## **تم يا عبدالعزيز ✅ لقد راجعت كل شيء بدقة وأعدت كتابة الجدول والمعايير بالكامل بناءً على ملاحظتك، وهنا النسخة النهائية الكاملة المحدثة لمراجعتها معًا:**

## 

## **🧭 الخطة الشاملة – معايير المشروع وجدول التعديلات**

## 

### **🧠 المعايير المعتمدة للتطوير الذكي في G-Assistant**

| **الفئة** | **المعايير** |
| --- | --- |
| **⚙️ هندسة الذكاء** | **- Dispatch Flow****- Prompt Classification****- Model Suggestion****- Context Chunking****- Streaming Responses****- Chain-of-Thought** |
| **🧪 مراجعة الكود** | **- Structured JSON Output****- Patch Generation****- Gemini Integration****- Feedback Cards****- Error Isolation** |
| **🛡️ الأمان والخصوصية** | **- Secure API Key Storage****- Access Control****- Function Calling Isolation****- Vulnerability Scanning** |
| **📊 البيانات والتحليل** | **- Usage Metrics****- Analytics Dashboard****- Auto Charts****- Sheet Formula Intelligence** |
| **🧬 التشغيل الذكي** | **- Retry with Backoff****- Modular Execution****- Memory Logging****- UI Sync with Scripts** |
| **💬 واجهة المستخدم** | **- CodeReviewSidebar****- Smart Buttons****- Diff View****- Function Suggestion Cards** |

## 

### **📑 جدول التعديلات حسب الشمولية والأولوية**

| **رقم** | **الملف** | **المعايير المرتبطة** | **التعديلات** | **الحالة** |
| --- | --- | --- | --- | --- |
| **1️⃣** | **ai\_core.gs** | **Dispatch, Context, Streaming** | **إدارة السياق الذكي، اختيار النموذج، دعم البث** | **جاهز** |
| **2️⃣** | **ai\_dispatcher.gs** | **Tool Execution, Patch** | **تنفيذ الأدوات، مراجعة diff patch** | **🔜 دعم patch** |
| **3️⃣** | **helpers.gs** | **Diff, Logging, Multimodal** | **توليد patch، إدارة الصور، سجل ذكي** |  |
| **4️⃣** | **gemini\_script\_doctor.gs** | **Code Review, Refactor** | **تحليل مهيكل، اقتراحات، دعم الاختبارات والأمان** | **أساسي** |
| **5️⃣** | **tools\_code\_review.gs** | **Gemini Bridge, API Trigger** | **واجهة تحليل Sidebar، دعم اختبارات** | **+ 🔜 اختبارات** |
| **6️⃣** | **ui.gs** | **Sidebar, Menu, Docs** | **القوائم الجانبية، زر النسخ الاحتياطي، مزامنة** | **تمل** |
| **7️⃣** | **CodeReviewSidebar.html** | **Feedback UI, Patch View** | **عرض الملاحظات، تطبيق التعديلات** | **ز** |
| **8️⃣** | **ai\_config.gs** | **Prompt Handling, Settings** | **اختيار النموذج الذكي، إعدادات chunk/stream** | **محدث بالكامل** |
| **9️⃣** | **env.gs** | **Key Store, Access** | **إدارة المفاتيح والصلاحيات** | **✅ مستقر** |
| **🔟** | **analytics\_dashboard.gs** | **Metrics, Graphs** | **لوحة تحليلات للأداء والاستدعاءات** | **🔜 قيد العمل** |
| **11** | **function\_declarations.gs** | **Function Metadata** | **توصيف الأدوات للاستدعاء الآلي** | **🔜 مطلوب** |
| **12** | **chunked\_context\_manager.gs** | **Context Optimization** | **تقسيم السياق الطويل** | **🔜 مطلوب** |
| **13** | **tools\_general.gs** | **Backup, Access View** | **نسخ احتياطي، عرض النموذج، إدارة الوصول** |  |
| **14** | **tools\_security\_review.gs** | **Static Scan** | **فحص ثغرات الكود** | **🔜 مطلوب** |
| **15** | **unit\_tests\_generator.gs** | **AI Testing** | **توليد اختبارات للوظائف تلقائيًا** | **🔜 مطلوب** |
| **16** | **context\_packer.gs** | **Prompt Reduction** | **تبسيط وإعداد السياق** | **🔜 مطلوب** |
| **17** | **copilot\_sheets\_tools.gs** | **Formula AI** | **توليد صيغ وتحليلات داخل Sheets** | **🔜 مطلوب** |
| **18** | **prompt\_engineering\_guide.gs** | **Prompt Patterns** | **موجهات ذكية للمراجعة والتحسين** | **🔜 مطلوب** |

