// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// --- START OF FILE: 30 إلى "مساعد نشط" يقوم بعمليتين مترابطتين في خطوة واحدة، وهذا هو جوهر الأ\_tools/4\_tools\_developer.gs ---

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\*

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

\* @file 30\_tools/4\_tools\_developer.gs

تمتة الذكية.

سأقوم الآن بتنفيذ هذا الاقتراح، بالإضافة إلى \*\*الاقتراح رقم \* @module GAssistant.Tools.Developer

\* @description

\* أدوات مخصصة لمساعدة المطورين في (2)\*\* لأنه مرتبط به بشكل مباشر ويكمل تدفق العمل. سأقوم بتحديث ملف `30\_ صيانة وتوسيع مشروع G-Assistant.

\* \*\*تحديث:\*\* تم إضافة سير عمل متكامل للمراجعة والتجهيز، ودعم التصنيفات، وحفظ التوثيق في الذاكرة.

\* =================================tools/4\_tools\_developer.gs` ليشمل هذه القدرات الجديدة.

---

### \*\*ملف `30\_tools/4\_tools\_developer.gs` (النسخة النهائية مع التحسينات)\*\*

---

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

\*/

var GAssistant = GAssistant || {};

GAssistant.Tools = GAssistant.Tools || {};

GAssistant.Tools.Developer = (() => {

'use strict';

/\*\*

\* ✅ \*\*تطبيق الاشرح التحديثات:\*\*

1. \*\*دالة جديدة `reviewAndStageCode`:\*\* تم إنشاء هذه الدالة الشاملة. عند استدعائها، ستقوم أولاً بإرسال الكود إلى `AIقتراح 1\*\*: دالة متكاملة تراجع الكود ثم تجهزه.

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

\* @param {string} category.Core` لمراجعته (بنفس منطق `reviewCode` الذي بنيناه)، ثم تأخذ الكود \*\*ونتيجة المراجعة\*\* وتضعهما في ورشة عمل المطور.

2. \*\*تط - تصنيف الكود (e.g., 'AI', 'UI', 'Tool').

\* @returns {UiResponse}

\*/

function reviewAndStageCode(functionCode, category = 'General') {

const {وير ورشة العمل:\*\* تم تعديل `stageCodeInWorkshop` (واستخدامها داخليًا بواسطة الد Utils, Agents, UI } = GAssistant.Utils.Injector.get('Utils', 'Agents', 'UI');

الة الجديدة) لتكون أكثر تنظيمًا. الآن تسجل الكود المقترح، الملف المقترح، \*\*ملخص الم

return Utils.executeSafely(() => {

// الخطوة 1: مراجعة الكود باستخدام وكراجعة من Gemini\*\*، والحالة، في صفوف منفصلة، مما يجعل ورشة العمل أكثر فائدة وقيمةيل المطور

const reviewResult = Agents.Developer.suggestRefactoring(functionCode);

