حالة المشروع الحالية

مرجع مشروع Elite Message

توثيق موحد للأعمال والمتطلبات والبنية التقنية والتصميم الحالي للمنصة.

12-04

التحكم بالمستند

الحقل	القيمة
اسم المستند	مرجع حالة مشروع Elite Message الحالية
نوع المستند	توثيق مشروع موحد
الجمهور	فريق مختلط: منتج، إدارة، هندسة، وتشغيل
نطاق الحالة	يعكس الوضع الحالي للمستودع حتى 2026-04-12
التقنيات الرئيسية	Next.js 16, React 19, NestJS 11, Prisma 7, MySQL, Redis, BullMQ, MinIO
مصدر الحقيقة	الشيفرة الحالية، المخطط، الإعدادات، والوثائق في docs/current-status

جدول المحتويات

1 الملخص التنفيذي

1.1 حالة التنفيذ الحالية

1.2 الملخص التقني

2 توثيق الأعمال والتخطيط

2.1 نظرة عامة على المشروع

2.2 مقترح البرمجية

2.3 وثيقة النطاق

2.4 دراسة الجدوى

2.4.1 الجدوى التقنية

2.4.2 الجدوى التشغيلية

2.4.3 الجدوى الاقتصادية

2.5 الميزانية والجدول الزمني

2.6 أصحاب المصلحة

2.7 سجل المخاطر

3 توثيق المتطلبات

3.1 وثيقة متطلبات الأعمال BRD

3.2 مواصفة المتطلبات البرمجية SRS

3.3 المتطلبات الوظيفية

3.4 المتطلبات غير الوظيفية

3.5 حالات الاستخدام

3.5.1 العميل يسجل حساباً

3.5.2 العميل ينشئ instance ويستخدم الـ API

3.5.3 الإدارة تحقق في مشكلة

3.6 قصص المستخدم

3.7 معايير القبول

4 توثيق المنتج والتصميم

4.1 وثيقة معمارية النظام

4.1.1 مخطط المعمارية التشغيلية

4.2 التصميم عالي المستوى HLD

4.3 التصميم منخفض المستوى LLD

4.3.1 تطبيق العميل

4.3.2 تطبيق الإدارة

4.3.3 الـ API
4.4 توثيق مخطط قاعدة البيانات
4.4.1 ملاحظات تصميمية
4.5 تصميم الـ API ومواصفاته
4.6 توثيق UI/UX
4.7 Wireframes و Mockups
4.7.1 بنية صفحة تسجيل الدخول للعميل
4.7.2 بنية لوحة العميل
4.8 المخططات الانسيابية وتسلسل العمليات
4.8.1 تدفق مصادقة العميل
4.8.2 من إنشاء الـ instance إلى استخدام الـ API
4.8.3 تسلسل عمليات الـ worker ودورة الحياة
4.8.4 تدفق تسليم الـ webhook
5 الملاحق والمراجع
5.1 بنية المستودع
5.2 البنية التحتية المحلية
5.3 مراجع داعمة
5.4 ملاحظة ختامية

1 الملخص التنفيذي

يمثل Elite Message حالياً منصة تشغيل رسائل متعددة التطبيقات وليست مجرد هيكل أولي للمشروع. يتكوّن النظام من:

- تطبيق ويب للعملاء لإدارة الحسابات ومساحات العمل والـ instances والرسائل والإعدادات والاشتراك ووثائق الـ API
- تطبيق ويب للإدارة لمتابعة المستخدمين ومساحات العمل والعاملين والرسائل والدعم والتدقيق
- واجهة API مبنية على `NestJS` تخدم مسارات العملاء والإدارة والواجهات العامة والداخلية
- خدمة عامل `worker` تدير العمليات غير المتزامنة وحالة التشغيل وربط WhatsApp
- بنية تحتية محلية تشمل `MySQL` و `Redis` و `MinIO` و `Prometheus` و `Grafana` و `Alertmanager`