## 

### **📘 ملاحظات مهمة**

## **الملفات من (1) إلى (4) تعتبر قلب المنظومة وتُفتح باستمرار لدمج التحسينات، لذا بدأنا بها أولًا.**

## **الملفات من (11) فأعلى هي وحدات توسعة ذكية يمكن جدولتها بعد استقرار البنية الأساسية.**

## **جميع المعايير مرتبطة مباشرة بالمصادر التي راجعتها: المقالات التعليمية، مشاركة Gemini، والمستندات النصية التي وفرتها.**

## 

## **هل ترغب نبدأ تنفيذ وحدة chunked\_context\_manager.gs التالية ضمن الجدول؟ أو تفضل توسعة gemini\_script\_doctor.gs لتشمل توليد الاختبارات؟**

## **أنت تقود وأنا أنفّذ 💼👨‍💻✨**

## 

## **أنت تقود وأنا أنفّذ 💼👨‍💻✨**

## 

1. **انتقال إلى Google Gen AI SDK**
   * استبدال UrlFetchApp بـ @google/genai أو مكتبة Google Cloud Gen AI
   * تبسيط بناء الطلبات ودعم الكاش والستريمنج
2. **تفعيل التفكير (Thinking) والتحكم به**
   * إضافة إعداد [thinkingConfig.thinkingBudget](https://thinkingconfig.thinkingbudget) في طلبات الـ SDK
   * واجهة اختيار ديناميكية للميزانية في [ui.gs](https://ui.gs)
3. **دعم الوسائط المتعددة (Multimodal)**
   * أداة لرفع الصور وفواتير الإيصالات في Sidebar
   * توسيع ai\_ask لقبول مدخلات image\_url وimage\_bytes
4. **تدفق المحادثات (Streaming Responses)**
   * استخدام streamGenerateContent لعرض الردود كلمة بكلمة
   * واجهة شفافة في Sidebar تظهر التقدم
5. **سياق طويل وذاكرة الجلسة (Explicit Caching)**
   * حفظ prompts الثابتة (instructions, templates) في Cache Service
   * إعادة استخدام المحتوى المخزن لتقليل تكاليف الـ API
6. **التطوير المستقبلي**
   * توسيع تحليل الفواتير (OCR + image understanding)
   * دعم الصوت/Vid​eo multimodal
   * جداول زمنية (Scheduler) لتشغيل مهام دورية

بذلك نضمن مشروعاً:

* أنظف في الكود وأكثر موثوقية
* أذكى في التفاعل والتحليل
* أسرع وأقل تكلفة
* جاهز للتوسع المستقبلي

أعلمني أي نقطة تود البدء بها، أو إذا تريد البدء بالتنفيذ التقني على مستوى الكود!

ممت .

