برنامج التعاملات الإلكترونية الحكومية
E-GOVERNMENT PROGRAM

تسريع عجلة التحول الرقمي بإستخدام الواجهات البرمجية

كيف يمكننا التغلب على بعض أبرز تحديات الأعمال بإتباع التطوير بمنهجية الواجهات البرمجية API-Centric

رقمي
DIGITAL SAUDI

رؤية VISION 2030
المملكة العربية السعودية
KINGDOM OF SAUDI ARABIA

7/9/2020

م. بندر إبراهيم الصانع

متخصص في مجال تطوير البرمجيات بخبرة أكثر من 13 سنة. قام بإدارة العديد من الفرق التقنية لتطوير منتجات و تطبيقات ذات حساسية عالية للأعمال. و في آخر 8 سنوات، عمل بشكل كبير على تطوير و بناء الفرق التقنية التي تعمل على تطوير البرمجيات بإتباع منهجية الـ API-Centric.

مهتم بـ

- التحول الرقمي وحل مشاكل الأعمال باستخدام التقنية
- بناء وقيادة الفرق التقنية متعددة الوظائف وذاتية الإدارة
- منهجية الآجايل وتطوير البرمجيات بالطرق الحديثة

المحاور

- تعريف التحول الرقمي و الفرق بينه وبين الأتمتة
- الطريقة التقليدية لتطوير الأنظمة التقنية و سلبياتها
- الطريقة الحديثة لتطوير الأنظمة التقنية و مميزاتها
- مقارنة بين الطريقة التقليدية و الحديثة
- مثال لتحدي شائع وكيف تساهم الواجهات البرمجية بحل التحدي

ماهو التحول الرقمي؟

التحول في طريقة العمل بالاستفادة من التطور التقني الكبير لخدمة المستفيدين بشكل أسرع وأفضل وذلك بإبتكار و تحسين إجراءات العمل داخلياً و طرق التواصل و إيصال الخدمات للعملاء

مثال

عمل يدوي	إجراءات تستخدم التقنية جزئياً	تحول رقمي
الذهاب لفرع الشركة		
تعبئة نموذج طلب قرض مالي • البيانات الشخصية • بيانات المؤسسة • بيانات دخل المؤسسة • بيانات الحساب البنكي و الدخل • إرفاق المستندات		
توقيع النموذج من العميل		
مراجعة صحة البيانات من قبل الموظف		
ارسال النموذج للمدير للمراجعة النهائية و الاعتماد (عن طريق مراسل)		
ارسال النموذج الى إدارة القروض للمراجعة وحساب معدل المخاطرة		
الموافقة النهائية		
تحويل القرض لحساب العميل يدوياً عن طريق الإدارة المالية		

مثال

عمل يدوي	إجراءات تستخدم التقنية جزئياً	تحول رقمي
الذهاب لفرع الشركة		
تعبئة نموذج طلب قرض مالي • البيانات الشخصية • بيانات المؤسسة • بيانات دخل المؤسسة • بيانات الحساب البنكي و الدخل • إرفاق المستندات	عن طريق الموقع	
توقيع النموذج من العميل	الذهاب لفرع الشركة لتوقيع النموذج	
مراجعة صحة البيانات من قبل الموظف	مراجعة صحة البيانات من قبل الموظف بإستخدام البرنامج التقني	
ارسال النموذج للمدير للمراجعة النهائية و الاعتماد (عن طريق مراسل)	يتم إرسال النموذج آلياً و اعتمادة من المدير عن طريق البرنامج التقني	
ارسال النموذج الى إدارة القروض للمراجعة وحساب معدل المخاطرة	يتم إرسال النموذج آلياً الى إدارة القروض للمراجعة وحساب معدل المخاطرة	
الموافقة النهائية	يتم إرسال طلب الموافقة آلياً للشخص المسؤول	
تحويل القرض لحساب العميل يدوياً عن طريق الإدارة المالية	يتم تحويل القرض لحساب العميل يدوياً عن طريق الإدارة المالية	