1.1 حالة التنفيذ الحالية

المجال	التنفيذ الحالي	الحالة	ملاحظة حالية
واجهة العميل	التسجيل وتسجيل الدخول ولوحة التحكم والرسائل والإعدادات والاشتراك وتفاصيل الـ instance ووثائق الـ API موجودة في التطبيق.	منفذ	سطح العميل يعمل فعلياً ويمكن استخدامه محلياً بشكل متكامل.
واجهة الإدارة	يوجد تسجيل دخول للإدارة مع دعم MFA وصفحات تشغيلية للمستخدمين ومساحات العمل والعاملين والرسائل والدعم والتدقيق.	منفذ	سطح الإدارة يدعم المتابعة التشغيلية اليومية والتحقيق والدعم.
التشغيل والرسائل	يوجد نموذج بيانات واضح لدورة حياة الـ instance وحالة التشغيل والرسائل الخارجة والداخلة وتسليمات الـ webhook.	منفذ	المخطط والـ API يغطيان دورة التشغيل والـ queue وحركة الرسائل الأساسية.
توثيق الـ API	يوجد توثيق داخل واجهة العميل مع أمثلة للغات backend الشائعة ودعم لرموز الحساب والـ instance.	منفذ	صفحة الوثائق أصبحت جزءاً عملياً من تجربة تفعيل الـ API لدى العميل.
الجاهزية التشغيلية	القياسات والتنبيهات وحدود المعدل والتوقيع على webhooks موجودة جزئياً، لكن الصلابة الإنتاجية ما زالت غير مكتملة.	جزئي	ما زال المشروع يحتاج إلى جولة إضافية من hardening قبل الاعتماد الإنتاجي الكامل.
مرجعية التوثيق	هذا المستند يوحّد حالة المشروع الحالية ويستبدل افتراضات المرحلة الصفرية السابقة.	منفذ	يوجد الآن مرجع حديث لحالة المشروع بصيغة PDF وبصيغة صفحات قابلة للتصفح.

1.2 الملخص التقني

المجال	التقنية الحالية
Monorepo	`Turborepo` + `pnpm`
تطبيقات الويب	Next.js 16 + React 19
الـ API	NestJS 11 + Express 5
العامل	`whatsapp-web.js` + BullMQ + Redis
قاعدة البيانات	MySQL + Prisma 7
التخزين	MinIO / S3-compatible
المراقبة	Prometheus + Grafana + Alertmanager

2 توثيق الأعمال والتخطيط

2.1 نظرة عامة على المشروع

يهدف Elite Message إلى توفير منصة متعددة المستأجرين لإدارة حسابات WhatsApp المرتبطة بـ instances قابلة للتشغيل والمتابعة من قبل العميل والإدارة. المشكلة الأساسية التي يعالجها النظام هي الحاجة إلى منصة موحدة تجمع:

- خدمة ذاتية للعميل
- تحكم تشغيلي للإدارة
- تتبع تشغيلي ورسائلي وتدقيقي
- واجهات API قابلة للاستهلاك من أنظمة backend

2.2 مقترح البرمجية

الحل المقترح، والذي أصبح جزء كبير منه منفذاً بالفعل، يقوم على فصل السطحين الرئيسيين:

- **سطح العميل** لإدارة الوصول وتهيئة الـ instances واستخدام الـ API
- **سطح الإدارة** للمتابعة والدعم والإشراف والتحقيق

مع الاعتماد على:

- API مركزية
- worker مستقل لإدارة جلسات WhatsApp
- قاعدة بيانات مركزية
- Queue-backed processing للعمليات غير المتزامنة

2.3 وثيقة النطاق

الأمور الداخلة ضمن النطاق الحالي:

- تسجيل العملاء وتسجيل دخولهم
- Google OAuth للعملاء
- تسجيل دخول الإدارة ودعم MFA
- إنشاء الـ instances ومتابعة حالتها وعملياتها
- الرسائل الخارجة والداخلة وعمليات الـ webhook
- رموز الـ API للحساب والـ instance
- واجهات الإدارة للمستخدمين ومساحات العمل والعاملين والدعم والتدقيق

الأمور الخارجة عن النطاق الحالي:

- تطبيقات موبايل أصلية
- تكاملات مؤسسية واسعة مع أنظمة خارجية
- نظام فوترة إنتاجي متكامل
- حزمة امتثال مؤسسية كاملة

2.4 دراسة الجدوى

2.4.1 الجدوى التقنية

مرتفعة، لأن النظام موجود فعلياً في صورة تطبيقات وصفحات ومسارات API ونموذج بيانات واضح.

2.4.2 الجدوى التشغيلية

متوسطة، لأن البيئة المحلية مدعومة جيداً لكن هناك فجوات متبقية في الجاهزية الإنتاجية والإجراءات التشغيلية.

2.4.3 الجدوى الاقتصادية

متوسطة إلى مرتفعة، لأن القسم الأكبر من البناء الأساسي موجود بالفعل، والتركيز القادم هو على الصقل والتوثيق والتقوية.

2.5 الميزانية والجدول الزمني

هذه الأرقام تقديرية وليست ناتجة عن سجلات مالية داخل المستودع:

- توثيق وتثبيت المتطلبات: 1 إلى 2 أسبوع
- تحسين تجربة العميل: 2 إلى 4 أسابيع
- تحسين تجربة الإدارة: 2 إلى 3 أسابيع
- التقوية التشغيلية والتحقق: 3 إلى 5 أسابيع
- جاهزية الإطلاق: 1 إلى 2 أسبوع

2.6 أصحاب المصلحة

- مالك المنتج أو الراعي
- العميل صاحب مساحة العمل
- مشغلو العميل
- مشرفو المنصة
- فريق الدعم والتشغيل
- فريق التطوير الخلفي والواجهات
- الجودة والتحقق
- DevOps / SRE

