/\*\*

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

\* @file tools\_ai\_helpers.gs – v1.2

\* @module TOOLS\_AI\_HELPERS

\* @description

\* وحدة مساعدة لدعم PROMPT\_ENGINE بوصف الورقة الذكي، بالإضافة إلى أدوات AI جديدة.

\* -----------------------------------------------------------------------------

\* تشمل:

\* - getSheetInsightForPrompt(): وصف Markdown

\* - getSheetInsightAsJson(): وصف JSON منظم

\* - summarizeActiveSheetWithGemini(): تلخيص الورقة النشطة باستخدام Gemini

\* - suggestFormulaWithGemini(): اقتراح صيغة Google Sheets باستخدام Gemini

\* - categorizeColumnWithGemini(): تصنيف عمود باستخدام Gemini

\* - translateCellWithGemini(): ترجمة خلية/نطاق باستخدام Gemini

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

\*/

var TOOLS\_AI\_HELPERS = (function (GA\_AI\_Core, GA\_AI\_Dialogue, GA\_Helpers, ENV) { // إضافة GA\_AI\_Core, GA\_AI\_Dialogue, ENV

'use strict';

// عدد الصفوف المراد أخذ عينة منها للتلخيص أو التحليل

const SAMPLE\_DATA\_ROW\_COUNT = ENV.SETTINGS.SAMPLE\_DATA\_ROW\_COUNT || 5;

/\*\*

\* 📝 [Markdown] توليد وصف مختصر للورقة الحالية بصيغة Markdown

\* @returns {string}

\*/

function getSheetInsightForPrompt() {

try {

const sheet = SpreadsheetApp.getActiveSheet();

const name = sheet.getName();

const data = sheet.getDataRange().getValues();

if (data.length < 1) return `📄 الورقة "${name}" فارغة.`;

const headers = data[0];

const rowCount = data.length - 1;

const colCount = headers.length;

let summary = `### 🧾 وصف الورقة "${name}"\\n`;

summary += `- 📌 عدد الأعمدة: ${colCount}\\n`;

summary += `- 📊 عدد الصفوف (بدون الرؤوس): ${rowCount}\\n`;

summary += `- 🏷️ الأعمدة:\\n`;

headers.forEach((h, i) => {

summary += ` - ${String.fromCharCode(65 + i)}: ${h}\\n`;

});

// إضافة عينة من البيانات

if (data.length > 1) {

summary += `\\n\*\*عينة من البيانات (أول ${Math.min(SAMPLE\_DATA\_ROW\_COUNT, rowCount)} صف):\*\*\\n`;

const sampleRows = data.slice(1, SAMPLE\_DATA\_ROW\_COUNT + 1);

sampleRows.forEach(row => {

summary += ` - ${row.join(' | ')}\\n`;

});

}

return summary;

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'getSheetInsightForPrompt:error', { error: e.message });

return `❌ فشل في توليد وصف Markdown: ${e.message}`;

}

}

/\*\*

\* 🧾 [JSON] وصف الورقة الحالية بشكل منظم

\* @returns {string}

\*/

function getSheetInsightAsJson() {

try {

const sheet = SpreadsheetApp.getActiveSheet();

const name = sheet.getName();

const data = sheet.getDataRange().getValues();

if (data.length < 1) return JSON.stringify({ sheet: name, message: "الورقة فارغة" }, null, 2);

const headers = data[0];

const sampleRows = data.slice(1, SAMPLE\_DATA\_ROW\_COUNT + 1); // نأخذ أول X صفوف فقط

const structure = {

sheetName: name,

columns: headers.map((h, i) => ({ name: h, index: i + 1 })),

sampleData: sampleRows

};

return JSON.stringify(structure, null, 2);

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'getSheetInsightAsJson:error', { error: e.message });

return JSON.stringify({ error: `فشل في توليد وصف JSON: ${e.message}` }, null, 2);

}

}

/\*\*

\* @function summarizeActiveSheetWithGemini

\* @description

\* تستخدم Gemini AI لتلخيص محتوى الورقة النشطة.

\* @returns {object} - كائن استجابة موحد (type: 'text' أو 'message').