برنامج التعاملات الإلكترونية الحكومية
E-GOVERNMENT PROGRAM

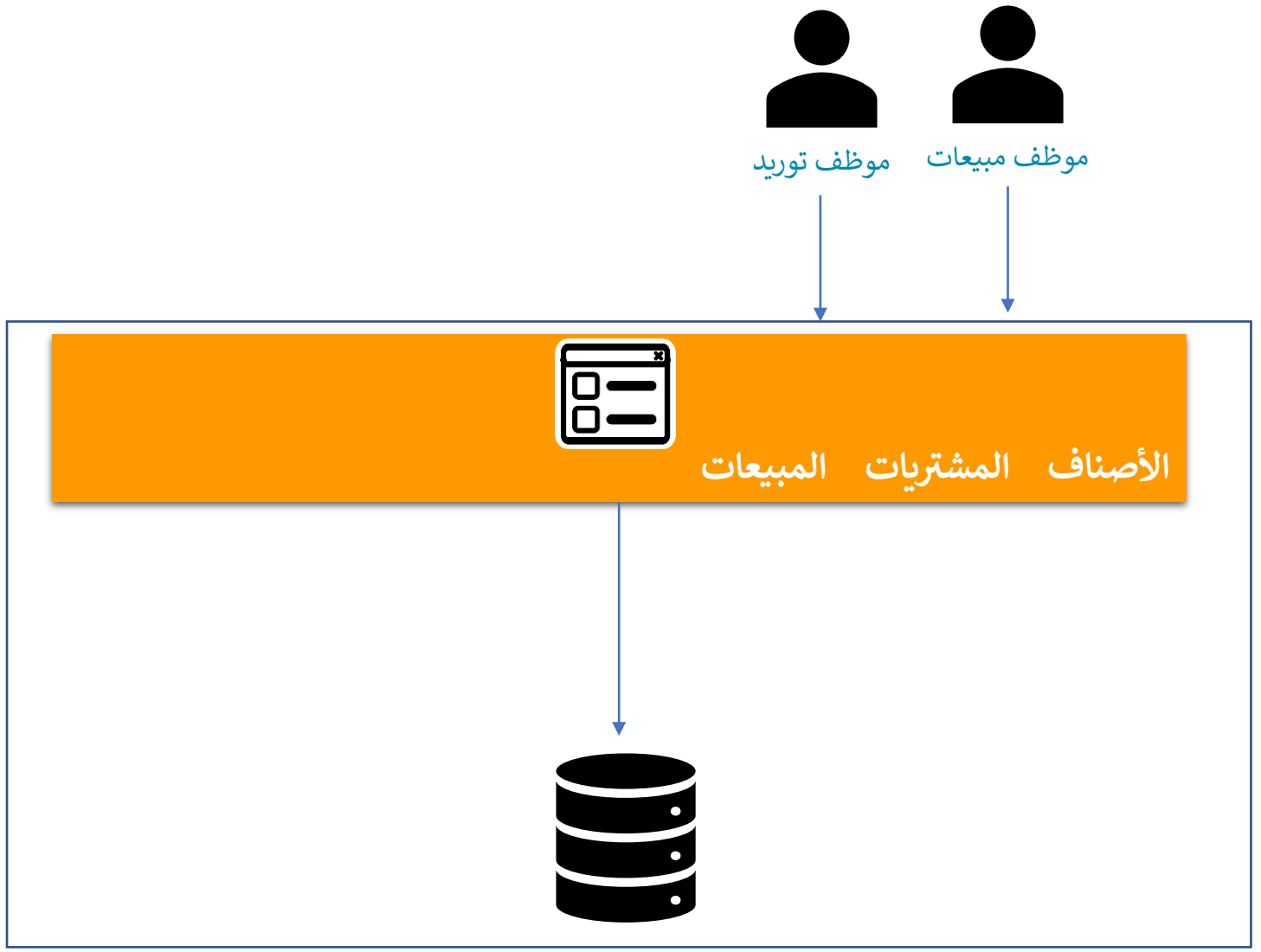
مثال

عمل يدوي	إجراءات تستخدم التقنية جزئياً	تحول رقمي
الذهاب لفرع الشركة		
تعبئة نموذج طلب قرض مالي • البيانات الشخصية • بيانات المؤسسة • بيانات دخل المؤسسة • بيانات الحساب البنكي و الدخل • إرفاق المستندات	عن طريق الموقع	عن طريق الموقع • يتم جلب البيانات الشخصية آلياً من وزارة الداخلية • يتم جلب بيانات المؤسسة آلياً من وزارة التجارة • بيانات دخل المؤسسة آلياً من البنك • يتم الإستعلام آلياً عن بيانات الدخل • ~~إرفاق المستندات~~
توقيع النموذج من العميل	الذهاب لفرع الشركة لتوقيع النموذج	
مراجعة صحة البيانات من قبل الموظف	مراجعة صحة البيانات من قبل الموظف بإستخدام البرنامج التقني	
ارسال النموذج للمدير للمراجعة النهائية و الاعتماد (عن طريق مراسل)	يتم إرسال النموذج آلياً و اعتمادة من المدير عن طريق البرنامج التقني	
ارسال النموذج الى إدارة القروض للمراجعة وحساب معدل المخاطرة	يتم إرسال النموذج آلياً الى إدارة القروض للمراجعة وحساب معدل المخاطرة	يقوم النظام آلياً بحساب معدل المخطارة بإستخدام خوارزميات حساب المخطارة و بيانات من شركة سمة و التاريخ المالي لصحاب الطلب
الموافقة النهائية	يتم إرسال طلب الموافقة آلياً للشخص المسؤول	تتم الموافقة آلياً على طلب القرض
تحويل القرض لحساب العميل يدوياً عن طريق الإدارة المالية	يتم تحويل القرض لحساب العميل يدوياً عن طريق الإدارة المالية	يتم التحويل المالي آلياً بالربط المباشر مع البنك.