2.7 سجل المخاطر

المعرف	الخطر	المعالجة
R1	اختلاف التوثيق عن التنفيذ	تحديث مرجع current-status مع أي تغيير رئيسي
R2	عدم استقرار runtime المرتبط بـ WhatsApp	متابعة heartbeat والحالة والأحداث التشغيلية
R3	مشاكل إعداد OAuth أو الجلسات	رسائل خطأ واضحة وتعليمات إعداد دقيقة
R4	بقاء الصلابة الإنتاجية جزئية	إنهاء خطة hardening والتحقق قبل الإطلاق
R5	توسع نطاق تحسينات UI/UX بلا ضبط	تنفيذ التحسينات على مراحل واضحة

3 توثيق المتطلبات

3.1 وثيقة متطلبات الأعمال BRD

المتطلبات التجارية الأساسية هي:

- تمكين العميل من تشغيل الخدمة ذاتياً
- تمكين الإدارة من الإشراف الكامل
- تمكين التكامل البرمجي عبر API
- ضمان إمكانية التتبع والتحقيق في الحالات التشغيلية

3.2 مواصفة المتطلبات البرمجية SRS

يتكوّن النظام من:

- تطبيق عميل
- تطبيق إدارة

- API مركزية
- worker runtime
- MySQL
- Redis
- MinIO
- منظومة مراقبة

المستخدمون الرئيسيون:

- عميل
- مدير منصة
- فريق دعم
- مطور backend خارجي يستهلك الـ API

3.3 المتطلبات الوظيفية

- يجب أن يسمح النظام بتسجيل العملاء وتسجيل دخولهم.
- يجب أن يدعم النظام Google OAuth عندما يكون مضبوطاً.
- يجب أن يدعم النظام تسجيل دخول الإدارة مع MFA.
- يجب أن يتيح إنشاء instances وربطها وتتبع حالتها.
- يجب أن يخزّن النظام الرسائل الخارجة والداخلة وتسليمات الـ webhook.
- يجب أن يوفّر رموز API على مستوى الحساب والـ instance.
- يجب أن يوفّر صفحات تشغيلية للإدارة تشمل المستخدمين ومساحات العمل والعاملين والرسائل والدعم والتدقيق.

3.4 المتطلبات غير الوظيفية

- يجب أن يبقى فصل الصلاحيات واضحاً بين العميل والإدارة.
- يجب أن تبقى الإعدادات والأنواع والـ contracts مشتركة وTyped.
- يجب أن تكون العمليات الطويلة غير متزامنة وتعتمد على Queue.
- يجب أن تكون القياسات والحالة الصحية متاحة عبر endpoints.
- يجب أن تكون واجهات العميل والإدارة واضحة وقابلة للاستخدام اليومي.

3.5 حالات الاستخدام

3.5.1 العميل يسجل حساباً

1. يفتح العميل التطبيق.
2. يدخل بياناته أو يستخدم Google.
3. يتم إنشاء الحساب والجلسة.
4. يصل إلى صفحات العميل.

3.5.2 العميل ينشئ instance ويستخدم الـ API

1. يسجّل العميل الدخول.
2. ينشئ instance جديدة.
3. يفتح صفحة الـ instance.
4. يربط حساب WhatsApp.
5. يحصل على `public instance ID` والـ token.
6. يستخدم مسارات الـ API من نظام backend.

3.5.3 الإدارة تحقق في مشكلة

1. يدخل المسؤول إلى صفحة الرسائل أو الـ instance.
2. يراجع الرسائل أو محاولات التسليم أو الـ webhook.
3. يربط المعلومات مع السجلات والأحداث.
4. يتخذ إجراء دعم أو تشغيل.

3.6 قصص المستخدم

- كعميل، أريد إنشاء instance بسرعة حتى أبدأ التكامل.
- كعميل، أريد رؤية الـ token والـ instance ID حتى أستخدم الـ API.
- كمسؤول، أريد البحث في الرسائل والويبهوكات عالمياً حتى أحقق في المشاكل.
- كمسؤول، أريد رؤية العاملين وحالاتهم حتى أتابع الأعطال التشغيلية.

3.7 معايير القبول

- يمكن للعميل التسجيل وتسجيل الدخول والوصول إلى لوحة التحكم.
- يمكنه إنشاء instance ورؤية حالتها وحالة الربط.
- تظهر وثائق الـ API وأمثلة الاستخدام.
- يمكن للإدارة الدخول والوصول إلى صفحات التشغيل الرئيسية.
- تتوفر endpoints للحالة الصحية والقياسات.

