```
/**
================================================================================
=
* 05_UI_Operaciones.gs — Menú + Sidebar + Handlers `ops_*`
* Panel unificado para operaciones, diagnósticos, cierre de mes y
facturación
*
================================================================================
= */

/* ============= MENÚ PRINCIPAL ============= */
function buildMenuOperaciones() {
 const ui = SpreadsheetApp.getUi();

 ui.createMenu('⚙️ Operaciones')
   .addItem('Abrir panel rápido', 'showSidebarOperaciones')
   .addSeparator()
   .addSubMenu(
     ui.createMenu('Sincronización')
       .addItem('Sincronizar ICS → Snapshot (30d)', 'ops_syncIcsSnapshot')
       .addItem('Aplicar Snapshot → Resumen', 'ops_applySnapshot')
       .addItem('ICS (Snapshot+Aplicar) ahora', 'ops_syncICS')
       .addItem('Watcher Calendar (30d) ahora', 'ops_syncWatcher')
       .addItem('Push Resumen → Calendarios', 'ops_pushCalendarios')
   )
   .addSeparator()
   .addSubMenu(
     ui.createMenu('Kawakan')
       .addItem('Generar mes siguiente (idempotente)',
'ops_kawakanMesSiguiente')
   )
   .addSeparator()
   .addSubMenu(
     ui.createMenu('Utilidades')
       .addItem('Quitar dominio en EventID', 'ops_quitarDominioID')
```

```
      .addItem('Diagnóstico rápido', 'ops_diagnosticoRapido')
      .addItem('Diagnóstico completo', 'ops_diagnosticoCompleto')
      .addItem('Auto-instalar todo', 'ops_installAll')
  )
  .addSeparator()
  .addSubMenu(
    ui.createMenu('Cierre / Facturación')
      .addItem('Generar Cierre mensual (híbrido)', 'ops_cerrarMes')
      .addItem('Generar Factura (Cliente + Mes)…', 'ops_facturarDialog')
  )
  .addSeparator()
  .addSubMenu(
    ui.createMenu('Triggers')
      .addItem('Instalar triggers base', 'ops_installTriggersAll')
      .addItem('Trigger Watcher diario (03:05)',
'ops_installWatcherDaily')
      .addItem('Trigger ICS cada 2h', 'ops_installIcsEach2h')
      .addItem('Trigger Kawakan (día 28 · 09:00)',
'ops_installKawakanMonthly')
  )
  .addToUi();
}

/* ============= SIDEBAR ============= */
function showSidebarOperaciones() {
 const html = HtmlService.createHtmlOutput(`
 <div style="font:14px system-ui,Segoe UI,Roboto,Arial; padding:14px 16px;
line-height:1.5">
  <h2 style="margin:0 0 10px; font-size:16px;">Panel de Operaciones</h2>

  <div style="display:grid; gap:8px;">

    <!-- Utilidades globales -->
    <button onclick="run('ops_installAll')"
style="padding:8px">🔧 Auto-instalar todo</button>
```

```
    <button onclick="runAndDump('ops_diagnosticoCompleto')"
style="padding:8px">🩺 Diagnóstico completo</button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- Diagnóstico rápido -->
    <button onclick="runAndDump('ops_diagnosticoRapido')"
style="padding:8px">🔎 Diagnóstico rápido</button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- ICS / Calendar -->
    <button onclick="run('ops_syncIcsSnapshot')"
style="padding:8px">📥 ICS → Snapshot (30d)</button>
    <button onclick="run('ops_applySnapshot')"
style="padding:8px">➡️ Snapshot → Resumen</button>
    <button onclick="run('ops_syncICS')"
style="padding:8px">🔄 ICS completo (dos pasos)</button>
    <button onclick="run('ops_syncWatcher')"
style="padding:8px">🛰️ Watcher Calendar (30d)</button>
