/\*\*

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

\* @file tests.gs – Intelligent Testing Module for G-Assistant (v2.5)

\* @description

\* Contains integration tests for all important modules in the system.

\* Separation between execution and display to support integration with the sidebar.

\* Added tests for built-in Apps Script function search and internal tool documentation functions.

\* Depends on: GAssistant.Helpers, GAssistant.AI.Dialogue, GAssistant.AI.Core,

\* GAssistant.Tools.Sheets, GAssistant.Tools.Accounting, GAssistant.Tools.General,

\* GAssistant.BuiltinDocs, GAssistant.Tools.DocGen, GAssistant.Tools.Workshop

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

\*/

var GAssistant = GAssistant || {};

// Inject dependencies: all dependent modules must be passed here with full namespace names

GAssistant.Tests = (function(

GA\_Helpers,

GA\_AI\_Dialogue,

GA\_AI\_Core,

GA\_Tools\_Sheets,

GA\_Tools\_Accounting,

GA\_Tools\_General,

GA\_BuiltinDocs,

GA\_Tools\_DocGen,

GA\_Tools\_Workshop // Added Workshop Tools

) {

'use strict';

/\*\*

\* @typedef {Object} TestResult

\* @property {string} name - اسم الاختبار.

\* @property {boolean} success - هل نجح الاختبار؟

\* @property {string} message - رسالة توضيحية (خطأ أو نجاح).

\*/

/\*\*

\* @const {Array<Object>} TEST\_CASES - مصفوفة تعريفات حالات الاختبار.

\* كل كائن يمثل اختبارًا، مع دالة التنفيذ والوصف.

\*/

const TEST\_CASES = [

{

name: "اختبار: جلب النماذج المتاحة",

description: "يتأكد من أن دالة getAvailableModels تُرجع قائمة بالنماذج.",

run: () => {

const result = GAssistant.Config.getAllowedModels(); // استخدام GAssistant.Config

if (Array.isArray(result) && result.length > 0) {

return { success: true, message: `تم جلب ${result.length} نموذج بنجاح.` };

}

throw new Error("لم يتم جلب أي نماذج أو الرد ليس مصفوفة.");

}

},

{

name: "اختبار: كتابة إلى خلية (TOOLS\_SHEETS)",

description: "يتأكد من أن دالة writeToCell تعمل بشكل صحيح.",

run: () => {

const sheetName = "TestSheet\_Write";

const cellA1 = "A1";

const value = "Hello Test!";

const result = GA\_Tools\_Sheets.writeToCell(sheetName, cellA1, value);

if (result.type === "success") {

// يمكننا هنا قراءة القيمة للتحقق، ولكن هذا يتطلب الوصول المباشر لـ SpreadsheetApp

// وهو ليس مثاليًا في اختبار الوحدة المعزول. نعتمد على رسالة النجاح في هذه الحالة.

return { success: true, message: `تم الكتابة بنجاح في ${sheetName}:${cellA1}.` };

}

throw new Error(`فشل الكتابة: ${result.text}`);

}

},

{

name: "اختبار: مسح نطاق (TOOLS\_SHEETS)",

description: "يتأكد من أن دالة clearRange تعمل.",

run: () => {

const sheetName = "TestSheet\_Clear";

const rangeA1 = "A1:B2";

// يجب أن نكتب شيئًا أولاً للمسح

GA\_Tools\_Sheets.writeToCell(sheetName, "A1", "Data");

GA\_Tools\_Sheets.writeToCell(sheetName, "B2", "More Data");

const result = GA\_Tools\_Sheets.clearRange(sheetName, rangeA1);

if (result.type === "success") {

return { success: true, message: `تم مسح النطاق بنجاح في ${sheetName}:${rangeA1}.` };

}

throw new Error(`فشل المسح: ${result.text}`);

}

},

{

name: "اختبار: إدراج صف (TOOLS\_SHEETS)",

description: "يتأكد من أن دالة insertRowAt تعمل.",

run: () => {

const sheetName = "TestSheet\_Insert";

const rowIndex = 1;

const result = GA\_Tools\_Sheets.insertRowAt(sheetName, rowIndex);

if (result.type === "success") {

return { success: true, message: `تم إدراج صف بنجاح في ${sheetName} في الصف ${rowIndex}.` };

}

throw new Error(`فشل الإدراج: ${result.text}`);

}

},

{

name: "اختبار: حذف صف (TOOLS\_SHEETS)",

description: "يتأكد من أن دالة deleteRow تعمل.",

run: () => {

const sheetName = "TestSheet\_Delete";

const rowIndex = 1;

// يجب أن ندرج صفًا أولاً للحذف

GA\_Tools\_Sheets.insertRowAt(sheetName, rowIndex);

const result = GA\_Tools\_Sheets.deleteRow(sheetName, rowIndex);

if (result.type === "success") {

return { success: true, message: `تم حذف صف بنجاح من ${sheetName} في الصف ${rowIndex}.` };

