Dashboard.html

<!doctype html>

<html lang="es">

<head>

  <meta charset="utf-8" />

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

  <title>Dashboard Administrador – SDCS</title>

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

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

  <!-- Supabase v2 -->

  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>

  <style>

    /\* realce de selección en listado \*/

    .list .item.sel{ outline:2px solid #1d4ed8; border-radius:10px; background:#eef2ff; }

    .controls-box{ border:1px solid var(--border); border-radius:12px; padding:12px }

    .controls-stack{ display:flex; flex-wrap:wrap; gap:10px }

    .btn-ghost{ padding:12px 14px; border:1px solid var(--border); border-radius:12px; background:#fff; cursor:pointer }

    .btn-accent{ background:#1d4ed8; border-color:#1d4ed8; color:#fff }

    .sep{ border:none; border-top:1px solid var(--border); margin:16px 0 }

    .editor{ min-height:220px; border:1px solid var(--border); border-radius:12px; padding:12px; background:#fff }

    .toolbar{ display:flex; gap:8px; flex-wrap:wrap; margin:10px 0 }

    .res-row{ display:grid; grid-template-columns: 140px 1fr 1fr 1fr; gap:10px; align-items:center }

    .res-list .item{ padding:8px 10px; border:1px solid var(--border); border-radius:10px; background:#fff }

    .modal{ position:fixed; inset:0; display:none; place-items:center; background:rgba(0,0,0,.4) }

    .modal.show{ display:grid }

    .modal\_\_inner{ width:min(960px,94vw); height:min(80vh,90vh); background:#fff; border-radius:14px; padding:18px; overflow:auto }

    .grid3{ display:grid; grid-template-columns: repeat(3, minmax(0,1fr)); gap:12px }

    .selects{ display:flex; gap:12px; align-items:end; flex-wrap:wrap; margin:10px 0 }

    .row{ display:grid; grid-template-columns: 1fr 1fr; gap:12px }

    @media (max-width:720px){ .row{grid-template-columns:1fr} .res-row{grid-template-columns:1fr} }

  </style>

</head>

<body>

  <main class="card dash card-dash">

    <header class="dash\_\_header">

      <h1 class="h">Dashboard Administrador</h1>

      <p class="p">Gestión de <strong>Qué vamos a aprender</strong>, <strong>Proyectos</strong> y <strong>Clases</strong>.</p>

    </header>

    <section class="grid3">

      <button id="btn-qa"    class="btn-ghost">🧭 Qué vamos a aprender</button>

      <button id="btn-proy"  class="btn-ghost">🧩 Proyecto</button>

      <button id="btn-class" class="btn-ghost">📚 Clases</button>

    </section>

    <section id="panel" class="panel"></section>

  </main>

  <script type="module">

  // --- Guard de sesión: si no hay admin en sessionStorage, vuelve al login ---

  try {

    const admin = JSON.parse(sessionStorage.getItem('sdcs\_admin\_user') || 'null');

    if (!admin) window.location.href = './administrador.html';

  } catch (\_) { window.location.href = './administrador.html'; }

  // --- Cliente Supabase exclusivo del Admin ---

  import { supabase } from './adminSupabaseClient.js';

  const $=(s,c=document)=>c.querySelector(s);

  const panel=$('#panel');

  const GRADOS=Array.from({length:11},(\_,i)=>i+1);

  const TRIMESTRES=[1,2,3];

  const FOLDERS={pdf:'pdf',img:'img',audio:'audio',video:'video'};

  $('#btn-class').addEventListener('click',uiClases);

  $('#btn-proy').addEventListener('click',uiProyecto);

  $('#btn-qa').addEventListener('click',uiQueAprender);

  /\* ============================== HELPERS ============================== \*/

  function esc(s=''){return s.replace(/[&<>"']/g,ch=>({'&':'&amp;','<':'&lt;','>':'&gt;','"':'&quot;',"'":'&#39;'}[ch]));}

  async function uploadToStorage(grado,tipo,file){

    const bucket=String(grado), folder=FOLDERS[tipo]||'misc';

    const path = `${folder}/${Date.now()}\_${file.name}`;

    const up=await supabase.storage.from(bucket).upload(path,file,{ upsert:true });

    if(up.error) throw up.error;

    return supabase.storage.from(bucket).getPublicUrl(path).data.publicUrl;