\*/

function summarizeActiveSheetWithGemini() {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'summarizeActiveSheetWithGemini:start');

try {

const sheetInsight = getSheetInsightForPrompt(); // جلب وصف الورقة بصيغة Markdown

if (sheetInsight.includes('فشل في توليد وصف')) {

return GA\_AI\_Dialogue.showError("خطأ في قراءة الورقة", sheetInsight);

}

if (sheetInsight.includes('الورقة فارغة')) {

return GA\_AI\_Dialogue.showMessage("info", sheetInsight);

}

const prompt = `بناءً على وصف ورقة Google Sheets التالي وعينة البيانات، قدم ملخصًا موجزًا وشاملاً للبيانات الموجودة في الورقة، والغرض المحتمل منها، وأي رؤى أولية يمكنك استنتاجها. استخدم تنسيق Markdown:\n\n${sheetInsight}\n\nملخص الورقة:`;

const result = GA\_AI\_Core.ask(prompt, "gemini-1.5-flash"); // استخدام نموذج flash للتلخيص السريع

if (result?.type === 'text' && result.content) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'summarizeActiveSheetWithGemini:success', { summaryLength: result.content.length });

return result;

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'summarizeActiveSheetWithGemini:no\_text\_response', { result });

return GA\_AI\_Dialogue.showError("خطأ", "لم يتمكن Gemini من تلخيص الورقة بشكل نصي.");

}

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'summarizeActiveSheetWithGemini:failure', { error: e.message });

return GA\_AI\_Dialogue.showError("خطأ", `فشل تلخيص الورقة: ${e.message}`);

}

}

/\*\*

\* @function suggestFormulaWithGemini

\* @description

\* تستخدم Gemini AI لاقتراح صيغة Google Sheets بناءً على وصف المستخدم.

\* @param {string} description - وصف المستخدم للصيغة المطلوبة (مثال: "مجموع العمود B").

\* @returns {object} - كائن استجابة موحد (type: 'text' أو 'message').

\*/

function suggestFormulaWithGemini(description) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'suggestFormulaWithGemini:start', { description: description });

try {

const sheetInsight = getSheetInsightForPrompt(); // لتوفير سياق حول الورقة الحالية

const prompt = `بناءً على وصف ورقة Google Sheets التالية وعينة البيانات، قم بتوليد صيغة Google Sheets (Formula) التي تحقق الوصف التالي: "${description}".

يجب أن تكون الصيغة دقيقة، قابلة للتطبيق مباشرة في Google Sheets، ولا تتضمن أي شرح إضافي أو نص قبلها أو بعدها.

إذا كانت الصيغة تتطلب نطاقًا، استخدم نطاقات افتراضية مثل A1:A أو B:B بناءً على الوصف.

الورقة الحالية:

${sheetInsight}

الصيغة المطلوبة:`;

const result = GA\_AI\_Core.ask(prompt, "gemini-1.5-flash"); // استخدام نموذج flash لاقتراح الصيغ

if (result?.type === 'text' && result.content) {

const formula = result.content.trim();

// التأكد من أن الرد يبدأ بعلامة =

if (formula.startsWith('=')) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'suggestFormulaWithGemini:success', { formula: formula });

return GA\_AI\_Dialogue.showMessage("success", `✅ الصيغة المقترحة: \`\`\`${formula}\`\`\``);

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'suggestFormulaWithGemini:invalid\_formula', { formula: formula });

return GA\_AI\_Dialogue.showError("خطأ", `لم يتمكن Gemini من توليد صيغة صالحة. الرد كان: \`\`\`${formula}\`\`\``);

}

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'suggestFormulaWithGemini:no\_text\_response', { result });

return GA\_AI\_Dialogue.showError("خطأ", "لم يتمكن Gemini من اقتراح صيغة بشكل نصي.");

}

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'suggestFormulaWithGemini:failure', { error: e.message });

return GA\_AI\_Dialogue.showError("خطأ", `فشل اقتراح الصيغة: ${e.message}`);

}

}

/\*\*

\* @function categorizeColumnWithGemini

\* @description

\* تستخدم Gemini AI لتصنيف البيانات في عمود محدد بناءً على وصف المستخدم.

\* @param {number} columnIndex - رقم العمود المراد تصنيفه (1-based).

\* @param {string} categoryPrompt - وصف التصنيفات المطلوبة (مثال: "إيجابي، سلبي، محايد").

\* @returns {object} - كائن استجابة موحد (type: 'summary\_table' أو 'message').