    <button onclick="run('ops_pushCalendarios')"
style="padding:8px">📤 Push Resumen → Calendarios</button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- Kawakan -->
    <button onclick="run('ops_kawakanMesSiguiente')"
style="padding:8px">🏷️ Kawakan: generar mes siguiente</button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- Cierre y facturación -->
    <div style="font-weight:600;margin-top:4px;">Cierre y
facturación</div>
    <button onclick="run('ops_cerrarMes')"          style="padding:8px">
```

```
      📁 Generar Cierre mensual (híbrido)
    </button>
    <button onclick="run('ops_facturarDialog')"      style="padding:8px">
      💼 Generar Factura (Cliente + Mes)…
    </button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- Otras utilidades -->
    <button onclick="run('ops_quitarDominioID')"
style="padding:8px">✂️ Limpiar dominio en EventID</button>

    <div style="height:1px;background:#ddd;margin:6px 0;"></div>

    <!-- Triggers -->
    <button onclick="run('ops_installTriggersAll')"
style="padding:8px">⚙️ Instalar triggers base</button>
    <button onclick="run('ops_installWatcherDaily')"
style="padding:8px">⏰ Instalar Watcher diario</button>
    <button onclick="run('ops_installIcsEach2h')"
style="padding:8px">⏰ Instalar ICS cada 2h</button>
    <button onclick="run('ops_installKawakanMonthly')"
style="padding:8px">⏰ Instalar Kawakan mensual</button>
    <button onclick="run('ops_installFormSubmit')"
style="padding:8px">📝 Instalar trigger de Form Submit</button>
    <button onclick="run('ops_uninstallFormSubmit')"
style="padding:8px">🗑️ Quitar trigger de Form Submit</button>
  </div>

  <pre id="log" style="margin-top:12px; background:#f8f8f8; border:1px
solid #eee; padding:8px; max-height:240px; overflow:auto;"></pre>

  <script>
    function run(fn){
      const log = document.getElementById('log');
```

```
      log.textContent = 'Ejecutando ' + fn + '…';
      google.script.run
        .withSuccessHandler(function(res){
          if (typeof res === 'string') log.textContent = res;
          else log.textContent = JSON.stringify(res, null, 2) || 'OK';
        })
        .withFailureHandler(function(err){
          log.textContent = (err && err.message) ? err.message :
String(err);
        })[fn]();
    }
    function runAndDump(fn){
      const log = document.getElementById('log');
      log.textContent = 'Ejecutando ' + fn + '…';
      google.script.run
        .withSuccessHandler(function(res){
          try { log.textContent = JSON.stringify(res, null, 2); }
          catch(e){ log.textContent = String(res); }
        })
        .withFailureHandler(function(err){
          log.textContent = (err && err.message) ? err.message :
String(err);
        })[fn]();
    }
  </script>
 </div>
 `);
 html.setTitle('Operaciones').setWidth(420);
 SpreadsheetApp.getUi().showSidebar(html);
}

/** Pequeño alias para usar en onOpen */
function openOpsSidebar() {
 showSidebarOperaciones();
}
```

```
/* ============= HANDLERS ops_* (envoltorios seguros) ============= */
/** 1) ICS */
function ops_syncIcsSnapshot(){
 if (typeof syncIcsSnapshotToResumenTest !== 'function') throw new
Error('Falta syncIcsSnapshotToResumenTest()');
 syncIcsSnapshotToResumenTest();
 return 'Hecho: ICS → RESUMEN_TEST.';