  }

  async function tryInsert(tabla, payload){

    try{

      const { error } = await supabase.from(tabla).insert([payload]);

      if(error) console.warn(`No se pudo insertar en ${tabla}:`, error.message);

    }catch(e){ console.warn(`No se pudo insertar en ${tabla}:`, e?.message||e); }

  }

  /\* =============================== CLASES ============================== \*/

  function uiClases(){

    panel.innerHTML=`

      <h2 class="h">📚 Clases</h2>

      <div class="selects selects--tight">

        <label>Grado <select id="gradoSel">${GRADOS.map(g=>`<option value="${g}">${g}°</option>`).join('')}</select></label>

        <label>Trimestre <select id="triSel">${TRIMESTRES.map(t=>`<option value="${t}">${t}</option>`).join('')}</select></label>

      </div>

      <fieldset class="controls-box">

        <legend>Controles</legend>

        <div class="controls-stack">

          <button id="btn-listar"   class="btn">Listar clases</button>

          <button id="btn-disenar"  class="btn btn-accent">Diseñar clase</button>

          <button id="btn-editar"   class="btn" disabled>Editar clase creada</button>

          <button id="btn-eliminar" class="btn" disabled>Eliminar clase</button>

          <button id="btn-preview"  class="btn btn-accent" disabled>Vista de estudiante</button>

        </div>

      </fieldset>

      <div id="info" class="p"></div>

      <hr class="sep"/>

      <div id="editorWrap" style="display:none"></div>

      <div id="lista"></div>

      <div id="modal" class="modal"><div class="modal\_\_inner" id="modalInner"></div></div>

    `;

    const gradoSel=$('#gradoSel'), triSel=$('#triSel');

    const btnListar=$('#btn-listar'), btnDis=$('#btn-disenar');

    const btnEdit=$('#btn-editar'), btnDel=$('#btn-eliminar'), btnPrev=$('#btn-preview');

    const listaEl=$('#lista'), infoEl=$('#info'), editorWrap=$('#editorWrap');

    const modal=$('#modal'), modalInner=$('#modalInner');

    let clases=[], claseAct=null, recursosTmp=[], recursosDB=[];

    function setHasClases(has){ btnEdit.disabled=!has; btnDel.disabled=!has; btnPrev.disabled=!has && !claseAct; }

    async function cargarListado(){

      const grado=parseInt(gradoSel.value,10), tri=parseInt(triSel.value,10);

      const {data,error}=await supabase.from('clases')

        .select('id,grado,trimestre,numero\_clase,nombre,descripcion,fecha\_ejecucion,publicado,fecha\_publicacion')

        .eq('grado',grado).eq('trimestre',tri).order('numero\_clase',{ascending:true});

      if(error){ infoEl.textContent='Error al listar clases.'; console.error(error); return; }

      clases=data||[]; renderListado(); setHasClases(clases.length>0);

    }

    function renderListado(){

      if(!clases.length){ listaEl.innerHTML='<div class="p">No hay clases creadas.</div>'; return; }

      listaEl.innerHTML = `<div class="list">

        ${clases.map(c=>`<div class="item" data-id="${c.id}">

            <strong>Clase ${c.numero\_clase}</strong> · ${esc(c.nombre)}

            ${c.descripcion?`<span class="badge">${esc(c.descripcion)}</span>`:''}

            ${c.publicado?'<span class="badge">PUBLICADA</span>':''}

        </div>`).join('')}

      </div>`;

    }

    async function numeroSiguiente(grado,tri){

      const {data,error}=await supabase.from('clases').select('numero\_clase')

        .eq('grado',grado).eq('trimestre',tri).order('numero\_clase',{ascending:false}).limit(1);

      if(error||!data?.length) return 1; return (data[0].numero\_clase||0)+1;

    }

    function buildEditorUI({grado,tri,clase}){

      const isEdit=!!clase, numero=clase?clase.numero\_clase:0, nombreSugerido=clase?clase.nombre:`Clase ${numero}`;

      editorWrap.style.display='block';

      editorWrap.innerHTML=`

        <h3 class="h">${isEdit?'Editar clase':'Diseñar clase'}</h3>

        <div class="row row--tight">

          <label>Título de la clase

            <input id="e-nombre" type="text" value="${esc(nombreSugerido)}" placeholder="Ej. Los carnívoros"/>

          </label>

          <label>Descripción corta

            <input id="e-desc" type="text" value="${esc(clase?.descripcion||'')}" placeholder="Resumen en una frase"/>