ماهي الطريقة التقليدية
Monolithic with no APIs

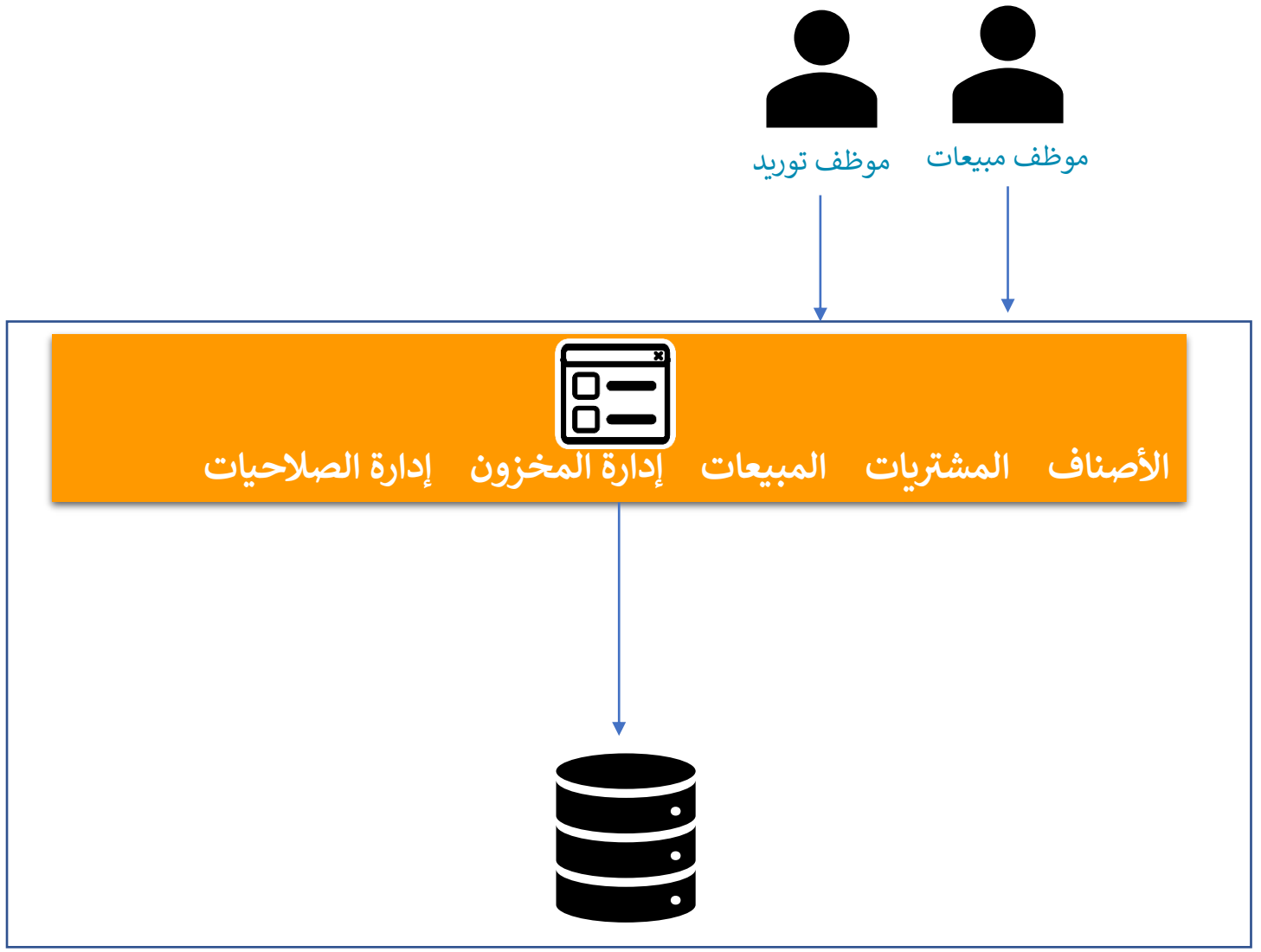
الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



