منصة OCP للصيانة بالذكاء الاصطناعي — التوثيق التقني

الإصدار: 1.0.0
آخر تحديث: فبراير 2026
المنصة: تطبيق ويب متكامل في حاويات
العميل: مجموعة OCP (المكتب الشريف للفوسفاط)

فهرس المحتويات

الرقم	القسم
1	نظرة عامة
2	هندسة النظام
3	المكدس التقني
4	هيكل المشروع
5	الواجهة الخلفية FastAPI
6	الواجهة الأمامية React + Vite
7	مخطط قاعدة البيانات
8	المصادقة والتفويض
9	مرجع واجهة API
10	التدويل i18n
11	تكامل SAP
12	والنشر Docker
13	الوحدات الوظيفية
14	سير العمل الرئيسية
15	الأمان
16	دليل التطوير المحلي
17	قائمة مراجعة الإنتاج
18	متغيرات البيئة

---

1. نظرة عامة

الوصف

للصيانة بالذكاء الاصطناعي هي نظام مؤسسي متكامل لإدارة الصيانة، يتألف من **أربع وحدات** مدعومة بالذكاء الاصطناعي. تم OCP منصة تصميم النظام خصيصاً لمجموعة OCP — أكبر مُصدّر للفوسفاط في العالم — للاستخدام في مجمّع الجرف الأصفر الصناعي (JFC1).

الأهداف الرئيسية

الهدف	الوصف
الصيانة التنبؤية	ذكاء اصطناعي للتنبؤ بالأعطال + Weibull تحليل
تحسين RCM	الصيانة المرتكزة على الموثوقية بقرارات آلية
تكامل SAP	تصدير حزم أوامر العمل إلى SAP PM
تعدد اللغات	دعم ثلاث لغات: الإنجليزية (EN)، الإسبانية (ES)، العربية (AR)
التحكم بالوصول	خمسة أدوار مبنية على الصلاحيات
التحليلات الآنية	لوحات معلومات ومؤشرات أداء في الوقت الفعلي

الوحدات الأربع

الوحدة	الاسم	المحتوى
1	التسلسل الهرمي للأصول والحرجية	إدارة هرمية المعدات وتقييم الحرجية
2	RCM واستراتيجية FMEA/FMECA	تحليل أنماط الأعطال والآثار وتحديد استراتيجيات الصيانة
3	العمليات والتخطيط	طلبات العمل، حزم العمل، الجدولة، التقاط البيانات الميدانية
4	التحليلات والتقارير والتحسين المستمر	لوحات المعلومات، التقارير، الموثوقية، تحليل السبب الجذري

2. هندسة النظام

مخطط الهندسة المعمارية

```
┌──────────────────────────────────────────────────────────────┐
│                    المتصفح (العميل)                          │
│                React 19 SPA + Tailwind CSS                   │
└──────────────────────────────────────────────────────────────┘
                          │
                          │ المنفذ 8080
                          ▼
┌──────────────────────────────────────────────────────────────┐
│                  Nginx (وكيل عكسي)                           │
│             8080 المنفذ (HTTP) / 8443 (HTTPS)                │
│     /api/*  →  FastAPI:8000    |    /*  →  React:5173        │
└──────────────────────────────────────────────────────────────┘
        │                                │
        ▼                                ▼
┌──────────────────┐        ┌──────────────────────┐
│  الواجهة الأمامية  |        |  الواجهة الخلفية      |
│  FastAPI + Gunicorn |        |   Vite + React 19     |
│   5173 المنفذ     |        |   8000 المنفذ         |
└──────────────────┘        └──────────────────────┘
          │
          ▼
┌──────────────────┐
│   قاعدة البيانات  |
│  SQLite (تطوير)   |
│  PostgreSQL (إنتاج) |
└──────────────────┘
```

منافذ الخدمات

الخدمة	المنفذ	الوصف
الواجهة الأمامية (Vite)	`5173`	React خادم تطوير
الواجهة الخلفية (FastAPI)	`8000`	REST واجهة برمجة التطبيقات
Nginx (HTTP)	`8080`	الوكيل العكسي
Nginx (HTTPS)	`8443`	SSL مشفّر اتصال

مسار الطلبات

```
المتصفح → Nginx:8080 → /api/* يُوجَّه إلى → FastAPI:8000
                     → /*     يُوجَّه إلى → React:5173
```

3. المكدس التقني

الواجهة الخلفية

التقنية	الإصدار	الغرض
Python	3.11	لغة البرمجة الأساسية
FastAPI	0.133.1	إطار عمل الواجهة البرمجية
Uvicorn	0.41.0	خادم ASGI للتطوير
Gunicorn	25.1.0	خادم الإنتاج
SQLAlchemy	2.0.47	مُعيّن الكائنات العلائقية (ORM)
Pydantic	2.12.5	التحقق من البيانات والمخططات
python-jose	3.3.0	معالجة رموز JWT
bcrypt	4.2.1	تشفير كلمات المرور
Anthropic SDK	0.83.0	تكامل الذكاء الاصطناعي (Claude)
openpyxl	3.1.5	تصدير ملفات Excel
httpx	0.28.1	عميل HTTP غير متزامن

الواجهة الأمامية

التقنية	الإصدار	الغرض
React	19.1.0	مكتبة واجهة المستخدم
Vite	7.3.1	أداة البناء والتجميع
Tailwind CSS	4.2.1	إطار CSS المرن
React Router DOM	7.13.1	التوجيه والملاحة
Radix UI	متعدد	مكونات واجهة المستخدم الأساسية
Recharts	3.7.0	مكتبة الرسوم البيانية
Lucide React	0.575.0	مكتبة الأيقونات
XLSX	0.18.5	معالجة ملفات Excel
file-saver	2.0.5	تحميل الملفات من المتصفح

البنية التحتية

التقنية	الغرض
Docker	بناء الحاويات
Docker Compose	تنسيق الخدمات المتعددة
Nginx Alpine	الوكيل العكسي وخادم الملفات الثابتة
SQLite	قاعدة بيانات التطوير
PostgreSQL	قاعدة بيانات الإنتاج

