import {

  showToast,

  showConfirm,

  showConfirmOkOnly,

  renderizarCabeceraContactos,

  renderizarFilasContactos

} from './utils.js';

// Estado global

let googleContactsMemory = null;

let terminoBusqueda = "";

let googleContactPage = 1;

const GOOGLE\_PAGE\_SIZE = 10;

let abrirGoogleContactsAlVolver = false;

function inicializarGoogleContactList() {

  const desde = (googleContactPage - 1) \* GOOGLE\_PAGE\_SIZE;

  const hasta = googleContactPage \* GOOGLE\_PAGE\_SIZE;

  const base = contactosGoogleCargadosCompleta || [];

  const filtrados = terminoBusqueda

    ? base.filter((c) => c.nombre.toLowerCase().includes(terminoBusqueda))

    : base;

  const visibles = filtrados.slice(desde, hasta);

  contactosGoogleCargados = filtrados;

  const container = document.getElementById("google-contacts-list");

let body = document.getElementById("google-contacts-list-body");

// (1) Si es la primera vez, agregamos botón Refrescar y el buscador:

if (!container.querySelector("#google-refresh-btn")) {

  // Botón Refrescar

  const refreshBtn = document.createElement("button");

  refreshBtn.id = "google-refresh-btn";

  refreshBtn.innerHTML = `<i class="fas fa-sync-alt"></i> Refrescar`;

  refreshBtn.title = "Volver a importar contactos desde Google";

  refreshBtn.style.padding = "0.4em 0.6em";

  refreshBtn.style.border = "none";

  refreshBtn.style.borderRadius = "0.3em";

  refreshBtn.style.backgroundColor = getComputedStyle(document.documentElement)

    .getPropertyValue("--color-fondo")

    .trim();

  refreshBtn.style.color = "white";

  refreshBtn.style.cursor = "pointer";

  refreshBtn.style.marginBottom = "0.5em";

  refreshBtn.style.display = "flex";

  refreshBtn.style.alignItems = "center";

  refreshBtn.style.gap = "0.4em";

  refreshBtn.addEventListener("click", () => {

    try {

      const clientId = `${GOOGLE\_CLIENT\_ID}`;

      const redirectUri = `${FRONT\_URL}/pages/googleCallback.html`;

      const scope = "https://www.googleapis.com/auth/contacts.readonly";

      const authUrl = `https://accounts.google.com/o/oauth2/v2/auth?response\_type=code&client\_id=${clientId}&redirect\_uri=${redirectUri}&scope=${encodeURIComponent(

        scope

      )}&access\_type=online&prompt=consent`;

      localStorage.setItem("abrirGoogleContactsAlVolver", "true");

      window.location.href = authUrl;

    } catch (err) {

      console.error("Error al redirigir a Google OAuth:", err);

      showToast("Error al conectar con Google. Intenta nuevamente.", "error");

    }

  });

  container.appendChild(refreshBtn);

  // Search

  const searchWrapper = document.createElement("div");

  searchWrapper.id = "google-search-wrapper";

  searchWrapper.style.margin = "0 auto 0.5em auto";

  searchWrapper.style.display = "flex";

  searchWrapper.style.alignItems = "center";

  searchWrapper.style.gap = "0.5em";

  searchWrapper.style.maxWidth = "440px";

  const searchIcon = document.createElement("i");

  searchIcon.className = "fas fa-search";

  searchIcon.style.color = "white";

  const searchInput = document.createElement("input");

  searchInput.id = "google-search-input";

  searchInput.type = "text";

  searchInput.placeholder = "Buscar contacto...";

  searchInput.style.flex = "1";

  searchInput.style.padding = "0.3em 0.5em";

  searchInput.style.borderRadius = "0.3em";

  searchInput.style.border = "1px solid #ccc";

  searchInput.style.width = "100%";

  searchInput.style.boxSizing = "border-box";

  searchInput.addEventListener("input", () => {

    terminoBusqueda = searchInput.value.trim().toLowerCase();

    googleContactPage = 1;

    inicializarGoogleContactList();

  });

  searchWrapper.appendChild(searchIcon);

  searchWrapper.appendChild(searchInput);

  container.appendChild(searchWrapper);

}

// (2) Cabecera de columnas (utils) + body

renderizarCabeceraContactos("google-contacts-list", [

  "Nombre",

  "Quiero que reciba mis mensajes",

  "Quiero que me vea",

  "Acción",

]);

body = document.getElementById("google-contacts-list-body");

// (3) Limpiar body

if (body) body.innerHTML = "";

// (4) Filas (utils)

renderizarFilasContactos(

  "google-contacts-list",

  visibles,

  (c, mensajes, visibilidad) => agregarContactoDesdeGoogle(c, mensajes, visibilidad)