}
function ops_applySnapshot(){
 if (typeof applyResumenTestToResumen !== 'function') throw new
Error('Falta applyResumenTestToResumen()');
 applyResumenTestToResumen();
 return 'Hecho: RESUMEN_TEST → Resumen.';
}
function ops_syncICS(){
 if (typeof runIcsEvery2h === 'function') {
   runIcsEvery2h();
 } else {
   ops_syncIcsSnapshot();
   ops_applySnapshot();
 }
 return 'Hecho: sincronización ICS completa.';
}

/** 2) Watcher y Push */
function ops_syncWatcher(){
 if (typeof syncCalendarChangesToResumen !== 'function') throw new
Error('Falta syncCalendarChangesToResumen()');
 syncCalendarChangesToResumen();
 return 'Hecho: relectura Calendar (30d) aplicada a Resumen.';
}
function ops_pushCalendarios(){
 if (typeof pushResumenToCalendars !== 'function') throw new Error('Falta
pushResumenToCalendars()');
```

```
 pushResumenToCalendars();
 return 'Hecho: Resumen → Calendarios.';
}

/** 3) Utilidades */
function ops_quitarDominioID(){
 if (typeof quitarDominioID !== 'function') throw new Error('Falta
quitarDominioID()');
 quitarDominioID();
 return 'Listo: EventID limpio (sin @google.com).';
}

/** 4) Kawakan */
function ops_kawakanMesSiguiente(){
 if (typeof generarServiciosKawakanNow !== 'function') throw new
Error('Falta generarServiciosKawakanNow()');
 const n = generarServiciosKawakanNow();
 return `Kawakan: filas nuevas añadidas = ${n || 0}.`;
}

/** 5) Triggers (set base) */
function ops_installTriggersAll(){
 if (typeof installTriggers !== 'function') throw new Error('Falta
installTriggers()');
 installTriggers();
 return 'Triggers base instalados: ICS cada 2h + Watcher diario.';
}
function ops_installWatcherDaily(){
 if (typeof createTrigger_syncCalendarDaily !== 'function') throw new
Error('Falta createTrigger_syncCalendarDaily()');
 createTrigger_syncCalendarDaily();
 return 'Trigger Watcher diario creado (03:05).';
}
function ops_installIcsEach2h(){
 if (typeof installTriggers === 'function') {
```

```
   installTriggers();
   return 'Reinstalado set de triggers base (incluye ICS cada 2h).';
 }
 throw new Error('No hay función installTriggers().');
}
function ops_installKawakanMonthly(){
 if (typeof createMonthlyTrigger_Kawakan !== 'function') throw new
Error('Falta createMonthlyTrigger_Kawakan()');
 createMonthlyTrigger_Kawakan();
 return 'Trigger Kawakan mensual creado (28 · 09:00).';
}

/** 6) Diagnósticos */
function ops_diagnosticoRapido(){
 const out = {};
 const ss = SpreadsheetApp.getActive();

 const mustSheets = (typeof SHEET_RESUMEN !== 'undefined')
   ? [SHEET_RESUMEN, (typeof
SHEET_RESUMEN_TEST!=='undefined'?SHEET_RESUMEN_TEST:'RESUMEN_TEST'),
      (typeof SHEET_SOURCES!=='undefined'?SHEET_SOURCES:'ICAL_SOURCES'),
      (typeof
SHEET_BASE_DATOS!=='undefined'?SHEET_BASE_DATOS:'BASE_DATOS')]
   : ['Resumen','RESUMEN_TEST','ICAL_SOURCES','BASE_DATOS'];

 out.sheets = mustSheets.map(n=>{
   const sh = ss.getSheetByName(n);
   return { name:n, exists: !!sh, rows: sh ? sh.getLastRow() : 0, cols: sh
? sh.getLastColumn() : 0 };
 });

 out.globals = {
   TZ: (typeof TZ!=='undefined') ? TZ : Session.getScriptTimeZone(),
   IMPORT_FUTURE_DAYS: (typeof IMPORT_FUTURE_DAYS!=='undefined') ?
IMPORT_FUTURE_DAYS : '(no definido)',
```

```
    CALENDARIOS: (typeof CALENDARIOS!=='undefined') ?
