/\*\*

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

\* @file ai\_context.gs - v15.7

\* @module AI\_CONTEXT

\* @description

\* وحدة مسؤولة عن جمع وإعداد كل المعلومات السياقية اللازمة للنموذج.

\* تستقبل كائن الحالة (state) وتقوم بإثرائه بالبيانات اللازمة قبل إرساله إلى Gemini.

\* الاعتمادات: ENV, AI\_MEMORY, AI\_DISPATCHER, HELPERS, AI\_TOOLS\_DIALOGUE, PROMPT\_ENGINE, AI\_CONFIG

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

\*/

var AI\_CONTEXT = (function(ENV, AI\_MEMORY, AI\_DISPATCHER, HELPERS, AI\_TOOLS\_DIALOGUE, PROMPT\_ENGINE, AI\_CONFIG) { // ✅ تم تحديث التبعيات

'use strict';

/\*\*

\* @function getUserRole

\* @description يقرأ دور المستخدم الحالي من الخلية B2 في ورقة التعليمات.

\* @returns {string} - دور المستخدم أو "مستخدم عام" كقيمة افتراضية.

\*/

function getUserRole() {

try {

const sheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName(ENV.SETTINGS.MASTER\_INSTRUCTIONS\_SHEET\_NAME); // ✅ استخدام ENV.SETTINGS

return sheet ? (String(sheet.getRange("B2").getValue() || "").trim() || "مستخدم عام") : "مستخدم عام";

} catch(e) {

HELPERS.logOperation('AI\_CONTEXT', 'getUserRole:error', { error: e.message });

return "مستخدم عام"; // قيمة افتراضية في حالة الخطأ

}

}

/\*\*

\* @function enrichContext

\* @description

\* تُثري كائن الحالة بالتعليمات السياقية (system instruction) وسجل المحادثة.

\* تستخدم PROMPT\_ENGINE لبناء الـ system prompt.

\* @param {object} state - كائن الحالة الذي سيتم إثراؤه.

\*/

function enrichContext(state) {

HELPERS.logOperation("AI\_CONTEXT", "enrichContext:start", { model: state.model, userPromptLength: state.userPrompt.length });

try {

state.role = getUserRole(); // جلب دور المستخدم

// جلب السياق الفوري (الخلية النشطة)

const immediateContext = (function() {

try {

const activeRange = SpreadsheetApp.getActiveRange();

if (!activeRange) return "";

const sheet = activeRange.getSheet();

const range = activeRange;

return `## 📝 سياق الورقة الحالي:\\n- اسم الورقة: \"${sheet.getName()}\"\\n- النطاق المحدد: ${range.getA1Notation()}\\n- القيمة الحالية للخلية: \"${range.getDisplayValue()}\"`;

} catch(e) {

HELPERS.logOperation('AI\_CONTEXT', 'immediateContext:error', { error: e.message }); // تسجيل الخطأ

return "";

}

})();

// استخدام AI\_TOOLS\_DIALOGUE.summarizeSession() لتقديم ملخص لسياق الجلسة

// بدلاً من stringify الكائن بالكامل. هذا يقلل من حجم السياق المرسل للنموذج.

// ملاحظة: summarizeSession هي دالة في AI\_TOOLS\_DIALOGUE

const sessionSummary = AI\_TOOLS\_DIALOGUE.summarizeSession().content; // يفترض أن summarizeSession ترجع كائن { type, content }

// جلب آخر 5 عمليات مسجلة من الذاكرة

const recentHistory = AI\_MEMORY.getRecentOperations(5).map(op => {

return `- [${new Date(op.timestamp).toLocaleTimeString()}] ${op.module}: ${op.action} - ${JSON.stringify(op.summary || {}).substring(0, 100)}...`;

}).join('\\n');

// ✅ بناء التعليمات النهائية للنموذج باستخدام PROMPT\_ENGINE

state.systemInstruction = PROMPT\_ENGINE.buildPrompt({

taskType: "مساعدة في جداول البيانات", // يمكن أن يكون ديناميكيًا بناءً على النية

userPrompt: state.userPrompt,

includeSheet: true, // دائمًا تضمين وصف الورقة

sheetFormat: "markdown", // يمكن تغييرها إلى "json" حسب الحاجة

style: state.role === "مطور" ? "تقني ومفصل" : "مباشر وواضح", // استنتاج أسلوب مبدئي من الدور

modelConfig: { // ✅ تمرير معلومات النموذج

name: state.model,

version: AI\_CONFIG.getConfig().GENERATION\_CONFIG.temperature // مثال على جلب معلومات إضافية

}

}) + `\n\n## 🧑‍💼 سياق المستخدم الحالي:\n- \*\*الدور:\*\* ${state.role}\n\n${immediateContext}\n\n## 🧠 ذاكرة الجلسة (بيانات مؤقتة):\n${sessionSummary}\n\n## 📜 آخر 5 عمليات مسجلة:\n${recentHistory}`;

HELPERS.logOperation("AI\_CONTEXT", "enrichContext:done", { model: state.model, role: state.role, finalSystemInstructionLength: state.systemInstruction.length });

} catch (e) {

HELPERS.logOperation("AI\_CONTEXT", "enrichContext:error", { error: e.message });

state.final = AI\_TOOLS\_DIALOGUE.showError(`فشل في بناء السياق: ${e.message}`);

state.halt = true; // إيقاف المعالجة إذا فشل بناء السياق

}

}

return {

getUserRole,

enrichContext

};

})(

GAssistant.Config, // ENV هو GAssistant.Config

GAssistant.AI.Memory,

GAssistant.AI.Dispatcher, // على الرغم من عدم استخدامه مباشرة هنا، فهو تبعية في الملف الأصلي

GAssistant.Helpers,

GAssistant.AI.Dialogue,

PROMPT\_ENGINE, // ✅ حقن PROMPT\_ENGINE

GAssistant.AI.Config // ✅ حقن GAssistant.AI.Config

);