\*/

function categorizeColumnWithGemini(columnIndex, categoryPrompt) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'categorizeColumnWithGemini:start', { columnIndex, categoryPrompt });

try {

const sheet = SpreadsheetApp.getActiveSheet();

const dataRange = sheet.getDataRange();

const values = dataRange.getValues();

if (values.length < 2) { // Less than 2 rows means no data beyond headers

return GA\_AI\_Dialogue.showMessage("info", "الورقة فارغة أو تحتوي على رؤوس فقط. لا توجد بيانات لتصنيفها.");

}

const header = values[0][columnIndex - 1]; // Get header of the column

const columnData = values.slice(1).map(row => row[columnIndex - 1]); // Get data from the column, excluding header

if (columnData.every(item => !item)) { // Check if all data in column is empty

return GA\_AI\_Dialogue.showMessage("info", `العمود ${columnIndex} فارغ. لا توجد بيانات لتصنيفها.`);

}

// Prepare data for Gemini, limit to SAMPLE\_DATA\_ROW\_COUNT for efficiency

const sampleData = columnData.slice(0, SAMPLE\_DATA\_ROW\_COUNT).filter(item => item !== null && item !== undefined && String(item).trim() !== '');

if (sampleData.length === 0) {

return GA\_AI\_Dialogue.showMessage("info", `العمود ${columnIndex} لا يحتوي على بيانات صالحة للتصنيف.`);

}

const prompt = `أمامك بيانات من عمود في جدول بيانات Google Sheets. رأس العمود هو "${header}".

المطلوب هو تصنيف كل عنصر من البيانات في هذا العمود بناءً على الوصف التالي للتصنيفات: "${categoryPrompt}".

أعد لي مصفوفة JSON من الكائنات، حيث يحتوي كل كائن على "original\_value" (القيمة الأصلية) و "category" (التصنيف الذي تم تحديده).

لا تقم بتضمين أي نص إضافي أو شرح قبل أو بعد مصفوفة JSON.

إليك عينة من البيانات من هذا العمود:

${JSON.stringify(sampleData)}

مثال على التنسيق المطلوب:

[

{"original\_value": "قيمة1", "category": "تصنيف1"},

{"original\_value": "قيمة2", "category": "تصنيف2"}

]

`;

const payload = {

contents: [{ role: "user", parts: [{ text: prompt }] }],

generationConfig: {

responseMimeType: "application/json",

responseSchema: {

type: "ARRAY",

items: {

type: "OBJECT",

properties: {

"original\_value": { "type": "STRING" },

"category": { "type": "STRING" }

},

"propertyOrdering": ["original\_value", "category"]

}

}

}

};

const result = GA\_AI\_Core.ask(payload, "gemini-1.5-pro"); // استخدام نموذج pro للاستجابات المنظمة

if (result?.type === 'structured\_data' && result.content) {

const categories = result.content;

const outputRows = categories.map(item => [item.original\_value, item.category]);

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'categorizeColumnWithGemini:success', { count: outputRows.length });

return GA\_AI\_Dialogue.showSummaryTable(outputRows, `نتائج تصنيف العمود ${columnIndex} ("${header}")`, ["القيمة الأصلية", "التصنيف"]);

} else if (result?.type === 'text' && result.content) {

// Fallback for text response if structured data fails

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'categorizeColumnWithGemini:fallback\_text\_response', { content: result.content.substring(0, 100) });

return GA\_AI\_Dialogue.showError("خطأ في التصنيف", `لم يتمكن Gemini من إرجاع بيانات منظمة، ولكن الرد كان: ${result.content}`);

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'categorizeColumnWithGemini:no\_structured\_response', { result });

return GA\_AI\_Dialogue.showError("خطأ", "لم يتمكن Gemini من تصنيف البيانات بشكل منظم.");

}

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'categorizeColumnWithGemini:failure', { error: e.message });

return GA\_AI\_Dialogue.showError("خطأ", `فشل تصنيف العمود: ${e.message}`);

}

}

/\*\*

\* @function translateCellWithGemini

\* @description

\* تستخدم Gemini AI لترجمة محتوى خلية أو نطاق من الخلايا.

\* @param {string} cellA1 - عنوان الخلية أو النطاق (مثال: "A1" أو "A1:A5").

\* @param {string} targetLanguage - اللغة المستهدفة للترجمة (مثال: "الإنجليزية", "الفرنسية").

\* @returns {object} - كائن استجابة موحد (type: 'text' أو 'summary\_table' أو 'message').