4 توثيق المنتج والتصميم

4.1 وثيقة معمارية النظام

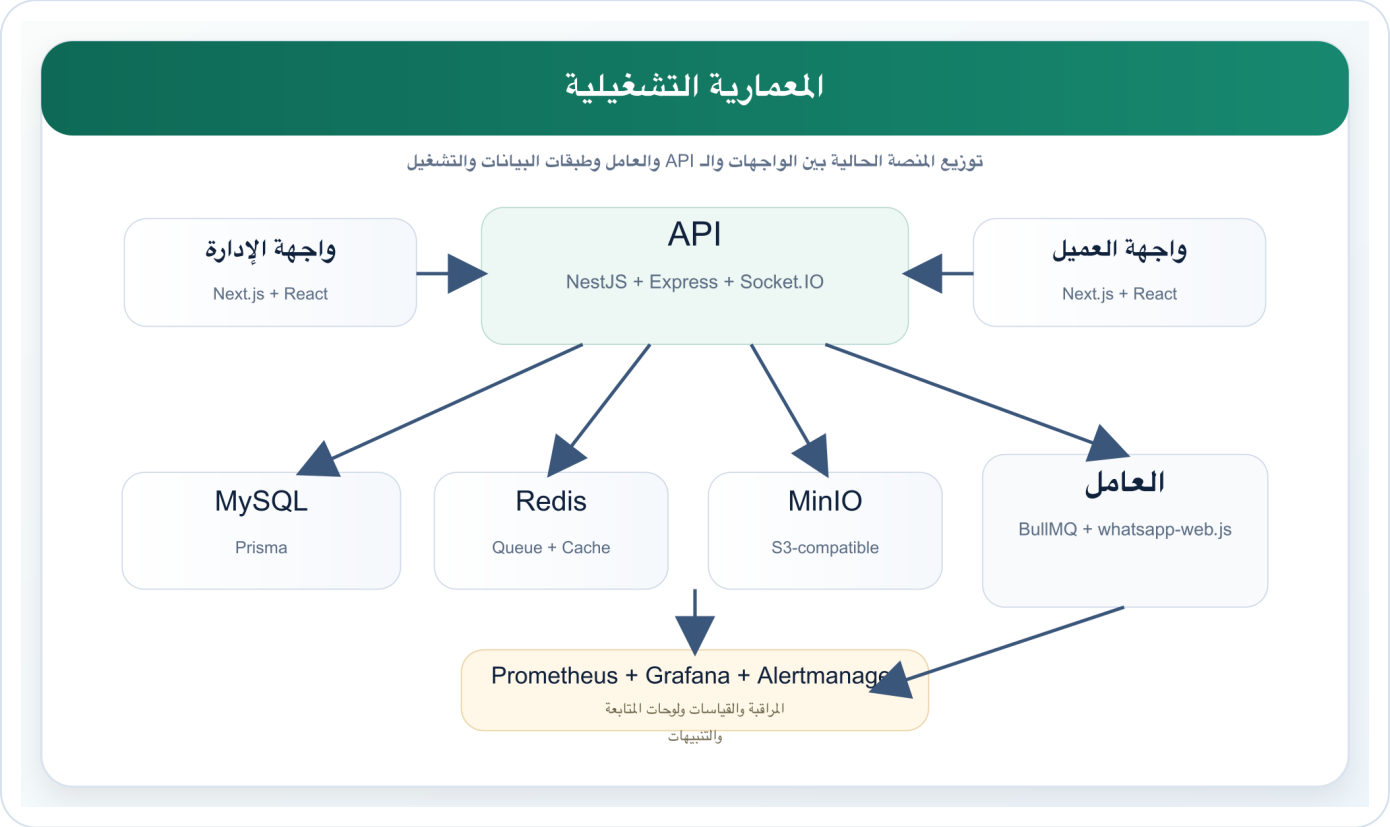
يمتلك النظام بنية مؤلفة من أربع تطبيقات تشغيلية رئيسية:

- `Customer Web`
- `Admin Web`
- `API`
- `Worker`

ويرتكز على الحزم المشتركة التالية:

- `packages/config`
- `packages/contracts`
- `packages/db`
- `packages/ui`

4.1.1 مخطط المعمارية التشغيلية



مخطط المعمارية التشغيلية

4.2 التصميم عالي المستوى HLD

يوزّع النظام المسؤوليات على ثلاث طبقات رئيسية:

1. **واجهات الاستخدام**: العميل والإدارة
2. **طبقة التحكم**: API
3. **طبقة التنفيذ**: worker + queue + runtime

كما توجد نطاقات بيانات رئيسية:

- الهوية والوصول
- المستأجرون ومساحات العمل
- الـ instances والحالة التشغيلية
- الرسائل والتسليمات
- الدعم والتدقيق والعاملون

4.3 التصميم منخفض المستوى LLD

4.3.1 تطبيق العميل

المسارات الحالية تشمل:

- `/`
- `signup/`
- `messages/`
- `settings/`
- `subscription/`
- `api-documents/`
- `instances/[id]/`
- `auth/google/`

4.3.2 تطبيق الإدارة

المسارات الحالية تشمل:

- `/`
- `messages/`
- `users/`
- `workspaces/`
- `workers/`
- `workers/[id]/`
- `instances/[id]/`
- `support/`
- `audit/`

4.3.3 الـ API

تتضمن:

- مسارات `auth`
- مسارات `customer`
- مسارات `admin`
- مسارات `internal`
- مسارات `instance` العامة
- `health/` و `ready/` و `metrics/`

4.4 توثيق مخطط قاعدة البيانات

تستخدم المنصة `MySQL` مع `Prisma 7` .

