/\*\*

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

\* @file ai\_config.gs – v1.1

\* @module GAssistant.AI.Config

\* @description

\* يحتوي على كل الإعدادات والثوابت الخاصة بسلوك محرك الذكاء الاصطناعي.

\* يعتمد على الثوابت الأساسية من GAssistant.Config ويوفر إعدادات متقدمة للـ API.

\* يعتمد على: GAssistant.Config

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

\*/

var GAssistant = GAssistant || {};

GAssistant.AI = GAssistant.AI || {};

// حقن التبعيات: يجب تمرير الوحدات المعتمدة هنا

GAssistant.AI.Config = (function(GA\_Config) {

'use strict';

/\*\*

\* @const {object} \_CONFIG

\* @description

\* كائن الإعدادات المركزي الخاص بوحدة الذكاء الاصطناعي.

\* يتم جلب بعض القيم من GAssistant.Config لضمان التوحيد.

\*/

const \_CONFIG = {

// ✨ تطوير: تم ربطها بالمتغيرات المركزية من GAssistant.Config

ALLOWED\_MODELS: GA\_Config.getAllowedModels(), // جلب النماذج المسموح بها من GAssistant.Config

DEFAULT\_MODEL: GA\_Config.getDefaultModel(), // جلب النموذج الافتراضي من GAssistant.Config

// ✨ تطوير: النموذج السريع للأوامر البسيطة

MODEL\_FOR\_SIMPLE\_TASKS: GA\_Config.getSetting('MODEL\_FOR\_SIMPLE\_TASKS') || "gemini-1.5-flash", // جلب من GAssistant.Config أو استخدام قيمة افتراضية

// أقصى عدد للأدوات في سلسلة واحدة

MAX\_TOOL\_CHAIN\_LENGTH: 5,

// مدة الانتظار الأولية (بالمللي ثانية) قبل إعادة المحاولة

RETRY\_DELAY\_MS: 1000,

// أقصى عدد لمحاولات الاتصال بالـ API

MAX\_API\_RETRIES: 3,

// نقطة النهاية المركزية لـ API

// هذه قيمة ثابتة، لذا لا يتم جلبها من PropertiesService مباشرة هنا.

// يمكن لـ GAssistant.Config توفيرها إذا كانت قابلة للتكوين.

API\_ENDPOINT: "https://generativelanguage.googleapis.com/v1beta/models/",

// إعدادات توليد النموذج (يمكن تجاوزها هنا إذا لزم الأمر)

GENERATION\_CONFIG: GA\_Config.getGenerationConfig(),

// إعدادات الأمان للنموذج (يمكن تجاوزها هنا إذا لزم الأمر)

SAFETY\_SETTINGS: GA\_Config.getSafetySettings(),

// إعدادات التفكير (Thinking) للنموذج (يمكن تجاوزها هنا إذا لزم الأمر)

THINKING\_CONFIG: GA\_Config.getThinkingConfig(),

};

/\*\*

\* @function \_getApiKey

\* @description

\* يجلب مفتاح API لخدمة Gemini من مخزن الأسرار الآمن عبر GAssistant.Config.

\* @returns {string|null} - مفتاح API أو null إذا لم يكن موجودًا.

\*/

function \_getApiKey() {

return GA\_Config.getSecret(GA\_Config.getSetting('API\_KEYS\_NAMES').GEMINI);

}

/\*\*

\* @function getConfig

\* @description

\* ترجع نسخة عميقة من كائن الإعدادات لمنع التعديل المباشر.

\* @returns {object} - كائن الإعدادات.

\*/

function getConfig() {

// استخدام JSON.stringify لإنشاء نسخة عميقة وآمنة

return JSON.parse(JSON.stringify(\_CONFIG));

}

/\*\*

\* @function getModelForTask

\* @description

\* يحدد النموذج المناسب للمهمة بناءً على نوع المهمة (بسيطة أو معقدة).

\* @param {boolean} isSimpleTask - صحيح إذا كانت المهمة بسيطة، خطأ إذا كانت معقدة.

\* @returns {string} - اسم النموذج المقترح.

\*/

function getModelForTask(isSimpleTask) {

return isSimpleTask ? \_CONFIG.MODEL\_FOR\_SIMPLE\_TASKS : \_CONFIG.DEFAULT\_MODEL;

}

/\*\*

\* @function getPreferredGeminiModel

\* @description

\* 🧠 تحليل محتوى prompt وتحديد النموذج الأنسب بناءً على طبيعته.

\* @param {string} prompt - نص الطلب من المستخدم.

\* @returns {object} - يحتوي على: model, complexity, type.

\*/

function getPreferredGeminiModel(prompt) {

if (!prompt || typeof prompt !== 'string') {

return { model: \_CONFIG.DEFAULT\_MODEL, complexity: 'low', type: 'chat' };

}

const lowerPrompt = prompt.toLowerCase();

// ⚙️ تصنيف نوع المهمة

const isVision = /(image|visual|vision|picture|photo|analyze image)/.test(lowerPrompt);

const isCode = /(function|loop|variable|class|script|code|gs|javascript|refactor)/.test(lowerPrompt);

const isChat = /(help|question|conversation|assist|explain|idea|feedback)/.test(lowerPrompt);

// 👁️ نموذج للصور

if (isVision) {

return {

model: 'gemini-1.5-pro-vision',

complexity: 'high',

type: 'vision'

};

}

// 💻 نموذج للكود البرمجي

if (isCode) {

return {

model: 'gemini-1.5-pro',

complexity: 'medium',

type: 'code'

};

}

// 💬 نموذج للمحادثات العامة

if (isChat) {

return {

model: \_CONFIG.DEFAULT\_MODEL, // استخدام \_CONFIG.DEFAULT\_MODEL

complexity: 'low',

type: 'chat'

};

}

// 🧩 في حال لم يتم التعرف على النمط

return {

model: \_CONFIG.MODEL\_FOR\_SIMPLE\_TASKS, // استخدام \_CONFIG.MODEL\_FOR\_SIMPLE\_TASKS

complexity: 'low',

type: 'generic'

};

}

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

return {

getConfig,

getApiKey: \_getApiKey, // تصدير دالة جلب المفتاح السري

getModelForTask,

getPreferredGeminiModel, // تصدير الدالة الجديدة

// يمكن تصدير إعدادات محددة مباشرة إذا لزم الأمر

getApiEndpoint: () => \_CONFIG.API\_ENDPOINT,

getGenerationConfig: () => \_CONFIG.GENERATION\_CONFIG,

getSafetySettings: () => \_CONFIG.SAFETY\_SETTINGS,

getThinkingConfig: () => \_CONFIG.THINKING\_CONFIG,

getMaxToolChainLength: () => \_CONFIG.MAX\_TOOL\_CHAIN\_LENGTH,

getRetryDelayMs: () => \_CONFIG.RETRY\_DELAY\_MS,

getMaxApiRetries: () => \_CONFIG.MAX\_API\_RETRIES,

};

})(

GAssistant.Config // حقن GAssistant.Config كوحدة تابعة

);