6 \*/

  #sec6Screen label {

    color: #d7d6d6; /\* Podes alterar aqui a cor do texto das etiquetas \*/

  }

  /\* 💡 Cor dos botões principais \*/

  button {

    background-color: #050505; /\* Azul principal, muda aqui para verde ou outro \*/

    border-radius: 50px;

  }

  .popup.show {

    display: flex;

  }

  .popup-content select,

  .popup-content button {

    width: 100%;

    margin-bottom: 10px;

  }

 .resultado-popup {

  max-height: 150px;

  overflow-y: auto;

  padding: 10px;

  margin: 15px auto;

  border: 1px solid #ccc;

  border-radius: 8px;

  background-color: #f9f9f9;

  width: 100%;

  max-width: 600px;

  box-sizing: border-box;

  font-size: 0.95em;

}

.resultado-popup table {

  width: 100%;

  border-collapse: collapse;

}

.resultado-popup th,

.resultado-popup td {

  border: 1px solid #ddd;

  padding: 8px;

  text-align: left;

}

.resultado-popup th {

  background-color: #f0f0f0;

  font-weight: bold;

}

.resultado-popup tr:nth-child(even) {

  background-color: #f9f9f9;

}

  /\*botões dentro do filtro\*/

  .botoes-acoes {

    display: flex;

    gap: 6px;

    margin-top: 4px;

  }

  .botoes-acoes button {

    background-color: #111111;

    border: none;

    border-radius: 4px;

    padding: 6px;

    font-size: 16px;

    cursor: pointer;

    transition: background-color 0.2s;

  }

  .botoes-acoes button:hover {

    background-color: #515050;

  }

  /\*Popup Reforço\*/

  .popup {

    display: flex;

    align-items: center;

    position: fixed;

    top: 0;

    left: 0;

    width: 100vw;

    height: 100vh;

    background-color: rgba(0, 0, 0, 0.6);

    justify-content: center;

    z-index: 9999;

  }

  /\*container top10\*/

  .resultado-container {

    max-height: 1000vh; /\* ou outro valor, consoante o teu layout \*/

    overflow-y: auto;

    padding-right: 10px; /\* espaço para scroll \*/

    margin-top: 1rem;

  }

  .popup-content {

    padding: 20px;

    margin: 50px auto;

    position: relative;

    max-height: 90vh;

    background: white;

    width: 90vw;

    max-width: 600px;

    border-radius: 10px;

    overflow-y: auto;

    display: flex;

    flex-direction: column;

  }

  .popup:not(.hidden) {

    display: block;

  }

  .hidden {

    display: none !important;

  }

.tabela-wrapper {

  overflow-x: auto;

  margin-top: 10px;

  border-radius: 8px;

  border: 1px solid #ccc;

  background-color: #f9f9f9;

  padding: 10px;

  box-sizing: border-box;

  max-width: 100%;

  font-size: 0.95em;

}

.tabela-wrapper table {

  width: 100%;

  border-collapse: collapse;

}

.tabela-wrapper th,

.tabela-wrapper td {

  border: 1px solid #ddd;

  padding: 8px;

  text-align: left;

}

.tabela-wrapper th {

  background-color: #f0f0f0;

  font-weight: bold;

}

.tabela-wrapper tr:nth-child(even) {

  background-color: #f9f9f9;

}

  .tabela-bloco {

    width: 100%;

    border-collapse: collapse;

    background-color: #f8f9fa;

    font-size: 0.95em;

    min-width: 500px;

    border-radius: 8px;

  }

  .tabela-bloco thead {

    background-color: #e0e0e0;

    color: #333;

  }

  .tabela-bloco th,

  .tabela-bloco td {

    padding: 12px 16px;

    text-align: left;

    border-bottom: 1px solid #ccc;

    white-space: nowrap;

  }

  .tabela-bloco tbody tr:hover {

    background-color: #eef5ff;

  }

  /\*investimento completo ou parcelar\*/

  input[type="checkbox"] {

    transform: scale(1.2);

    margin-right: 0.4em;

    cursor: pointer;

  }

  label {

    font-family: Arial, sans-serif;

    font-size: 0.95em;

  }

  table {

    width: 100%;

    border-collapse: collapse;

    margin-top: 1em;

  }

  th, td {

    border: 1px solid #ccc;

    padding: 0.5em;

    text-align: center;

  }

  th {

    background-color: #f0f0f0;

  }

  #resultadoDistribuicao {

  position: relative;

  overflow: visible;

  z-index: 0;

}

#resultadoDistribuicao table {

  margin-bottom: 1em;

  z-index: 1;

}

#resultadoDistribuicao p {

  margin-top: 0.5em;

  font-size: 1.1em;

  font-weight: bold;

  color: #333;

}

/\*Popup Analise dos ativos comprados\*/

.modal {

  position: fixed;

  z-index: 9999;

  left: 0;

  top: 0;

  width: 100%;

  height: 100%;

  overflow: auto;

  background-color: rgba(0,0,0,0.5);

}

.modal-content {

  background-color: #fff;

  margin: 10% auto;

  padding: 20px;

  border-radius: 10px;

  width: 90%;

  max-width: 600px;

  box-shadow: 0 0 10px #444;

}

.close {

  float: right;

  font-size: 24px;

  cursor: pointer;

}

#sec7Screen button {

  display: inline-block !important;

  visibility: visible !important;

  opacity: 1 !important;

}

/\*tabela progressos\*/

.tabela-ativos {

  width: 100%;

  border-collapse: collapse;

  margin-top: 10px;

}

.tabela-ativos th,

.tabela-ativos td {

  border: 1px solid #ccc;

  padding: 8px;

  text-align: center;

}

.tabela-ativos th {

  background-color: #f4f4f4;

}

.tabela-ativos td {

  background-color: #fff;

}

JS

// Firebase Configurações (substituir pelos teus dados reais)

const firebaseConfig = {

  apiKey: "AIzaSyDbSYjVwsOOnBjZe\_X8y7gS-W4DhYqHEnE",

  authDomain: "appfinance-812b2.firebaseapp.com",

  projectId: "appfinance-812b2",

  storageBucket: "appfinance-812b2.firebasestorage.app",

  messagingSenderId: "383837988480",

  appId: "1:383837988480:web:dd114574838c6a9dbb2865",

};

// Registo do Service Worker (PWA)

if ("serviceWorker" in navigator) {

  navigator.serviceWorker

    .register("sw.js")

    .then((reg) => console.log("SW registado!", reg))

    .catch((err) => console.error("SW falhou:", err));

}

// Inicializar Firebase

firebase.initializeApp(firebaseConfig);

const db = firebase.firestore();

// Navegação entre screens

function goToScreen2() {

  document.getElementById("screen1").classList.add("hidden");

  document.getElementById("screen2").classList.remove("hidden");

  document

    .getElementById("botoesSimulacaoRapida")

    .classList.add("hidden buttons");

}

function goToScreen1() {

  carregarTop10Crescimento();

  document.getElementById("screen2").classList.add("hidden");

  document.getElementById("screen1").classList.remove("hidden");

}

// Secção 1 - Lucro e lucro total com dividendos

setInterval(() => {

  carregarTop10Crescimento();

}, 30000); // 30 segundos (30000 milissegundos)

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

  carregarTop10Crescimento();

});

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

  carregarTop7("1s"); // Carrega ao abrir

  // Atualiza a cada 30 segundos

  setInterval(() => {

    carregarTop10Crescimento();

  }, 30000);

});

//top 10 empresas da Firebase Crescimento

function carregarTop10Crescimento(periodo = "1s") {

  const lista = document.getElementById("listaTop10");

  lista.innerHTML = "🔄 A carregar...";

  db.collection("acoesDividendos")

    .get()

    .then((querySnapshot) => {

      const acoes = [];

      querySnapshot.forEach((doc) => {

        const dados = doc.data();

        let crescimento = 0;

        if (periodo === "1s") crescimento = parseFloat(dados.taxaCrescimento\_1s || 0);

        else if (periodo === "1m") crescimento = parseFloat(dados.taxaCrescimento\_1m || 0);

        else if (periodo === "1ano") crescimento = parseFloat(dados.taxaCrescimento\_1ano || 0);

        if (crescimento > 0) {

          acoes.push({

            nome: dados.nome,

            ticker: dados.ticker,

            crescimento: crescimento.toFixed(2),

          });

        }

      });

      const top10 = acoes

        .sort((a, b) => b.crescimento - a.crescimento)

        .slice(0, 10);

      if (top10.length === 0) {

        lista.innerHTML = "<li>😕 Nenhuma ação com crescimento positivo.</li>";

        return;

      }

      lista.innerHTML = top10

        .map((acao) => `<li><strong>${acao.nome}</strong> (${acao.ticker}) — +${acao.crescimento}%</li>`)

        .join("");

    })

    .catch((error) => {

      console.error("Erro ao carregar Top 10:", error);

      lista.innerHTML = "<li style='color:red;'>Erro ao carregar dados.</li>";

    });

}

//Botões filtro top semana/mes/ano

function carregarTop7(periodo) {

  const lista = document.getElementById("listaTopAcoes");

  lista.innerHTML = "🔄 A carregar...";

  db.collection("acoesDividendos")

    .get()