4. هيكل المشروع

```
ASSET-MANAGEMENT-SOFTWARE-master/
├── api/                              # الواجهة الخلفية (FastAPI)
│   ├── main.py                       # نقطة الدخول وإعداد التطبيق
│   ├── config.py                     # إعدادات التطبيق (متغيرات البيئة)
│   ├── schemas.py                    # مخططات Pydantic (أكثر من 50 مخططاً)
│   ├── seed.py                       # ملء قاعدة البيانات ببيانات أولية
│   ├── database/
│   │   ├── connection.py             # اتصال SQLAlchemy وجلسات العمل
│   │   └── models.py                 # نماذج ORM (أكثر من 25 نموذجاً)
│   ├── routers/                      # نقاط نهاية API (18 موجّه)
│   │   ├── auth.py                   # المصادقة وإدارة المستخدمين
│   │   ├── hierarchy.py              # التسلسل الهرمي للأصول
│   │   ├── criticality.py            # تقييمات الحرجية
│   │   ├── fmea.py                   # FMEA/FMECA وقرارات RCM
│   │   ├── tasks.py                  # مهام الصيانة
│   │   ├── work_packages.py          # حزم العمل
│   │   ├── sap.py                    # تكامل SAP
│   │   ├── analytics.py              # التحليلات والمؤشرات
│   │   ├── work_requests.py          # طلبات العمل
│   │   ├── capture.py                # التقاط البيانات الميدانية
│   │   ├── planner.py                # توصيات المُخطّط
│   │   ├── backlog.py                # قائمة الأعمال المتراكمة
│   │   ├── scheduling.py             # الجدولة الأسبوعية
│   │   ├── reliability.py            # هندسة الموثوقية
│   │   ├── reporting.py              # التقارير والتصدير
│   │   ├── dashboard.py              # لوحة المعلومات التنفيذية
│   │   ├── rca.py                    # تحليل السبب الجذري
│   │   └── admin.py                  # إدارة النظام
│   ├── services/                     # طبقة منطق الأعمال (23 خدمة)
│   │   ├── auth_service.py           # خدمة المصادقة و JWT
│   │   ├── hierarchy_service.py      # خدمة التسلسل الهرمي
│   │   ├── hierarchy_builder_service.py  # بناء الهرمية من بيانات المصنّع
│   │   ├── criticality_service.py    # خدمة تقييم الحرجية
│   │   ├── fmea_service.py           # خدمة FMEA/FMECA
│   │   ├── task_service.py           # خدمة المهام
│   │   ├── work_package_service.py   # خدمة حزم العمل
│   │   ├── sap_service.py            # خدمة تكامل SAP
│   │   ├── analytics_service.py      # خدمة التحليلات
│   │   ├── work_request_service.py   # خدمة طلبات العمل
│   │   ├── capture_service.py        # خدمة التقاط البيانات
│   │   ├── planner_service.py        # خدمة المُخطّط
│   │   ├── backlog_service.py        # خدمة الأعمال المتراكمة
│   │   ├── scheduling_service.py     # خدمة الجدولة
│   │   ├── reliability_service.py    # خدمة الموثوقية
│   │   ├── reporting_service.py      # خدمة التقارير
│   │   ├── rca_service.py            # خدمة تحليل السبب الجذري
│   │   ├── agent_service.py          # خدمة الذكاء الاصطناعي
│   │   ├── capa_service.py           # خدمة الإجراءات التصحيحية والوقائية
│   │   ├── audit_service.py          # خدمة سجل التدقيق
│   │   └── validation_service.py     # خدمة التحقق من الصحة
│   └── dependencies/
│       └── auth.py                   # حراس المصادقة والأدوار
├── frontend/                         # الواجهة الأمامية (React + Vite)
│   ├── package.json                  # الحزم والاعتماديات
│   ├── vite.config.js                # إعدادات Vite
│   ├── postcss.config.mjs            # إعدادات PostCSS
│   ├── Dockerfile                    # بناء حاوية الواجهة الأمامية
│   ├── public/
│   │   └── OCP_LOGO.png              # شعار OCP
│   └── src/
│       ├── main.jsx                  # نقطة الدخول والتوجيه
│       ├── App.jsx                   # المكوّن الجذري للتطبيق
│       ├── api.js                    # عميل API (أكثر من 50 دالة)
│       ├── pages/                    # صفحات التطبيق (23 صفحة)
│       │   ├── Login.jsx             # صفحة تسجيل الدخول
│       │   ├── Dashboard.jsx         # لوحة المعلومات الرئيسية
│       │   ├── ExecutiveDashboard.jsx # لوحة المعلومات التنفيذية
│       │   ├── Hierarchy.jsx         # التسلسل الهرمي للأصول
```

```
│       │   ├── Criticality.jsx       # تقييم الحرجية
│       │   ├── FMEA.jsx              # تحليل أنماط الأعطال
│       │   ├── FMECA.jsx             # ورقة عمل FMECA
│       │   ├── Strategy.jsx          # استراتيجية الصيانة
│       │   ├── WorkRequests.jsx      # طلبات العمل
│       │   ├── WorkPackages.jsx      # حزم العمل
│       │   ├── Backlog.jsx           # الأعمال المتراكمة
│       │   ├── Scheduling.jsx        # الجدولة
│       │   ├── Planning.jsx          # التخطيط
│       │   ├── Planner.jsx           # المُخطِّط
│       │   ├── FieldCapture.jsx      # التقاط البيانات الميدانية
│       │   ├── Analytics.jsx         # التحليلات
│       │   ├── Reports.jsx           # التقارير
│       │   ├── Reliability.jsx       # الموثوقية
│       │   ├── RCA.jsx               # تحليل السبب الجذري
│       │   ├── DefectElimination.jsx  # إزالة العيوب
│       │   ├── SAPReview.jsx         # مراجعة SAP
│       │   ├── Admin.jsx             # إدارة النظام
│       │   └── Profile.jsx           # الملف الشخصي
│       ├── components/               # مكونات مشتركة
│       │   ├── Sidebar.jsx           # الشريط الجانبي
│       │   ├── Header.jsx            # شريط العنوان
│       │   ├── ErrorBoundary.jsx     # حدود الخطأ
│       │   ├── ConfirmDialog.jsx     # مربع حوار التأكيد
│       │   ├── Toast.jsx             # إشعارات منبثقة
│       │   ├── Shared.jsx            # مكونات مشتركة
│       │   ├── ProtectedRoute.jsx    # حارس المسارات المحمية
│       │   ├── UpdateBanner.jsx      # شريط إشعار التحديثات
│       │   └── ui/                   # مكونات Radix UI مخصصة
│       │       ├── accordion.jsx
│       │       ├── alert-dialog.jsx
│       │       ├── badge.jsx
│       │       ├── button.jsx
│       │       ├── card.jsx
│       │       ├── checkbox.jsx
│       │       ├── dialog.jsx
│       │       ├── input.jsx
│       │       ├── label.jsx
│       │       ├── progress.jsx
│       │       ├── scroll-area.jsx
│       │       ├── select.jsx
│       │       ├── separator.jsx
│       │       ├── skeleton.jsx
│       │       ├── table.jsx
│       │       ├── tabs.jsx
│       │       ├── textarea.jsx
│       │       ├── tooltip.jsx
│       │       └── utils.js
│       ├── contexts/                 # سياقات React
│       │   ├── AuthContext.jsx       # سياق المصادقة (JWT)
│       │   └── LanguageContext.jsx   # سياق اللغة (i18n)
│       ├── i18n/                     # ملفات الترجمة
│       │   ├── en.js                 # الإنجليزية
│       │   ├── es.js                 # الإسبانية
│       │   └── ar.js                 # العربية
│       ├── data/
│       │   └── mockData.js           # بيانات وهمية للتطوير
│       ├── utils/
│       │   └── exportFile.js         # أداة تصدير الملفات
│       └── styles/                   # أنماط CSS مخصصة
├── sap_mock/                         # بيانات SAP الوهمية
│   └── data/
│       ├── work_orders.json          # أوامر العمل
│       ├── functional_locations.json  # المواقع الوظيفية
│       ├── materials_bom.json        # قائمة المواد
│       ├── equipment_master.json     # البيانات الرئيسية للمعدات
│       └── maintenance_plans.json    # خطط الصيانة
├── Dockerfile                        # بناء حاوية الواجهة الخلفية
├── docker-compose.yml                # تنسيق الخدمات
├── nginx.conf                        # إعدادات الوكيل العكسي
```

```
├── requirements.txt                # اعتماديات Python
└── start.sh                        # سكربت بدء التشغيل
```

