import {

showToast,

showConfirm,

showConfirmOkOnly,

renderizarCabeceraContactos,

renderizarFilasContactos,

renderizarContactosConPaginado,

renderizarBuscadorYBotonRefrescar,

sanitizarTelefonoE164

} from "./utils.js";

let contactosGoogleCargados = [];

let contactosGoogleCargadosCompleta = [];

// Estado global

let googleContacts = [];

let paginaGoogle = 1;

const PAGE\_SIZE\_GOOGLE = 10;

let terminoGoogle = "";

async function actualizarGoogleContacts() {

try {

// 2.1) Traer lista de Google

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

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

});

if (!res.ok) throw new Error("No se pudieron cargar contactos de Google");

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

googleContacts = contactos;

// 2.2) Renderizar buscador + refrescar

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

renderizarBuscadorYBotonRefrescar({

container,

onBuscar: termino => {

terminoGoogle = termino;

paginaGoogle = 1;

renderGoogle();

},

onRefrescar: actualizarGoogleContacts

});

// 2.3) Filtrar por término

const filtrados = terminoGoogle

? googleContacts.filter(c =>

c.nombre.toLowerCase().includes(terminoGoogle)

)

: googleContacts;

// 2.4) Renderizar tabla con paginado y callback de “Agregar”

renderizarContactosConPaginado({

containerId: "google-contacts-list",

contactos: filtrados,

paginaActual: paginaGoogle,

pageSize: PAGE\_SIZE\_GOOGLE,

columnas: [

"Nombre",

"Quiero que reciba mis mensajes",

"Quiero que me vea",

"Acción"

],

onAccionClick: (contacto, mensajes, visibilidad) =>

agregarContactoDesdeGoogle(contacto, mensajes, visibilidad),

onPaginaChange: nueva => {

paginaGoogle = nueva;

renderGoogle();

}

});

} catch (err) {

showToast(err.message, "error");

}

}

// 3) Separar render para reusarlo en paginado

function renderGoogle() {

actualizarGoogleContacts();

}

// 4) Iniciar al cargar la página y al hacer click en el toggle

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

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

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

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

const visible = lista.style.display === "block";

lista.style.display = visible ? "none" : "block";

toggle.querySelector("i")?.classList.toggle("rotate", !visible);

if (!visible) {

// si abrimos, primero refrescamos de Google

await actualizarGoogleContacts();

}

});

});

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

paginaGoogle = 1;

renderGoogle();

// 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;

paginaGoogle = 1;

renderGoogle();

} 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;

paginaGoogle = 1;

renderGoogle();

} 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;