/\*\*

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

\* @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 // ✅ تم تحديث التبعيات

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

\*/

var AI\_CONTEXT = (function(ENV, AI\_MEMORY, AI\_DISPATCHER, HELPERS, AI\_TOOLS\_DIALOGUE){ // ✅ تم تمرير AI\_TOOLS\_DIALOGUE

'use strict';

// ❌ تم إزالة const SETTINGS = ENV.SETTINGS; لتوحيد استخدام ENV.SETTINGS مباشرةً

/\*\*

\* @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";

}

}

// ❌ تم إزالة دالة getPreferredGeminiModel بالكامل من هنا.

// هذه الدالة يجب أن تكون موجودة فقط في AI\_CONFIG.gs لتجنب التضارب.

// AI\_CORE سيستدعيها من AI\_CONFIG.getPreferredGeminiModel.

/\*\*

\* @function enrichContext

\* @description

\* تدمج تعليمات النظام، دور المستخدم، السياق الفوري، الذاكرة، وسجل العمليات لإنشاء systemPrompt مركّب.

\* يتم استخدامه من قبل AI\_CORE قبل إرسال الطلب إلى Gemini.

\* @param {object} state - كائن الحالة الحالي.

\* @returns {object} - كائن الحالة المحدث مع `systemInstruction`.

\*/

function enrichContext(state) {

if (state.halt) return state;

HELPERS.logOperation("AI\_CONTEXT", "enrichContext:start", {});

try {

const ss = state.ss;

const instructionsSheet = ss.getSheetByName(ENV.SETTINGS.MASTER\_INSTRUCTIONS\_SHEET\_NAME); // ✅ استخدام ENV.SETTINGS

const systemPrompt = instructionsSheet ? instructionsSheet.getRange("A1").getValue() : "أنت مساعد ذكي وفعال.";

// تجميع سجل العمليات

let recentHistory = "لا يوجد سجل سابق للعمليات.";

const logSheet = ss.getSheetByName(ENV.SETTINGS.LOG\_SHEET\_NAME); // ✅ استخدام ENV.SETTINGS

if (logSheet && logSheet.getLastRow() > 1) {

// قراءة آخر 10 صفوف من السجل، مع التأكد من عدم تجاوز الصف 2 (بعد الرأس)

const logsRange = logSheet.getRange(Math.max(2, logSheet.getLastRow() - 9), 1, Math.min(10, logSheet.getLastRow() - 1), 3);

const logs = logsRange.getDisplayValues();

// تصفية الصفوف الفارغة وإنشاء ملخص لآخر 5 عمليات

recentHistory = logs.filter(r => r[0]).map(r => `[${r[0]}] طلب: "${r[1]}" ← إجراء: ${r[2]}`).slice(-5).join("\n");

}

// تجميع السياق الفوري من الواجهة (النطاق المحدد حالياً)

const immediateContext = (() => {

try {

const range = ss.getActiveRange();

if (!range) return "";

return `\n## سياق فوري من الواجهة:\n- الورقة النشطة: "${range.getSheet().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.

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

// بناء التعليمات النهائية للنموذج

state.systemInstruction = [

systemPrompt,

`## 🧑‍💼 سياق المستخدم الحالي:\n- \*\*الدور:\*\* ${state.role}`,

immediateContext,

`## 🧠 ذاكرة الجلسة (بيانات مؤقتة):\n${sessionSummary}`, // ✅ استخدام الملخص

`## 📜 آخر 5 عمليات مسجلة:\n${recentHistory}`

].join("\n\n");

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

} catch (e) {

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

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

state.halt = true;

}

return state;

}

// 📦 التصدير

return {

getUserRole,

enrichContext

};

})(ENV, AI\_MEMORY, AI\_DISPATCHER, HELPERS, AI\_TOOLS\_DIALOGUE); // ✅ تم تمرير AI\_TOOLS\_DIALOGUE