---

5. الواجهة الخلفية FastAPI

تهيئة التطبيق

يتم إنشاء تطبيق FastAPI عبر دالة `create_app()` في ملف `api/main.py`:

```
def create_app() -> FastAPI:
    app = FastAPI(
        title=settings.PROJECT_NAME,
        version="1.0.0",
        description="OCP Maintenance AI MVP — 4-module maintenance strategy platform",
        lifespan=lifespan,
    )
```

خطوات التهيئة

1. **إنشاء نسخة FastAPI** مع إعدادات المشروع
2. **إضافة وسيط CORS** للسماح بالطلبات من نطاقات محددة
3. **إضافة وسيط مفتاح API** لحماية طلبات التعديل
4. **تسجيل 18 موجّهاً** تحت البادئة `/api/v1`
5. **إنشاء جداول قاعدة البيانات** عند بدء التشغيل
6. **نقطة نهاية** `/health` لمراقبة صحة النظام

الموجّهات (18 موجّهاً)

الموجّه	البادئة	الوسم	الوصف
`auth`	`/api/v1/auth`	`auth`	المصادقة وإدارة المستخدمين
`hierarchy`	`/api/v1/hierarchy`	`hierarchy`	التسلسل الهرمي للأصول
`criticality`	`/api/v1/criticality`	`criticality`	تقييمات الحرجية
`fmea`	`/api/v1/fmea`	`fmea`	RCM وقرارات FMEA/FMECA
`tasks`	`/api/v1/tasks`	`tasks`	مهام الصيانة
`work_packages`	`/api/v1/work-packages`	`work-packages`	حزم العمل
`sap`	`/api/v1/sap`	`sap`	تكامل SAP
`analytics`	`/api/v1/analytics`	`analytics`	التحليلات والمؤشرات
`work_requests`	`/api/v1/work-requests`	`work-requests`	طلبات العمل
`capture`	`/api/v1/capture`	`capture`	التقاط البيانات الميدانية
`planner`	`/api/v1/planner`	`planner`	توصيات المُخطّط
`backlog`	`/api/v1/backlog`	`backlog`	قائمة الأعمال المتراكمة
`scheduling`	`/api/v1/scheduling`	`scheduling`	الجدولة الأسبوعية
`reliability`	`/api/v1/reliability`	`reliability`	هندسة الموثوقية
`reporting`	`/api/v1/reporting`	`reporting`	التقارير والتصدير
`dashboard`	`/api/v1/dashboard`	`dashboard`	لوحة المعلومات التنفيذية
`rca`	`/api/v1/rca`	`rca`	تحليل السبب الجذري
`admin`	`/api/v1/admin`	`admin`	إدارة النظام

طبقة الخدمات

طبقة الخدمات تفصل منطق الأعمال عن الموجّهات. تتواصل الخدمات مع قاعدة البيانات عبر SQLAlchemy ORM.

الخدمة	الملف	الوصف
خدمة المصادقة	`auth_service.py`	إدارة المستخدمين، JWT، تسجيل الدخول
خدمة FMEA	`fmea_service.py`	RCM، FMECA أنماط الأعطال، قرارات
خدمة التحليلات	`analytics_service.py`	Weibull، المؤشرات، صحة الأصول
خدمة الحرجية	`criticality_service.py`	تقييمات مصفوفة المخاطر
خدمة حزم العمل	`work_package_service.py`	إنشاء وإدارة حزم العمل
خدمة التقارير	`reporting_service.py`	التقارير الأسبوعية والشهرية والربعية
خدمة تحليل السبب الجذري	`rca_service.py`	5 تحليلW2H والسبب والأثر
خدمة التقاط البيانات	`capture_service.py`	معالجة البيانات الميدانية
خدمة الذكاء الاصطناعي	`agent_service.py`	تكامل Anthropic Claude
خدمة التسلسل الهرمي	`hierarchy_service.py`	عمليات شجرة الأصول
خدمة بناء الهرمية	`hierarchy_builder_service.py`	بناء من بيانات المصنّع
خدمة المهام	`task_service.py`	إدارة مهام الصيانة
خدمة SAP	`sap_service.py`	تصدير البيانات إلى SAP
خدمة طلبات العمل	`work_request_service.py`	دورة حياة طلبات العمل
خدمة المُخطّط	`planner_service.py`	توصيات التخطيط
خدمة الأعمال المتراكمة	`backlog_service.py`	إدارة قائمة الانتظار
خدمة الجدولة	`scheduling_service.py`	البرنامج الأسبوعي
خدمة الموثوقية	`reliability_service.py`	التحليل المتقدم
خدمة CAPA	`capa_service.py`	الإجراءات التصحيحية والوقائية
خدمة التدقيق	`audit_service.py`	سجل العمليات
خدمة التحقق	`validation_service.py`	التحقق من صحة البيانات

الإعدادات

يتم تحميل الإعدادات من متغيرات البيئة عبر الملف `api/config.py` :

```
class Settings:
    DATABASE_URL: str = os.getenv("DATABASE_URL", "sqlite:///./ocp_maintenance.db")
    SAP_MOCK_DIR: str = os.getenv("SAP_MOCK_DIR", "sap_mock/data")
    ANTHROPIC_API_KEY: str = os.getenv("ANTHROPIC_API_KEY", "")
    API_V1_PREFIX: str = "/api/v1"
    PROJECT_NAME: str = "OCP Maintenance AI MVP"
    DEBUG: bool = os.getenv("DEBUG", "false").lower() == "true"
    CORS_ORIGINS: str = os.getenv("CORS_ORIGINS", "http://localhost:5173,...")
    JWT_SECRET_KEY: str = os.getenv("JWT_SECRET_KEY", "...")
    JWT_ALGORITHM: str = os.getenv("JWT_ALGORITHM", "HS256")
    JWT_ACCESS_TOKEN_EXPIRE_MINUTES: int = 240  # 4 ساعات
    JWT_REFRESH_TOKEN_EXPIRE_DAYS: int = 7       # 7 أيام
```

6. الواجهة الأمامية React + Vite

المسارات (23 مساراً)