}

throw new Error(`فشل الحذف: ${result.text}`);

}

},

{

name: "اختبار: تلخيص ورقة نشطة (TOOLS\_SHEETS)",

description: "يتأكد من أن summarizeActiveSheet تُرجع ملخصًا.",

run: () => {

// إنشاء ورقة اختبار مؤقتة

const tempSheetName = "TempSummaryTest";

const ss = SpreadsheetApp.getActiveSpreadsheet();

let tempSheet = ss.getSheetByName(tempSheetName);

if (!tempSheet) {

tempSheet = ss.insertSheet(tempSheetName);

}

tempSheet.clear();

tempSheet.appendRow(["Header1", "Header2", "Header3"]);

tempSheet.appendRow(["Value1", 123, new Date()]);

tempSheet.appendRow(["Value2", "", "Text"]);

ss.setActiveSheet(tempSheet); // تفعيل الورقة للاختبار

const result = GA\_Tools\_Sheets.summarizeActiveSheet();

ss.deleteSheet(tempSheet); // حذف الورقة بعد الاختبار

if (result.type === "info" && result.text.includes("وصف الورقة")) {

return { success: true, message: "تم توليد ملخص الورقة النشطة بنجاح." };

}

throw new Error(`فشل تلخيص الورقة النشطة: ${result.text}`);

}

},

{

name: "اختبار: إنشاء ورقة اختبار بسيطة (TOOLS\_SHEETS)",

description: "يتأكد من أن createMinimalTestSheet تنشئ ورقة بالبيانات الصحيحة.",

run: () => {

const result = GA\_Tools\_Sheets.createMinimalTestSheet();

if (result.type === "success") {

// يمكن التحقق من وجود الورقة ومحتواها هنا إذا أردت اختبارًا أعمق

return { success: true, message: "تم إنشاء ورقة الاختبار البسيطة بنجاح." };

}

throw new Error(`فشل إنشاء ورقة الاختبار: ${result.text}`);

}

},

{

name: "اختبار: حساب إجمالي الربح (TOOLS\_ACCOUNTING)",

description: "يتأكد من أن calculateGrossProfit تُرجع بيانات ربح صالحة.",

run: () => {

// هذا الاختبار يتطلب بيانات في أوراق الإيرادات والمصروفات

// لتبسيط الاختبار، سنقوم بمحاكاة بيانات بسيطة أو نفترض وجودها

// في بيئة حقيقية، ستحتاج إلى إعداد بيانات اختبار مسبقًا أو استخدام بيانات وهمية

const result = GA\_Tools\_Accounting.calculateGrossProfit("2024-01-01", "2024-01-31");

if (result && typeof result.totalRevenue === 'number' && typeof result.totalExpenses === 'number') {

return { success: true, message: "تم حساب إجمالي الربح بنجاح (افتراضي)." };

}

throw new Error("فشل حساب إجمالي الربح أو الرد غير صالح.");

}

},

{

name: "اختبار: توليد ملخص يومي (TOOLS\_ACCOUNTING)",

description: "يتأكد من أن generateDailySummary تُرجع ملخصًا يوميًا.",

run: () => {

const result = GA\_Tools\_Accounting.generateDailySummary();

if (result && result.date && typeof result.net === 'number') {

return { success: true, message: "تم توليد الملخص اليومي بنجاح (افتراضي)." };

}

throw new Error("فشل توليد الملخص اليومي أو الرد غير صالح.");

}

},

{

name: "اختبار: اقتراح ملف من الكود (WORKSHOP\_TOOLS)",

description: "يتأكد من أن suggestFileFromSidebar يقترح ملفًا بناءً على الكود.",

run: () => {

const sampleCode = `function myFunction() { SpreadsheetApp.getActiveSpreadsheet(); }`;

const result = GA\_Tools\_Workshop.suggestFileFromSidebar(sampleCode);

if (result && result.suggestedFile && result.suggestedFile.includes('sheets')) { // يتوقع أن يقترح ملف الأوراق

return { success: true, message: `تم اقتراح ملف بنجاح: ${result.suggestedFile}.` };

}

throw new Error(`فشل اقتراح الملف: ${JSON.stringify(result)}`);

}

},

{

name: "اختبار: جلب توثيق دالة مدمجة (BUILTIN\_DOCS)",

description: "يتأكد من أن getBuiltinFunctionDoc تُرجع توثيقًا لدالة Apps Script.",

run: () => {

const functionName = "SpreadsheetApp.getActiveSpreadsheet";

const result = GA\_BuiltinDocs.getBuiltinFunctionDoc(functionName);

if (result && result.text && result.text.includes("SpreadsheetApp")) {

return { success: true, message: `تم جلب توثيق ${functionName} بنجاح.` };

}

throw new Error(`فشل جلب توثيق الدالة: ${JSON.stringify(result)}`);