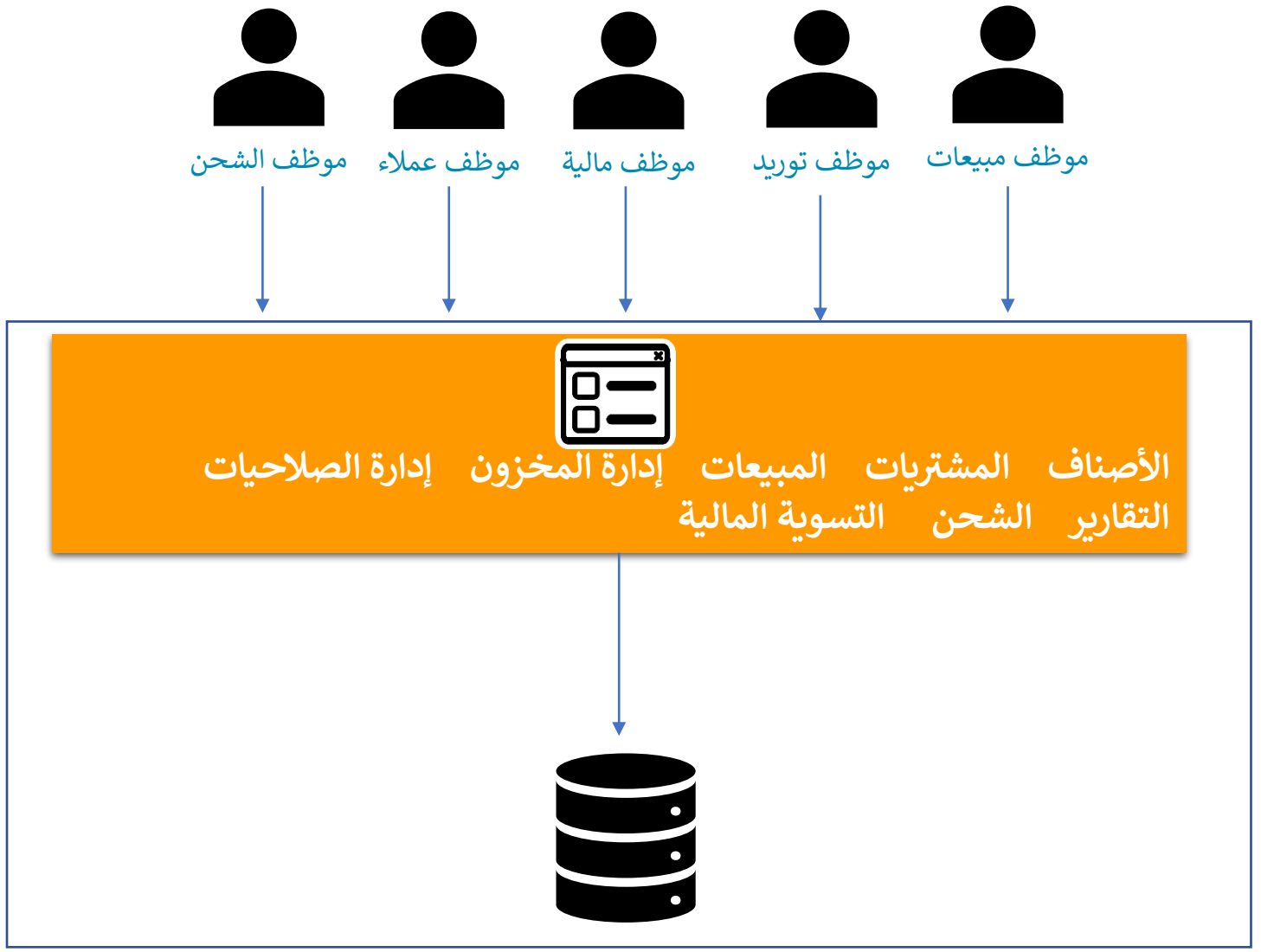
الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