## **🧩 بند المقترحات الذكية – مخصّص للتوسعات المستقبلية داخل G-Assistant**

| **الرقم** | **المقترح** | **وصف الفكرة** | **الموقع المناسب** |
| --- | --- | --- | --- |
| 1️⃣ | generateUnitTestsForFunction(functionName) | توليد اختبارات تلقائية للدالة المحددة بصيغة GAS عبر Gemini | unit\_tests\_generator.gs |
| 2️⃣ | reviewSecurityRisks(scriptText) | تحليل الكود لاكتشاف ثغرات محتملة وإعطاء تنبيه أو توصية أمنية | tools\_security\_review.gs |
| 3️⃣ | chunkCodeAutomatically(scriptText) | تقسيم الأكواد الطويلة إلى أجزاء قابلة للتحليل ثم تجميع النتيجة | chunked\_context\_manager.gs |
| 4️⃣ | getPromptTemplate(taskType) | جلب قالب موجه مناسب لنوع المهمة: مراجعة، شرح، تحليل | prompt\_engineering\_guide.gs |
| 5️⃣ | generateFormulaFromNaturalText(text) | تحويل وصف بالعربية إلى صيغة SUMIFS, QUERY, وغيرها | copilot\_sheets\_tools.gs |
| 6️⃣ | summarizeContextForGemini(rawCode, history) | تلخيص محتوى السكربت وسجل العمليات لإرسال أفضل | context\_packer.gs |
| 7️⃣ | suggestFunctionCallsFromGeminiReply(reply) | تحليل رد Gemini واستخراج functionCall وتنفيذه مباشرة | function\_declarations.gs + ai\_dispatcher.gs |
| 8️⃣ | generatePatchAndApply(original, suggestion) | توليد الفرق بين الكودين وتطبيق التصحيح | helpers.gs + CodeReviewSidebar.html |
| 9️⃣ | dashboard.getUsageInsights() | واجهة تعرض عدد طلبات Gemini وحالة كل نموذج | analytics\_dashboard.gs |
| 🔟 | GeminiScriptDoctor.refactorCodeAdvanced(script) | تقسيم وتحسين الكود تلقائيًا، إعادة التسمية والتنسيق | gemini\_script\_doctor.gs |

أعدت لك مجموعة اقتراحات عملية تستفيد من تكامل Gemini في Google Sheets لتعزيز تطوير جافا سكربت داخل مشروعك. راجعها أدناه، وعندما توافق على أيٍ منها أبدأ بكتبيتها ودمجها في الكود:

| **الاقتراح** | **الفائدة** | **طريقة التكامل** | **موافقتك؟** |
| --- | --- | --- | --- |
| 1. دالة =GAS\_GENERATE(prompt) | توليد شفرات Apps Script تلقائياً من أوصاف باللغة الطبيعية | إنشاء Custom Function في copilot\_sheets\_tools.gs يستقبل نصاً ويدعو Gemini ليرجع snippet جاهز للنسخ | ☐ |
| 2. زر “اقتراح كود” في الشريط الجانبي | واجهة تفاعلية لصياغة دوال جديدة أو توسيع دوال قائمة | في ui.gs وSidebar.html تضيف زر يفتح مدخل نصي ثم يستدعي gemini\_script\_doctor.analyzeAndSuggestFix | ☐ |
| 3. اقتراح الصيغ الذكية داخل الخلايا | تحويل الوصفغة الطبيعية إلى صيغ SUMIFS/QUERY/etc بصورة فورية | Custom Function أخرى =AI\_FORMULA(desc) تستدعي generateFormulaFromNaturalText() وتُدرج الصيغة في الخلية | ☐ |
| 4. مراجعة سكربت كامل بضغطة زر | تحليل جميع ملفات المشروع واقتراح إصلاحات دفعة واحدة | قائمة “مطور” جديدة في ui.gs تنفذ tools\_code\_review.reviewAllScripts() وتعرض النتائج في Sidebar | ☐ |
| 5. تنبيهات فورية عند حفظ السكربت | يرسلك Gemini ملاحظات أمان وجودة قبل نشر التعديلات | Trigger onSave في Apps Script يستدعي reviewCodeForSidebar() ويعرض نافذة Alerts مع ملخص الأخطاء | ☐ |

بعد مراجعتك للجدول، علّم على الاقتراحات اللي تحب أدرجها أولاً، وأنا أبدأ بصياغتها ودمجها في ملفات الـ JavaScript الخاصة بالمشروع.

--- ui.gs

+++ ui.gs

@@ function onOpen() {

spreadsheet.addMenu('🛠️ أدوات المطوّر', [

- 'showCodeReviewSidebar',

+ 'showCodeReviewSidebar',

'showDeveloperToolsSidebar'

]);

}

// دالة جديدة لعرض Sidebar لأدوات المطوّر

+/\*\*

+ \* تفتح Sidebar بعنوان "أدوات المطوّر"

+ \*/

+function showDeveloperToolsSidebar() {

+ const html = HtmlService

+ .createTemplateFromFile('DeveloperToolsSidebar')

+ .evaluate()

+ .setTitle('أدوات المطوّر');

+ SpreadsheetApp.getUi().showSidebar(html);

