JS | SETTINGS

// js/screens/settings.js

// ⚠️ AJUSTA ESTE CAMINHO conforme a estrutura do teu projeto:

// - Se este ficheiro está em js/screens/, usa "../auth.js" (como abaixo).

// - Se estiver lado a lado com auth.js, usa "./auth.js".

import { doLogout } from "./auth.js";

const SETTINGS\_STORAGE\_KEY = "app.settings";

const defaultSettings = {

  // Notificações

  emailNotifications: true,

  pushNotifications: false,

  weeklyReports: true,

  // Segurança

  twoFactor: false,

  loginNotifications: true,

  // Interface

  darkMode: false,          // mantém compat com versões antigas

  language: "pt-PT",

  currency: "EUR",

};

/\* ---------------- helpers de storage ---------------- \*/

function loadSettings() {

  try {

    const raw = localStorage.getItem(SETTINGS\_STORAGE\_KEY);

    if (!raw) return { ...defaultSettings };

    const parsed = JSON.parse(raw);

    return { ...defaultSettings, ...parsed };

  } catch {

    return { ...defaultSettings };

  }

}

function saveSettings(s) {

  try {

    localStorage.setItem(SETTINGS\_STORAGE\_KEY, JSON.stringify(s));

  } catch {}

}

/\* ---------------- tema (light/dark) ---------------- \*/

function applyTheme(dark) {

  const mode = dark ? "dark" : "light";

  document.documentElement.setAttribute("data-theme", mode);

  // atualizar theme-color do mobile (opcional)

  const meta = document.querySelector('meta[name="theme-color"]');

  if (meta) meta.content = dark ? "#0b0e13" : "#ffffff";

  // notificar a app inteira (outros screens podem ouvir este evento)

  window.dispatchEvent(new CustomEvent("app:theme-changed", { detail: { dark } }));

}

/\* ---------------- init do screen ---------------- \*/

export function initScreen() {

  // ⚠️ NUNCA fazer querySelector/getElementById fora do initScreen,

  // porque o HTML do screen só existe depois da navegação injetar o markup.

  // Elementos

  const elLanguage = document.getElementById("cfgLanguage");

  const elCurrency = document.getElementById("cfgCurrency");

  const elDark     = document.getElementById("cfgDarkMode");

  const elEmailN = document.getElementById("cfgEmailNotifications");

  const elPush   = document.getElementById("cfgPushNotifications");

  const elWeekly = document.getElementById("cfgWeeklyReports");

  const el2FA   = document.getElementById("cfgTwoFactor");

  const elLogin = document.getElementById("cfgLoginNotifications");

  const btnSave   = document.getElementById("cfgSave");

  const btnCancel = document.getElementById("cfgCancel");

  const btnLogout = document.getElementById("btnLogout");

  if (!elLanguage || !elCurrency || !elDark || !btnSave || !btnCancel) {

    console.warn("⚠️ settings.js: elementos não encontrados. Confirma o HTML dos IDs.");

    return;

  }

  // 🔒 Logout — liga AQUI (agora o botão existe no DOM)

  if (btnLogout) {

    btnLogout.addEventListener("click", (e) => {

      e.preventDefault();

      doLogout(); // exportado do auth.js

    });

  }

  // Carrega estado atual (se não houver, segue sistema e grava já)

  let state = loadSettings();

  if (!("darkMode" in state)) {

    state.darkMode =

      window.matchMedia &&

      window.matchMedia("(prefers-color-scheme: dark)").matches;

    saveSettings(state);

  }

  // Preenche UI

  elLanguage.value = state.language;

  elCurrency.value = state.currency;

  elDark.checked   = !!state.darkMode;

  if (elEmailN) elEmailN.checked = !!state.emailNotifications;

  if (elPush)   elPush.checked   = !!state.pushNotifications;

  if (elWeekly) elWeekly.checked = !!state.weeklyReports;

  if (el2FA)   el2FA.checked   = !!state.twoFactor;

  if (elLogin) elLogin.checked = !!state.loginNotifications;

  // Aplica tema no arranque deste screen

  applyTheme(!!state.darkMode);

  // Listeners (atualizam o estado em memória)

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

    state.language = elLanguage.value;

  });

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

    state.currency = elCurrency.value;

  });

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

    state.darkMode = !!elDark.checked;

    applyTheme(state.darkMode); // aplica imediatamente

  });

  elEmailN?.addEventListener("change", () => {

    state.emailNotifications = !!elEmailN.checked;

  });

  elPush?.addEventListener("change", () => {

    state.pushNotifications = !!elPush.checked;

  });

  elWeekly?.addEventListener("change", () => {

    state.weeklyReports = !!elWeekly.checked;

  });

  el2FA?.addEventListener("change", () => {

    state.twoFactor = !!el2FA.checked;

  });

  elLogin?.addEventListener("change", () => {

    state.loginNotifications = !!elLogin.checked;

  });

  // Botões

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

    saveSettings(state);

    // showToast?.("Configurações guardadas!");

  });

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

    // Recarrega do storage e volta a preencher/Aplicar

    state = loadSettings();

    elLanguage.value = state.language;

    elCurrency.value = state.currency;

    elDark.checked   = !!state.darkMode;

    if (elEmailN) elEmailN.checked = !!state.emailNotifications;

    if (elPush)   elPush.checked   = !!state.pushNotifications;

    if (elWeekly) elWeekly.checked = !!state.weeklyReports;

    if (el2FA)   el2FA.checked   = !!state.twoFactor;

    if (elLogin) elLogin.checked = !!state.loginNotifications;

    applyTheme(state.darkMode);

  });

  // (Opcional) Seguir alterações do sistema se o user nunca “forçou”

  try {

    const mq = window.matchMedia("(prefers-color-scheme: dark)");

    const handler = (e) => {

      // só respeita o sistema se o utilizador nunca mudou manualmente

      const saved = loadSettings();

      if (!("userForcedTheme" in saved) || !saved.userForcedTheme) {

        state.darkMode = e.matches;

        applyTheme(state.darkMode);

        saveSettings(state);

        elDark.checked = state.darkMode;

      }

    };

    // marca que o user escolheu manualmente quando mexer no switch

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

      const s = loadSettings();

      s.userForcedTheme = true;

      saveSettings(s);

    });

    mq.addEventListener?.("change", handler);

  } catch {}

}