);

  const totalPaginas = Math.ceil(filtrados.length / GOOGLE\_PAGE\_SIZE);

  // Contenedor de paginación

  const paginacion = document.createElement("div");

  paginacion.id = "paginacion-contactos";

  paginacion.style.display = "flex";

  paginacion.style.justifyContent = "center";

  paginacion.style.alignItems = "center";

  paginacion.style.gap = "0.5em";

  paginacion.style.marginTop = "1em";

  paginacion.style.color = "white";

  // Botón ⏮ (Primera página)

  const btnPrimera = document.createElement("button");

  btnPrimera.innerHTML = "⏮";

  btnPrimera.disabled = googleContactPage === 1;

  btnPrimera.style.opacity = btnPrimera.disabled ? "0.5" : "1";

  btnPrimera.addEventListener("click", () => {

    googleContactPage = 1;

    inicializarGoogleContactList();

  });

  paginacion.appendChild(btnPrimera);

  // Botón ◀ (Anterior)

  const btnAnterior = document.createElement("button");

  btnAnterior.innerHTML = "◀";

  btnAnterior.disabled = googleContactPage === 1;

  btnAnterior.style.opacity = btnAnterior.disabled ? "0.5" : "1";

  btnAnterior.addEventListener("click", () => {

    if (googleContactPage > 1) {

      googleContactPage--;

      inicializarGoogleContactList();

    }

  });

  paginacion.appendChild(btnAnterior);

  // Cuadro editable de número de página

  const inputPagina = document.createElement("input");

  inputPagina.type = "number";

  inputPagina.value = googleContactPage;

  inputPagina.min = 1;

  inputPagina.max = totalPaginas;

  inputPagina.style.width = "3em";

  inputPagina.style.textAlign = "center";

  inputPagina.style.padding = "0.3em";

  inputPagina.style.borderRadius = "0.3em";

  inputPagina.style.border = "1px solid #ccc";

  inputPagina.style.backgroundColor = "#f8f9fa";

  inputPagina.addEventListener("change", () => {

    let nuevaPagina = parseInt(inputPagina.value);

    if (isNaN(nuevaPagina) || nuevaPagina < 1) nuevaPagina = 1;

    if (nuevaPagina > totalPaginas) nuevaPagina = totalPaginas;

    googleContactPage = nuevaPagina;

    inicializarGoogleContactList();

  });

  paginacion.appendChild(inputPagina);

  // Total de páginas

  const totalSpan = document.createElement("span");

  totalSpan.textContent = ` / ${totalPaginas}`;

  paginacion.appendChild(totalSpan);

  // Botón ▶ (Siguiente)

  const btnSiguiente = document.createElement("button");

  btnSiguiente.innerHTML = "▶";

  btnSiguiente.disabled = googleContactPage >= totalPaginas;

  btnSiguiente.style.opacity = btnSiguiente.disabled ? "0.5" : "1";

  btnSiguiente.addEventListener("click", () => {

    if (googleContactPage < totalPaginas) {

      googleContactPage++;

      inicializarGoogleContactList();

    }

  });

  paginacion.appendChild(btnSiguiente);

  // Botón ⏭ (Última página)

  const btnUltima = document.createElement("button");

  btnUltima.innerHTML = "⏭";

  btnUltima.disabled = googleContactPage >= totalPaginas;

  btnUltima.style.opacity = btnUltima.disabled ? "0.5" : "1";

  btnUltima.addEventListener("click", () => {

    googleContactPage = totalPaginas;

    inicializarGoogleContactList();

  });

  paginacion.appendChild(btnUltima);

  const body1 = document.getElementById("google-contacts-list-body");

if (body1) body1.appendChild(paginacion);

}

function sanitizarTelefonoE164(input) {

  const limpio = input.trim().replace(/[^\d+]/g, ""); // 🔧 Elimina todo excepto dígitos y "+"

  const soloNumeros = limpio.replace(/\D/g, ""); // solo números

  // 🌍 Si empieza con 00 → internacional

  if (soloNumeros.startsWith("00")) {

    return "+" + soloNumeros.slice(2);

  }

  // ✅ Si empieza con +54 (Argentina), forzamos +549...

  if (limpio.startsWith("+54")) {

    const sinMas = soloNumeros; // Ej: "541134560947" o "5491151227864"

    if (sinMas.startsWith("54") && !sinMas.startsWith("549")) {

      return "+549" + sinMas.slice(2); // fuerza el 9 después de 54

    }

    return "+" + sinMas;

  }

  // 📱 Si empieza con 15 y tiene 11 dígitos → +549...