          </label>

        </div>

        <div class="row row--tight">

          <label>Fecha de ejecución

            <input id="e-fecha" type="date" value="${clase?.fecha\_ejecucion||''}"/>

          </label>

          <div>

            <div class="p"><strong>Clase #:</strong> ${numero}</div>

            <div class="p"><strong>Grado:</strong> ${grado}° · <strong>Trimestre:</strong> ${tri}</div>

          </div>

        </div>

        <div class="toolbar">

          <button data-cmd="bold" title="Negrita"><b>B</b></button>

          <button data-cmd="italic" title="Itálica"><i>I</i></button>

          <button data-cmd="underline" title="Subrayado"><u>U</u></button>

          <select id="e-size" title="Tamaño">

            <option value="3">Tamaño</option>

            <option value="2">Pequeño</option>

            <option value="3">Normal</option>

            <option value="5">Grande</option>

            <option value="6">Muy grande</option>

          </select>

          <input id="e-color" type="color" title="Color de texto"/>

          <button id="e-link" title="Insertar enlace">Link</button>

        </div>

        <div id="e-editor" class="editor" contenteditable="true">${clase?.contenido||''}</div>

        <hr class="sep"/>

        <h4 class="h">Recursos</h4>

        <div class="res-row">

          <select id="r-tipo">

            <option value="pdf">PDF</option><option value="img">Imagen</option>

            <option value="audio">Audio</option><option value="video">Video (archivo o YouTube)</option>

          </select>

          <input id="r-etiqueta" type="text" placeholder="Etiqueta (ej. lectura, robot, intro)"/>

          <input id="r-url" type="text" placeholder="URL (YouTube u otro enlace)"/>

          <input id="r-file" type="file"/>

        </div>

        <div class="controls controls--tight">

          <button id="r-add" class="btn">Agregar recurso</button>

        </div>

        <div id="r-list" class="res-list"></div>

        <hr class="sep"/>

        <fieldset class="controls-box">

          <legend>Controles</legend>

          <div class="controls-stack">

            <button id="e-guardar"  class="btn">Guardar borrador</button>

            <button id="e-publicar" class="btn btn-accent">Publicar clase</button>

            <button id="e-cerrar"   class="btn">Cerrar editor</button>

          </div>

        </fieldset>

      `;

      // Toolbar

      editorWrap.querySelectorAll('.toolbar [data-cmd]').forEach(b=>b.addEventListener('click',()=>document.execCommand(b.dataset.cmd,false,null)));

      $('#e-size').addEventListener('change',e=>document.execCommand('fontSize',false,e.target.value));

      $('#e-color').addEventListener('change',e=>document.execCommand('foreColor',false,e.target.value));

      $('#e-link').addEventListener('click',()=>{ const url=prompt('URL (incluye https://)'); if(url) document.execCommand('createLink',false,url); });

      function renderRecursos(){

        const todos=[...recursosTmp,...recursosDB];

        $('#r-list').innerHTML = todos.length

          ? todos.map(r=>`<div class="item">${r.tipo\_recurso.toUpperCase()} · ${esc(r.nombre)} <small class="badge">${r.url?'URL':'Archivo'}</small></div>`).join('')

          : `<div class="p">Sin recursos.</div>`;

      }

      renderRecursos();

      $('#r-add').addEventListener('click', async ()=>{

        const tipo=$('#r-tipo').value, etiqueta=$('#r-etiqueta').value.trim(), urlText=$('#r-url').value.trim(), file=$('#r-file').files[0];

        if(!etiqueta){ alert('Ingresa una etiqueta.'); return; }

        let finalUrl='';

        if(urlText){ finalUrl=urlText; }

        else if(file){

          try{ finalUrl=await uploadToStorage(grado,tipo,file); }

          catch(err){ console.error(err); alert('No se pudo subir el archivo'); return; }

        } else { alert('Archivo o URL requerido.'); return; }

        recursosTmp.push({ tipo\_recurso:tipo, nombre:etiqueta, url:finalUrl });

        $('#r-etiqueta').value=''; $('#r-url').value=''; $('#r-file').value='';

        renderRecursos();

      });