الكيانات الأساسية:

- `User`
- `Workspace`
- `Membership`
- `RefreshSession`
- `ApiToken`
- `Instance`
- `InstanceSettings`
- `InstanceRuntimeState`
- `InstanceOperation`
- `InstanceLifecycleEvent`
- `OutboundMessage`
- `OutboundMessageAttempt`
- `InboundMessage`
- `WebhookDelivery`
- `WorkerHeartbeat`
- `AuditLog`
- `SupportCase`

4.4.1 ملاحظات تصميمية

- يتم حفظ الـ tokens بشكل Hash مع Prefix للعرض
- لكل instance معرف داخلي ومعرف عام للاستخدام البرمجي
- حالة التشغيل منفصلة عن بيانات الـ instance الأساسية
- سجلات التدقيق والدعم كيانات مستقلة وليست ملاحظات حرة فقط

4.5 تصميم الـ API ومواصفاته

الواجهة تعتمد على:

- `HTTP/JSON`
- إصدار `v1`
- فصل واضح بين مسارات العميل والإدارة والداخلية والعامة

أنماط المصادقة الحالية:

- جلسات dashboard
- Google OAuth للعملاء
- MFA للإدارة
- account tokens
- instance tokens

4.6 توثيق UI/UX

الاتجاه الحالي للتجربة:

- Light theme أولاً
- فصل بصري واضح بين العميل والإدارة
- Sidebar + Topbar للمسارات المصادق عليها
- صفحات استكشاف وتشغيل قابلة للمسح السريع
- شاشات signin/signup حديثة مع validation ومساعدة سياقية

4.7 Wireframes و Mockups

4.7.1 بنية صفحة تسجيل الدخول للعميل

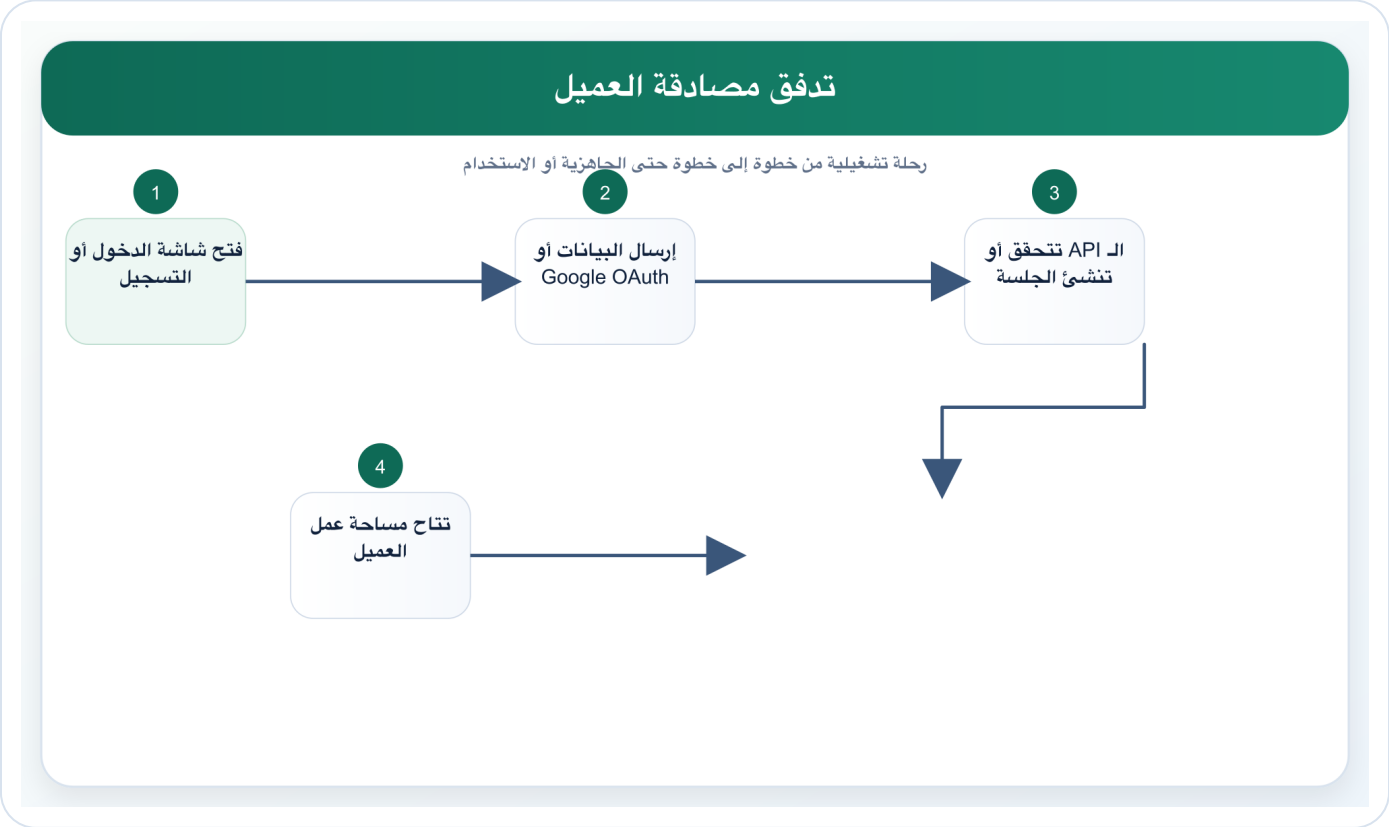
```
+-----------------------------------------------------------------+
|                                                   Brand / topbar |
+--------------------------------+-----------------------------+
|                Visual brand panel         | Signin form card |
|       email / password / Google |                             |
|                   help / submit |                             |
+--------------------------------+-----------------------------+
```