يتم تعريف جميع المسارات في `frontend/src/main.jsx` مع حراس الوصول المبنية على الأدوار:

المسار	المكوّن	الأدوار المسموحة	الوصف
`/login`	`Login`	عام	صفحة تسجيل الدخول
`/`	`Dashboard`	الكل	لوحة المعلومات الرئيسية
`/executive-dashboard`	`ExecutiveDashboard`	admin, manager	لوحة المعلومات التنفيذية
`/hierarchy`	`Hierarchy`	admin, manager, engineer	التسلسل الهرمي
`/criticality`	`Criticality`	admin, manager, engineer	تقييم الحرجية
`/fmea`	`FMEA`	admin, planner, engineer	تحليل أنماط الأعطال
`/fmeca`	`FMECA`	admin, planner, engineer	ورقة عمل FMECA
`/strategy`	`Strategy`	admin, planner, engineer	استراتيجية الصيانة
`/work-requests`	`WorkRequests`	الكل	طلبات العمل
`/work-packages`	`WorkPackages`	admin, planner	حزم العمل
`/backlog`	`Backlog`	admin, planner	الأعمال المتراكمة
`/scheduling`	`Scheduling`	admin, planner	الجدولة
`/planning`	`Planning`	admin, planner	التخطيط
`/planner`	`Planner`	admin, planner	المُخطّط
`/field-capture`	`FieldCapture`	admin, tecnico	التقاط الميداني
`/analytics`	`Analytics`	admin, manager, engineer	التحليلات
`/reports`	`Reports`	admin, manager	التقارير
`/reliability`	`Reliability`	admin, engineer	الموثوقية
`/rca`	`RCA`	admin, engineer	تحليل السبب الجذري
`/defect-elimination`	`DefectElimination`	admin, engineer	إزالة العيوب
`/sap-review`	`SAPReview`	admin, manager	مراجعة SAP
`/admin`	`Admin`	admin	إدارة النظام
`/profile`	`Profile`	الكل	الملف الشخصي

التخطيط العام

```
┌──────────────────────────────────────────────────────────────────────┐
│ UpdateBanner (شريط إشعار التحديثات)                                  │
├──────────┬──────────────────────────────────────────────────────────┤
│          │  Header (شريط العنوان)                                    │
│          │  ┌──────────────────────────────────────────────┐        │
│ Sidebar  │  │ تبديل السمة | اختيار اللغة | اختيار المصنع   |        |
│ الشريط   │  └──────────────────────────────────────────────┘        │
│ الجانبي  │                                                           │
│          │  ┌──────────────────────────────────────────────┐        │
│ اللون:   │  │                                              |        |
│ #1B5E20  │  |          منطقة المحتوى (Outlet)              |        |
│ أخضر     |  |                                              |        |
│ OCP      |  |         ErrorBoundary يحيط بالمحتوى         |        |
│          |  |                                              |        |
│ شعار     |  └──────────────────────────────────────────────┘        │
│ OCP      |                                                           |
└──────────┴──────────────────────────────────────────────────────────┘
```

إدارة الحالة

السياق	الملف	الوظيفة
`AuthContext`	`contexts/AuthContext.jsx`	معلومات المستخدم، تسجيل الدخول/الخروج، JWT إدارة جلسة
`LanguageContext`	`contexts/LanguageContext.jsx`	تبديل اللغة، دعم RTL ، `t()` التدويل عبر دالة

عميل API

ملف `frontend/src/api.js` يحتوي على أكثر من **50 دالة** للتواصل مع الواجهة الخلفية:

- **إضافة رمز Bearer تلقائياً** من `AuthContext`
- **التعامل مع خطأ 401** تلقائياً: محاولة تحديث الرمز ثم إعادة التوجيه إلى `/login`
- **عنوان URL أساسي قابل للتكوين** عبر متغير `VITE_API_BASE_URL`

```
// مثال على دالة API
export async function listPlants() {
    const res = await fetch(`${BASE}/api/v1/hierarchy/plants`, { headers: authHeaders() });
    return res.json();
}
```

مكونات واجهة المستخدم

لتوفير **Tailwind CSS** مع **Radix UI** مبنية على:

- مكونات قابلة للوصول (Accessible) بشكل افتراضي
- تخصيص كامل عبر Tailwind
- 19 مكوّن UI في مجلد `components/ui/`

السمات

الوصف	الوضع
خلفية فاتحة مع نصوص داكنة	وضع النهار
خلفية داكنة مع نصوص فاتحة	وضع الليل
أخضر OCP `#1B5E20`	اللون الرئيسي

---

7. مخطط قاعدة البيانات

نظرة عامة

في بيئة PostgreSQL في بيئة التطوير و SQLite قاعدة البيانات تحتوي على أكثر من **25 جدولاً** تغطي جميع وحدات النظام. يتم استخدام الإنتاج.

التسلسل الهرمي للمعدات

```
PLANT (مصنع)
  └── AREA (منطقة)
        └── SYSTEM (نظام)
              └── EQUIPMENT (معدة)
                    └── SUB_ASSEMBLY (تجميع فرعي)
                          └── MAINTAINABLE_ITEM (عنصر قابل للصيانة)
```

الجداول الرئيسية

الجدول	الوصف	الحقول الرئيسية
`UserModel`	المستخدمون	`user_id` , `email` , `username` , `role` , `hashed_password`
`PlantModel`	المصانع	`plant_id` , `name` , `name_fr` , `name_ar` , `location`
`HierarchyNodeModel`	عقد التسلسل الهرمي	`node_id` , `node_type` , `name` , `parent_node_id` , `plant_id` , `criticality`
`CriticalityAssessmentModel`	تقييمات الحرجية	`assessment_id` , `node_id` , `criteria_scores` , `risk_class` , `overall_score`
`FunctionModel`	الوظائف	`function_id` , `node_id` , `function_type` , `description`
`FunctionalFailureModel`	الأعطال الوظيفية	`failure_id` , `function_id` , `failure_type` , `description`
`FailureModeModel`	أنماط الأعطال	`failure_mode_id` , `functional_failure_id` , `mechanism` , `cause` , `strategy_type`
`MaintenanceTaskModel`	مهام الصيانة	`task_id` , `failure_mode_id` , `name` , `task_type` , `frequency_value`
`WorkPackageModel`	حزم العمل	`work_package_id` , `name` , `node_id` , `frequency_value` , `status`
`WorkOrderModel`	أوامر العمل	`work_order_id` , `order_type` , `equipment_id` , `priority` , `status`
`HealthScoreModel`	نقاط صحة الأصول	`score_id` , `node_id` , `composite_score` , `health_class` , `trend`
`KPIMetricsModel`	مؤشرات الأداء	`metrics_id` , `plant_id` , `mtbf_days` , `mttr_hours` , `availability_pct`
`FailurePredictionModel`	توقعات الأعطال	`prediction_id` , `equipment_id` , `weibull_params` , `risk_score`
`SAPUploadPackageModel`	حزم رفع SAP	`package_id` , `plant_code` , `maintenance_plan` , `status`
`CAPAItemModel`	إجراءات CAPA	`capa_id` , `capa_type` , `title` , `status` , `root_cause`