      $('#e-guardar').addEventListener('click', async ()=>{

        const payload={ grado, trimestre:tri, numero\_clase:numero,

          nombre:$('#e-nombre').value.trim()||`Clase ${numero}`,

          descripcion:$('#e-desc').value.trim(),

          fecha\_ejecucion:$('#e-fecha').value||null,

          contenido:$('#e-editor').innerHTML };

        if(clase&&clase.id){

          const {error}=await supabase.from('clases').update(payload).eq('id',clase.id);

          if(error){ console.error(error); alert('No se pudo guardar'); return; }

          for(const r of recursosTmp){ await supabase.from('recursos').insert([{ clase\_id:clase.id, ...r }]); }

          recursosDB=[...recursosDB,...recursosTmp]; recursosTmp=[];

          alert('Cambios guardados.');

        } else {

          const ins=await supabase.from('clases').insert([payload]).select('id');

          if(ins.error){ console.error(ins.error); alert('No se pudo guardar'); return; }

          clase={ id:ins.data[0].id, ...payload }; claseAct=clase;

          for(const r of recursosTmp){ await supabase.from('recursos').insert([{ clase\_id:clase.id, ...r }]); }

          recursosDB=recursosTmp.splice(0,recursosTmp.length);

          alert(`Borrador guardado. Clase #${numero}.`);

        }

        await cargarListado();

      });

      $('#e-publicar').addEventListener('click', async ()=>{

        if(!clase?.id){ $('#e-guardar').click(); return; }

        const {error}=await supabase.from('clases').update({ publicado:true, fecha\_publicacion:new Date().toISOString() }).eq('id',clase.id);

        if(error){ console.error(error); alert('No se pudo publicar'); return; }

        alert(`Clase publicada: Clase ${numero} · ${$('#e-nombre').value}`);

        await cargarListado();

      });

      $('#e-cerrar').addEventListener('click',()=>{ editorWrap.style.display='none'; claseAct=null; recursosTmp=[]; recursosDB=[]; });

      btnPrev.onclick=()=>{

        const recursos=[...recursosTmp,...recursosDB];

        modalInner.innerHTML=`

          <h3 class="h">${$('#e-nombre').value}</h3>

          <p class="p">${$('#e-desc').value}</p>

          <div>${$('#e-editor').innerHTML}</div>

          <hr class="sep"/><h4 class="h">Recursos</h4>

          <div class="grid3">${recursos.map(r=>`<a class="item" href="${r.url}" target="\_blank" rel="noopener"><strong>${esc(r.nombre)}</strong><div class="badge">${r.tipo\_recurso}</div></a>`).join('')}</div>`;

        modal.classList.add('show');

      };

      modal.onclick=(e)=>{ if(e.target===modal) modal.classList.remove('show'); };

    } // buildEditorUI

    // === Listar ===

    btnListar.addEventListener('click',cargarListado);

    // === Diseñar ===

    btnDis.addEventListener('click', async ()=>{

      const grado=parseInt($('#gradoSel').value,10), tri=parseInt($('#triSel').value,10);

      const numero=await (async(g,t)=>{ const r=await supabase.from('clases').select('numero\_clase').eq('grado',g).eq('trimestre',t).order('numero\_clase',{ascending:false}).limit(1); return r.error||!r.data?.length?1:(r.data[0].numero\_clase||0)+1; })(grado,tri);

      $('#info').innerHTML=`Creando <strong>Clase ${numero}</strong> para <strong>${grado}°</strong>, trimestre <strong>${tri}</strong>.`;

      buildEditorUI({ grado, tri, clase:{ numero\_clase:numero, nombre:`Clase ${numero}` } });

      // Eliminar deshabilitado hasta seleccionar/crear algo

      btnDel.disabled = true;

    });

    // === Seleccionar de la lista para Editar / Eliminar ===

    $('#lista').addEventListener('click', async (e)=>{

      const item=e.target.closest('.item'); if(!item) return;

      // marcar selección visual

      $('#lista .item.sel')?.classList.remove('sel');

      item.classList.add('sel');

      const id=Number(item.dataset.id);

      const l=await supabase.from('clases').select('\*').eq('id',id).single();

      const rs=await supabase.from('recursos').select('\*').eq('clase\_id',id);

      const grado=parseInt($('#gradoSel').value,10), tri=parseInt($('#triSel').value,10);

      claseAct=l.data || null;             // <<< clave para eliminar

      recursosDB=rs.data||[];

      buildEditorUI({ grado, tri, clase:claseAct });