4.7.2 بنية لوحة العميل

```
+-----------------------------------------+-------------------+
|                           Sidebar           | Sticky topbar |
+-----------------------------------------+            routes |
|                  workspace         | breadcrumb / page title |
+-----------------------------------------+     account chips |
|     dashboard cards / explorers / detail |                   |
+-----------------------------------------+-------------------+
```

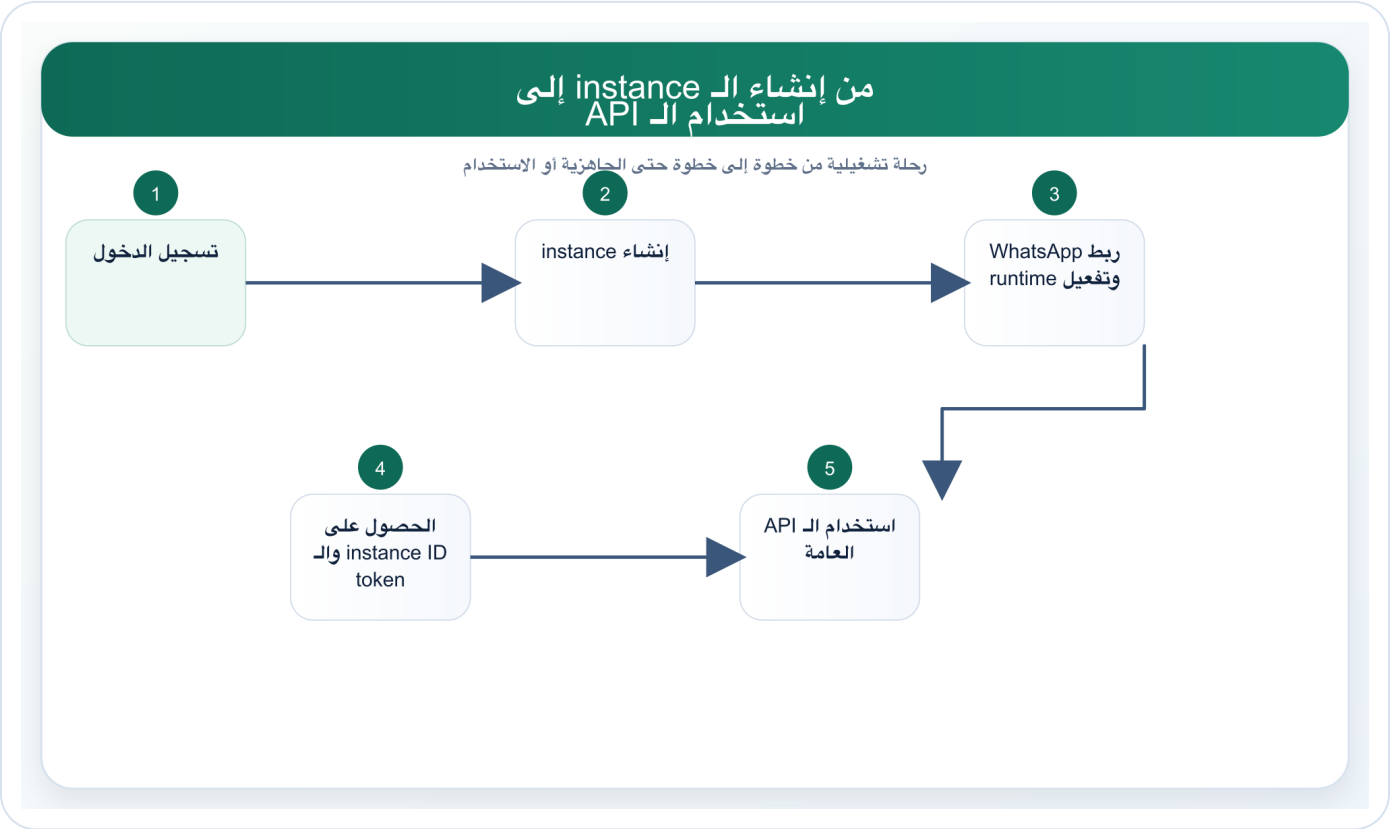