Object.keys(CALENDARIOS).length : 0
  };

  const fnExists = n => (typeof this[n] === 'function');
  out.functions = {
    syncIcsSnapshotToResumenTest: fnExists('syncIcsSnapshotToResumenTest'),
    applyResumenTestToResumen   : fnExists('applyResumenTestToResumen'),
    syncCalendarChangesToResumen: fnExists('syncCalendarChangesToResumen'),
    pushResumenToCalendars      : fnExists('pushResumenToCalendars'),
    onFormSubmit                : fnExists('onFormSubmit'),
  };

  out.triggers = ScriptApp.getProjectTriggers()
    .map(t => ({ handler: t.getHandlerFunction ? t.getHandlerFunction() :
'(?)', type: String(t.getEventType ? t.getEventType() : '') }));

  try{
    const sh = ss.getSheetByName(mustSheets[0]);
    if (sh && sh.getLastRow()>=2) {
      const last = Math.min(200, sh.getLastRow()-1);
      const H = sh.getRange(1,1,1,sh.getLastColumn()).getDisplayValues()[0];
      const iFecha = H.findIndex(h => String(h).toLowerCase()==='fecha');
      const vals = sh.getRange(sh.getLastRow()-last+1, 1, last,
sh.getLastColumn()).getValues();
      const cutoff = Date.now() - 48*3600*1000;
      let recent = 0;
      vals.forEach(r=>{
        const v = r[iFecha];
        const d = (v instanceof Date && !isNaN(v)) ? v : new
Date(String(v||''));
        if (!isNaN(d) && d.getTime()>=cutoff) recent++;
      });
      out.resumenUltimas48h = recent;
    }
```

```
 }catch(_){}

 return out;
}

function ops_diagnosticoCompleto(){
 if (typeof diagnosticoProfundo !== 'function') throw new Error('Falta
diagnosticoProfundo() en 01_Principal.gs');
 return diagnosticoProfundo(); // devuelve JSON; el sidebar lo vuelca en
<pre id="log">
}

function ops_installAll(){
 if (typeof installAll !== 'function') throw new Error('Falta installAll()
en 01_Principal.gs');
 installAll();
 return 'Auto-instalado: ICS cada 2h + Watcher diario + Kawakan mensual.';
}

function ops_installFormSubmit(){ installFormSubmitTrigger(); return
'Trigger de Form Submit instalado.'; }
function ops_uninstallFormSubmit(){ uninstallFormSubmitTrigger(); return
'Trigger de Form Submit eliminado.'; }

/** 7) Cierre de mes y Facturación */
function ops_cerrarMes() {
 if (typeof cerrarMesHibrido !== 'function') {
   throw new Error('Falta cerrarMesHibrido() en 03_Final de mes.gs');
 }
 cerrarMesHibrido();
 return 'Cierre mensual generado. Revisa la carpeta 00_RESUMEN DATOS.';
}

function ops_facturarDialog() {
 if (typeof solicitarFactura_DriveWatch !== 'function') {
```

```
    throw new Error('Falta solicitarFactura_DriveWatch() en
04_Facturacion.gs');
  }
  solicitarFactura_DriveWatch(); // abre el diálogo Cliente+Mes
  return 'Abriendo selector de Factura…';
}

/***** === onOpen ÚNICO (menú maestro + triggers + panel) === *****/
function onOpen(e){
  // 1) Asegurar cabecera de RESUMEN al abrir (lógica antigua)
  try {
    if (typeof ensureResumenHeader_ === 'function') {
      ensureResumenHeader_();
    }
  } catch (err) {
    Logger.log('Error en ensureResumenHeader_ desde onOpen: ' + err);
  }

  // 2) Menú ⚙️ Operaciones
  try {
    if (typeof buildMenuOperaciones === 'function') {
      buildMenuOperaciones();
    }
  } catch (err) {
    Logger.log('Error en buildMenuOperaciones: ' + err);
  }

  // 3) Menú ⏱ Triggers
  try {
    if (typeof buildMenuTriggers === 'function') {
      buildMenuTriggers();
    }
  } catch (err) {
    Logger.log('Error en buildMenuTriggers: ' + err);
  }
```

```
 // 4) Abrir automáticamente el panel de Operaciones (si está activado en la config)
 try {
   if (typeof UI_CFG !== 'undefined' && UI_CFG.AUTO_OPEN_SIDEBAR_ON_OPEN) {
     if (typeof openOpsSidebar === 'function') {
       openOpsSidebar();
     }
   }
 } catch (err) {
   Logger.log('Error al abrir sidebar de Operaciones: ' + err);
 }
}
```