      $('#info').innerHTML=`Editando <strong>Clase ${claseAct.numero\_clase}</strong>.`;

      // Habilitar acciones

      btnEdit.disabled=false; btnDel.disabled=false; btnPrev.disabled=false;

    });

    // === Eliminar clase seleccionada ===

    btnDel.addEventListener('click', async ()=>{

      if(!claseAct?.id){ alert('Primero selecciona una clase de la lista.'); return; }

      const confirmMsg = `¿Eliminar definitivamente la Clase ${claseAct.numero\_clase}: "${claseAct.nombre}"?\nSe borrarán también sus recursos.`;

      if(!confirm(confirmMsg)) return;

      // 1) Borrar recursos vinculados (si la tabla existe / tiene RLS de delete)

      const delRec = await supabase.from('recursos').delete().eq('clase\_id', claseAct.id);

      if(delRec.error && delRec.error.code !== 'PGRST116'){ // PGRST116 = table not found (por si no existe)

        console.warn('Error borrando recursos:', delRec.error);

        alert('No se pudieron borrar los recursos (revisa políticas RLS).');

        return;

      }

      // 2) Borrar la clase

      const delClase = await supabase.from('clases').delete().eq('id', claseAct.id);

      if(delClase.error){

        console.error(delClase.error);

        alert('No se pudo eliminar la clase (revisa políticas RLS).');

        return;

      }

      // Limpieza de UI

      editorWrap.style.display='none';

      $('#modal').classList.remove('show');

      claseAct=null; recursosDB=[]; recursosTmp=[];

      await cargarListado();

      $('#info').innerHTML=`<strong>Clase eliminada.</strong>`;

      btnDel.disabled=true; btnEdit.disabled=true; btnPrev.disabled=true;

      alert('Clase eliminada correctamente.');

    });

  } // uiClases

  /\* =============================== PROYECTO ============================= \*/

  function uiProyecto(){

    panel.innerHTML=`

      <h2 class="h">🧩 Proyecto</h2>

      <div class="selects selects--tight">

        <label>Grado <select id="p-grado">${GRADOS.map(g=>`<option value="${g}">${g}°</option>`).join('')}</select></label>

        <label>Trimestre <select id="p-tri">${TRIMESTRES.map(t=>`<option value="${t}">${t}</option>`).join('')}</select></label>

      </div>

      <div class="row row--tight">

        <label>Nombre <input id="p-nombre" type="text"/></label>

        <label>Descripción <input id="p-desc" type="text"/></label>

      </div>

      <div class="row row--tight">

        <label>Fecha inicio <input id="p-fi" type="date"/></label>

        <label>Fecha final  <input id="p-ff" type="date"/></label>

      </div>

      <h4 class="h">Recursos</h4>

      <div class="res-row">

        <select id="pr-tipo">

          <option value="pdf">PDF</option><option value="img">Imagen</option>

          <option value="audio">Audio</option><option value="video">Video</option>

        </select>

        <input id="pr-etiqueta" type="text" placeholder="Etiqueta"/>

        <input id="pr-url" type="text" placeholder="URL (YouTube u otro enlace)"/>

        <input id="pr-file" type="file"/>

      </div>

      <div class="controls controls--tight">

        <button id="pr-add" class="btn">Agregar recurso</button>

      </div>

      <div id="pr-list" class="res-list"></div>

      <fieldset class="controls-box">

        <legend>Controles</legend>

        <div class="controls-stack">

          <button id="p-guardar" class="btn">Guardar</button>

          <button id="p-publicar" class="btn btn-accent">Publicar</button>

          <button id="p-listar"  class="btn">Listar proyectos</button>

        </div>

      </fieldset>

      <div id="p-lista"></div>`;

    const grado=()=>parseInt($('#p-grado').value,10), tri=()=>parseInt($('#p-tri').value,10);

    const prList=$('#pr-list'); let prTmp=[];

    function renderPR(){ prList.innerHTML = prTmp.length? prTmp.map(r=>`<div class="item">${r.tipo\_recurso.toUpperCase()} · ${esc(r.nombre)} <small class="badge">${r.url?'URL':'Archivo'}</small></div>`).join('') : `<div class="p">Sin recursos.</div>`; }

    renderPR();