    .then((querySnapshot) => {

      const acoes = [];

      querySnapshot.forEach((doc) => {

        const dados = doc.data();

        let crescimento = null;

        if (periodo === "1s") crescimento = dados.taxaCrescimento\_1s;

        else if (periodo === "1m") crescimento = dados.taxaCrescimento\_1m;

        else if (periodo === "1ano") crescimento = dados.taxaCrescimento\_1ano;

        if (typeof crescimento === "number" && crescimento > 0) {

          acoes.push({

            nome: dados.nome,

            ticker: dados.ticker,

            crescimento: crescimento.toFixed(2),

          });

        }

      });

      const top7 = acoes

        .sort((a, b) => b.crescimento - a.crescimento)

        .slice(0, 7);

      if (top7.length === 0) {

        lista.innerHTML = "<li>😕 Nenhuma ação com crescimento positivo.</li>";

        return;

      }

      lista.innerHTML = top7

        .map(

          (acao) =>

            `<li><strong>${acao.nome}</strong> (${acao.ticker}) — 📈 +${acao.crescimento}%</li>`

        )

        .join("");

    })

    .catch((error) => {

      console.error("Erro ao carregar top ações:", error);

      lista.innerHTML =

        "<li style='color:red;'>Erro ao carregar ações.</li>";

    });

}

function calcularLucro1() {

  let tp1 = parseFloat(document.getElementById("tp1\_1").value);

  let tp2 = parseFloat(document.getElementById("tp2\_1").value);

  let invest = parseFloat(document.getElementById("invest\_1").value);

  let dividendo = parseFloat(document.getElementById("dividendo\_1").value);

  if (!isNaN(tp1) && !isNaN(tp2) && !isNaN(invest) && !isNaN(dividendo)) {

    let lucro\_valorizacao = (invest \* (tp2 - tp1)) / tp1;

    let numero\_acoes = invest / tp1;

    let lucro\_dividendos = numero\_acoes \* dividendo;

    let lucro\_total = lucro\_valorizacao + lucro\_dividendos;

    document.getElementById("lucro\_valorizacao1").textContent =

      lucro\_valorizacao.toFixed(2);

    document.getElementById("lucro\_total1").textContent =

      lucro\_total.toFixed(2);

  } else {

    document.getElementById("lucro\_valorizacao1").textContent =

      "Preencha todos os campos.";

    document.getElementById("lucro\_total1").textContent = "-";

  }

}

// Secção 2 - Investimento para lucro desejado (sem dividendos)

function calcularInvestimento2() {

  let tp1 = parseFloat(document.getElementById("tp1\_2").value);

  let tp2 = parseFloat(document.getElementById("tp2\_2").value);

  let lucro = parseFloat(document.getElementById("lucro\_2").value);

  if (!isNaN(tp1) && !isNaN(tp2) && !isNaN(lucro)) {

    let invest = (lucro \* tp1) / (tp2 - tp1);

    if (invest < 0) invest = 0;

    document.getElementById("invest2").textContent = invest.toFixed(2);

  } else {

    document.getElementById("invest2").textContent =

      "Preencha todos os campos.";

  }

}

// Atualiza % crescimento dinamicamente com base em TP1 e TP2 Crescimento

function atualizarTP2\_2() {

  const tp1 = parseFloat(document.getElementById("tp1\_2").value);

  const tp2 = parseFloat(document.getElementById("tp2\_2").value);

  if (!isNaN(tp1) && !isNaN(tp2) && tp1 !== 0) {

    const percent = ((tp2 - tp1) / tp1) \* 100;

    document.getElementById("percent\_2").textContent = percent.toFixed(2) + "%";

  } else {

    document.getElementById("percent\_2").textContent = "-";

  }

}

// Secção 3 - Lucro e lucro total com dividendos (percentagem crescimento)

function calcularLucro3() {

  let tp1 = parseFloat(document.getElementById("tp1\_3").value);

  let percent = parseFloat(document.getElementById("percent\_3").value);

  let invest = parseFloat(document.getElementById("invest\_3").value);

  let dividendo = parseFloat(document.getElementById("dividendo\_3").value);

  if (!isNaN(tp1) && !isNaN(percent) && !isNaN(invest) && !isNaN(dividendo)) {

    let tp2 = tp1 \* (1 + percent / 100);

    let lucro\_valorizacao = (invest \* (tp2 - tp1)) / tp1;

    let numero\_acoes = invest / tp1;

    let lucro\_dividendos = numero\_acoes \* dividendo;

    let lucro\_total = lucro\_valorizacao + lucro\_dividendos;

    document.getElementById("lucro\_valorizacao3").textContent =

      lucro\_valorizacao.toFixed(2);

    document.getElementById("lucro\_total3").textContent =

      lucro\_total.toFixed(2);

  } else {

    document.getElementById("lucro\_valorizacao3").textContent =

      "Preencha todos os campos.";

    document.getElementById("lucro\_total3").textContent = "-";

  }

}

// Secção 4 - Lucro e lucro total (percentagem crescimento)

// Atualiza TP2 dinamicamente com base em TP1 e % Crescimento

function atualizarTP2\_4() {

  const tp1 = parseFloat(document.getElementById("tp1\_4").value);

  const percent = parseFloat(document.getElementById("percent\_4").value);

  if (!isNaN(tp1) && !isNaN(percent)) {

    const tp2 = tp1 \* (1 + percent / 100);

    document.getElementById("tp2\_4").textContent = tp2.toFixed(2);

  } else {

    document.getElementById("tp2\_4").textContent = "-";

  }

}

// Calcula o investimento necessário para obter o lucro desejado

function calcularInvestimento4() {

  const tp1 = parseFloat(document.getElementById("tp1\_4").value);

  const percent = parseFloat(document.getElementById("percent\_4").value);

  const lucro = parseFloat(document.getElementById("lucro\_4").value);

  if (!isNaN(tp1) && !isNaN(percent) && !isNaN(lucro)) {

    const tp2 = tp1 \* (1 + percent / 100);

    const invest = (lucro \* tp1) / (tp2 - tp1);

    document.getElementById("tp2\_4").textContent = tp2.toFixed(2);

    document.getElementById("invest4").textContent =

      invest < 0 ? "0.00" : invest.toFixed(2);

  } else {

    document.getElementById("invest4").textContent =

      "Preencha todos os campos.";

  }

}

//Botão +Info

function toggleInfo(id) {

  const info = document.getElementById(id);

  if (info.style.display === "none" || info.style.display === "") {

    info.style.display = "block";

  } else {

    info.style.display = "none";

  }

}

function mostrarSecao(secaoId) {

  document.getElementById("menuSecoes").classList.add("hidden");

  document.querySelectorAll("#screen2 > div").forEach((div) => {

    if (div.id.startsWith("secao")) {

      div.classList.add("hidden");

    }

  });

  document.getElementById(secaoId).classList.remove("hidden");

}

function voltarMenuSecoes() {

  document.querySelectorAll("#screen2 > div").forEach((div) => {

    if (div.id.startsWith("secao")) {

      div.classList.add("hidden");

    }

  });

  document.getElementById("menuSecoes").classList.remove("hidden");

}

function abrirSecao(num) {

  document.getElementById("screen2").classList.add("hidden");

  for (let i = 1; i <= 7; i++) {

    document.getElementById(`sec${i}Screen`).classList.add("hidden");

  }

  document.getElementById(`sec${num}Screen`).classList.remove("hidden");

}

function voltarMenu() {

  for (let i = 1; i <= 4; i++) {

    document.getElementById(`sec${i}Screen`).classList.add("hidden");

  }

  document.getElementById("screen2").classList.remove("hidden");

}

//Simulações

const simulacoes = [];

//Fórmulas de Lucros

function guardarSimulacao(nomeAcao, tp1, tp2, valorInvestido, dividendo = 0) {

  const crescimento = ((tp2 - tp1) / tp1) \* 100;

  const numeroAcoes = valorInvestido / tp1;

  const lucroValorizacao = (tp2 - tp1) \* numeroAcoes;

  const lucroDividendos = numeroAcoes \* dividendo;

  const lucroTotal = lucroValorizacao + lucroDividendos;

  const novaSimulacao = {

    nomeAcao,

    tp1,

    tp2,

    valorInvestido,

    lucro: parseFloat(lucroTotal.toFixed(2)),

    crescimentoPercentual: parseFloat(crescimento.toFixed(2)),

  };

  simulacoes.push(novaSimulacao);

  atualizarTabela();

  atualizarGrafico();

}

//Atualizar a Tabela

function atualizarTabela() {

  const corpoTabela = document.querySelector("#tabelaSimulacoes tbody");

  corpoTabela.innerHTML = "";

  simulacoes.forEach((sim, index) => {

    const linha = document.createElement("tr");

    linha.innerHTML = `

      <td>${sim.nomeAcao}</td>

      <td>${sim.tp1}</td>

      <td>${sim.tp2}</td>

      <td>${sim.valorInvestido}</td>

      <td>${sim.lucro.toFixed(2)}</td>

      <td>${sim.crescimentoPercentual}%</td>

      <td>

        <button onclick="removerSimulacao(${index})">❌</button>

      </td>

      <td>

        <input type="checkbox" onchange="atualizarSomaLucros()" class="checkbox-lucro" data-lucro="${

          sim.lucro

        }">

      </td>

    `;

    corpoTabela.appendChild(linha);