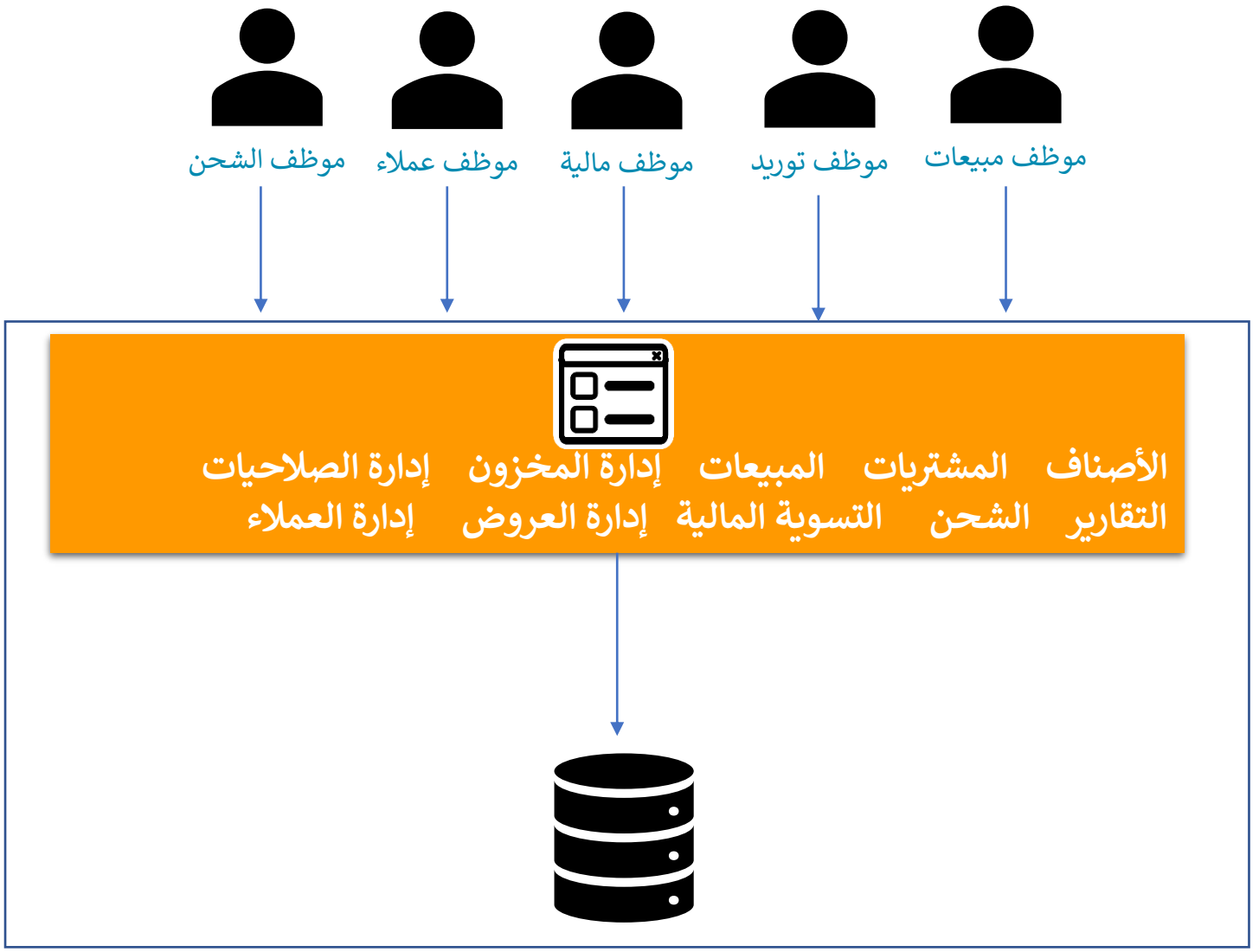
الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



