/\*\*

\* =============================================================================

\* @file Code.gs

\* @description

\* هذا الملف يحتوي على الدوال التي يمكن استدعاؤها مباشرة من الواجهة الأمامية (Sidebar.html)

\* عبر google.script.run.

\* يعمل كنقطة دخول مركزية، حيث يقوم بتوجيه الاستدعاءات إلى الوحدات المناحدة ضمن مساحة الاسم GAssistant.

\* يعتمد على جميع وحدات GAssistant لتقديم الوظائف.

\* =============================================================================

\*/

// تأكد من أن جميع وحدات GAssistant معرفة ومحملة قبل هذا الملف في مشروع Apps Script.

// هذا الملف يعمل كـ "Proxy" للدوال العامة.

// -----------------------------------------------------------------------------

// دوال عامة لـ google.script.run

// -----------------------------------------------------------------------------

/\*\*

\* @function onOpen

\* @description

\* تُنشئ القوائم المخصصة في Google Sheets عند فتح المستند.

\* يتم استدعاؤها من GAssistant.UI.onOpen.

\*/

function onOpen() {

GAssistant.UI.onOpen();

}

/\*\*

\* @function showSidebar

\* @description

\* تعرض الشريط الجانبي الرئيسي للمساعد.

\*/

function showSidebar() {

GAssistant.UI.showSidebar();

}

/\*\*

\* @function askGemini

\* @description

\* ترسل طلبًا إلى نموذج Gemini وتجلب الرد.

\* @param {string} userPrompt - طلب المستخدم.

\* @param {string} [modelOverride] - نموذج محدد لاستخدامه.

\* @param {string|null} [imagePart=null] - بيانات الصورة بصيغة Base64.

\* @param {boolean} [structuredOutputRequested=false] - هل طلب المستخدم إخراجًا مهيكلًا (JSON)؟

\* @returns {object} - كائن الرد النهائي.

\*/

function askGemini(userPrompt, modelOverride, imagePart, structuredOutputRequested) {

return GAssistant.AI.Core.ask(userPrompt, modelOverride, imagePart, structuredOutputRequested);

}

/\*\*

\* @function getAvailableModels

\* @description

\* تُرجع قائمة بالنماذج المتاحة للواجهة الأمامية من إعدادات AI\_CONFIG.

\* @returns {object} - كائن يحتوي على نوع الرد وقائمة النماذج.

\*/

function getAvailableModels() {

return GAssistant.UI.showAvailableModels(); // UI تعرضها كجدول ملخص

}

/\*\*

\* @function getSheetInsightForPrompt

\* @description

\* تجلب ملخصًا ذكيًا لسياق الورقة النشطة.

\* @returns {string} - نص الـ prompt المقترح.

\*/

function getSheetInsightForPrompt() {

return GAssistant.Tools.Sheets.generateSheetAnalysisPrompt('markdown'); // افتراضي Markdown

}

/\*\*

\* @function createMinimalTestSheet

\* @description

\* تنشئ ورقة اختبار بسيطة ببيانات وهمية.

\* @returns {object} - رسالة نجاح أو خطأ.

\*/

function createMinimalTestSheet() {

return GAssistant.Tools.Sheets.createMinimalTestSheet();

}

/\*\*

\* @function summarizeActiveSheet

\* @description

\* يلخص البيانات في ورقة العمل النشطة.

\* @returns {object} - ملخص للورقة.

\*/

function summarizeActiveSheet() {

return GAssistant.Tools.Sheets.summarizeActiveSheet();

}

/\*\*

\* @function suggestTargetFile

\* @description

\* يحلل محتوى الكود ويقترح اسم الملف الأنسب لوضعه فيه.

\* @param {string} functionCode - الكود البرمجي للدالة المقترحة.

\* @returns {string} - اسم الملف المقترح.

\*/

function suggestTargetFile(functionCode) {

return GAssistant.WorkshopTools.suggestTargetFile(functionCode);

}

/\*\*

\* @function showBuiltinDocsSidebar

\* @description

\* تعرض الشريط الجانبي للبحث في دوال Apps Script المدمجة.