  });

  atualizarSomaLucros(); // Atualiza o total sempre que a tabela é renderizada

}

function removerSimulacao(index) {

  simulacoes.splice(index, 1); // Remove 1 elemento na posição index

  atualizarTabela(); // Re-renderiza a tabela

  atualizarGrafico(); // Atualiza o gráfico com as simulações restantes

}

function atualizarSomaLucros() {

  const checkboxes = document.querySelectorAll(".checkbox-lucro");

  let total = 0;

  checkboxes.forEach((cb) => {

    if (cb.checked) {

      total += parseFloat(cb.dataset.lucro);

    }

  });

  mostrarTotalLucro(total);

}

function mostrarTotalLucro(valor) {

  let totalRow = document.getElementById("linha-total-lucro");

  if (!totalRow) {

    totalRow = document.createElement("tr");

    totalRow.id = "linha-total-lucro";

    totalRow.innerHTML = `

      <td colspan="4"><strong>Total Lucro Selecionado:</strong></td>

      <td colspan="4" id="valorTotalLucro"><strong>${valor.toFixed(

        2

      )} €</strong></td>

    `;

    document.querySelector("#tabelaSimulacoes tbody").appendChild(totalRow);

  } else {

    totalRow.querySelector(

      "#valorTotalLucro"

    ).innerHTML = `<strong>${valor.toFixed(2)} €</strong>`;

  }

}

let grafico;

function atualizarGrafico() {

  const labels = simulacoes.map((s) => s.nomeAcao);

  const dadosLucro = simulacoes.map((s) => s.lucro);

  if (grafico) grafico.destroy();

  const ctx = document.getElementById("graficoLucro").getContext("2d");

  grafico = new Chart(ctx, {

    type: "bar",

    data: {

      labels: labels,

      datasets: [

        {

          label: "Lucro (€)",

          data: dadosLucro,

          backgroundColor: "rgba(54, 162, 235, 0.5)",

          borderColor: "rgba(54, 162, 235, 1)",

          borderWidth: 1,

        },

      ],

    },

    options: {

      scales: {

        y: { beginAtZero: true },

      },

    },

  });

}

//Botão Simular e Guardar para passar para o gráfico

function simularEGUardar() {

  document.querySelector(".tabela-scroll-wrapper")?.classList.remove("hidden");

  const nome = document.getElementById("nomeAcao").value;

  const tp1 = parseFloat(document.getElementById("tp1").value);

  const tp2 = parseFloat(document.getElementById("tp2").value);

  const investimento = parseFloat(

    document.getElementById("investimento").value

  );

  const dividendo = parseFloat(document.getElementById("dividendo").value || 0);

  if (!nome || isNaN(tp1) || isNaN(tp2) || isNaN(investimento)) {

    alert("Preenche todos os campos!");

    return;

  }

  guardarSimulacao(nome, tp1, tp2, investimento, dividendo);

}

function voltarMenu() {

  // Esconde todas as secções

  const secoes = document.querySelectorAll(".screen");

  secoes.forEach((secao) => secao.classList.add("hidden"));

  // Mostra apenas o menu principal

  document.getElementById("screen2").classList.remove("hidden");

  secoes.forEach((secao) => secao.classList.add("hidden"));

  document.getElementById("screen2").classList.remove("hidden");

  // 🔒 Esconde os botões de simulação rápida se estiverem visíveis

  const botoes = document.getElementById("botoesSimulacaoRapida");

  if (botoes) botoes.classList.add("hidden");

}

function limparCampos() {

  // Encontra a secção visível (sem a classe "hidden")

  const secaoVisivel = document.querySelector(".screen:not(.hidden)");

  if (secaoVisivel) {

    // Limpa todos os inputs do tipo number e text dentro da secção visível

    const inputs = secaoVisivel.querySelectorAll(

      "input[type=number], input[type=text]"

    );

    inputs.forEach((input) => (input.value = ""));

    // Limpa todos os spans com resultados dentro da secção visível

    const spans = secaoVisivel.querySelectorAll("span");

    spans.forEach((span) => {

      if (span.id) {

        span.textContent = "-";

      }

    });

  }

}

//secção 5 - Simulador

function abrirSecao(num) {

  document.getElementById("screen2").classList.add("hidden");

  for (let i = 1; i <= 6; i++) {

    document.getElementById(`sec${i}Screen`).classList.add("hidden");

  }

  document.getElementById(`sec${num}Screen`).classList.remove("hidden");

  // 🔄 Garante que os botões de simulação rápida estão sempre escondidos ao entrar no screen 5

  if (num === 5) {

    const botoes = document.getElementById("botoesSimulacaoRapida");

    if (botoes && !botoes.classList.contains("hidden")) {

      botoes.classList.add("hidden");

    }

  }

}

function prepararSimulacao(nome, valorStock, dividendo) {

  console.log(

    "Preparar simulação para:",

    nome,

    "a TP1",

    valorStock,

    "com dividendo:",

    dividendo

  );

  abrirSecao(5); // Vai para screen simulação

  document.getElementById("nomeAcao").value = nome;

  // Mostra os botões de simulação rápida

  const botoes = document.getElementById("botoesSimulacaoRapida");

  if (botoes) {

    botoes.classList.remove("hidden");

  }

}

// 🚨 NOVO: Limpar Gráfico

function limparGrafico() {

  simulacoes.length = 0; // Apaga todas as simulações

  atualizarTabela(); // Atualiza a tabela (fica vazia)

  if (grafico) {

    grafico.destroy(); // Destrói o gráfico existente

    grafico = null;

  }

}

//Enviar Email

function abrirPopupEmail() {

  document.getElementById("popupSimulacao").classList.add("hidden");

  const popup = document.getElementById("popupEmail");

  popup.classList.remove("hidden");

  const dadosDiv = document.getElementById("dadosSimulacao");

  if (simulacoes.length > 0) {

    let html = "<h4>Simulações Guardadas:</h4>";

    simulacoes.forEach((sim, index) => {

      html += `

        <p><strong>Simulação ${index + 1}</strong></p>

        <p>Ação: ${sim.nomeAcao}</p>

        <p>Preço Inicial: €${sim.tp1}</p>

        <p>Preço Final: €${sim.tp2}</p>

        <p>Investimento: €${sim.valorInvestido}</p>

        <p>Lucro: €${sim.lucro}</p>

        <p>Crescimento: ${sim.crescimentoPercentual}%</p>

        <hr>

      `;

    });

    dadosDiv.innerHTML = html;

  } else {

    dadosDiv.innerHTML = "<p>Sem simulações disponíveis.</p>";

  }

}

function fecharPopupEmail() {

  document.getElementById("popupEmail").classList.add("hidden");

}

//Enviar Email

function enviarEmail() {

  const emailDestino = document.getElementById("emailDestino").value;

  if (!emailDestino || simulacoes.length === 0) {

    alert("Preenche o email e faz pelo menos uma simulação.");

    return;

  }

  const assunto = encodeURIComponent("Resumo de Simulações Financeiras");

  let corpoTexto = "Resumo das Simulações:\n\n";

  simulacoes.forEach((sim, index) => {

    corpoTexto += `Simulação ${index + 1}:\n`;

    corpoTexto += `Ação: ${sim.nomeAcao}\n`;

    corpoTexto += `Preço Inicial: €${sim.tp1}\n`;

    corpoTexto += `Preço Final: €${sim.tp2}\n`;

    corpoTexto += `Investimento: €${sim.valorInvestido}\n`;

    corpoTexto += `Lucro: €${sim.lucro}\n`;

    corpoTexto += `Crescimento: ${sim.crescimentoPercentual}%\n\n`;

  });

  const corpo = encodeURIComponent(corpoTexto);

  const mailtoLink = `mailto:${emailDestino}?subject=${assunto}&body=${corpo}`;

  window.location.href = mailtoLink;

}

//Menu 6 - Registar/Editar e Filtrar Empresas com Dividendos

let acoesSelecionadasParaBloco = [];

function selecionarAcao(acao) {

  acoesSelecionadasParaBloco.push(acao);

}

function simularAcoesSelecionadas() {

  calcularDistribuicao(); // já usa a variável global

}

//Botão secção dos filtros

function toggleFiltrosMes() {

  const filtrosDiv = document.getElementById("filtrosMesContainer");

  const botao = document.getElementById("btnMostrarFiltros");

  if (filtrosDiv.classList.contains("hidden")) {

    filtrosDiv.classList.remove("hidden");

    botao.textContent = "▲";

  } else {

    filtrosDiv.classList.add("hidden");

    botao.textContent = "▼";

  }

}

// Filtrar Base Dados Firebase por múltiplos critérios combinados

function filtrarAcoes() {

  const setor = document.getElementById("filtroSetor").value;

  const mercado = document.getElementById("filtroMercado").value;

  const mes = document.getElementById("filtroMes").value;

  const periodicidade = document.getElementById("filtroPeriodicidade").value;

  const nomeInput = document

    .getElementById("filtroNome")

    .value.trim()

    .toLowerCase();

  const tickerInput = document

    .getElementById("filtroTicker")

    .value.trim()

    .toLowerCase();

  const resultadoDiv = document.getElementById("resultadoFiltroMes");

  resultadoDiv.innerHTML = "A carregar...";

  db.collection("acoesDividendos")

    .get()