    $('#pr-add').onclick=async ()=>{

      const tipo=$('#pr-tipo').value, etiqueta=$('#pr-etiqueta').value.trim(), url=$('#pr-url').value.trim(), file=$('#pr-file').files[0];

      if(!etiqueta){ alert('Ingresa una etiqueta.'); return; }

      let finalUrl=url;

      if(!finalUrl && file){

        try{ finalUrl=await uploadToStorage(grado(),tipo,file); }

        catch(e){ console.error(e); alert('No se pudo subir'); return; }

      }

      if(!finalUrl){ alert('Archivo o URL requerido.'); return; }

      prTmp.push({ tipo\_recurso:tipo, nombre:etiqueta, url:finalUrl });

      $('#pr-etiqueta').value=''; $('#pr-url').value=''; $('#pr-file').value='';

      renderPR();

    };

    $('#p-guardar').onclick=async ()=>{

      const payload={ grado:grado(), trimestre:tri(), nombre:$('#p-nombre').value.trim(), descripcion:$('#p-desc').value.trim(), fecha\_inicio:$('#p-fi').value||null, fecha\_final:$('#p-ff').value||null };

      const ins=await supabase.from('proyectos').insert([payload]).select('id');

      if(ins.error){ console.error(ins.error); alert('No se pudo guardar'); return; }

      const proyecto\_id=ins.data[0].id;

      for(const r of prTmp){ await tryInsert('recursos\_proyecto',{ proyecto\_id, ...r }); }

      alert('Proyecto guardado.');

    };

    $('#p-publicar').onclick=async ()=>{

      const {data,error}=await supabase.from('proyectos').update({ publicado:true, fecha\_publicacion:new Date().toISOString() }).eq('grado',grado()).eq('trimestre',tri()).select('nombre').limit(1);

      if(error){ console.error(error); alert('No se pudo publicar'); return; }

      alert(`Proyecto publicado ${data?.[0]?.nombre?'· '+data[0].nombre:''}`);

    };

    $('#p-listar').onclick=async ()=>{

      const {data,error}=await supabase.from('proyectos').select('\*').eq('grado',grado()).eq('trimestre',tri()).order('id');

      if(error){ console.error(error); return; }

      $('#p-lista').innerHTML = data.length ? `<div class="list">${data.map(p=>`<div class="item"><strong>${esc(p.nombre)}</strong> · ${p.descripcion||''} ${p.publicado?'<span class="badge">PUBLICADO</span>':''}</div>`).join('')}</div>` : `<div class="p">Sin proyectos.</div>`;

    };

  }

  /\* ======================== QUÉ VAMOS A APRENDER ======================= \*/

  function uiQueAprender(){

    panel.innerHTML=`

      <h2 class="h">🧭 Qué vamos a aprender</h2>

      <div class="selects selects--tight">

        <label>Grado <select id="q-grado">${GRADOS.map(g=>`<option value="${g}">${g}°</option>`).join('')}</select></label>

        <label>Trimestre <select id="q-tri">${TRIMESTRES.map(t=>`<option value="${t}">${t}</option>`).join('')}</select></label>

      </div>

      <div class="toolbar">

        <button data-cmd="bold"><b>B</b></button><button data-cmd="italic"><i>I</i></button><button data-cmd="underline"><u>U</u></button>

        <select id="q-size"><option value="3">Tamaño</option><option value="2">Pequeño</option><option value="3">Normal</option><option value="5">Grande</option><option value="6">Muy grande</option></select>

        <input id="q-color" type="color"/><button id="q-link">Link</button>

      </div>

      <div id="q-editor" class="editor" contenteditable="true"></div>

      <h4 class="h">Recursos</h4>

      <div class="res-row">

        <select id="qa-tipo">

          <option value="pdf">PDF</option><option value="img">Imagen</option>

          <option value="audio">Audio</option><option value="video">Video</option>

        </select>

        <input id="qa-etiqueta" type="text" placeholder="Etiqueta"/>

        <input id="qa-url" type="text" placeholder="URL (YouTube u otro enlace)"/>

        <input id="qa-file" type="file"/>

      </div>

      <div class="controls controls--tight"><button id="qa-add" class="btn">Agregar recurso</button></div>

      <div id="qa-list" class="res-list"></div>

      <fieldset class="controls-box">

        <legend>Controles</legend>

        <div class="controls-stack">

          <button id="q-guardar"  class="btn">Guardar</button>

          <button id="q-publicar" class="btn btn-accent">Publicar</button>

          <button id="q-ver"      class="btn">Listar</button>

        </div>

      </fieldset>

      <div id="q-lista"></div>`;

    document.querySelectorAll('.toolbar [data-cmd]').forEach(b=>b.addEventListener('click',()=>document.execCommand(b.dataset.cmd,false,null)));