\*/

function showBuiltinDocsSidebar() {

GAssistant.UI.showBuiltinDocsSidebar();

}

/\*\*

\* @function getBuiltinFunctionDoc

\* @description

\* يجلب توثيق دالة Apps Script مدمجة باستخدام نموذج Gemini.

\* @param {string} functionName - اسم الدالة المراد البحث عنها.

\* @returns {object} - كائن يحتوي على HTML منسق و/أو نص عادي للتوثيق.

\*/

function getBuiltinFunctionDoc(functionName) {

return GAssistant.BuiltinDocs.getBuiltinFunctionDoc(functionName);

}

/\*\*

\* @function showCodeReviewSidebar

\* @description

\* تعرض الشريط الجانبي لمراجعة الكود.

\*/

function showCodeReviewSidebar() {

GAssistant.UI.showCodeReviewSidebar();

}

/\*\*

\* @function reviewCodeForSidebar

\* @description

\* تُستخدم بواسطة CodeReviewSidebar.html لجلب تغذية راجعة حول الكود.

\* @param {string} code - الكود البرمجي المراد مراجعته.

\* @param {string} reviewType - نوع المراجعة المطلوبة.

\* @returns {object} - كائن جاهز للعرض في الشريط الجانبي.

\*/

function reviewCodeForSidebar(code, reviewType) {

return GAssistant.Tools.CodeReview.reviewCodeForSidebar(code, reviewType);

}

/\*\*

\* @function getCodeFromActiveCell

\* @description

\* يجلب الكود من الخلية النشطة.

\* @returns {string} - محتوى الخلية النشطة.

\*/

function getCodeFromActiveCell() {

return GAssistant.UI.getCodeFromActiveCell();

}

/\*\*

\* @function updateActiveCellWithCode

\* @description

\* يحدث الخلية النشطة بكود جديد.

\* @param {string} code - الكود المراد كتابته.

\*/

function updateActiveCellWithCode(code) {

GAssistant.UI.updateActiveCellWithCode(code);

}

/\*\*

\* @function showToolsDocumentationSidebar

\* @description

\* تعرض الشريط الجانبي لتوثيق الأدوات المخصصة.

\*/

function showToolsDocumentationSidebar() {

GAssistant.UI.showToolsDocumentationSidebar();

}

/\*\*

\* @function exportToolsDocumentationToDoc

\* @description

\* يصدر توثيق الأدوات إلى مستند Google Docs.

\*/

function exportToolsDocumentationToDoc() {

GAssistant.UI.exportToolsDocumentationToDoc();

}

/\*\*

\* @function summarizeSession

\* @description

\* يلخص سياق الجلسة الحالي.

\*/

function summarizeSession() {

return GAssistant.UI.summarizeSession();

}

/\*\*

\* @function manageAccessRestrictions

\* @description

\* تعرض واجهة لإدارة قيود الوصول. (Placeholder)

\*/

function manageAccessRestrictions() {

GAssistant.UI.manageAccessRestrictions();

}

/\*\*

\* @function refreshEnvSettingsWithNotice

\* @description

\* يعيد تحميل إعدادات البيئة مع إشعار للمستخدم.

\*/

function refreshEnvSettingsWithNotice() {

GAssistant.UI.refreshEnvSettingsWithNotice();

}

/\*\*

\* @function backupEnvSettingsToDrive

\* @description

\* يحفظ نسخة احتياطية من إعدادات البيئة إلى Google Drive.

\*/

function backupEnvSettingsToDrive() {

GAssistant.UI.backupEnvSettingsToDrive();

}

/\*\*

\* @function clearSessionContext

\* @description

\* تمسح ذاكرة الجلسة الحالية للمساعد.

\*/

function clearSessionContext() {

GAssistant.UI.clearSessionContext();

}

/\*\*

\* @function calculateGrossProfit

\* @description

\* يحسب إجمالي الربح لفترة محددة.

\* @param {object} options - خيارات الحساب.

\* @param {string} options.startDate - تاريخ البدء.

\* @param {string} options.endDate - تاريخ الانتهاء.

\* @returns {object} - كائن يحتوي على إجمالي الربح أو رسالة خطأ.