    .then((querySnapshot) => {

      let html = "<ul>";

      let count = 0;

      querySnapshot.forEach((doc) => {

        const dados = doc.data();

        const matchSetor =

          !setor ||

          dados.setor?.trim().toLowerCase() === setor.trim().toLowerCase();

        const matchMercado =

          !mercado ||

          dados.mercado?.trim().toLowerCase() === mercado.trim().toLowerCase();

        const matchMes =

          !mes || dados.mes?.trim().toLowerCase() === mes.trim().toLowerCase();

        const matchPeriodicidade =

          !periodicidade ||

          dados.periodicidade?.trim().toLowerCase() ===

            periodicidade.trim().toLowerCase();

        const matchNome =

          !nomeInput || dados.nome?.toLowerCase().includes(nomeInput);

        const matchTicker =

          !tickerInput || dados.ticker?.toLowerCase().includes(tickerInput);

        if (

          matchSetor &&

          matchMercado &&

          matchMes &&

          matchPeriodicidade &&

          matchNome &&

          matchTicker

        ) {

          const jaSelecionada = acoesSelecionadasParaBloco.some(

            (a) => a.ticker === dados.ticker

          );

          html += `

              <li>

                <input type="checkbox" class="checkbox-selecao"

                      onchange='atualizarSelecao(this)'

                      value='${JSON.stringify(dados)}' />

                <strong>${dados.nome}</strong> (${dados.ticker})<br>

                Setor: ${dados.setor} | Mercado: ${dados.mercado} | Dividendo: €${dados.dividendo} |

                Mês: ${dados.mes} | Periodicidade: ${dados.periodicidade} | Valor da Ação: €${dados.valorStock || "N/D"}<br>

              </li>

              `;

          count++;

        }

      });

      html += "</ul>";

      resultadoDiv.innerHTML =

        count > 0

          ? html

          : "<p>Nenhuma ação encontrada com os filtros aplicados.</p>";

    })

    .catch((error) => {

      console.error("Erro ao filtrar:", error);

      resultadoDiv.innerHTML =

        "<p style='color:red;'>Erro ao carregar dados.</p>";

    });

}

//Atualizar Seleccionadas

function atualizarSelecao(checkbox) {

  const dados = JSON.parse(checkbox.value);

  // Verifica se já está selecionada

  const index = acoesSelecionadasParaBloco.findIndex(

    (acao) => acao.ticker === dados.ticker

  );

  if (checkbox.checked) {

    if (index === -1) {

      acoesSelecionadasParaBloco.push(dados);

    }

  } else {

    if (index > -1) {

      acoesSelecionadasParaBloco.splice(index, 1);

    }

  }

  console.log("Selecionadas:", acoesSelecionadasParaBloco);

}

//Simular Ações Selecionadas

function prepararSimulacaoBloco() {

  if (!acoesSelecionadasParaBloco || acoesSelecionadasParaBloco.length === 0) {

    alert("⚠️ Nenhuma ação selecionada");

    return;

  }

  // Montar a tabela de ações

  const tabelaContainer = document.getElementById("tabelaAcoesSelecionadas");

  let html = `

    <table>

      <thead>

        <tr><th>Nome</th><th>Ticker</th><th>Valor (€)</th><th>Dividendo (€)</th></tr>

      </thead><tbody>

  `;

  acoesSelecionadasParaBloco.forEach((dados) => {

    html += `

      <tr>

        <td>${dados.nome}</td>

        <td>${dados.ticker}</td>

        <td>${dados.valorStock}</td>

        <td>${dados.dividendo}</td>

      </tr>

    `;

  });

  html += "</tbody></table>";

  tabelaContainer.innerHTML = html;

  // Abrir popup

  document.getElementById("popupSimulacaoBloco").classList.remove("hidden");

  // Forçar o cálculo inicial (caso já haja valor)

  calcularDistribuicao();

}

//Mostrar as ações selecionada

function preencherTabelaSimulacaoBloco(acoes) {

  const tbody = document.querySelector("#tabelaAcoesSelecionadas tbody");

  tbody.innerHTML = "";

  acoes.forEach((acao) => {

    const linha = document.createElement("tr");

    linha.innerHTML = `

      <td>${acao.nome}</td>

      <td>${acao.ticker}</td>

      <td>€${acao.valorStock.toFixed(2)}</td>

      <td>€${acao.dividendo.toFixed(2)}</td>

    `;

    tbody.appendChild(linha);

  });

}

//Fechar popup Bloco

function fecharPopupSimulacaoBloco() {

  // 1. Fechar o popup

  document.getElementById("popupSimulacaoBloco").classList.add("hidden");

  // 2. Limpar a lista de ações selecionadas

  acoesSelecionadasParaBloco = [];

  // 3. Limpar o conteúdo da tabela de ações selecionadas

  document.getElementById("tabelaAcoesSelecionadas").innerHTML = "";

  // 4. Limpar o resultado da distribuição

  document.getElementById("resultadoDistribuicao").innerHTML = "";

  // 5. (Opcional) desmarcar checkboxes

  const checkboxes = document.querySelectorAll(".checkbox-selecao");

  checkboxes.forEach(cb => cb.checked = false);

}

let acoesParaSimulacao = []; // <-- isto deve estar fora das funções, no topo do ficheiro .js

//checkboxes para selecionar investimento completo ou parcelar

document.getElementById("usarTotal").addEventListener("change", () => {

    if (usarTotal.checked) {

      acoesCompletas.checked = false;

    }

    calcularDistribuicao();

  });

  document.getElementById("acoesCompletas").addEventListener("change", () => {

    if (acoesCompletas.checked) {

      usarTotal.checked = false;

    }

    calcularDistribuicao();

  });

  document.getElementById("investimentoTotal").addEventListener("input", calcularDistribuicao);

//Lucro máximo

function calcularDistribuicao() {

  const investimentoTotal = parseFloat(document.getElementById("investimentoTotal").value);

  const resultadoDiv = document.getElementById("resultadoDistribuicao");

  const usarTotal = document.getElementById("usarTotal").checked;

  const apenasCompletas = document.getElementById("acoesCompletas").checked;

  if (!investimentoTotal || investimentoTotal <= 0) {

    resultadoDiv.innerHTML = "<p style='color:red;'>⚠️ Introduz um valor de investimento válido.</p>";

    return;

  }

  if (!acoesSelecionadasParaBloco || acoesSelecionadasParaBloco.length === 0) {

    resultadoDiv.innerHTML = "<p style='color:red;'>⚠️ Nenhuma ação selecionada.</p>";

    return;

  }

  const tipoCrescimento = document.getElementById("periodoCrescimento").value || "taxaCrescimento\_1s";

  // 1. Enriquecer ações com dados calculados

  const acoesComLucro = acoesSelecionadasParaBloco.map((acao) => {

    const preco = parseFloat(acao.valorStock || 0);

    const dividendo = parseFloat(acao.dividendo || 0);

    const taxa = parseFloat(acao[tipoCrescimento] || 0);

    const dividendoAnual = dividirPeriodicidade(dividendo, acao.periodicidade);

    const lucroUnidade = dividendoAnual + (preco \* taxa / 100);

    const retornoPorEuro = lucroUnidade / preco;

    return {

      ...acao,

      preco,

      dividendoAnual,

      taxa,

      lucroUnidade,

      retornoPorEuro

    };

  });

  // 2. Calcular soma total de retorno por euro

  const totalRetornoPorEuro = acoesComLucro.reduce((sum, a) => sum + a.retornoPorEuro, 0);

  if (totalRetornoPorEuro === 0) {

    resultadoDiv.innerHTML = "<p style='color:red;'>⚠️ As ações selecionadas não têm retorno estimado.</p>";

    return;

  }

  let totalLucro = 0;

  let distribuicao = [];

  // 3. Distribuir o investimento proporcionalmente

  acoesComLucro.forEach((acao) => {

    const proporcao = acao.retornoPorEuro / totalRetornoPorEuro;

    let valorInvestido = investimentoTotal \* proporcao;

    let qtd = valorInvestido / acao.preco;

    if (apenasCompletas) {

      qtd = Math.floor(qtd);

      valorInvestido = qtd \* acao.preco;

    }

    const lucro = qtd \* acao.lucroUnidade;

    totalLucro += lucro;

    if (qtd > 0) {

      distribuicao.push({

        nome: acao.nome,

        ticker: acao.ticker,

        quantidade: apenasCompletas ? qtd : qtd.toFixed(4),

        investido: valorInvestido.toFixed(2),

        lucro: lucro.toFixed(2),

        crescimento: `${acao.taxa.toFixed(2)}%`,

        dividendo: acao.dividendoAnual.toFixed(2)

      });

    }

  });

  // 4. Apresentar resultados

  let html = `

    <table>

      <thead>

        <tr>

          <th>Ação</th><th>Ticker</th><th>Qtd</th><th>Investido (€)</th><th>Lucro (€)</th><th>Tx Crescimento</th><th>Dividendo Anual</th>

        </tr>

      </thead>

      <tbody>

  `;

  distribuicao.forEach((linha) => {

    html += `<tr>

      <td>${linha.nome}</td>

      <td>${linha.ticker}</td>

      <td>${linha.quantidade}</td>

      <td>${linha.investido}</td>

      <td>${linha.lucro}</td>

      <td>${linha.crescimento}</td>

      <td>${linha.dividendo}</td>

    </tr>`;

  });

  html += `</tbody></table><p><strong>Lucro Total Estimado: €${totalLucro.toFixed(2)}</strong></p>`;

  resultadoDiv.innerHTML = html;