const reviewText.

```javascript

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// --- START OF FILE: 30\_tools/4 = reviewResult.type === 'info' ? reviewResult.text : "فشلت المراجعة التلقائية.";\_tools\_developer.gs ---

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*\*

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

\* @file 30\_tools/4\_tools\_developer.gs

\* @module GAssistant.Tools.Developer

\* @description

\* أدوات متقدمة لمساعدة المطورين.

\*

// الخطوة 2: تجهيز الكود مع نتيجة المراجعة

const sheetName = GAssistant.Config.get('DEVELOPER\_WORKSHOP\_SHEET');

const headers = ['الفئة', 'الك \*\*تحديث:\*\* تم إضافة دالة `reviewAndStageCode` لربط المراجعة بالتجهيز،

\* وتطوير ورشة العمل لتشمل ملخص المراجعة.

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

ود المقترح', 'نتيجة المراجعة', 'الحالة', 'التاريخ'];

const sheet = Utils.getSheet(sheetName, headers);

if (!sheet) return UI.Dialogue.createError("فشل الوصول إلى ورشة عمل المطور.");

const newRow = [category, functionCode, reviewText, ' \*/

var GAssistant = GAssistant || {};

GAssistant.Tools = GAssistant.Tools || {};

GAssistant.Tools.Developer = (() => {

'use strict';

/\*\*

\* ✅ \*\*الدللمراجعة البشرية', new Date()];

sheet.appendRow(newRow);

Utils.log('Tools.Developer.reviewAndStageCode: Code reviewed and staged.');

return UI.Dialogue.createالة الجديدة والشاملة (تطبيق الاقتراح 1)\*\*

\* تراجع الكود، تقترح ملفًا، ثم تجهز كل شيء في ورشة العمل.

\* @param {string} functionCode - الكودSuccess(`✅ تمت مراجعة الكود وتجهيزه في ورشة العمل.`);

}, [], 'reviewAndStageCode');

}

/\*\*

\* يجلب توثيق دالة Apps Script مدمجة، مع استخدام البرمجي المراد مراجعته وتجهيزه.

\* @returns {UiResponse}

\*/

function reviewAndStageCode(functionCode) {

const { Utils, UI, Tools, AI الكاش أولاً.

\* @param {object} args - { functionName: string }.

\* @returns { } = GAssistant.Utils.Injector.get('Utils', 'UI', 'Tools', 'AI');

returnUiResponse}

\*/

function getBuiltinFunctionDoc(args) {

const { functionName } = args;

const { Utils, AI, Config } = GAssistant.Utils.Injector.get('Utils', Utils.executeSafely(() => {

Utils.log('Developer.reviewAndStageCode: Starting...', { codeLength: functionCode.length });

// الخطوة 1: اقتراح الملف المناسب

const suggestedFile = suggest 'AI', 'Config');

return Utils.executeSafely(() => {

// ✅ \*\*TargetFile(functionCode).suggestedFile;

// الخطوة 2: مراجعة الكود باستخدام أداة المراجعة

const reviewResponse = Tools.CodeReview.reviewCode({ code: functionCode, reviewType: 'تطبيق الاقتراح 3\*\*: التحقق من الذاكرة طويلة المدى أولاً

const memoryKey = `doc\_${functionName}`;

const cachedDoc = AI.LongTermMemory.load(memoryKey); // نفترض أن load يمكنها جلب مفتاح واحد

if (cachedDoc && cachedDoc.documentation) {

general' });

let reviewSummary = "فشلت عملية المراجعة التلقائية.";

if (reviewResponse.type === 'success' && reviewResponse.data) {

reviewSummary = `التقييم:Utils.log('Tools.Developer.getBuiltinFunctionDoc: Doc retrieved from cache.', { functionName });

return AI.Core.ask(`أعد صياغة هذا التوثيق بصيغة Markdown:\n${cachedDoc.documentation}`);

${reviewResponse.data.overallScore}/100. ملخص: ${reviewResponse.data.summary}`;

}

// الخطوة 3: تجهيز كل شيء في ورشة العمل

const stageResponse = \_stageCode }

// إذا لم يكن في الكاش، قم باستدعاء Gemini

const prompt = `قدم توثيقًا شاملاً لدالة Google Apps Script التالية: \`${functionName}\`. أعد الرد كنص منسق بصيغة Markdown.`;

const result = AI.Core.ask(prompt, {InWorkshop(functionCode, suggestedFile, reviewSummary);

Utils.log('Developer.reviewAndStageCode: modelOverride: Config.get('GEMINI\_FLASH\_MODEL') });

// حفظ النتيجة في الذاكرة للم Completed successfully.');

return stageResponse; // إرجاع نتيجة التجهيز النهائية