+}

<!-- ملف: DeveloperToolsSidebar.html -->

<!DOCTYPE html>

<html>

<head>

<base target="\_top">

<style>

.btn { margin: 8px 0; padding: 6px 12px; }

</style>

</head>

<body>

<h2>🛠️ أدوات المطوّر</h2>

<p>اضغط أسفل لإجراء مسح ذكي للكود واقتراح التحسينات:</p>

<button class="btn" onclick="runInspection()">ابحث عن تحسينات</button>

<div id="results"></div>

<script>

function runInspection() {

const btn = document.querySelector('button');

btn.disabled = true;

btn.textContent = 'جارٍ المسح...';

google.script.run

.withSuccessHandler(renderResults)

.withFailureHandler(err => {

document.getElementById('results').innerText = err;

btn.disabled = false;

btn.textContent = 'ابحث عن تحسينات';

})

.GAssistant\_Tools\_Developer\_inspectProject();

}

function renderResults(items) {

const container = document.getElementById('results');

if (!items.length) {

container.innerText = 'لا توجد اقتراحات حالياً.';

} else {

items.forEach(it => {

const div = document.createElement('div');

div.innerHTML = `

<h4>${it.file}</h4>

<ul>

${it.suggestions.map(s => `<li>${s.description}</li>`).join('')}

</ul>`;

container.appendChild(div);

});

}

}

</script>

</body>

</html>

// ملف جديد: tools\_developer.gs

/\*\*

\* @module GAssistant.Tools.Developer

\* @description

\* يجمع السكربتات في المشروع، يمرّرها إلى Gemini Script Doctor،

\* ويرجع قائمة بالاقتراحات لكل ملف.

\*/

var GAssistant = GAssistant || {};

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

GAssistant.Tools.Developer = (function(DRIVE, SCRIPT, DOCTOR) {

'use strict';

/\*\*

\* يبحث في ملفّات السكربت (”.gs“) في المشروع ويحلّلها.

\* @returns {Array<{file:string,suggestions:Array<{description:string}>>}

\*/

function inspectProject() {

// 1. جلب جميع ملفات السكربت من المشروع

const files = DriveApp.getRootFolder()

.getFilesByType(MimeType.GOOGLE\_APPS\_SCRIPT)

.next().getId();

// ملاحظة: قد تحتاج API خاص لجلب ملفات السكربت. هذا مثال تمثيلي.

const projectFiles = SCRIPT.getProjectFiles();

const results = projectFiles.map(f => {

const text = SCRIPT.getFileContent(f.id);

const review = DOCTOR.analyzeAndSuggestFix(text);

const suggestions = (review.content.feedbackItems||[])

.filter(i=>i.type!=='PRAISE')

.map(i=>({ description: i.description }));

return { file: f.name, suggestions };

});

return results;

}

return {

inspectProject: inspectProject

};

})(

DriveApp, // للوصول لمجلدات الملفات

GAssistant.Scripts || {}, // واجهة وهمية لجلب ملفات المشروع

GEMINI\_SCRIPT\_DOCTOR // وحدة التحليل

);

/\*\*

\* ربط الدالة للاستدعاء من Client-side

\*/

function GAssistant\_Tools\_Developer\_inspectProject() {

return GAssistant.Tools.Developer.inspectProject();

}

1. أضفنا في ui.gs قائمة جديدة “أدوات المطوّر” تستدعي Sidebar مخصّص. صممّا DeveloperToolsSidebar.html فيه زر “ابحث عن تحسينات” ويعرض الاقتراحات.
2. أنشأنا tools\_developer.gs مع منطق inspectProject() الذي يجمع ملفات السكربت، يمرّرها لـ Gemini لتحليل الكود، ويجمع الاقتراحات.

بعد دمج هذه الملفات:

* اضغط على قائمة🛠️ “أدوات المطوّر” → “ابحث عن تحسينات” → سترى قائمة بالاقتراحات لكل ملف.
* يمكنك بعد ذلك مراجعة الوصف وتطبيق التعديلات يدويًا أو توسيع الميزة لتطبيقها آليًا.

قل لي إذا حابّب نجربها مباشرة أو نعدّل أي تفاصيل إضافية!