}

function dividirPeriodicidade(dividendo, periodicidade) {

  switch ((periodicidade || "").toLowerCase()) {

    case "mensal": return dividendo \* 12;

    case "trimestral": return dividendo \* 4;

    case "semestral": return dividendo \* 2;

    case "anual": return dividendo;

    default: return dividendo; // fallback

  }

}

function abrirPopupFiltro() {

  filtrarAcoes(); // mostra todas as empresas

  const popup = document.getElementById("popupFiltro");

  popup.classList.remove("hidden");

  popup.classList.add("show");

}

function fecharPopupFiltro() {

  const popup = document.getElementById("popupFiltro");

  popup.classList.remove("show");

  popup.classList.add("hidden");

}

// preparar simulação do screen 6 para o 5

function prepararSimulacao(nome, valorStock, dividendo) {

  fecharPopupFiltro(); // fecha o popup ao simular

  console.log(

    "Preparar simulação para:",

    nome,

    "com dividendo:",

    dividendo,

    "e TP1:",

    valorStock

  );

  abrirSecao(5); // Vai para screen simulação

  document.getElementById("nomeAcao").value = nome;

  document.getElementById("dividendo").value = dividendo || 0;

  document.getElementById("tp1").value = valorStock || "";

  document.getElementById("tp2").value = "";

  document.getElementById("investimento").value = "";

  document.getElementById("botoesSimulacaoRapida").classList.remove("hidden");

}

// editar ação

function editarAcao(id, dados) {

  fecharPopupFiltro(); // fecha o popup ao editar

  idAcaoEmEdicao = id;

  document.getElementById("nomeAcaoReg").value = dados.nome || "";

  document.getElementById("tickerAcaoReg").value = dados.ticker || "";

  document.getElementById("Setor").value = dados.setor || "";

  document.getElementById("Mercado").value = dados.mercado || "";

  document.getElementById("valorDividendoReg").value = dados.dividendo || "";

  document.getElementById("mesDividendoReg").value = dados.mes || "";

  document.getElementById("periodicidade").value = dados.periodicidade || "";

}

//Função simularValorRapido(valor)

function simularValorRapido(valor) {

  const tp1 = parseFloat(document.getElementById("tp1").value);

  const tp2 = parseFloat(document.getElementById("tp2").value);

  const dividendo = parseFloat(document.getElementById("dividendo").value || 0);

  if (isNaN(tp1) || isNaN(tp2)) {

    alert("⚠️ Introduz TP1 e TP2 antes de simular.");

    return;

  }

  document.getElementById("investimento").value = valor;

  guardarSimulacao(

    document.getElementById("nomeAcao").value,

    tp1,

    tp2,

    valor,

    dividendo

  );

  // Esconde os botões após simular

  //document.getElementById("botoesSimulacaoRapida").classList.add("hidden");

}

//Quando clico no Botão Voltar no menu Simular

function voltarMenu() {

  // Esconde todas as secções

  const secoes = document.querySelectorAll(".screen");

  secoes.forEach((secao) => secao.classList.add("hidden"));

  // Mostra apenas o menu principal

  document.getElementById("screen2").classList.remove("hidden");

  // Esconde os botões de simulação rápida

  const botoes = document.getElementById("botoesSimulacaoRapida");

  if (botoes) botoes.classList.add("hidden");

}

//Use o ticker como ID do documento no Firestore (evita duplicações).

//Adicione o campo origem: "webapp" para rastrear a origem dos dados

//Use .set(..., { merge: true }) para atualizar ou criar o documento de forma segura.

let idAcaoEmEdicao = null;

function guardarOuAtualizarAcaoFirebase() {

  const nome = document.getElementById("nomeAcaoReg").value.trim();

  const ticker = document.getElementById("tickerAcaoReg").value.trim();

  const setor = document.getElementById("Setor").value.trim();

  const mercado = document.getElementById("Mercado").value.trim();

  const dividendo = parseFloat(

    document.getElementById("valorDividendoReg").value

  );

  const valorStock = parseFloat(document.getElementById("valorStock").value);

  const mes = document.getElementById("mesDividendoReg").value;

  const periodicidade = document.getElementById("periodicidade").value;

  if (

    !nome ||

    !ticker ||

    !setor ||

    !mercado ||

    isNaN(dividendo) ||

    isNaN(valorStock) ||

    !mes ||

    !periodicidade

  ) {

    alert("⚠️ Preenche todos os campos corretamente.");

    return;

  }

  const dadosAcao = {

    nome,

    ticker,

    setor,

    mercado,

    valorStock,

    dividendo,

    mes,

    periodicidade,

    origem: "webapp",

    timestamp: new Date(),

  };

  // Se estiver a editar, usa o ID do documento em edição; senão, usa o ticker como ID

  const docId = idAcaoEmEdicao || ticker;

  db.collection("acoesDividendos")

    .doc(docId)

    .set(dadosAcao, { merge: true }) // Cria ou atualiza sem apagar campos existentes

    .then(() => {

      alert(

        idAcaoEmEdicao

          ? "✅ Ação atualizada com sucesso!"

          : "✅ Ação guardada com sucesso na Firebase!"

      );

      limparCamposSec6();

      idAcaoEmEdicao = null;

    })

    .catch((error) => {

      console.error("Erro ao guardar/atualizar:", error);

      alert("❌ Erro ao guardar/atualizar. Verifica a ligação com a Firebase.");

    });

}

if (idAcaoEmEdicao) {

  // Atualizar ação existente

  db.collection("acoesDividendos")

    .doc(idAcaoEmEdicao)

    .update(dadosAcao)

    .then(() => {

      alert("✅ Ação atualizada com sucesso!");

      limparCamposSec6();

      idAcaoEmEdicao = null;

    })

    .catch((error) => {

      console.error("Erro ao atualizar:", error);

      alert("❌ Erro ao atualizar. Tenta novamente.");

    });

} else {

  // Guardar nova ação

  db.collection("acoesDividendos")

    .add(dadosAcao)

    .then(() => {

      alert("✅ Ação guardada com sucesso na Firebase!");

      limparCamposSec6();

    })

    .catch((error) => {

      console.error("Erro ao guardar ação:", error);

      alert(

        "❌ Ocorreu um erro ao guardar. Verifica a ligação com a Firebase."

      );

    });

}

//Eliminar um Registo

function eliminarAcao(id) {

  if (confirm("⚠️ Tens a certeza que queres eliminar esta ação?")) {

    db.collection("acoesDividendos")

      .doc(id)

      .delete()

      .then(() => {

        alert("✅ Ação eliminada com sucesso.");

        filtrarAcoes(); // Recarrega os resultados após eliminar

      })

      .catch((error) => {

        console.error("Erro ao eliminar ação:", error);

        alert("❌ Erro ao eliminar. Tenta novamente.");

      });

  }

}

function mostrarSecao(id) {

  document

    .querySelectorAll(".screen")

    .forEach((sec) => sec.classList.add("hidden"));

  document.getElementById(id).classList.remove("hidden");

}

//Secção Reforço

function abrirPopupReforco() {

  document.getElementById("popupReforco").classList.remove("hidden");

}

function fecharPopupReforco() {

  document.getElementById("popupReforco").classList.add("hidden");

}

function calcularMediaPonderada() {

  const invest1 = parseFloat(

    document.getElementById("invest1").value.replace(",", ".")

  );

  const preco1 = parseFloat(

    document.getElementById("preco1").value.replace(",", ".")

  );

  const invest22 = parseFloat(

    document.getElementById("invest22").value.replace(",", ".")

  );

  const preco2 = parseFloat(

    document.getElementById("preco2").value.replace(",", ".")

  );

  if (

    !isNaN(invest1) &&

    !isNaN(preco1) &&

    !isNaN(invest22) &&

    !isNaN(preco2) &&

    preco1 > 0 &&

    preco2 > 0

  ) {

    const qtd1 = invest1 / preco1;

    const qtd2 = invest22 / preco2;

    const totalQtd = qtd1 + qtd2;

    const totalInvestido = invest1 + invest22;

    const precoMedio = totalInvestido / totalQtd;

    const html = `

      <p>📊 <strong>Preço Médio:</strong> ${precoMedio.toFixed(2)} €</p>

      <p>📦 <strong>Total de Ações:</strong> ${totalQtd.toFixed(2)}</p>

      <p>💰 <strong>Total Investido:</strong> ${totalInvestido.toFixed(2)} €</p>

    `;

    document.getElementById("resultadoReforco").innerHTML = html;

  } else {

    document.getElementById("resultadoReforco").innerHTML =

      "❌ Preencha todos os campos corretamente.";

  }

}

//Objetivo: Identificar, entre várias ações (com base em dados reais), qual é a mais provável de atingir

// um lucro X com um investimento Y, considerando o preço da ação, o dividendo anual e eventualmente uma previsão de valorização.