جداول إضافية

الجدول	الوصف
`VarianceAlertModel`	تنبيهات الانحراف
`ExpertCardModel`	بطاقات الخبراء
`AuditLogModel`	سجل التدقيق
`FieldCaptureModel`	التقاط البيانات الميدانية
`WorkRequestModel`	طلبات العمل
`PlannerRecommendationModel`	توصيات المُخطّط
`BacklogItemModel`	عناصر الأعمال المتراكمة
`OptimizedBacklogModel`	الأعمال المتراكمة المُحسّنة
`WorkforceModel`	القوى العاملة
`ShutdownCalendarModel`	تقويم التوقفات
`InventoryItemModel`	عناصر المخزون
`WeeklyProgramModel`	البرامج الأسبوعية
`ShutdownEventModel`	أحداث التوقف
`MoCRequestModel`	طلبات إدارة التغيير
`SparePartAnalysisModel`	تحليل قطع الغيار
`RBIAssessmentModel`	تقييمات الفحص المبني على المخاطر
`ReportModel`	التقارير
`NotificationModel`	الإشعارات
`RCAAnalysisModel`	تحليلات السبب الجذري
`PlanningKPISnapshotModel`	لقطات مؤشرات التخطيط
`DEKPISnapshotModel`	لقطات مؤشرات إزالة العيوب
`UserFeedbackModel`	ملاحظات المستخدمين
`FMECAWorksheetModel`	أوراق عمل FMECA

---

8. المصادقة والتفويض

تدفق JWT

```
1. المستخدم يرسل  →  POST /api/v1/auth/login  (username + password)
2. الخادم يُعيد   ←  { access_token (4 ساعات), refresh_token (7 أيام) }
3. كل طلب يحمل    →  Authorization: Bearer <access_token>
4. 401 → عند انتهاء الصلاحية → POST /api/v1/auth/refresh
5. إعادة التوجيه إلى →  عند فشل التحديث /login
```

الأدوار الخمسة

الدور	الاسم	الصلاحيات
`admin`	المدير	صلاحيات كاملة على جميع الوحدات والإعدادات
`manager`	المشرف	العمليات، التحليلات، لوحة المعلومات التنفيذية، التقارير
`planner`	المُخطّط	حزم العمل، الجدولة، التخطيط، الأعمال المتراكمة، FMEA
`tecnico`	الفني	لوحة المعلومات، التقاط البيانات الميدانية، طلبات العمل
`engineer`	المهندس	التحليلات، FMEA، الموثوقية، تحليل السبب الجذري

المستخدمون التجريبيون

اسم المستخدم	كلمة المرور	الدور
`admin`	`admin123`	مدير
`manager`	`manager123`	مشرف
`planner`	`planner123`	مُخطّط
`tecnico`	`tecnico123`	فني

أمان كلمات المرور

- **خوارزمية التشفير**: bcrypt مع عامل تكلفة تلقائي
- **التخزين**: فقط القيمة المُشفّرة (`hashed_password`) تُخزّن في قاعدة البيانات
- **التحقق**: مقارنة آمنة زمنياً عبر `bcrypt.checkpw()`

---

9. مرجع واجهة API

المصادقة (`/api/v1/auth`)

الطريقة	نقطة النهاية	الوصف	المصادقة
`POST`	`/auth/login`	تسجيل الدخول والحصول على رمز JWT	لا
`POST`	`/auth/refresh`	تحديث رمز الوصول	لا
`POST`	`/auth/register`	تسجيل مستخدم جديد	admin
`GET`	`/auth/me`	الحصول على بيانات المستخدم الحالي	نعم
`PUT`	`/auth/me`	تحديث الملف الشخصي	نعم
`PUT`	`/auth/change-password`	تغيير كلمة المرور	نعم
`GET`	`/auth/users`	قائمة جميع المستخدمين	admin
`PUT`	`/auth/users/{user_id}/role`	تغيير دور المستخدم	admin
`PUT`	`/auth/users/{user_id}/deactivate`	تعطيل حساب المستخدم	admin
`PUT`	`/auth/users/{user_id}/activate`	تفعيل حساب المستخدم	admin

التسلسل الهرمي (`/api/v1/hierarchy`)

الطريقة	نقطة النهاية	الوصف	المصادقة
GET	/hierarchy/plants	قائمة المصانع	لا
POST	/hierarchy/plants	إنشاء مصنع جديد	نعم
GET	/hierarchy/nodes	قائمة العقد (مع فلترة)	لا
POST	/hierarchy/nodes	إنشاء عقدة جديدة	نعم
GET	/hierarchy/nodes/{node_id}	الحصول على تفاصيل عقدة	لا
GET	/hierarchy/nodes/{node_id}/tree	الحصول على الشجرة الفرعية	لا
POST	/hierarchy/build-from-vendor	بناء هرمية من بيانات المصنّع	نعم
GET	/hierarchy/stats	إحصائيات العقد حسب النوع	لا

FMEA (`/api/v1/fmea`)

الطريقة	نقطة النهاية	الوصف	المصادقة
POST	/fmea/failure-modes	إنشاء نمط عطل جديد	نعم
GET	/fmea/failure-modes/{fm_id}	الحصول على تفاصيل نمط عطل	لا
GET	/fmea/failure-modes	قائمة أنماط الأعطال	لا
GET	/fmea/validate-fm	التحقق من صحة مجموعة آلية/سبب	لا
GET	/fmea/fm-combinations	قائمة المجموعات الصالحة	لا
POST	/fmea/rcm-decide	تنفيذ قرار شجرة RCM	لا
GET	/fmea/functions	قائمة الوظائف	لا
POST	/fmea/functions	إنشاء وظيفة جديدة	نعم
GET	/fmea/functional-failures	قائمة الأعطال الوظيفية	لا
POST	/fmea/functional-failures	إنشاء عطل وظيفي جديد	نعم
GET	/fmea/fmeca/worksheets	قائمة أوراق عمل FMECA	لا
POST	/fmea/fmeca/worksheets	إنشاء ورقة عمل FMECA	نعم
GET	/fmea/fmeca/worksheets/{id}	الحصول على ورقة عمل FMECA	لا
POST	/fmea/fmeca/rpn	حساب رقم أولوية المخاطر	لا
PUT	/fmea/fmeca/worksheets/{id}/run-decisions	تنفيذ قرارات FMECA	نعم
GET	/fmea/fmeca/worksheets/{id}/summary	ملخص ورقة عمل FMECA	لا

التحليلات (`/api/v1/analytics`)

