

المحور الثالث: الهيكلية والمكدس التقني

Platform Architecture and Technology Stack

المؤلف: Manus AI **التاريخ:** 3 نوفمبر 2025 **الهدف:** تحديد المكدس التقني (Technology Stack) الأمثل الذي يوازن بين سرعة التطوير، وقابلية التوسع، والكفاءة المطلوبة لمعالجة الذكاء الاصطناعي الثقيلة (Heavy AI Processing).

1. الهيكلية العامة (General Architecture)

تعتمد الهيكلية المقترحة على مبدأ **فصل الاهتمامات (Separation of Concerns)**، حيث يتم فصل الواجهة الأمامية (Frontend) عن الواجهة الخلفية (Backend)، ويتم التعامل مع مهام الذكاء الاصطناعي الثقيلة بشكل غير متزامن (Asynchronously) عبر نظام قائمة انتظار.

أ. المكندس التقني الموصى به

المكون	التوصية التقنية	التبرير الاستراتيجي والتقني
الواجهة الأمامية (Frontend)	React/Next.js	يوفر مجتمعاً ضخماً، أدوات قوية، وقابلية ممتازة للتوسع. Next.js يضيف ميزات مثل العرض من جانب الخادم (SSR) وتحسين الأداء.
الواجهة الخلفية (Backend)	Python مع FastAPI	Python هو الخيار الأمثل لمعالجة الذكاء الاصطناعي (AI/ML) نظراً لتوفر مكتبات مثل TensorFlow, FastAPI و PyTorch يوفر أداءً عالياً جداً (أسرع من Flask و Django) وهو مثالي لخدمة استدعاءات الـ API الثقيلة لمعالجة الصور.
قواعد البيانات (Database)	PostgreSQL (SQL)	موثوقية عالية، تكامل بيانات، وقدرات متقدمة مثل تخزين JSONB لإدارة بيانات الصور والمفاتيح (API Keys) بشكل مرن.
تخزين الملفات (File Storage)	Amazon S3 أو Google Cloud Storage	ضروري لتخزين الصور الأصلية والصور المُعالجة. يوفر قابلية توسع لا نهائية وتكلفة فعالة.
إدارة المهام (Task Queue)	Celery مع Redis	حاسم لمعالجة الصور بشكل غير متزامن (Asynchronous). يضمن أن المستخدم لا ينتظر نتيجة المعالجة، بل يتم إخطاره عند الانتهاء.

2. إدارة المهام غير المتزامنة (Asynchronous Task Management)

تعتبر معالجة الصور بالذكاء الاصطناعي عملية تستغرق وقتاً طويلاً (قد تصل إلى عدة دقائق). لذلك، فإن استخدام نظام قائمة انتظار غير متزامن هو أمر **حاسم** لضمان تجربة مستخدم سلسة وموثوقية عالية.

أ. آلية العمل المقترحة (Celery/Redis)

- الطلب (Request): يرفع المستخدم الصورة عبر الواجهة الأمامية.
- الواجهة الخلفية (FastAPI): تستقبل الطلب، وتتحقق من صلاحية المستخدم، ثم ترسل مهمة المعالجة إلى قائمة انتظار Celery.
- الاستجابة الفورية (Immediate Response): ترسل FastAPI استجابة فورية للمستخدم (HTTP 202 Accepted) مع رقم تعريف المهمة (Job ID).
- المعالجة (Processing): تقوم وحدات العمل (Celery Workers) بتنفيذ مهمة الذكاء الاصطناعي على خوادم الـ GPU.

5. **الإخطار (Notification):** عند الانتهاء، يتم تخزين الصورة الناتجة في S3، ويتم تحديث حالة المهمة في قاعدة البيانات، وإرسال إخطار للمستخدم (عبر WebSockets أو Polling) بأن الصورة جاهزة للتحميل.

ب. الفوائد التقنية

- تجربة مستخدم محسنة:** لا يضطر المستخدم للانتظار على صفحة واحدة.
- قابلية التوسع:** يمكن إضافة المزيد من وحدات العمل (Celery Workers) بسهولة لتلبية الطلب المتزايد (Auto-Scaling).
- الموثوقية:** يضمن Celery أن المهام التي تفشل يمكن إعادة محاولتها تلقائياً (Automatic Retry) دون تدخل يدوي.

3. متطلبات البنية التحتية للذكاء الاصطناعي (AI Infrastructure Requirements)

يتطلب تشغيل نماذج الانتشار (Diffusion Models) عالية الدقة (مثل SDXL) وحدات معالجة رسومات (GPU) متخصصة.

المتطلب	التوصية	التبرير
وحدات المعالجة	NVIDIA A100 أو H100	هذه الوحدات هي المعيار الصناعي لعمليات الاستدلال (Inference) والتدريب الدقيق (Fine-Tuning) لنماذج الذكاء الاصطناعي الكبيرة، وتوفر أداءً عالياً مقابل التكلفة.
التوسع	التوسع التلقائي (Auto-Scaling)	يجب إعداد نظام التوسع التلقائي (Kubernetes أو ما يعادله) لزيادة أو تقليل عدد وحدات الـ GPU العاملة بناءً على حجم قائمة انتظار Celery، لضمان الكفاءة في التكلفة والسرعة في الخدمة.
البيئة	Docker/Containerization	استخدام حاويات Docker لضمان بيئة تشغيل متسقة وموثوقة لجميع مكونات الذكاء الاصطناعي، مما يسهل عملية النشر (Deployment) والإدارة.