    $('#q-size').addEventListener('change',e=>document.execCommand('fontSize',false,e.target.value));

    $('#q-color').addEventListener('change',e=>document.execCommand('foreColor',false,e.target.value));

    $('#q-link').addEventListener('click',()=>{ const url=prompt('URL (incluye https://)'); if(url) document.execCommand('createLink',false,url); });

    const grado=()=>parseInt($('#q-grado').value,10), tri=()=>parseInt($('#q-tri').value,10);

    const qaList=$('#qa-list'); let qaTmp=[];

    function renderQA(){ qaList.innerHTML = qaTmp.length? qaTmp.map(r=>`<div class="item">${r.tipo\_recurso.toUpperCase()} · ${esc(r.nombre)} <small class="badge">${r.url?'URL':'Archivo'}</small></div>`).join('') : `<div class="p">Sin recursos.</div>`; }

    renderQA();

    $('#qa-add').onclick=async ()=>{

      const tipo=$('#qa-tipo').value, etiqueta=$('#qa-etiqueta').value.trim(), url=$('#qa-url').value.trim(), file=$('#qa-file').files[0];

      if(!etiqueta){ alert('Ingresa una etiqueta.'); return; }

      let finalUrl=url;

      if(!finalUrl && file){

        try{ finalUrl=await uploadToStorage(grado(),tipo,file); }

        catch(e){ console.error(e); alert('No se pudo subir'); return; }

      }

      if(!finalUrl){ alert('Archivo o URL requerido.'); return; }

      qaTmp.push({ tipo\_recurso:tipo, nombre:etiqueta, url:finalUrl });

      $('#qa-etiqueta').value=''; $('#qa-url').value=''; $('#qa-file').value='';

      renderQA();

    };

    $('#q-guardar').onclick=async ()=>{

      const payload={ grado:grado(), trimestre:tri(), contenido:$('#q-editor').innerHTML };

      const ins=await supabase.from('que\_aprender').insert([payload]).select('id');

      if(ins.error){ console.error(ins.error); alert('No se pudo guardar'); return; }

      const qa\_id=ins.data[0].id;

      for(const r of qaTmp){ await tryInsert('recursos\_qa',{ qa\_id, ...r }); }

      alert('Guardado.');

    };

    $('#q-publicar').onclick=async ()=>{

      const {error}=await supabase.from('que\_aprender').update({ publicado:true, fecha\_publicacion:new Date().toISOString() }).eq('grado',grado()).eq('trimestre',tri());

      if(error){ console.error(error); alert('No se pudo publicar'); return; }

      alert('Publicado.');

    };

    $('#q-ver').onclick=async ()=>{

      const {data,error}=await supabase.from('que\_aprender').select('\*').eq('grado',grado()).eq('trimestre',tri()).order('id');

      if(error){ console.error(error); return; }

      $('#q-lista').innerHTML = data.length ? `<div class="list">${data.map(q=>`<div class="item">${q.publicado?'<span class="badge">PUBLICADO</span> ':''}<span>${q.contenido}</span></div>`).join('')}</div>` : `<div class="p">Sin contenidos.</div>`;

    };

  }

  </script>

</body>

</html>

Dashboard-administrador.html

<!doctype html>

<html lang="es">

<head>

  <meta charset="utf-8" />

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

  <title>Acceso Administrador – SDCS</title>

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

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

  <!-- Supabase JS v2 (requerido por adminSupabaseClient.js) -->

  <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>

</head>

<body>

  <main class="card card-auth">

    <h1 class="h">Administrador</h1>

    <p class="p">Ingresa tu <strong>código</strong> de administrador.</p>

    <label for="codigo">Código</label>

    <input id="codigo" class="input" type="text" inputmode="numeric" placeholder="Ej. 943009" maxlength="6" />

    <button id="btn">Entrar</button>

    <div id="msg" class="msg" style="display:none"></div>

  </main>

  <!-- Lógica de acceso del administrador (importa desde ./adminAuth.js) -->

  <script type="module" src="./administrador.js"></script>

</body>

</html>

Administrador.js

// /administrador/administrador.js

import {

  validarAccesoPorCodigoAdmin,

  redirigirAdmin,

  setUserSession

} from './adminAuth.js';

const $ = (id) => document.getElementById(id);

const btn = $('btn');

const msg = $('msg');

const input = $('codigo');

function showMsg(text, kind = 'err') {

  msg.style.display = 'block';

  msg.className = 'msg ' + (kind || '');

  msg.textContent = text;