الطريقة	نقطة النهاية	الوصف	المصادقة
POST	/analytics/health-score	حساب نقاط صحة الأصول	نعم
POST	/analytics/kpis	حساب مؤشرات الأداء الرئيسية	نعم
POST	/analytics/weibull-fit	ملاءمة توزيع Weibull	لا
POST	/analytics/weibull-predict	التنبؤ بالأعطال عبر Weibull	نعم
POST	/analytics/variance-detect	كشف الانحرافات الإحصائية	لا
GET	/analytics/variance-alerts	قائمة تنبيهات الانحراف	لا
GET	/analytics/asset-health	نقاط صحة جميع المعدات	لا

التقارير (`/api/v1/reporting`)

الطريقة	نقطة النهاية	الوصف	المصادقة
POST	/reporting/reports/weekly	إنشاء تقرير أسبوعي	نعم
POST	/reporting/reports/monthly	إنشاء تقرير شهري	نعم
POST	/reporting/reports/quarterly	إنشاء تقرير ربعي	نعم
GET	/reporting/reports	قائمة التقارير	لا
GET	/reporting/reports/{report_id}	الحصول على تفاصيل تقرير	لا
POST	/reporting/de-kpis/calculate	حساب مؤشرات إزالة العيوب	نعم
POST	/reporting/de-kpis/program-health	تقييم صحة برنامج DE	نعم
POST	/reporting/notifications/generate	توليد الإشعارات	نعم
GET	/reporting/notifications	قائمة الإشعارات	لا
PUT	/reporting/notifications/{id}/ack	تأكيد استلام إشعار	نعم
POST	/reporting/import/validate	التحقق من بيانات الاستيراد	نعم
POST	/reporting/export	تصدير البيانات	نعم
POST	/reporting/cross-module/analyze	تحليل عبر الوحدات	نعم

10. التدويل i18n

اللغات المدعومة

اللغة	الرمز	الاتجاه	الملف
الإنجليزية	`en`	من اليسار إلى اليمين (LTR)	`frontend/src/i18n/en.js`
الإسبانية	`es`	من اليسار إلى اليمين (LTR)	`frontend/src/i18n/es.js`
العربية	`ar`	من اليمين إلى اليسار (RTL)	`frontend/src/i18n/ar.js`

التنفيذ عبر React Context

```
// استخدام LanguageContext
import { useLanguage } from '../contexts/LanguageContext';

function MyComponent() {
    const { t, language, setLanguage } = useLanguage();
    return <h1>{t('dashboard.title')}</h1>;
}
```

هيكل الترجمات

كل ملف ترجمة يحتوي على الأقسام التالية:

القسم	المفتاح	الوصف
العام	`common`	نصوص عامة (حفظ، إلغاء، حذف...)
المصادقة	`auth`	تسجيل الدخول، كلمة المرور
التنقل	`nav`	عناوين القائمة الجانبية
التسلسل الهرمي	`hierarchy`	إدارة الأصول
الحرجية	`criticality`	تقييمات المخاطر
FMEA	`fmea`	أنماط الأعطال
المهام	`tasks`	مهام الصيانة
حزم العمل	`workPackages`	حزم العمل
التحليلات	`analytics`	المؤشرات والرسوم البيانية
التقارير	`reports`	التقارير والتصدير
المُخطّط	`planner`	توصيات التخطيط
الاستراتيجية	`strategy`	استراتيجيات الصيانة
الإدارة	`admin`	إعدادات النظام

دعم RTL

عند اختيار اللغة العربية، يتم تلقائياً:

- تطبيق `dir="rtl"` على عنصر `<html>`

- عكس اتجاه التخطيط (يظهر على اليمين Sidebar)
- ضبط محاذاة النصوص والأيقونات

---

11. تكامل SAP

ملفات البيانات الوهمية

يوفر النظام بيانات SAP وهمية للتطوير والاختبار في مجلد `sap_mock/data/` :

الملف	المعاملة SAP	الوصف
`work_orders.json`	IW31/IW32	أوامر العمل
`functional_locations.json`	IL01	المواقع الوظيفية
`materials_bom.json`	CS01	قائمة المواد وفاتورة المواد
`equipment_master.json`	IE01	البيانات الرئيسية للمعدات
`maintenance_plans.json`	IP10	خطط الصيانة

تدفق الرفع إلى SAP

```
1. إنشاء حزمة عمل (Work Package)
   ↓
2. الموافقة على حزمة العمل
   ↓
3. توليد حزمة SAP تلقائياً
   ↓
4. مراجعة الحزمة في صفحة SAPReview
   ↓
5. الموافقة النهائية على الحزمة
   ↓
6. تصدير إلى SAP PM (ملف أو واجهة برمجية)
```

ربط الحقول مع SAP

حقل النظام	حقل SAP	الوصف
`work_order_id`	`AUFNR`	رقم أمر العمل
`equipment_tag`	`EQUNR`	رقم المعدة
`task_type`	`AUART`	نوع أمر العمل
`priority`	`PRIOK`	أولوية أمر العمل
`description`	`KTEXT`	وصف مختصر
`planned_hours`	`ARBEI`	ساعات العمل المخططة

---

12. Docker والنشر

الخدمات الثلاث

الخدمة	الحاوية	المنفذ	الوصف
`ocp-backend`	FastAPI + Gunicorn	`8000`	الواجهة الخلفية
`ocp-frontend`	React + Nginx	`5173`	الواجهة الأمامية
`ocp-nginx`	Nginx Alpine	`8080` / `8443`	الوكيل العكسي

عملية البناء

الواجهة الخلفية

```
FROM python:3.11-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
CMD ["gunicorn", "api.main:app", "-k", "uvicorn.workers.UvicornWorker", "-b", "0.0.0.0:8000"]
```

الواجهة الأمامية

```
# مرحلة البناء
FROM node:20-alpine AS build
WORKDIR /app
COPY package*.json .
RUN npm ci
COPY . .
RUN npm run build

# مرحلة الإنتاج
FROM nginx:alpine
COPY --from=build /app/dist /usr/share/nginx/html
```

أوامر Docker

الأمر	الوصف
`docker-compose up -d --build`	بناء وتشغيل جميع الخدمات
`docker-compose logs -f ocp-backend`	عرض سجلات الواجهة الخلفية
`docker-compose stop`	إيقاف جميع الخدمات
`docker-compose down`	إيقاف وحذف الحاويات
`docker-compose up -d --build ocp-backend`	إعادة بناء خدمة واحدة فقط

الشبكة والحجوم

```
networks:
  ocp-network:
    name: ocp-maintenance-net
    driver: bridge

volumes:
  ocp_db_data:      # بيانات قاعدة البيانات الدائمة
  ocp_sap_data:     # بيانات SAP الدائمة
```

---

13. الوحدات الوظيفية

الوحدة 1: التسلسل الهرمي للأصول والحرجية

الصفحة	المكوّن	الوظيفة
التسلسل الهرمي	`Hierarchy.jsx`	عرض وإدارة شجرة الأصول بستة مستويات (مصنع → منطقة → نظام → معدة → تجميع فرعي → عنصر قابل للصيانة)
الحرجية	`Criticality.jsx`	تقييم حرجية المعدات باستخدام مصفوفة المخاطر (الاحتمالية × الأثر)