}, [], 'reviewAndStageCodeرة القادمة

if (result.type === 'info') {

AI.LongTermMemory.');

}

/\*\*

\* يجهز الكود المقترح وملخص المراجعة في ورقة "Developer Workshop".

\* @private

\* @param {string} functionCode

\* @param {string} suggestedsave('FunctionDoc', {

functionName: functionName,

documentation: result.text

});

File

\* @param {string} reviewSummary

\* @returns {UiResponse}

\*/

function \_stageCodeInWorkshop(functionCode, suggestedFile, reviewSummary) {

const { Utils, Config }

Utils.log('Tools.Developer.getBuiltinFunctionDoc: Doc fetched from API.', { functionName });

return result;

}, [], `getBuiltinFunctionDoc for ${functionName}`);

}

/\*\*

\* يصدر توثيق الأدوات إلى مستند Google Docs.

\*, UI } = GAssistant.Utils.Injector.get('Utils', 'Config', 'UI');

const sheetName = Config.get('DEVELOPER\_WORKSHOP\_SHEET');

// ✅ \*\*تطبيق الاقتراح 2 (يمكن تطويره لاحقًا لدعم الفلترة كما اقترحت).

\*/

function exportToolsDocumentationToDoc() {

const { Utils, Tools } = GAssistant.Utils.Injector.get('Utils', 'Tools');

Utils.executeSafely(() => {

const doc = DocumentApp.create(`توثيق\*\*: إضافة عمود "الفئة"

const sheet = Utils.getSheet(sheetName, ['التاريخ', 'الملف المقترح', 'ملخص المراجعة', 'الكود المقترح', 'الحالة', 'الف أدوات G-Assistant - ${new Date().toLocaleDateString()}`);

const body = doc.getBody();

body.appendParagraph('توثيق أدوات G-Assistant').setHeading(DocumentApp.ParagraphHeading.TITLE);ئة']);

if (!sheet) return UI.Dialogue.createError("فشل الوصول إلى ورشة عمل المطور.");

const newRow = [

new Date(),

suggestedFile,

reviewSummary,

const declarations = Tools.Catalog.getDeclarations();

declarations.forEach(tool => {

body.appendParagraph(tool.name).setHeading(DocumentApp.ParagraphHeading.HEADING1);

body.appendParagraph(tool.description).setItalic(true);

const paramsfunctionCode,

'للمراجعة',

\_categorizeCode(functionCode) // دالة مساعدة لتحديد الفئة

];

sheet.appendRow(newRow);

return UI = tool.parameters.properties;

if(params && Object.keys(params).length > 0) {

body.appendParagraph("المعاملات:").setHeading(DocumentApp.ParagraphHeading.HEADING2);

.Dialogue.createInfo(`✅ تم مراجعة الكود وتجهيزه في ورشة العمل: "${sheetName}".`);

}

/\*\*

\* دالة مساعدة بسيطة لتصنيف الكود بناءً على الك const table = body.appendTable([['المعامل', 'النوع', 'الوصف', 'مطلوب؟']]);

// ... (منطق بناء الجدول) ...

}

body.appendHorizontalRule();

لمات المفتاحية.

\* @private

\*/

function \_categorizeCode(code) {

const lowerCode = code.toLowerCase();

if (lowerCode.includes('accounting') || lowerCode.});

Utils.log('Tools.Developer.exportToolsDocumentationToDoc: Success', { url: doc.includes('profit')) return 'أدوات مالية';

if (lowerCode.includes('sheet') || lowerCode.includes('range')) return 'أدوات Sheets';

if (lowerCode.includes('sidebar') || lowerCode.getUrl() });

SpreadsheetApp.getUi().alert(`✅ تم تصدير التوثيق بنجاح. الرابط: ${doc.getUrl()}`);

}, [], 'exportToolsDocumentationToDoc');

}

// الدوال القديمة مثل suggestTargetFile و stageCodeInWorkshop يمكن الإبقاء عليها كأدوات مسincludes('menu')) return 'واجهة مستخدم';

if (lowerCode.includes('gemini') || lowerCode.includes('prompt')) return 'ذكاء اصطناعي';

return 'وظائف عامة';

}اعدة داخلية إذا لزم الأمر

// أو يمكن اعتبار reviewAndStageCode هو البديل الرسمي لها.

return {

reviewAndStageCode, // <-- الدالة الجديدة المتكاملة

getBuiltinFunctionDoc,

/\*\*

\* يحلل الكود ويقترح اسم الملف الأنسب لوضعه فيه.

\* (تبقى هذه الدالة متاحة للاستخدام المنفصل إذا لزم الأمر)

\*/

exportToolsDocumentationToDoc

};

})();

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// --- END OF FILE: 30\_tools/4\_tools\_developer.gs ---

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*