\*/

function calculateGrossProfit(options) {

return GAssistant.Tools.Accounting.calculateGrossProfit(options);

}

/\*\*

\* @function generateDailySummary

\* @description

\* يولد ملخصًا يوميًا للإيرادات والمصروفات.

\* @param {object} options - خيارات الملخص.

\* @param {string} options.date - التاريخ المطلوب.

\* @returns {object} - كائن يحتوي على الملخص اليومي أو رسالة خطأ.

\*/

function generateDailySummary(options) {

return GAssistant.Tools.Accounting.generateDailySummary(options);

}

/\*\*

\* @function calculateCategoryTotals

\* @description

\* يحلل الإيرادات/المصاريف حسب الفئة والزمن.

\* @param {object} options - خيارات التحليل.

\* @param {string} options.type - نوع التحليل.

\* @param {string} [options.category] - الفئة المراد تحليلها.

\* @param {string} [options.startDate] - تاريخ البدء.

\* @param {string} [options.endDate] - تاريخ الانتهاء.

\* @returns {object} - كائن يحتوي على الإجماليات أو رسالة خطأ.

\*/

function calculateCategoryTotals(options) {

return GAssistant.Tools.Accounting.calculateCategoryTotals(options);

}

/\*\*

\* @function generateIncomeStatement

\* @description

\* يولد قائمة دخل مبسطة لفترة محددة.

\* @param {object} options - خيارات التقرير.

\* @param {string} options.startDate - تاريخ البدء.

\* @param {string} options.endDate - تاريخ الانتهاء.

\* @returns {object} - كائن يحتوي على قائمة الدخل أو رسالة خطأ.

\*/

function generateIncomeStatement(options) {

return GAssistant.Tools.Accounting.generateIncomeStatement(options);

}

/\*\*

\* @function detectAnomalies

\* @description

\* يكتشف القيم الشاذة في ورقة مالية.

\* @param {object} options - خيارات الكشف.

\* @param {string} options.sheetName - اسم الورقة المالية.

\* @param {string} options.columnName - اسم العمود الذي سيتم فحصه.

\* @returns {object} - كائن يحتوي على القيم الشاذة أو رسالة خطأ.

\*/

function detectAnomalies(options) {

return GAssistant.Tools.Accounting.detectAnomalies(options);

}

/\*\*

\* @function generateProfitabilityAnalysis

\* @description

\* يولد تحليلًا شاملاً للربحية.

\* @param {object} options - خيارات التحليل.

\* @param {string} options.startDate - تاريخ البدء.

\* @param {string} options.endDate - تاريخ الانتهاء.

\* @returns {object} - كائن يحتوي على التحليل أو رسالة خطأ.

\*/

function generateProfitabilityAnalysis(options) {

return GAssistant.Tools.Accounting.generateProfitabilityAnalysis(options);

}

/\*\*

\* @function runAllTests

\* @description

\* يشغل جميع الاختبارات المحددة في المشروع.

\* @returns {object} - كائن يحتوي على ملخص ونتائج الاختبارات.

\*/

function runAllTests() {

return GAssistant.Tests.runAllTests();

}

/\*\*

\* @function runEdgeCaseTests

\* @description

\* يشغل اختبارات الحالات الهامشية المحددة.

\* @returns {object} - كائن يحتوي على ملخص ونتائج الاختبارات.

\*/

function runEdgeCaseTests() {

return GAssistant.Tests.runEdgeCaseTests();

}

/\*\*

\* @function GAssistant\_Tools\_Developer\_inspectProject

\* @description

\* يفحص جميع ملفات السكربت في المشروع ويقترح تحسينات.

\* (تم تسميتها هكذا لتجنب التضارب مع الاسم العام للدالة في GAssistant.Tools.Developer)

\* @returns {Array<object>} - قائمة بالاقتراحات لكل ملف.

\*/

function GAssistant\_Tools\_Developer\_inspectProject() {

return GAssistant.Tools.Developer.inspectProject();

}

/\*\*

\* @function showDeveloperToolsSidebar

\* @description

\* تعرض الشريط الجانبي لأدوات المطور.