4.8 المخططات الانسيابية وتسلسل العمليات

4.8.1 تدفق مصادقة العميل



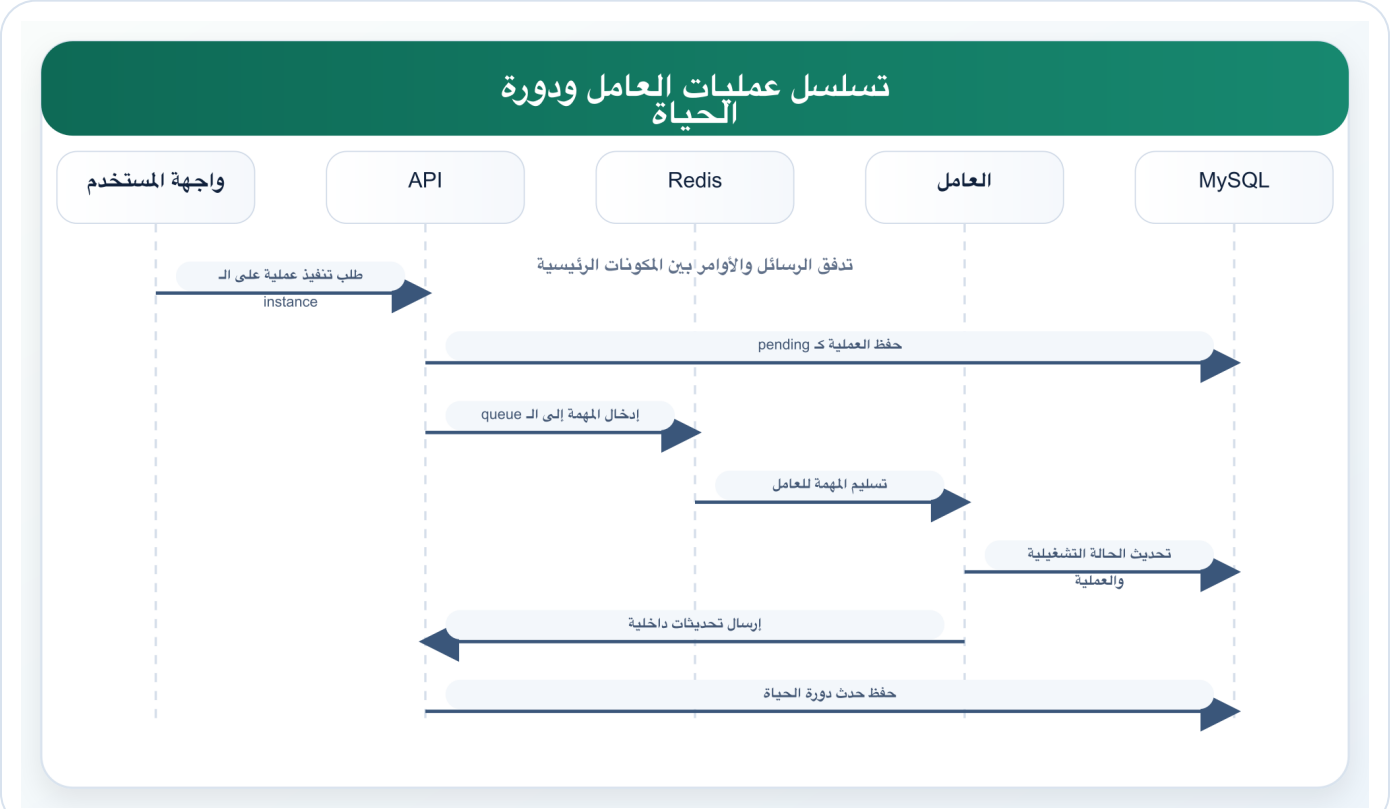
تدفق مصادقة العميل

4.8.2 من إنشاء الـ instance إلى استخدام الـ API



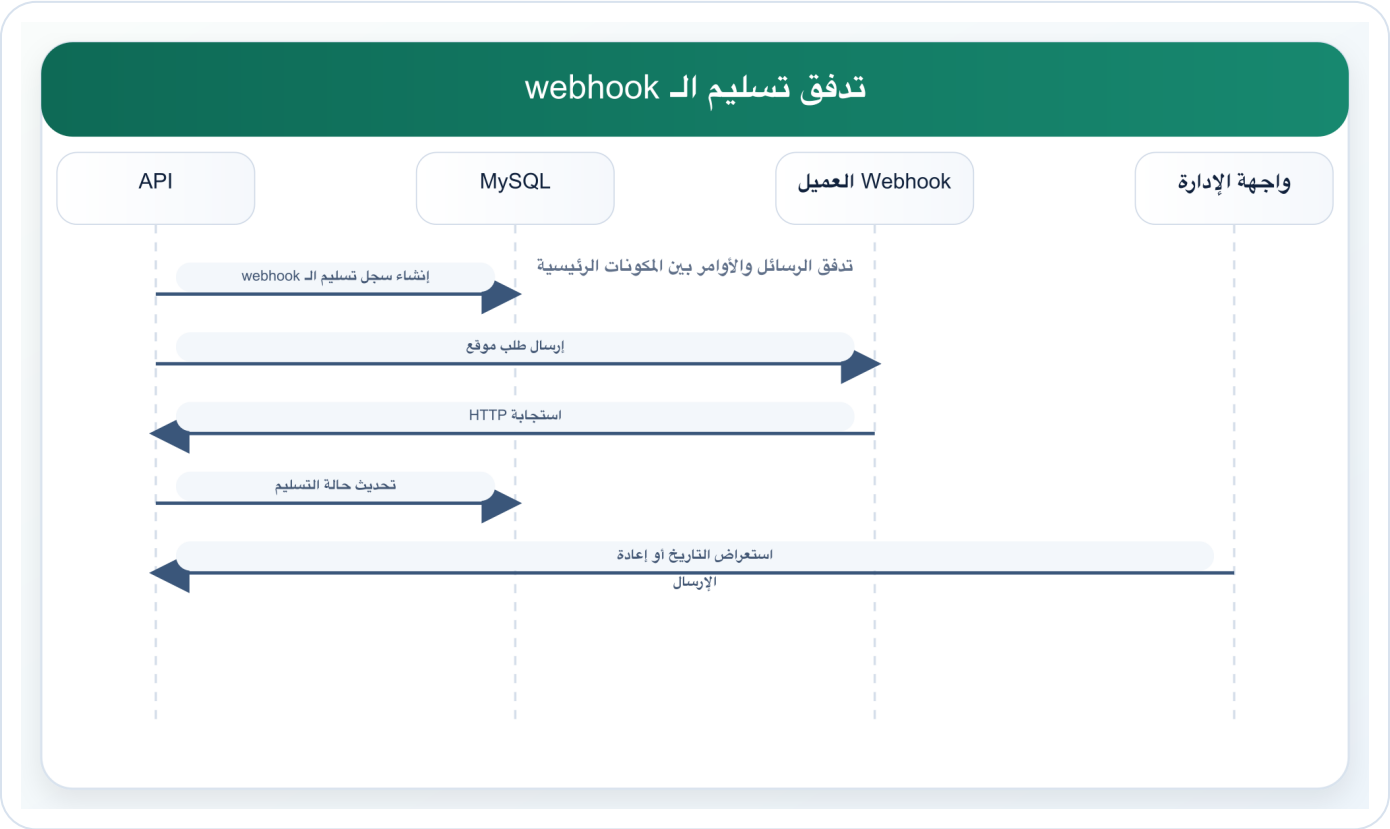
من إنشاء الـ instance إلى استخدام الـ API

4.8.3 تسلسل عمليات الـ worker ودورة الحياة



تسلسل عمليات الـ worker

4.8.4 تدفق تسليم الـ webhook



تدفق تسليم الـ webhook

5 الملاحق والمراجع

5.1 بنية المستودع

- `apps/customer-web`
- `apps/admin-web`
- `apps/api`
- `apps/worker`
- `packages/config`
- `packages/contracts`
- `packages/db`
- `packages/ui`

5.2 البنية التحتية المحلية

- MySQL 8.4
- Redis 8
- MinIO
- Prometheus

- Alertmanager
- Grafana

5.3 مراجع داعمة

- `docs/elite-message-full-module-architecture.md`
- `docs/elite-message-api-route-list.md`
- `docs/elite-message-database-schema.md`
- `docs/elite-message-mvp-phase-plan.md`

5.4 ملاحظة ختامية

هذا المستند يوثّق حالة المشروع الحالية كما هي في المستودع، ويُميز بوضوح بين ما هو منفذ فعلياً وما يزال جزئياً أو مستقبلياً. وهو مرجع موحد موجّه لفريق مختلط من أصحاب المصلحة التجاريين والمنتج والهندسة والتشغيل.