}

},

{

name: "اختبار: مراجعة الكود (ScriptDoctor)",

description: "يتأكد من أن ScriptDoctor يُرجع مراجعة للكود.",

run: () => {

const sampleCode = `function sum(a, b) { return a + b; }`;

const result = GAssistant.AI.ScriptDoctor.reviewCode(sampleCode, 'general');

if (result && typeof result.overallScore === 'number' && result.summary) {

return { success: true, message: `تمت مراجعة الكود بنجاح (الدرجة: ${result.overallScore}).` };

}

throw new Error(`فشل مراجعة الكود: ${JSON.stringify(result)}`);

}

}

// أضف المزيد من حالات الاختبار هنا

];

/\*\*

\* @function runAllTests

\* @description

\* تُشغل جميع حالات الاختبار المحددة وتُرجع النتائج.

\* @returns {Array<TestResult>} - مصفوفة بكائنات نتائج الاختبار.

\*/

function runAllTests() {

GA\_Helpers.logOperation('GAssistant.Tests', 'runAllTests:start', { testCount: TEST\_CASES.length });

const results = [];

TEST\_CASES.forEach(testCase => {

try {

const testResult = testCase.run();

results.push({ name: testCase.name, success: true, message: testResult.message || "نجح." });

} catch (e) {

results.push({ name: testCase.name, success: false, message: e.message });

}

});

GA\_Helpers.logOperation('GAssistant.Tests', 'runAllTests:end', { totalTests: results.length, failed: results.filter(r => !r.success).length });

return results;

}

/\*\*

\* @function showTestResultsSidebar

\* @description

\* تعرض الشريط الجانبي لنتائج الاختبارات.

\*/

function showTestResultsSidebar() {

GA\_Helpers.logOperation('GAssistant.Tests', 'showTestResultsSidebar:start', {});

try {

const html = HtmlService.createHtmlOutputFromFile("TestResultsSidebar")

.setTitle("🧪 نتائج الاختبارات")

.setWidth(420);

SpreadsheetApp.getUi().showSidebar(html);

GA\_Helpers.logOperation('GAssistant.Tests', 'showTestResultsSidebar:end', {});

} catch (e) {

GA\_Helpers.logOperation('GAssistant.Tests', 'showTestResultsSidebar:error', { error: e.message });

GA\_AI\_Dialogue.showError(`❌ فشل في فتح شريط نتائج الاختبارات: ${e.message}`);

}

}

/\*\*

\* @function showErrorLoadingTests

\* @description دالة مساعدة لعرض رسالة خطأ إذا فشلت الاختبارات في التحميل.

\* @returns {void}

\*/

function showErrorLoadingTests() {

// افتراض وجود GAssistant.AI.Dialogue

if (typeof GAssistant.AI.Dialogue !== 'undefined' && GAssistant.AI.Dialogue.showError) {

GAssistant.AI.Dialogue.showError('فشل تحميل وحدة الاختبارات. يرجى التحقق من ملف tests.gs.');

} else {

SpreadsheetApp.getUi().alert('فشل تحميل وحدة الاختبارات. يرجى التحقق من ملف tests.gs.');

}

}

/\*\*

\* @function getAllTestResults

\* @description دالة عامة تُستخدم بواسطة google.script.run لجلب جميع نتائج الاختبارات.

\* @returns {Array<{name:string, success:boolean, message:string}>} - مصفوفة كائنات نتائج الاختبار.

\*/

function getAllTestResults() {

// التأكد من أن الوحدة GAssistant.Tests جاهزة قبل تشغيل الاختبارات

if (typeof GAssistant.Tests !== 'undefined' && GAssistant.Tests.runAllTests) {

return GAssistant.Tests.runAllTests();

} else {

showErrorLoadingTests();

return [];

}

}

// --- تصدير الواجهة العامة للوحدة ---

return {

runAllTests,

showTestResultsSidebar,

getAllTestResults // تصدير الدالة العامة لـ google.script.run

};

})(

GAssistant.Helpers,

GAssistant.AI.Dialogue,

GAssistant.AI.Core,

GAssistant.Tools.Sheets,

GAssistant.Tools.Accounting,

GAssistant.Tools.General, // افتراض وجود هذه الوحدة

GAssistant.BuiltinDocs,

GAssistant.Tools.DocGen,

GAssistant.Tools.Workshop

);

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

// الدوال العامة (Global Functions)

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

// هذه الدوال ضرورية لتمكين google.script.run من استدعاء الدوال من HTML.

// يجب أن تكون هذه الدوال في النطاق العام (global scope) لكي يتمكن google.script.run من رؤيتها.

/\*\*

\* @function runAllTestsInSidebar

\* @description

\* دالة عامة تُستخدم لفتح الشريط الجانبي وتشغيل جميع الاختبارات.

\*/

function runAllTestsInSidebar() {

GAssistant.Tests.showTestResultsSidebar();

}