  if (soloNumeros.startsWith("15") && soloNumeros.length === 11) {

    return "+549" + soloNumeros.slice(2);

  }

  // 📲 Si empieza con 9 y tiene 11 dígitos → ya está bien

  if (soloNumeros.startsWith("9") && soloNumeros.length === 11) {

    return "+54" + soloNumeros;

  }

  // 🏠 Si empieza con 0 y tiene 11 dígitos → forzamos +549...

  if (soloNumeros.startsWith("0") && soloNumeros.length === 11) {

    return "+549" + soloNumeros.slice(1);

  }

  // 🧼 Si tiene 10 dígitos → asumimos móvil sin 0 ni 9 → le agregamos ambos

  if (soloNumeros.length === 10) {

    return "+549" + soloNumeros;

  }

  // Fallback genérico (último recurso)

  return "+" + soloNumeros;

}

function esTelefonoValido(numero) {

  // Debe empezar con + y tener entre 10 y 15 dígitos (ej: +541112345678)

  return /^\+\d{10,15}$/.test(numero);

}

// ⚠️ Evitar mostrar si el usuario ya tildó "No volver a mostrar"

async function esperarOk(mensaje) {

  if (window.\_\_noMostrarMensajeDePrefijo9) return;

  return new Promise((resolve) => {

    const modal = document.getElementById("info-modal");

    const mensajeEl = document.getElementById("info-message");

    const okBtn = document.getElementById("info-ok");

    const checkbox = document.getElementById("info-dont-show-again");

    mensajeEl.textContent = mensaje;

    checkbox.checked = false;

    modal.classList.remove("hidden");

    okBtn.onclick = () => {

      if (checkbox.checked) {

        window.\_\_noMostrarMensajeDePrefijo9 = true;

      }

      modal.classList.add("hidden");

      resolve();

    };

  });

}

// Función reutilizable para el click de "Agregar" en cada fila

async function agregarContactoDesdeGoogle(c, mensajes, visibilidad) {

  const endpoint = `${ROOT\_URL}/api/contacts`;

  const telefonoOriginal = c.telefono || "";

  const telefonoSanitizado = sanitizarTelefonoE164(telefonoOriginal);

  // Advertencia por prefijo 9

  if (

    telefonoSanitizado.startsWith("+549") &&

    !telefonoOriginal.replace(/\D/g, "").startsWith("9") &&

    !telefonoOriginal.replace(/\D/g, "").startsWith("15") &&

    !telefonoOriginal.trim().startsWith("+549")

  ) {

    await esperarOk(

      "El número agregado no tenía prefijo 9. Beepsafe lo ha agregado automáticamente. Por favor, verificá que sea un número de celular con WhatsApp válido. De no serlo, el destinatario no recibirá los mensajes."

    );

  }

  if (!esTelefonoValido(telefonoSanitizado)) {

    showToast(

      `El número de teléfono "${telefonoOriginal}" de este contacto no es válido`,

      "error"

    );

    return;

  }

  const payload = {

    nombre: c.nombre,

    telefono: telefonoSanitizado || "",

    mensajes,

    visibilidad,

  };

  try {

    const response = await fetch(endpoint, {

      method: "POST",

      headers: {

        "Content-Type": "application/json",

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

      body: JSON.stringify(payload),

    });

    if (!response.ok) {

      const err = await response.json();

      showToast("Error: " + err.error, "error");

      return;

    }

    // Sacarlo de la lista local

    contactosGoogleCargados = contactosGoogleCargados.filter(

      (contacto) =>

        contacto.nombre.trim().toLowerCase() !==

        c.nombre.trim().toLowerCase()

    );

    contactosGoogleCargadosCompleta = contactosGoogleCargados.slice();

    // Reiniciar a página 1 y re-render

    googleContactPage = 1;

    inicializarGoogleContactList();

    // Refrescar "Mis Contactos"

    const res = await fetch(`${ROOT\_URL}/api/contacts`, {

      headers: {

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

    });

    const { contactos: nuevosContactos } = await res.json();

    if (typeof renderizarMisContactos === "function") {

      renderizarMisContactos(nuevosContactos);

    }

  } catch (err) {

    showToast("Error al conectar con el servidor", "error");

    console.error(err);

  }

}

async function refrescarContactosGoogle() {

  try {

    const response = await fetch(`${ROOT\_URL}/api/google/tempContacts`, {

      headers: {

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

    });

    const { contactos: googleContacts } = await response.json();

    const res = await fetch(`${ROOT\_URL}/api/contacts`, {

      headers: {

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

    });

    const { contactos: contactosGuardados } = await res.json();

    const nombresGuardados = new Set(

      contactosGuardados.map((c) => c.nombre.trim().toLowerCase())