}

btn.addEventListener('click', async () => {

  showMsg('', '');

  btn.disabled = true;

  const codigo = input.value.trim();

  if (!/^\d{6}$/.test(codigo)) {

    showMsg('El código debe tener 6 dígitos.');

    btn.disabled = false;

    return;

  }

  const { data, error } = await validarAccesoPorCodigoAdmin(codigo);

  if (error) {

    showMsg('Error: ' + (error.message || 'No se pudo validar.'));

    btn.disabled = false;

    return;

  }

  if (!data) {

    showMsg('Código no válido.');

    btn.disabled = false;

    return;

  }

  if (data.perfil !== 'ADMINISTRADOR') {

    showMsg(`Acceso denegado: tu perfil es ${data.perfil}.`);

    btn.disabled = false;

    return;

  }

  // Guardar “sesión” de admin y redirigir

  setUserSession({ codigo: data.codigo, nombre: data.nombre, perfil: data.perfil });

  showMsg('Acceso concedido. Redirigiendo…', 'ok');

  try {

    redirigirAdmin(data.perfil);

  } catch (e) {

    showMsg(e.message || 'No autorizado.');

  } finally {

    btn.disabled = false;

  }

});

// Enter para enviar

input.addEventListener('keydown', (e) => { if (e.key === 'Enter') btn.click(); });

supabaseClient.js

// /administrador/adminSupabaseClient.js

// Cliente Supabase exclusivo del área Administrador (no interfiere con /js/supabaseClient.js).

// Requiere que el HTML cargue primero el CDN v2:

//   <script src="https://cdn.jsdelivr.net/npm/@supabase/supabase-js@2"></script>

const SUPABASE\_URL = 'https://trfiejowmfgzudhsdpdx.supabase.co';

const SUPABASE\_ANON\_KEY = 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6InRyZmllam93bWZnenVkaHNkcGR4Iiwicm9sZSI6ImFub24iLCJpYXQiOjE3NTY0MjM2MjksImV4cCI6MjA3MTk5OTYyOX0.SCNuSO9uKtaKfGBSi3WMIpvNFmsE66aRROJEgTUrx34';

// Creamos un cliente independiente y lo exportamos con nombre.

export const supabase = window.supabase.createClient(SUPABASE\_URL, SUPABASE\_ANON\_KEY);

// (Opcional) exposición de solo-lectura para depuración

window.\_\_supabaseAdmin\_\_ = supabase;

adminAuth.js

// /administrador/adminAuth.js

import { supabase } from './adminSupabaseClient.js';

/\*\* Guarda la sesión del admin en sessionStorage (evita colisiones con otros módulos). \*/

export function setUserSession(user) {

  try {

    sessionStorage.setItem('sdcs\_admin\_user', JSON.stringify(user || null));

  } catch (\_) {}

}

export function getUserSession() {

  try {

    const raw = sessionStorage.getItem('sdcs\_admin\_user');

    return raw ? JSON.parse(raw) : null;

  } catch (\_) {

    return null;

  }

}

/\*\* Valida el código contra la función segura en la BD (fn\_validar\_codigo). \*/

export async function validarAccesoPorCodigoAdmin(codigoStr) {

  const clean = String(codigoStr).trim();

  if (!/^\d{1,20}$/.test(clean)) {

    return { data: null, error: new Error('El código debe ser numérico.') };

  }

  const codigoNum = Number(clean);

  const { data, error } = await supabase.rpc('fn\_validar\_codigo', { codigo\_in: codigoNum });

  if (error) return { data: null, error };

  if (!data || data.length === 0) return { data: null, error: null };

  // data[0] esperado: { codigo, nombre, perfil }

  return { data: data[0], error: null };

}

/\*\* Redirige solo si el perfil es ADMINISTRADOR. \*/

export function redirigirAdmin(perfil) {

  if (perfil === 'ADMINISTRADOR') {

    window.location.href = './dashboard.html';

  } else {

    throw new Error('Acceso denegado: perfil no autorizado para Administrador.');

  }

}