\*/

function showDeveloperToolsSidebar() {

GAssistant.UI.showDeveloperToolsSidebar();

}

/\*\*

\* @function showWorkshopSidebar

\* @description

\* تعرض الشريط الجانبي لورشة عمل المبرمج.

\*/

function showWorkshopSidebar() {

GAssistant.UI.showWorkshopSidebar();

}

/\*\*

\* @function proposeAndStageCodeFromSidebar

\* @description

\* يقترح كود دالة ويجهزه في ورقة Staging.

\* @param {string} functionCode - الكود البرمجي للدالة المقترحة.

\* @returns {object} - رسالة نجاح أو خطأ.

\*/

function proposeAndStageCodeFromSidebar(functionCode) {

return GAssistant.WorkshopTools.proposeAndStageCodeFromSidebar(functionCode);

}

/\*\*

\* @function suggestFileFromSidebar

\* @description

\* ترسل كود الدالة المقترحة إلى جانب الخادم لاقتراح الملف الأنسب.

\* @param {string} functionCode - كود الدالة المقترحة من المستخدم.

\* @returns {{suggestedFile: string}} - كائن يحتوي على اسم الملف المقترح.

\*/

function suggestFileFromSidebar(functionCode) {

return GAssistant.WorkshopTools.suggestFileFromSidebar(functionCode);

}

/\*\*

\* @function PROMPT\_ENGINE\_buildPrompt

\* @description

\* تُنشئ systemPrompt مهيأ استنادًا إلى المعطيات.

\* (تم تسميتها هكذا لتجنب التضارب مع الاسم العام للدالة في GAssistant.PromptEngine)

\* @param {Object} options - خيارات بناء الـ prompt.

\* @returns {string} systemPrompt مهيأ للإرسال.

\*/

function PROMPT\_ENGINE\_buildPrompt(options) {

return GAssistant.PromptEngine.buildPrompt(options);

}

/\*\*

\* @function logUserFeedback

\* @description

\* يسجل ملاحظات المستخدم حول أداء المساعد.

\* @param {string} feedbackText - نص الملاحظة.

\* @param {string} [type='general'] - نوع الملاحظة.

\* @returns {object} - رسالة تأكيد.

\*/

function logUserFeedback(feedbackText, type) {

return GAssistant.AI.Memory.logUserFeedback(feedbackText, type);

}

/\*\*

\* @function presentDecisionWithJustification

\* @description

\* تعرض الإجراء المقترح ومنطقه قبل تنفيذ حرج.

\* @param {string} proposedAction - وصف الإجراء المقترح.

\* @param {string} justification - المنطق أو التبرير وراء الاقتراح.

\* @param {object} payload - الحمولة التي سيتم تمريرها إذا وافق المستخدم.

\* @returns {object} - كائن طلب التأكيد.

\*/

function presentDecisionWithJustification(proposedAction, justification, payload) {

return GAssistant.AI.Dialogue.presentDecisionWithJustification(proposedAction, justification, payload);

}

/\*\*

\* @function executeInternalAction

\* @description

\* نقطة دخول عامة لتنفيذ الإجراءات الداخلية بناءً على اسم الإجراء.

\* تُستخدم غالبًا بواسطة `Code.gs` بعد تلقي تأكيد من الواجهة الأمامية.

\* @param {string} actionName - اسم الإجراء الداخلي المراد تنفيذه.

\* @param {object} params - المعاملات التي سيتم تمريرها للإجراء.

\* @returns {object} - نتيجة تنفيذ الإجراء.

\*/

function executeInternalAction(actionName, params) {

return GAssistant.Tools.Internal.executeInternalAction(actionName, params);

}

/\*\*

\* @function GAssistant\_UI\_getConstitutionText

\* @description

\* يجلب نص دستور مساعد Gemini AI لعرضه في الشريط الجانبي.

\* (تم تسميتها هكذا لتجنب التضارب مع الاسم العام للدالة في GAssistant.UI)

\* @returns {string} - نص الدستور.

\*/

function GAssistant\_UI\_getConstitutionText() {

return GAssistant.UI.getConstitutionText();

}