// Simulador de ações com base em investimento, crescimento esperado e lucro desejado

function abrirSimulador() {

  document.getElementById("simuladorModal").classList.remove("hidden");

}

function fecharSimulador() {

  document.getElementById("simuladorModal").classList.add("hidden");

}

async function simular() {

  const investimento = parseFloat(

    document.getElementById("inputInvestimento").value

  );

  const crescimentoEstimado =

    parseFloat(document.getElementById("inputCrescimento").value) || 0;

  const lucroDesejado =

    parseFloat(document.getElementById("inputLucro").value) || 0;

  const db = firebase.firestore();

  const acoesRef = db.collection("acoesDividendos");

  const snapshot = await acoesRef.get();

  let resultados = [];

  snapshot.forEach((doc) => {

    const acao = doc.data();

    let preco = parseFloat(acao.valorStock);

    let dividendo = parseFloat(acao.dividendo);

    if (!preco || preco <= 0 || isNaN(preco)) return;

    if (isNaN(dividendo)) dividendo = 0;

    const quantidade = Math.floor(investimento / preco);

    if (quantidade === 0) return;

    const lucroValorizacao = quantidade \* preco \* (crescimentoEstimado / 100);

    const lucroDividendos = quantidade \* dividendo;

    const lucroTotal = lucroValorizacao + lucroDividendos;

    resultados.push({

      nome: acao.nome,

      preco,

      dividendo,

      quantidade,

      lucroValorizacao,

      lucroDividendos,

      lucroTotal,

      diferenca: Math.abs(lucroTotal - lucroDesejado),

    });

  });

  const resultado = document.getElementById("resultadoSimulacao");

  if (resultados.length === 0) {

    resultado.innerHTML =

      "<p>⚠️ Nenhuma ação válida para este investimento.</p>";

    return;

  }

  // Ordenar pelas mais próximas do lucro desejado (menor diferença absoluta)

  const topMaisProximas = resultados

    .sort((a, b) => a.diferenca - b.diferenca)

    .slice(0, 10);

  let html = `<h3>🔍 Top 10 mais próximas do lucro desejado (${lucroDesejado.toFixed(

    2

  )}€)</h3>`;

  topMaisProximas.forEach((acao, i) => {

    html += `

      <hr />

      <p><strong>Top ${i + 1}: ${acao.nome}</strong></p>

      <p>Preço atual: €${acao.preco.toFixed(2)}</p>

      <p>Dividendos: €${acao.dividendo.toFixed(2)} por ação</p>

      <p>Quantidade a comprar: ${acao.quantidade}</p>

      <p>Lucro com valorização: €${acao.lucroValorizacao.toFixed(2)}</p>

      <p>Lucro com dividendos: €${acao.lucroDividendos.toFixed(2)}</p>

      <p><strong>Lucro total estimado: €${acao.lucroTotal.toFixed(

        2

      )}</strong> (diferença: €${acao.diferenca.toFixed(2)})</p>

    `;

  });

  resultado.innerHTML = html;

}

function abrirPopupSimuladorGrafico() {

  document

    .getElementById("popupSimuladorGrafico")

    .classList.remove("popup hidden");

}

function fecharPopupSimuladorGrafico() {

  document

    .getElementById("popupSimuladorGrafico")

    .classList.add("popup hidden");

}

//TP2

function abrirSimuladorTP2() {

  document.getElementById("popupTP2").classList.remove("hidden");

}

function fecharSimuladorTP2() {

  document.getElementById("popupTP2").classList.add("hidden");

  document.getElementById("resultadoTP2").innerHTML = "";

  document.getElementById("tp1Input").value = "";

  document.getElementById("investimentoInput").value = "";

  document.getElementById("lucroDesejadoInput").value = "";

}

function calcularTP2() {

  const tp1 = parseFloat(document.getElementById("tp1Input").value);

  const investimento = parseFloat(document.getElementById("investimentoInput").value);

  const lucroDesejado = parseFloat(document.getElementById("lucroDesejadoInput").value);

  if (isNaN(tp1) || isNaN(investimento) || isNaN(lucroDesejado) || tp1 <= 0) {

    document.getElementById("resultadoTP2").innerHTML = "⚠️ Preenche todos os campos corretamente.";

    return;

  }

  const quantidade = investimento / tp1;

  const tp2 = calcularTp2(tp1, lucroDesejado, quantidade);

  const crescimento = ((tp2 - tp1) / tp1) \* 100;

  document.getElementById("resultadoTP2").innerHTML = `

    💰 <strong>TP2:</strong> €${tp2.toFixed(2)}<br>

    📈 <strong>Taxa de crescimento:</strong> ${crescimento.toFixed(2)}%

  `;

}

function calcularTp2(precoCompra, lucroAlvo, quantidade) {

  if (!precoCompra || !lucroAlvo || !quantidade) return 0;

  return precoCompra + lucroAlvo / quantidade;

}

function abrirPopupSimulacao() {

  document.getElementById("popupSimulacao").classList.remove("hidden");

}

function fecharPopupSimulacao() {

  document.getElementById("popupSimulacao").classList.add("hidden");

}

//botão Somar Lucros

function somarLucros() {

  const corpoTabela = document.querySelector("#tabelaSimulacoes tbody");

  if (!corpoTabela) return;

  // Soma os lucros

  let totalLucro = 0;

  simulacoes.forEach((sim) => {

    totalLucro += Number(sim.lucro || 0);

  });

  // Remove linha de total anterior, se já existir

  const linhaAnterior = document.querySelector(".linha-total");

  if (linhaAnterior) linhaAnterior.remove();

  // Cria nova linha de total

  const linhaTotal = document.createElement("tr");

  linhaTotal.classList.add("linha-total");

  linhaTotal.style.fontWeight = "bold";

  linhaTotal.style.backgroundColor = "#d1f0d1"; // verde claro

  linhaTotal.innerHTML = `

    <td colspan="4" style="text-align:right;">Total de Lucros:</td>

    <td>${totalLucro.toFixed(2)} €</td>

    <td>-</td>

  `;

  corpoTabela.appendChild(linhaTotal);

}

function selecionarTudoCheckboxes(checkboxSelecionarTudo) {

  const checkboxes = document.querySelectorAll("#listaAcoesCheckbox .checkboxAcao");

  checkboxes.forEach(cb => {

    cb.checked = checkboxSelecionarTudo.checked;

  });

}

  //Registo de compra de acçoes

  collection(db, "ativos")

  function guardarCompraAcaoOuEtf() {

    const nome = document.getElementById("nomeAtivo").value.trim();

    const ticker = document.getElementById("tickerAtivo").value.trim().toUpperCase();

    const tipoAtivo = document.getElementById("tipoAtivo").value;

    const precoCompra = parseFloat(document.getElementById("precoCompra").value);

    const quantidade = parseFloat(document.getElementById("quantidade").value);

    const setor = document.getElementById("setorCompra").value.trim();

    const mercado = document.getElementById("mercadoCompra").value.trim();

    const tipoObjetivo = document.getElementById("tipoObjetivo").value;

    const objetivoFinanceiro = parseFloat(document.getElementById("objetivoFinanceiro").value);

    if (!nome || !ticker || isNaN(precoCompra) || isNaN(quantidade) || isNaN(objetivoFinanceiro)) {

      alert("⚠️ Preenche todos os campos obrigatórios.");

      return;

    }

    const dados = {

      nome,

      ticker,

      tipoAtivo,

      precoCompra,

      quantidade,

      setor,

      mercado,

      tipoObjetivo,

      objetivoFinanceiro

    };

    db.collection("ativos")

      .add(dados)

      .then(() => {

        alert("✅ Compra registada com sucesso!");

        // Limpar campos

        document.querySelectorAll("#sec7Screen input").forEach((el) => (el.value = ""));

      })

      .catch((error) => {

        console.error("Erro ao guardar:", error);

        alert("❌ Erro ao guardar dados.");

      });

  }

  //Analisar progresso do objetivo (por ticker)

  async function verificarProgressoObjetivo(tickerAtual) {

    try {

      // 1. Buscar a compra em 'ativos'

      const ativosSnapshot = await db

        .collection("ativos")

        .where("ticker", "==", tickerAtual.toUpperCase())

        .get();

      if (ativosSnapshot.empty) {

        alert("⚠️ Nenhum ativo encontrado com esse ticker.");

        return;

      }

      let totalInvestido = 0;

      let totalQuantidade = 0;

      let tipoObjetivo = "";

      let objetivoFinanceiro = 0;

      let nome = "";

      ativosSnapshot.forEach((doc) => {

        const data = doc.data();

        totalInvestido += data.precoCompra \* data.quantidade;

        totalQuantidade += data.quantidade;

        tipoObjetivo = data.tipoObjetivo;

        objetivoFinanceiro = data.objetivoFinanceiro;

        nome = data.nome;

      });

      // 2. Buscar o preço atual na coleção 'acoesDividendos'

      const cotacaoSnap = await db.collection("acoesDividendos").doc(tickerAtual).get();

      if (!cotacaoSnap.exists) {

        alert("⚠️ Cotação não encontrada para esse ticker.");

        return;

      }

      const valorAtual = cotacaoSnap.data().valorStock;

      const valorTotalAtual = valorAtual \* totalQuantidade;

      let progresso = 0;

      let mensagem = "";

      if (tipoObjetivo === "lucro") {

        const lucroAtual = valorTotalAtual - totalInvestido;

        progresso = (lucroAtual / objetivoFinanceiro) \* 100;

        const lucroEmFalta = objetivoFinanceiro - lucroAtual;

        const tpNecessario = (totalInvestido + objetivoFinanceiro) / totalQuantidade;

        mensagem = `

          🎯 Objetivo: Lucro de €${objetivoFinanceiro.toFixed(2)}<br>

          📊 Progresso: ${progresso.toFixed(1)}%<br>

          💰 Preço atual: €${valorAtual.toFixed(2)}<br>

          🎯 Preço alvo necessário (TP2): €${tpNecessario.toFixed(2)}

        `;