\*/

function translateCellWithGemini(cellA1, targetLanguage) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:start', { cellA1, targetLanguage });

try {

const sheet = SpreadsheetApp.getActiveSheet();

const range = sheet.getRange(cellA1);

const values = range.getValues();

if (values.length === 0 || values[0].length === 0) {

return GA\_AI\_Dialogue.showMessage("info", "الخلية أو النطاق المحدد فارغ. لا توجد بيانات للترجمة.");

}

if (values.length === 1 && values[0].length === 1) {

// ترجمة خلية واحدة

const textToTranslate = String(values[0][0]);

if (textToTranslate.trim() === '') {

return GA\_AI\_Dialogue.showMessage("info", "الخلية المحددة فارغة. لا توجد بيانات للترجمة.");

}

const prompt = `ترجم النص التالي إلى ${targetLanguage}. أعد النص المترجم فقط، بدون أي شرح إضافي:\n\n"${textToTranslate}"`;

const result = GA\_AI\_Core.ask(prompt, "gemini-1.5-flash"); // استخدام نموذج flash للترجمة السريعة

if (result?.type === 'text' && result.content) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:single\_cell\_success', { translatedText: result.content.substring(0, 50) });

return GA\_AI\_Dialogue.showMessage("success", `✅ الترجمة لـ "${textToTranslate}": \`\`\`${result.content}\`\`\``);

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:single\_cell\_no\_text\_response', { result });

return GA\_AI\_Dialogue.showError("خطأ", "لم يتمكن Gemini من ترجمة الخلية بشكل نصي.");

}

} else {

// ترجمة نطاق من الخلايا

const textsToTranslate = values.flat().filter(item => item !== null && item !== undefined && String(item).trim() !== '');

if (textsToTranslate.length === 0) {

return GA\_AI\_Dialogue.showMessage("info", "النطاق المحدد فارغ. لا توجد بيانات للترجمة.");

}

const prompt = `ترجم كل عنصر من القائمة التالية إلى ${targetLanguage}. أعد مصفوفة JSON من الكائنات، حيث يحتوي كل كائن على "original\_text" و "translated\_text". لا تقم بتضمين أي نص إضافي أو شرح قبل أو بعد مصفوفة JSON.

\nالقائمة:\n${JSON.stringify(textsToTranslate)}

مثال على التنسيق المطلوب:

[

{"original\_text": "Hello", "translated\_text": "مرحباً"},

{"original\_text": "World", "translated\_text": "العالم"}

]

`;

const payload = {

contents: [{ role: "user", parts: [{ text: prompt }] }],

generationConfig: {

responseMimeType: "application/json",

responseSchema: {

type: "ARRAY",

items: {

type: "OBJECT",

properties: {

"original\_text": { "type": "STRING" },

"translated\_text": { "type": "STRING" }

},

"propertyOrdering": ["original\_text", "translated\_text"]

}

}

}

};

const result = GA\_AI\_Core.ask(payload, "gemini-1.5-pro"); // استخدام نموذج pro للاستجابات المنظمة

if (result?.type === 'structured\_data' && result.content) {

const translations = result.content;

const outputRows = translations.map(item => [item.original\_text, item.translated\_text]);

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:range\_success', { count: outputRows.length });

return GA\_AI\_Dialogue.showSummaryTable(outputRows, `نتائج ترجمة النطاق ${cellA1} إلى ${targetLanguage}`, ["النص الأصلي", "النص المترجم"]);

} else if (result?.type === 'text' && result.content) {

// Fallback for text response if structured data fails

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:range\_fallback\_text\_response', { content: result.content.substring(0, 100) });

return GA\_AI\_Dialogue.showError("خطأ في الترجمة", `لم يتمكن Gemini من إرجاع بيانات منظمة، ولكن الرد كان: ${result.content}`);

} else {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:range\_no\_structured\_response', { result });

return GA\_AI\_Dialogue.showError("خطأ", "لم يتمكن Gemini من ترجمة النطاق بشكل منظم.");

}

}

} catch (e) {

GA\_Helpers.logOperation('TOOLS\_AI\_HELPERS', 'translateCellWithGemini:failure', { error: e.message });

return GA\_AI\_Dialogue.showError("خطأ", `فشل الترجمة: ${e.message}`);

}

}

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

return {

getSheetInsightForPrompt: getSheetInsightForPrompt,

getSheetInsightAsJson: getSheetInsightAsJson,

summarizeActiveSheetWithGemini: summarizeActiveSheetWithGemini,

suggestFormulaWithGemini: suggestFormulaWithGemini,

categorizeColumnWithGemini: categorizeColumnWithGemini, // تصدير الدالة الجديدة

translateCellWithGemini: translateCellWithGemini // تصدير الدالة الجديدة

};

})(

GAssistant.AI.Core, // تبعية لـ AI\_Core

GAssistant.AI.Dialogue, // تبعية لـ AI\_Dialogue

GAssistant.Helpers, // تبعية لـ Helpers

ENV // تبعية لـ ENV

);