رائع، دعنا ندمج “أدوات المطوّر” مباشرة في مشروعك الموجود على Drive. بما أنني لا أستطيع الوصول للمجلد مباشرة، أتبع هذه الخطوات:

1. احصل على Script ID  
    – افتح أي ملف سكربت في مشروعك (Code Editor)  
    – من الشريط العلوي “ملف → إعدادات المشروع” وابحث عن “معرف المشروع” (Project ID)
2. استنساخ المشروع محليًا (اختياري)  
    – ثبّت clasp (npm install -g @google/clasp)  
    – سجل دخولك (clasp login)  
    – نفّذ clasp clone <PROJECT\_ID> لاستنساخ كل الملفات

إضافة ملفات “أدوات المطوّر”  
 أ. افتح المجلد (محلي أو عبر محرر Apps Script)  
 ب. أنشئ ملفًا جديدًا باسم **tools\_developer.gs** وألصق الكود التالي:  
  
 /\*\*

\* @module GAssistant.Tools.Developer

\* @description

\* يجمع سكربتات المشروع، يمرّرها إلى Gemini Script Doctor،

\* ويرجع قائمة بالاقتراحات لكل ملف.

\*/

var GAssistant = GAssistant || {};

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

GAssistant.Tools.Developer = (function(DRIVE, SCRIPTS, DOCTOR) {

'use strict';

function inspectProject() {

// جلب جميع الملفات من مشروع Apps Script

const files = SCRIPTS.listFiles(); // تأكد من تعريف دالة listFiles

return files.map(f => {

const text = SCRIPTS.getFileContent(f.id);

const review = DOCTOR.analyzeAndSuggestFix(text);

const suggestions = (review.content.feedbackItems||[])

.filter(i=>i.type!=='PRAISE')

.map(i=>({ description: i.description }));

return { file: f.name, suggestions };

});

}

return { inspectProject };

})(

DriveApp,

GAssistant.Scripts, // تأكد من وجود وحدة GAssistant.Scripts مسجّلة

GEMINI\_SCRIPT\_DOCTOR

);

function GAssistant\_Tools\_Developer\_inspectProject() {

return GAssistant.Tools.Developer.inspectProject();

}

ج. أنشئ ملف HTML جديد باسم **DeveloperToolsSidebar.html** وألصق:  
  
 <!DOCTYPE html>

<html>

<head><base target="\_top"><style>.btn{margin:8px 0;padding:6px 12px;}</style></head>

<body>

<h2>🛠️ أدوات المطوّر</h2>

<button class="btn" onclick="runInspection()">ابحث عن تحسينات</button>

<div id="results"></div>

<script>

function runInspection() {

const btn = document.querySelector('button');

btn.disabled = true; btn.textContent = 'جارٍ المسح...';

google.script.run

.withSuccessHandler(renderResults)

.withFailureHandler(err => {

document.getElementById('results').innerText = err;

btn.disabled = false; btn.textContent = 'ابحث عن تحسينات';

})

.GAssistant\_Tools\_Developer\_inspectProject();

}

function renderResults(items) {

const c = document.getElementById('results');

if (!items.length) return c.innerText = 'لا اقتراحات.';

items.forEach(it => {

const div = document.createElement('div');

div.innerHTML = `<h4>${it.file}</h4><ul>${

it.suggestions.map(s=>`<li>${s.description}</li>`).join('')

}</ul>`;

c.appendChild(div);

});

}

</script>

</body>

</html>

تعديل القائمة الرئيسية في **ui.gs** افتح ui.gs وأضف بند القائمة:  
  
 function onOpen() {

const ui = SpreadsheetApp.getUi();

ui.createMenu('🛠️ أدوات المطوّر')

.addItem('ابحث عن تحسينات', 'showDeveloperToolsSidebar')

.addToUi();

}

function showDeveloperToolsSidebar() {

const html = HtmlService

.createHtmlOutputFromFile('DeveloperToolsSidebar')

.setTitle('أدوات المطوّر');

SpreadsheetApp.getUi().showSidebar(html);

}

1. نشر واختبار  
    – احفظ كل التعديلات  
    – افتح جداولك واضغط على قائمة **🛠️ أدوات المطوّر → ابحث عن تحسينات** – انتظر النتائج ثم راجع الاقتراحات

بعد هذه الخطوات ستكون لديك لوحة “أدوات المطوّر” متصلة بـ Gemini لتحليل مشروعك.  
 إذا احتجت مساعدة في أي نقطة أو في تعريف وحدة GAssistant.Scripts لجلب الملفات، أعطني الهيكل الحالي لملفاتك وأساعدك بالتحديد.