      } else if (tipoObjetivo === "valorFinal") {

        progresso = (valorTotalAtual / objetivoFinanceiro) \* 100;

        mensagem = `

          🎯 Objetivo: Valor final de €${objetivoFinanceiro.toFixed(2)}<br>

          📊 Progresso: ${progresso.toFixed(1)}%<br>

          💰 Valor atual da posição: €${valorTotalAtual.toFixed(2)}

        `;

      } else if (tipoObjetivo === "quantidade") {

        progresso = (totalQuantidade / objetivoFinanceiro) \* 100;

        mensagem = `

          🎯 Objetivo: ${objetivoFinanceiro} unidades<br>

          📊 Progresso: ${progresso.toFixed(1)}%<br>

          💼 Já tens: ${totalQuantidade} unidades

        `;

      }

      document.getElementById("analiseResultados").innerHTML = `

        <h3>${nome} (${tickerAtual})</h3>

        <p>${mensagem}</p>

      `;

    } catch (error) {

      console.error("Erro ao analisar objetivo:", error);

      alert("❌ Erro ao consultar Firestore.");

    }

  }

  //função para mostrar situação atual:

  function analisarObjetivos(tickerAtual, precoAtual) {

    db.collection("ativos")

      .where("ticker", "==", tickerAtual.toUpperCase())

      .get()

      .then((querySnapshot) => {

        if (querySnapshot.empty) {

          alert("⚠️ Nenhuma compra encontrada para esse ticker.");

          return;

        }

        let totalInvestido = 0;

        let totalQuantidade = 0;

        let objetivoFinanceiro = 0;

        let nomeAtivo = "";

        querySnapshot.forEach((doc) => {

          const data = doc.data();

          totalInvestido += data.valorInvestido || 0; // corrigido: era 'totalInvestido'

          totalQuantidade += data.quantidade || 0;

          objetivoFinanceiro = data.objetivoFinanceiro || 0;

          nomeAtivo = data.nome;

        });

        const valorAtual = totalQuantidade \* precoAtual;

        const lucroAtual = valorAtual - totalInvestido;

        const faltaParaObjetivo = objetivoFinanceiro - valorAtual;

        let html = `

          <h3>📈 Situação Atual: ${nomeAtivo} (${tickerAtual.toUpperCase()})</h3>

          <ul>

            <li><strong>Total Investido:</strong> €${totalInvestido.toFixed(2)}</li>

            <li><strong>Quantidade Total:</strong> ${totalQuantidade}</li>

            <li><strong>Preço Atual:</strong> €${precoAtual.toFixed(2)}</li>

            <li><strong>Valor Atual:</strong> €${valorAtual.toFixed(2)}</li>

            <li><strong>Lucro/Perda:</strong> €${lucroAtual.toFixed(2)}</li>

            <li><strong>Objetivo Financeiro:</strong> €${objetivoFinanceiro.toFixed(2)}</li>

        `;

        if (faltaParaObjetivo > 0) {

          const quantidadeNecessaria = faltaParaObjetivo / precoAtual;

          html += `<li style="color:orange;"><strong>⚠️ Para atingir o objetivo, investe mais €${faltaParaObjetivo.toFixed(2)} (${quantidadeNecessaria.toFixed(2)} unidades)</strong></li>`;

        } else {

          html += `<li style="color:green;"><strong>🎯 Objetivo já atingido!</strong></li>`;

        }

        html += `</ul>`;

        document.getElementById("analiseResultados").innerHTML = html;

      })

      .catch((error) => {

        console.error("❌ Erro na análise:", error);

        alert("Erro ao consultar dados do Firestore.");

      });

  }

  function obterPrecoAtualDoAtivo(ticker) {

    return fetch(`https://api.exemplo.com/preco?ticker=${ticker}`)

      .then((res) => res.json())

      .then((data) => data.precoAtual || null)

      .catch(() => null);

  }

  //lista de progressos

  async function listarProgressoDosAtivos() {

    const container = document.getElementById("listaProgressoAtivos");

    container.innerHTML = "🔄 A carregar...";

    try {

      const snapshot = await db.collection("ativos").get();

      if (snapshot.empty) {

        container.innerHTML = "❗ Não há ativos registados.";

        return;

      }

      const promessas = [];

      snapshot.forEach((doc) => {

        const data = doc.data();

        promessas.push(verificarProgressoDeItem(data));

      });

      const resultados = await Promise.all(promessas);

      container.innerHTML = resultados.join("<hr>");

    } catch (error) {

      console.error("Erro ao listar progresso:", error);

      container.innerHTML = "❌ Erro ao carregar progresso.";

    }

  }

async function verificarProgressoDeItem(data) {

    try {

      const cotacaoSnap = await db.collection("acoesDividendos").doc(data.ticker).get();

      if (!cotacaoSnap.exists) {

        return `<strong>${data.nome} (${data.ticker})</strong><br>❌ Cotação não encontrada.`;

      }

      const cotacao = cotacaoSnap.data();

      const valorAtual = cotacao.valorStock;

      const totalInvestido = data.precoCompra \* data.quantidade;

      const valorTotalAtual = valorAtual \* data.quantidade;

      let progresso = 0;

      let mensagem = "";

      // ➕ Novos cálculos de crescimento

      let crescimentoNecessario = 0;

      let tpNecessario = 0;

      const estimativas = [];

      if (data.tipoObjetivo === "lucro") {

        const lucroAtual = valorTotalAtual - totalInvestido;

        progresso = (lucroAtual / data.objetivoFinanceiro) \* 100;

        tpNecessario = (totalInvestido + data.objetivoFinanceiro) / data.quantidade;

        crescimentoNecessario = ((tpNecessario - valorAtual) / valorAtual) \* 100;

        // Obter taxas de crescimento do documento

        const crescimentoSemanal = cotacao.taxaCrescimento\_1s;

        const crescimentoMensal = cotacao.taxaCrescimento\_1m;

        const crescimentoAnual = cotacao.taxaCrescimento\_1ano;

        if (crescimentoSemanal > 0) {

          const semanas = Math.ceil(crescimentoNecessario / crescimentoSemanal);

          estimativas.push(`📅 TP2 em ~${semanas} semanas`);

        } else {

          estimativas.push("❌ Sem taxa semanal");

        }

        if (crescimentoMensal > 0) {

          const meses = Math.ceil(crescimentoNecessario / crescimentoMensal);

          estimativas.push(`📅 TP2 em ~${meses} meses`);

        } else {

          estimativas.push("❌ Sem taxa mensal");

        }

        if (crescimentoAnual > 0) {

          const anos = Math.ceil(crescimentoNecessario / crescimentoAnual);

          estimativas.push(`📅 TP2 em ~${anos} anos`);

        } else {

          estimativas.push("❌ Sem taxa anual");

        }

        mensagem = `

          🎯 Lucro alvo: €${data.objetivoFinanceiro.toFixed(2)}<br>

          📈 Progresso: ${progresso.toFixed(1)}%<br>

          💰 Preço atual: €${valorAtual.toFixed(2)}<br>

          🎯 TP2 necessário: €${tpNecessario.toFixed(2)}<br>

          📊 Crescimento necessário: ${crescimentoNecessario.toFixed(2)}%<br>

          ${estimativas.join("<br>")}

        `;

      }

      // ⚠️ Podes depois replicar a lógica das estimativas para os outros tipos de objetivo se quiseres.

      return `<strong>${data.nome} (${data.ticker})</strong><br>${mensagem}`;

    } catch (err) {

      return `<strong>${data.nome} (${data.ticker})</strong><br>❌ Erro ao consultar cotação.`;

    }

}

  function abrirPopupProgresso() {

    document.getElementById("popupProgresso").classList.remove("hidden");

    listarProgressoDosAtivos();

  }

  function fecharPopupProgresso() {

    document.getElementById("popupProgresso").classList.add("hidden");

  }

  //JS para gerar linhas dinamicamente

  function mostrarProgressoAtivos(dadosAtivos) {

    const container = document.getElementById("listaProgressoAtivos");

    container.innerHTML = "";

    dadosAtivos.forEach((ativo) => {

      const {

        nomeAtivo,

        tickerAtivo,

        precoCompra,

        quantidade,

        tipoObjetivo,

        objetivoFinanceiro,

        valorStockAtual, // cotação atual

      } = ativo;

      // Calcula TP2 com base no objetivo

      const tp2 = tipoObjetivo === "lucro"

        ? calcularTp2(precoCompra, objetivoFinanceiro, quantidade)

        : objetivoFinanceiro;

      const progresso = valorStockAtual && tp2

        ? ((valorStockAtual - precoCompra) / (tp2 - precoCompra)) \* 100

        : 0;

      // Cria o bloco HTML com os dados

      const html = `

        <div class="ativo-box">

          <h4>${nomeAtivo} (${tickerAtivo})</h4>

          <ul>

            <li>Preço de compra: €${precoCompra}</li>

            <li>Quantidade: ${quantidade}</li>

            <li>Objetivo: ${tipoObjetivo === "lucro" ? `Lucro de €${objetivoFinanceiro}` : `€${objetivoFinanceiro}`}</li>

            <li>TP2 (Preço objetivo): <strong>€${tp2.toFixed(2)}</strong></li>

            <li>Cotação atual: ${valorStockAtual ? `€${valorStockAtual}` : "❌ Não encontrada"}</li>

            <li>Progresso: ${valorStockAtual ? `${progresso.toFixed(1)}%` : "N/A"}</li>

          </ul>

        </div>

      `;

      container.innerHTML += html;

    });