    );

    const contactosFiltrados = googleContacts.filter(

      (gc) => !nombresGuardados.has(gc.nombre.trim().toLowerCase())

    );

    contactosGoogleCargadosCompleta = contactosFiltrados;

    googleContactPage = 1;

    inicializarGoogleContactList();

  } catch (err) {

    console.warn("Error al refrescar contactos de Google:", err.message);

  }

}

async function cargarContactosGoogleDesdeBackend() {

  try {

    const res = await fetch(`${ROOT\_URL}/api/google/tempContacts`, {

      headers: {

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

    });

    const { contactos } = await res.json();

    return contactos;

  } catch (err) {

    console.warn("No se pudieron recuperar contactos desde el backend:", err);

    return [];

  }

}

window.addEventListener("DOMContentLoaded", async () => {

  // 🔒 Verifica que esté logueado

  try {

    const response = await fetch(`${ROOT\_URL}/api/sessions/current`, {

      method: "GET",

      headers: {

        Authorization: `Bearer ${localStorage.getItem("token")}`,

      },

    });

    if (!response.ok) {

      throw new Error("Token inválido");

    }

  } catch (err) {

    window.location.href = "iniciarSesion.html";

    return;

  }

  // 🔽 Referencias

  const toggleGoogleContactos = document.getElementById(

    "toggleGoogleContactos"

  );

  const googleContactsList = document.getElementById("google-contacts-list");

  // Ejecuta cuando el usuario hace click en el botón de importar contactos desde Google

  toggleGoogleContactos?.addEventListener("click", async () => {

    try {

      // Cambia el estado de visibilidad del contenedor de contactos de Google

      const isVisible = googleContactsList.style.display === "block";

      googleContactsList.style.display = isVisible ? "none" : "block";

      // Rota el icono de chevron

      const icon = toggleGoogleContactos.querySelector("i");

      if (icon) {

        icon.classList.toggle("rotate", !isVisible);

      }

      // Busca los contactos de Google en la base de datos

      const response = await fetch(`${ROOT\_URL}/api/google/tempContacts`, {

        headers: {

          Authorization: `Bearer ${localStorage.getItem("token")}`,

        },

      });

      const { contactos } = await response.json();

      let contactosGoogle = contactos;

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

        try {

          const clientId = `${GOOGLE\_CLIENT\_ID}`;

          const redirectUri = `${FRONT\_URL}/pages/googleCallback.html`;

          const scope = "https://www.googleapis.com/auth/contacts.readonly";

          const authUrl = `https://accounts.google.com/o/oauth2/v2/auth?response\_type=code&client\_id=${clientId}&redirect\_uri=${redirectUri}&scope=${encodeURIComponent(

            scope

          )}&access\_type=online&prompt=consent`;

          localStorage.setItem("abrirGoogleContactsAlVolver", "true");

          window.location.href = authUrl;

          return;

        } catch (authErr) {

          console.error("Error al iniciar autenticación con Google:", authErr);

          showToast(

            "Error al conectar con Google. Verifica la configuración.",

            "error"

          );

          return;

        }

      }

      // Trae los contactos del usuario

      const res = await fetch(`${ROOT\_URL}/api/contacts`, {

        headers: {

          Authorization: `Bearer ${localStorage.getItem("token")}`,

        },

      });

      const { contactos: contactosGuardados } = await res.json();

      const nombresGuardados = new Set(

        contactosGuardados.map((c) => c.nombre.trim().toLowerCase())

      );

      const contactosFiltrados = contactosGoogle.filter(

        (c) => !nombresGuardados.has(c.nombre.trim().toLowerCase())

      );

      contactosGoogleCargadosCompleta = contactosFiltrados;

      googleContactPage = 1;

      inicializarGoogleContactList();

    } catch (err) {

      console.error("Error al procesar contactos de Google:", err);

      showToast("No se pudieron cargar los contactos desde Google.", "error");

    }

  });

  // ✅ Si volvimos de Google y tenemos bandera activa

  const abrirGoogle = localStorage.getItem("abrirGoogleContactsAlVolver");

  if (abrirGoogle === "true") {

    localStorage.removeItem("abrirGoogleContactsAlVolver");

    // ✅ Simular clic en el botón para abrir contactos de Google

    const btnGoogle = document.getElementById("toggleGoogleContactos");

    if (btnGoogle) {

      btnGoogle.click(); // Esto ejecuta la lógica de carga y renderizado

    }

  }

});

// Pone la función refrescarContactosGoogle disponible globalmente

window.refrescarContactosGoogle = refrescarContactosGoogle;