المميزات:
- تلقائياً OEM/بناء الهرمية من بيانات المصنّع
- تصنيف الحرجية: A (حرج)، B (مهم)، C (عام)
- اقتراحات الذكاء الاصطناعي لتصنيف الحرجية
- ربط المعدات بالمواقع الوظيفية في SAP

الوحدة 2: FMEA/FMECA واستراتيجية RCM

الصفحة	المكوّن	الوظيفة
FMEA	`FMEA.jsx`	تحليل أنماط الأعطال والآثار مع شجرة قرارات RCM
FMECA	`FMECA.jsx`	ورقة عمل FMECA مع حساب RPN (الخطورة × الحدوث × الكشف)
الاستراتيجية	`Strategy.jsx`	عرض وإدارة استراتيجيات الصيانة المُختارة

أنواع استراتيجيات الصيانة:

الرمز	النوع	الوصف
CBM	صيانة مبنية على الحالة	مراقبة مستمرة للحالة واتخاذ القرار بناءً على المؤشرات
TBM	صيانة مبنية على الوقت	تنفيذ دوري بفترات زمنية محددة
FTM	صيانة حتى العطل	تشغيل حتى حدوث العطل (للمعدات غير الحرجة)
RTF	التشغيل حتى الفشل	لا صيانة وقائية (عند عدم وجود عواقب خطيرة)

الوحدة 3: العمليات والتخطيط

الصفحة	المكوّن	الوظيفة
طلبات العمل	`WorkRequests.jsx`	إنشاء وتتبع طلبات العمل
حزم العمل	`WorkPackages.jsx`	تجميع المهام في حزم عمل
الأعمال المتراكمة	`Backlog.jsx`	إدارة قائمة الانتظار وتحديد الأولويات
الجدولة	`Scheduling.jsx`	البرنامج الأسبوعي وتخصيص الموارد
التخطيط	`Planning.jsx`	نظرة عامة على التخطيط
المُخطّط	`Planner.jsx`	توصيات الذكاء الاصطناعي للمُخطّط
التقاط الميداني	`FieldCapture.jsx`	تسجيل الملاحظات الميدانية (نص/صوت/صورة)

الوحدة 4: التحليلات والتقارير والتحسين المستمر

الصفحة	المكوّن	الوظيفة
التحليلات	`Analytics.jsx`	مؤشرات الأداء الرئيسية والرسوم البيانية
لوحة المعلومات التنفيذية	`ExecutiveDashboard.jsx`	ملخص تنفيذي لجميع المصانع
التقارير	`Reports.jsx`	إنشاء تقارير أسبوعية/شهرية/ربعية
الموثوقية	`Reliability.jsx`	تحليل Weibull، RBI، قطع الغيار
تحليل السبب الجذري	`RCA.jsx`	تحليل 5W2H والسبب والأثر
إزالة العيوب	`DefectElimination.jsx`	برنامج إزالة العيوب ومؤشرات CAPA
مراجعة SAP	`SAPReview.jsx`	مراجعة والموافقة على حزم SAP

---

14. سير العمل الرئيسية

1. سير عمل تقييم الحرجية

```
اختيار المعدة من التسلسل الهرمي
              ↓
تعبئة مصفوفة المخاطر (معايير الأثر × الاحتمالية)
              ↓
حساب الدرجة الإجمالية تلقائياً
              ↓
اقتراح الذكاء الاصطناعي لفئة المخاطر
              ↓
مراجعة واعتماد التقييم
              ↓
تحديث حالة الحرجية في التسلسل الهرمي (A/B/C)
```

2. سير عمل من FMEA إلى حزمة العمل

```
تحديد الوظائف للمعدة
        ↓
تحديد الأعطال الوظيفية لكل وظيفة
        ↓
تحديد أنماط الأعطال (آلية + سبب)
        ↓
تشغيل شجرة قرارات RCM
        ↓
اختيار استراتيجية الصيانة (CBM/TBM/FTM/RTF)
        ↓
إنشاء مهام الصيانة مع الموارد
        ↓
تجميع المهام في حزمة عمل
        ↓
الموافقة والتصدير إلى SAP
```

3. سير عمل دورة حياة طلب العمل

```
التقاط بيانات ميدانية (نص/صوت/صورة)
        ↓
تصنيف آلي بالذكاء الاصطناعي (نوع العطل، المعدة، الأولوية)
        ↓
إنشاء طلب عمل تلقائياً
        ↓
مراجعة المُخطّط مع توصيات AI
        ↓
الموافقة / التعديل / التأجيل / التصعيد
        ↓
إضافة إلى قائمة الأعمال المتراكمة
        ↓
جدولة في البرنامج الأسبوعي
        ↓
تنفيذ وإغلاق
```

4. سير عمل إنشاء التقارير والتصدير

```
اختيار نوع التقرير (أسبوعي/شهري/ربعي)
        ↓
تحديد المصنع والفترة الزمنية
        ↓
توليد التقرير تلقائياً مع المؤشرات
        ↓
مراجعة المحتوى والرسوم البيانية
        ↓
تصدير (Excel / PDF)
```

---

15. الأمان

تدابير الحماية

التدبير	التفاصيل
مصادقة JWT	رمز وصول صالح لـ 4 ساعات، رمز تحديث صالح لـ 7 أيام، HS256 خوارزمية
تشفير كلمات المرور	مع عامل تكلفة تلقائي bcrypt
قائمة CORS البيضاء	نطاقات محددة فقط مسموح بها في الإعدادات
مفتاح API	عند تفعيله (POST/PUT/DELETE) مطلوب لطلبات التعديل `X-API-Key` رأس
تحديد المعدل	30 طلب/ثانية للطلبات العامة، 2 طلب/ثانية لطلبات الإدارة
رؤوس الأمان	عبر Nginx `X-XSS-Protection` , `X-Frame-Options` , `X-Content-Type-Options`
جاهزية HTTPS	دعم SSL/TLS عبر Nginx 8443 على المنفذ
التحكم بالوصول المبني على الأدوار	5 أدوار مع صلاحيات محددة لكل نقطة نهاية
حاوية غير جذرية	الحاويات تعمل بمستخدم غير جذري (non-root)
حماية من حقن SQL	استخدام SQLAlchemy ORM مع استعلامات مُعلّمة (parameterized)
حماية CSRF	رموز SameSite في ملفات تعريف الارتباط ورؤوس مخصصة

---

16. دليل التطوير المحلي

المتطلبات الأساسية

المتطلب	الإصدار المطلوب
Python	3.11 أو أحدث
Node.js	20 أو أحدث
npm	9 أو أحدث
Git	أحدث إصدار