  }

**AppsScript**

**Appscript.json**

{

  "timeZone": "Europe/Lisbon",

  "dependencies": {},

  "exceptionLogging": "STACKDRIVER",

  "runtimeVersion": "V8",

  "oauthScopes": [

    "https://www.googleapis.com/auth/spreadsheets",

    "https://www.googleapis.com/auth/script.external\_request",

    "https://www.googleapis.com/auth/datastore"

  ]

}

**Botões.gs**

function onOpen() {

  const ui = SpreadsheetApp.getUi();

  ui.createMenu('🔁 Firebase Sync')

    .addItem('1️⃣ Gravar/Atualizar na Firebase', 'enviarParaFirebase')

    .addItem('2️⃣ Importar só os que faltam', 'importarFirebaseParaSheet')

    .addItem('3️⃣ Eliminar incompletos em ambos', 'eliminarIncompletos')

    .addItem('[4] Eliminar BD Firebase', 'apagarTodosDocumentosFirebase')

    .addToUi();

}

**EnviarParaFirebse**

function enviarParaFirebase() {

  const ss = SpreadsheetApp.getActiveSpreadsheet();

  const sheet = ss.getSheetByName("Firebase");

  if (!sheet) {

    Logger.log("❌ A folha 'Firebase' não foi encontrada.");

    return;

  }

  const data = sheet.getRange(2, 1, sheet.getLastRow() - 1, 11).getValues();

  const headers = {

    Authorization: 'Bearer ' + ScriptApp.getOAuthToken()

  };

  const firestoreUrl = 'https://firestore.googleapis.com/v1/projects/appfinance-812b2/databases/(default)/documents/acoesDividendos?pageSize=1000';

  const response = UrlFetchApp.fetch(firestoreUrl, { method: 'get', headers });

  const firestoreDocs = JSON.parse(response.getContentText()).documents || [];

  const firestoreMap = {};

  firestoreDocs.forEach(doc => {

    const fields = doc.fields;

    const ticker = fields.ticker?.stringValue;

    if (ticker) {

      firestoreMap[ticker] = {

        docName: doc.name,

        fields

      };

    }

  });

  data.forEach(row => {

    const [nome, ticker, setor, mercado, valorStock, dividendo, periodicidade, mes, taxa1s, taxa1m, taxa1a] = row;

    // Ignorar linhas com dados essenciais em branco

    if (!ticker || !nome || !setor || !mercado || !periodicidade || !mes) {

      Logger.log(`⚠️ Linha ignorada por dados incompletos: ${ticker}`);

      return;

    }

  const payloadFields = {

  nome: { stringValue: String(nome || "") },

  ticker: { stringValue: String(ticker || "") },

  setor: { stringValue: String(setor || "") },

  mercado: { stringValue: String(mercado || "") },

  valorStock: { doubleValue: parseFloat(valorStock) || 0 },

  dividendo: { doubleValue: parseFloat(dividendo) || 0 },

  periodicidade: { stringValue: String(periodicidade || "") },

  mes: { stringValue: String(mes || "") },

  taxaCrescimento\_1s: { doubleValue: parseFloat(taxa1s) || 0 },

  taxaCrescimento\_1m: { doubleValue: parseFloat(taxa1m) || 0 },

  taxaCrescimento\_1ano: { doubleValue: parseFloat(taxa1a) || 0 },

  origem: { stringValue: "sheets" }

};

    const existing = firestoreMap[ticker];

    if (!existing) {

      const createUrl = `https://firestore.googleapis.com/v1/projects/appfinance-812b2/databases/(default)/documents/acoesDividendos?documentId=${ticker}`;

      UrlFetchApp.fetch(createUrl, {

        method: 'POST',

        contentType: 'application/json',

        payload: JSON.stringify({ fields: payloadFields }),

        headers

      });

      Logger.log(`📄 Criado: ${ticker}`);

    } else {

      const isDifferent = Object.keys(payloadFields).some(key => {

        const newVal = JSON.stringify(payloadFields[key]);

        const oldVal = JSON.stringify(existing.fields[key]);

        return newVal !== oldVal;

      });

      if (isDifferent) {

        const updateUrl = `https://firestore.googleapis.com/v1/${existing.docName}`;

        UrlFetchApp.fetch(updateUrl, {

          method: 'PATCH',

          contentType: 'application/json',

          payload: JSON.stringify({ fields: payloadFields }),

          headers

        });

        Logger.log(`🔄 Atualizado: ${ticker}`);

      } else {

        Logger.log(`✅ Sem alterações: ${ticker}`);

      }

    }

    if (!["Mensal", "Trimestral", "Semestral", "Anual"].includes(String(periodicidade))) {

        Logger.log(`⚠️ Periodicidade inválida para ${ticker}: ${periodicidade}`);

    }

  });

  Logger.log("✅ Sincronização concluída.");

}

importarFirebaseParaSheet

//Importar da Firebase

function importarFirebaseParaSheet() {

  const ss = SpreadsheetApp.getActiveSpreadsheet();

  const sheet = ss.getSheetByName("Firebase");

  if (!sheet) {

    Logger.log("❌ A folha 'Firebase' não foi encontrada.");

    return;

  }

  const headers = {

    Authorization: 'Bearer ' + ScriptApp.getOAuthToken()

  };

  const firestoreUrl = 'https://firestore.googleapis.com/v1/projects/appfinance-812b2/databases/(default)/documents/acoesDividendos?pageSize=1000';

  const response = UrlFetchApp.fetch(firestoreUrl, { method: 'get', headers });

  const firestoreDocs = JSON.parse(response.getContentText()).documents || [];

  const existingTickers = sheet.getRange(2, 2, sheet.getLastRow() - 1).getValues().flat();

  firestoreDocs.forEach(doc => {

    const fields = doc.fields;

    const ticker = fields.ticker?.stringValue;

    if (ticker && !existingTickers.includes(ticker)) {

      const novaLinha = [

        fields.nome?.stringValue || "",

        ticker,

        fields.setor?.stringValue || "",

        fields.mercado?.stringValue || "",

        fields.valorStock?.doubleValue || 0,

        fields.dividendo?.doubleValue || 0,

        fields.periodicidade?.stringValue || "",

        fields.mes?.stringValue || ""

      ];

      sheet.appendRow(novaLinha);

      Logger.log(`⬇️ Importado para Sheet: ${ticker}`);

    }

  });

  Logger.log("✅ Importação da Firebase concluída.");

}

EliminarIncompletos

//Importar da Firebase

function importarFirebaseParaSheet() {

  const ss = SpreadsheetApp.getActiveSpreadsheet();

  const sheet = ss.getSheetByName("Firebase");

  if (!sheet) {

    Logger.log("❌ A folha 'Firebase' não foi encontrada.");

    return;

  }

  const headers = {

    Authorization: 'Bearer ' + ScriptApp.getOAuthToken()

  };

  const firestoreUrl = 'https://firestore.googleapis.com/v1/projects/appfinance-812b2/databases/(default)/documents/acoesDividendos?pageSize=1000';

  const response = UrlFetchApp.fetch(firestoreUrl, { method: 'get', headers });

  const firestoreDocs = JSON.parse(response.getContentText()).documents || [];

  const existingTickers = sheet.getRange(2, 2, sheet.getLastRow() - 1).getValues().flat();

  firestoreDocs.forEach(doc => {

    const fields = doc.fields;

    const ticker = fields.ticker?.stringValue;

    if (ticker && !existingTickers.includes(ticker)) {

      const novaLinha = [

        fields.nome?.stringValue || "",

        ticker,

        fields.setor?.stringValue || "",

        fields.mercado?.stringValue || "",

        fields.valorStock?.doubleValue || 0,

        fields.dividendo?.doubleValue || 0,

        fields.periodicidade?.stringValue || "",

        fields.mes?.stringValue || ""

      ];

      sheet.appendRow(novaLinha);

      Logger.log(`⬇️ Importado para Sheet: ${ticker}`);

    }

  });

  Logger.log("✅ Importação da Firebase concluída.");

}

Eliminar BD Firebase

async function apagarTodosDocumentosFirebase() {

  const projectId = "appfinance-812b2"; // ⚠️ Altera se usares outro ID

  const urlBase = `https://firestore.googleapis.com/v1/projects/${projectId}/databases/(default)/documents/acoesDividendos`;

  const token = ScriptApp.getOAuthToken();

  // Obter todos os documentos da coleção

  const response = UrlFetchApp.fetch(urlBase, {

    method: "get",

    headers: {

      Authorization: `Bearer ${token}`

    },

    muteHttpExceptions: true

  });

  const dados = JSON.parse(response.getContentText());

  if (!dados.documents || dados.documents.length === 0) {

    Logger.log("❌ Nenhum documento encontrado na Firebase.");

    return;

  }

  let contador = 0;

  // Apagar cada documento individualmente

  dados.documents.forEach(doc => {

    const deleteUrl = `https://firestore.googleapis.com/v1/${doc.name}`;

    const deleteResponse = UrlFetchApp.fetch(deleteUrl, {

      method: "delete",

      headers: {

        Authorization: `Bearer ${token}`

      },

      muteHttpExceptions: true

    });

    if (deleteResponse.getResponseCode() === 200) {

      contador++;

      Logger.log(`✅ Documento apagado: ${doc.name}`);

    } else {

      Logger.log(`⚠️ Erro ao apagar: ${doc.name}`);

    }

  });

  Logger.log(`✅ ${contador} documentos apagados com sucesso.`);

}