إعداد الواجهة الخلفية

```
# 1. إنشاء بيئة افتراضية
python -m venv venv

# 2. تفعيل البيئة الافتراضية
# في Windows:
venv\Scripts\activate
# في Linux/macOS:
source venv/bin/activate

# 3. تثبيت الاعتماديات
pip install -r requirements.txt

# 4. إنشاء ملف .env (نسخ من المثال)
cp .env.example .env
# أو إنشاء يدوياً مع المتغيرات المطلوبة

# 5. تشغيل خادم التطوير
uvicorn api.main:app --reload --port 8000
```

إعداد الواجهة الأمامية

```
# 1. الانتقال إلى مجلد الواجهة الأمامية
cd frontend

# 2. تثبيت الاعتماديات
npm install

# 3. تشغيل خادم التطوير
npm run dev
```

عناوين الوصول

الوصف	العنوان	الخدمة
تطبيق React	`http://localhost:5173`	الواجهة الأمامية
خادم API	`http://localhost:8000`	الواجهة الخلفية
واجهة Swagger UI	`http://localhost:8000/docs`	توثيق API التفاعلي
حالة النظام	`http://localhost:8000/health`	فحص الصحة

ملء قاعدة البيانات ببيانات أولية

```
# عبر curl
curl -X POST http://localhost:8000/api/v1/admin/seed

# أو عبر Python
python -c "from api.seed import seed_database; seed_database()"
```

---

17. قائمة مراجعة الإنتاج

قبل النشر في بيئة الإنتاج، تأكد من إنجاز جميع العناصر التالية:

الرقم	العنصر	الحالة	الوصف
1	مفتاح JWT السري	مطلوب	تغيير `JWT_SECRET_KEY` إلى مفتاح عشوائي قوي (256 بت على الأقل)
2	PostgreSQL	مطلوب	التبديل من SQLite إلى PostgreSQL عبر `DATABASE_URL`
3	HTTPS	مطلوب	تفعيل SSL/TLS عبر Nginx على المنفذ 8443
4	شهادات SSL	مطلوب	تثبيت شهادات SSL صالحة في مجلد `./certs/`
5	مفاتيح API	موصى به	تعيين `API_KEY` لحماية طلبات التعديل
6	نطاقات CORS	مطلوب	تحديد `CORS_ORIGINS` بالنطاقات المسموحة فقط
7	مفتاح Anthropic	اختياري	تعيين `ANTHROPIC_API_KEY` لتفعيل ميزات الذكاء الاصطناعي
8	وضع التصحيح	مطلوب	تعيين `DEBUG=false` في بيئة الإنتاج
9	تحديد المعدل	موصى به	التأكد من إعدادات تحديد المعدل في Nginx
10	حجوم Docker	مطلوب	التأكد من تكوين حجوم دائمة لقاعدة البيانات (`ocp_db_data`)
11	المراقبة	موصى به	إعداد مراقبة نقطة النهاية `/health` ومراقبة السجلات
12	تشغيل الخدمات	مطلوب	تنفيذ `docker-compose up -d --build` والتحقق من جميع الخدمات

18. متغيرات البيئة

جدول المتغيرات الكامل

المتغير	القيمة الافتراضية	الوصف
`DATABASE_URL`	`sqlite:///./ocp_maintenance.db`	عنوان اتصال قاعدة البيانات (SQLite للتطوير، PostgreSQL للإنتاج)
`SAP_MOCK_DIR`	`sap_mock/data`	مسار مجلد البيانات الوهمية لـ SAP
`ANTHROPIC_API_KEY`	`""` (فارغ)	مفتاح API لخدمة Anthropic Claude
`GOOGLE_AI_API_KEY`	`""` (فارغ)	مفتاح API لخدمة Google AI
`API_KEY`	`""` (فارغ)	مفتاح API لحماية طلبات التعديل (اختياري)
`DEBUG`	`false`	تفعيل وضع التصحيح (يجب أن يكون `false` في الإنتاج)
`CORS_ORIGINS`	`http://localhost:5173,http://localhost:8501,http://localhost`	النطاقات المسموحة لـ CORS (مفصولة بفاصلة)
`LOG_LEVEL`	`INFO`	مستوى التسجيل (`DEBUG` , `INFO` , `WARNING` , `ERROR`)
`JWT_SECRET_KEY`	`ocp-maintenance-dev-secret-key-change-in-production-2024`	المفتاح السري لتوقيع رموز JWT (يجب

المتغير	القيمة الافتراضية	الوصف
		تغييره في الإنتاج)
`JWT_ALGORITHM`	`HS256`	خوارزمية توقيع JWT
`JWT_ACCESS_TOKEN_EXPIRE_MINUTES`	`240`	مدة صلاحية رمز الوصول بالدقائق (4 ساعات)
`JWT_REFRESH_TOKEN_EXPIRE_DAYS`	`7`	مدة صلاحية رمز التحديث بالأيام
`BACKEND_PORT`	`8000`	منفذ خدمة الواجهة الخلفية
`FRONTEND_PORT`	`5173`	منفذ خدمة الواجهة الأمامية
`NGINX_PORT`	`8080`	منفذ HTTP لـ Nginx
`NGINX_SSL_PORT`	`8443`	منفذ HTTPS لـ Nginx
`PYTHONPATH`	`/app`	مسار Python داخل الحاوية
`VITE_API_BASE_URL`	`""` (فارغ)	عنوان URL الأساسي لـ API في الواجهة الأمامية

مثال ملف `.env`

```
# قاعدة البيانات
DATABASE_URL=postgresql://ocp_user:ocp_password@db:5432/ocp_maintenance

# المصادقة
JWT_SECRET_KEY=your-very-secure-random-secret-key-256-bits-minimum
JWT_ALGORITHM=HS256
JWT_ACCESS_TOKEN_EXPIRE_MINUTES=240
JWT_REFRESH_TOKEN_EXPIRE_DAYS=7

# الذكاء الاصطناعي
ANTHROPIC_API_KEY=sk-ant-xxxxxxxxxxxx

# الأمان
API_KEY=your-api-key-for-mutations
CORS_ORIGINS=https://your-domain.com
DEBUG=false
LOG_LEVEL=INFO

# المنافذ
BACKEND_PORT=8000
FRONTEND_PORT=5173
NGINX_PORT=8080
NGINX_SSL_PORT=8443
```

إحصائيات المشروع

المقياس	القيمة
موجّهات API	18
نقاط نهاية API	أكثر من 100
نماذج قاعدة البيانات	أكثر من 25
خدمات الواجهة الخلفية	23
صفحات React	23
مكونات واجهة المستخدم	أكثر من 20
مخططات Pydantic	أكثر من 50
دوال عميل API	أكثر من 50
اللغات المدعومة	3 (الإنجليزية، الإسبانية، العربية)
خدمات Docker	3
اعتماديات Python	27
اعتماديات npm	أكثر من 30

(المكتب الشريف للفوسفاط) OCP ***للصيانة بالذكاء الاصطناعي*** *-- مُطوَّرة لمجموعة* ***منصة OCP***
جميع الحقوق محفوظة 2026