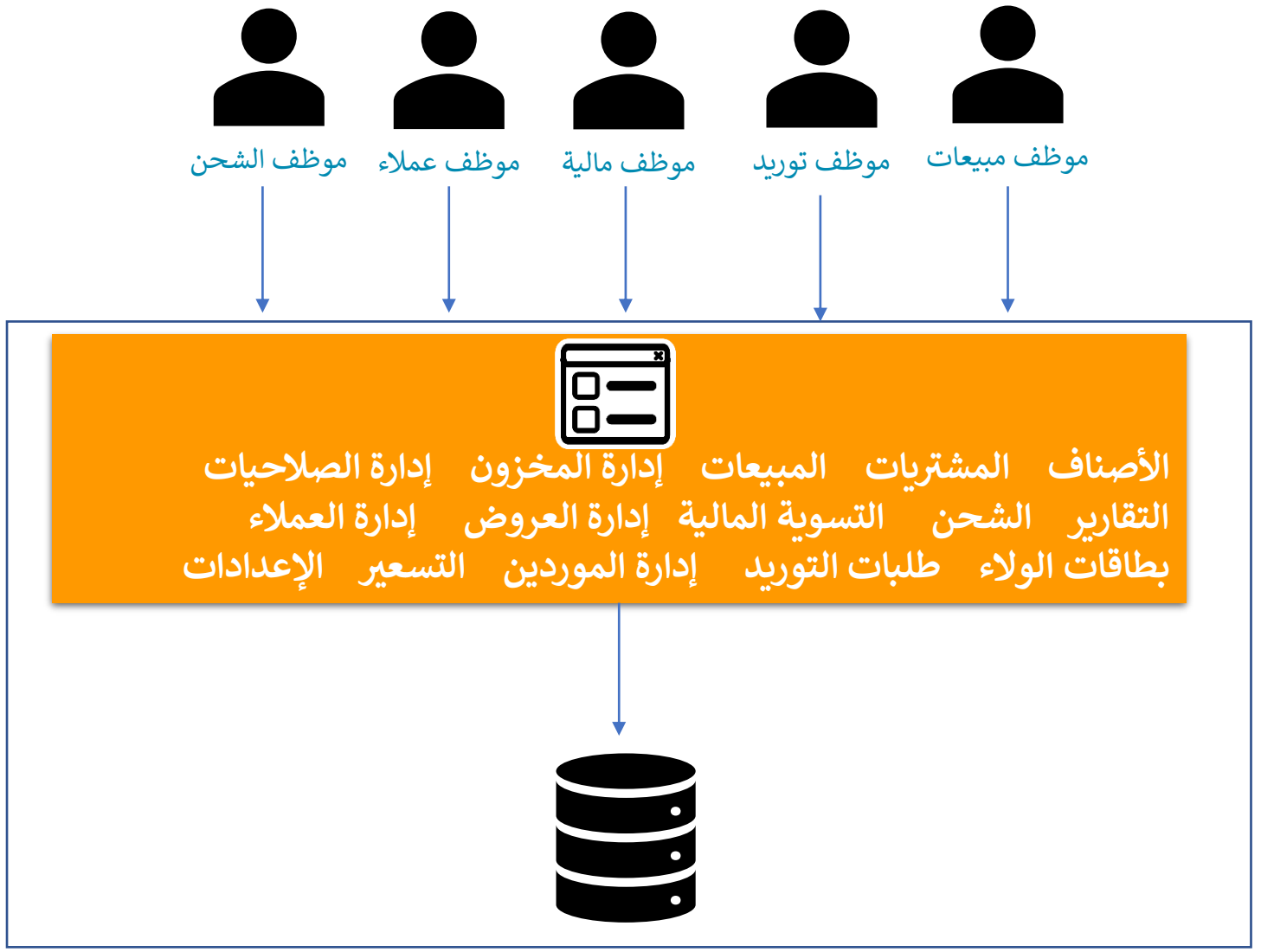
الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



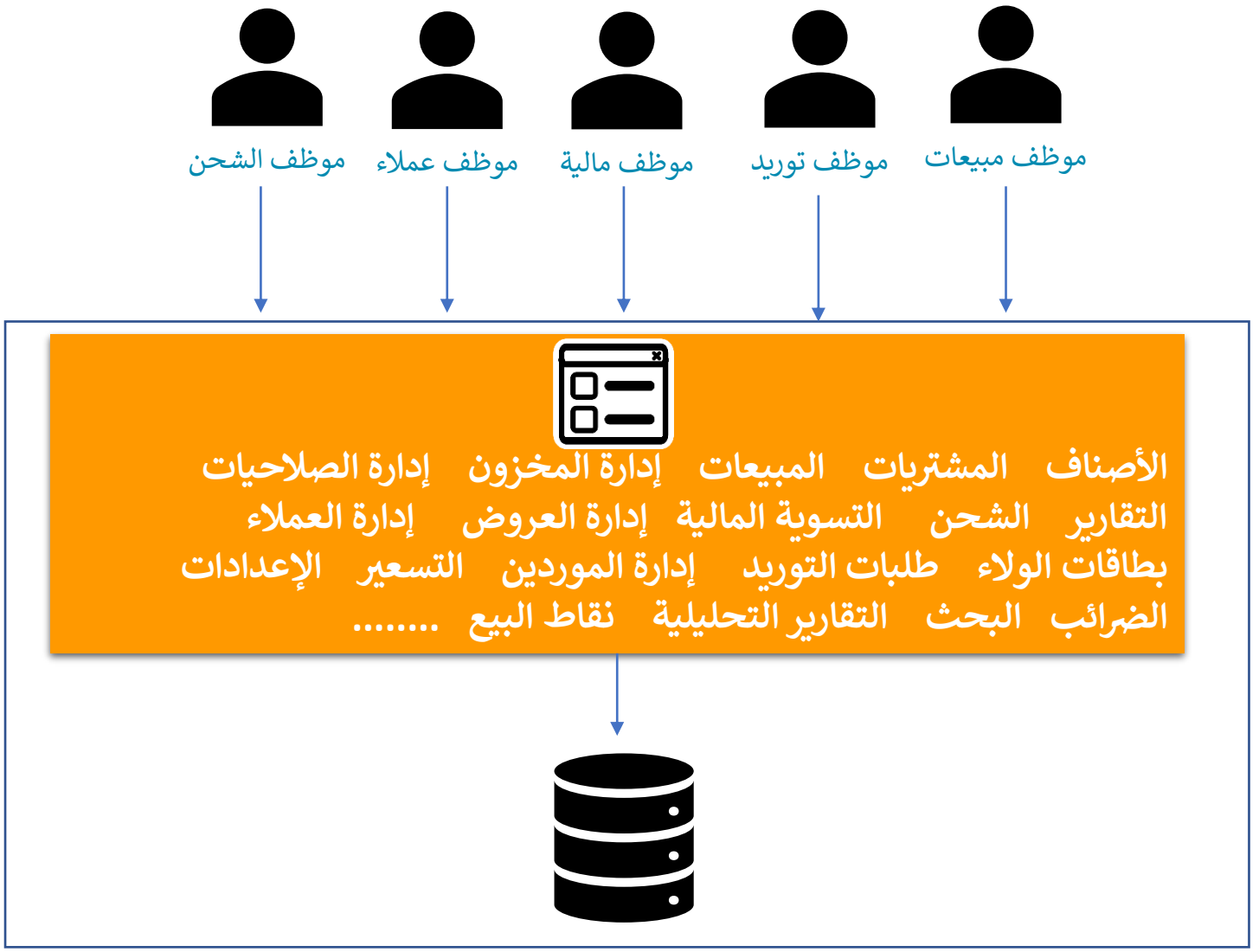
الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs



الطريقة التقليدية لتطوير أنظمة الأعمال

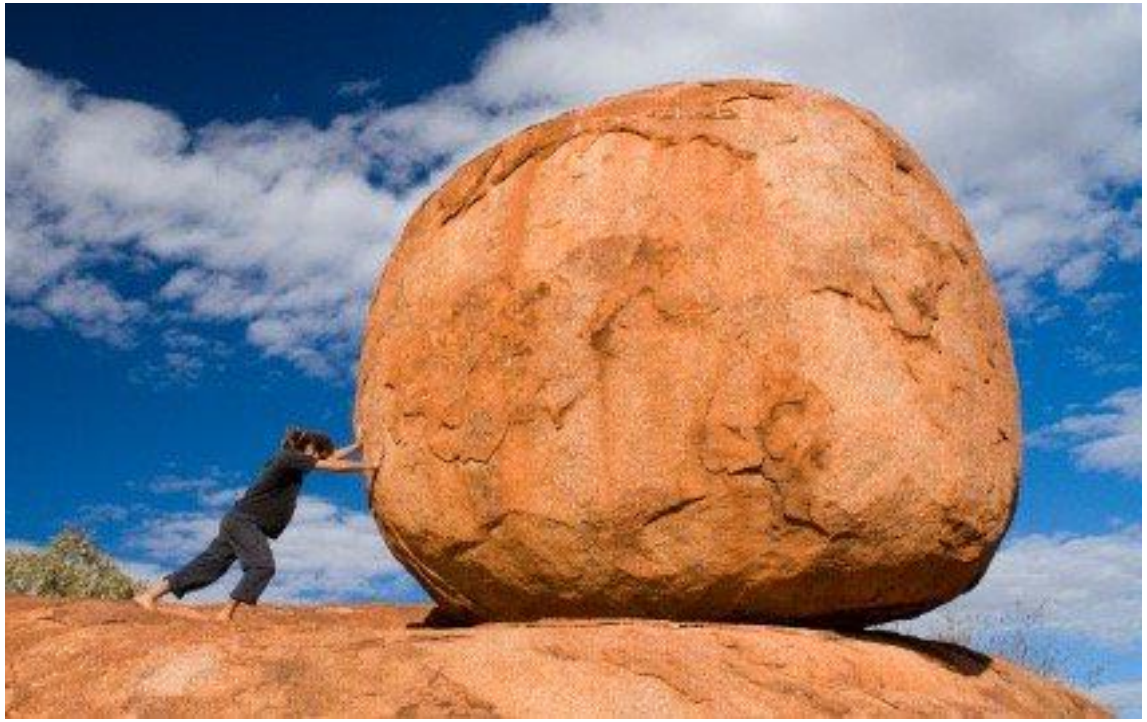
Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه



الطريقة التقليدية لتطوير أنظمة الأعمال

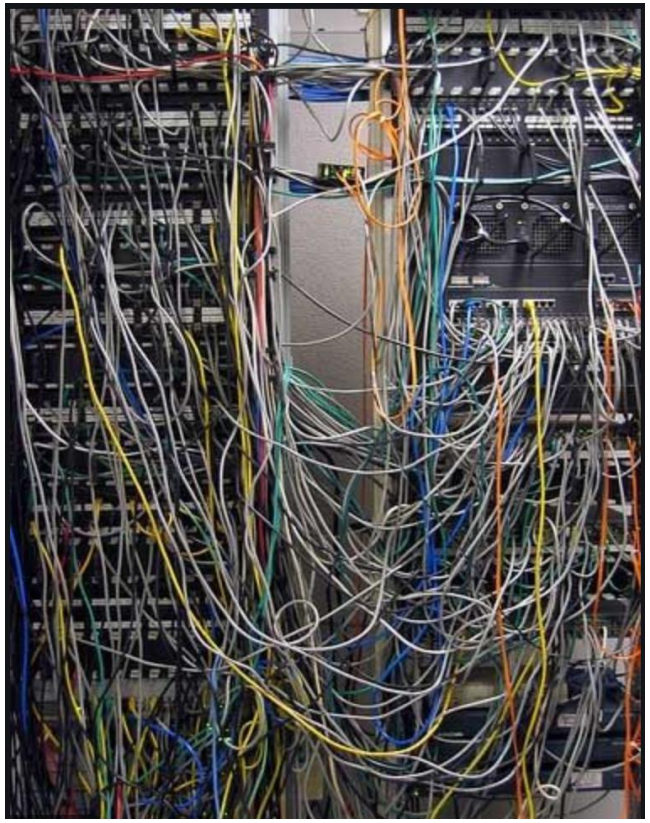
Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام



الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق

الطريقة التقليدية لتطوير أنظمة الأعمال

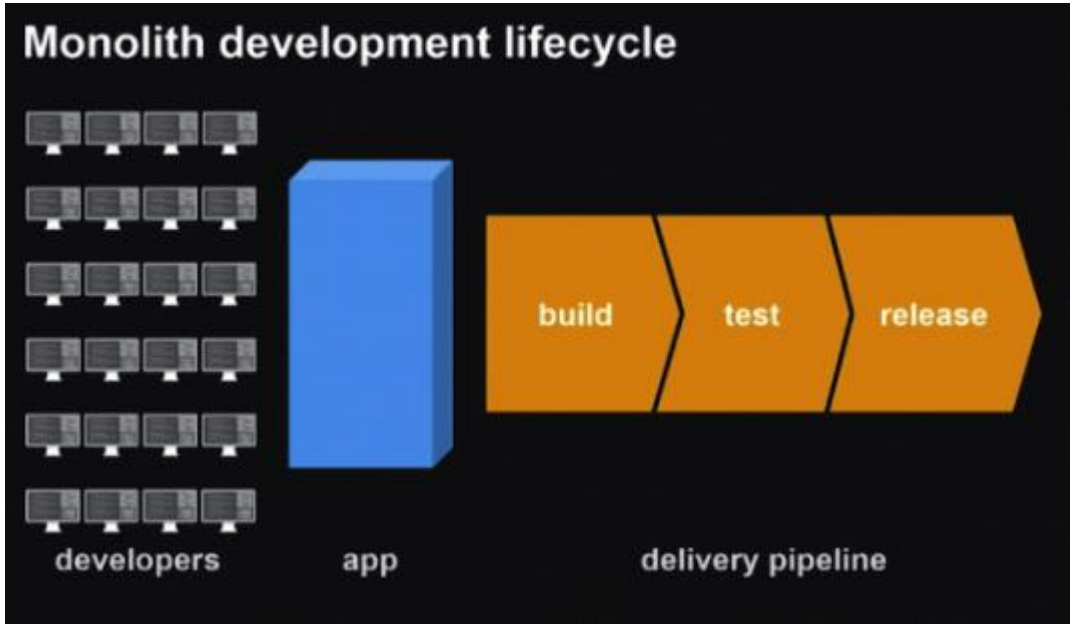
Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment



الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment
- مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment
- مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر
- الاعتمادية على فريق التطوير الداخلي

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment
- مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر
- الاعتمادية على فريق التطوير الداخلي
- من الصعب اختبار النظام

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment
- مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر
- الاعتمادية على فريق التطوير الداخلي
- من الصعب اختبار النظام
- صعوبة التزمين

الطريقة التقليدية لتطوير أنظمة الأعمال

Monolithic, No APIs

- النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير عليه
- لتشعب و تداخل أجزاء النظام و كبير حجمة، يصعب فهم النظام
- صعوبة إضافة مطورين الى الفريق
- يتم نشر النظام ككتلة واحدة Single package deployment
- مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر
- الاعتمادية على فريق التطوير الداخلي
- من الصعب اختبار النظام
- صعوبة التزمين
- صعوبة الربط Hard to integrate with

الطريقة الحديثة لتطوير أنظمة الأعمال

Service-Based, API-Centric

ماهي الواجهة البرمجية API؟

ماهي الواجهة البرمجية API؟

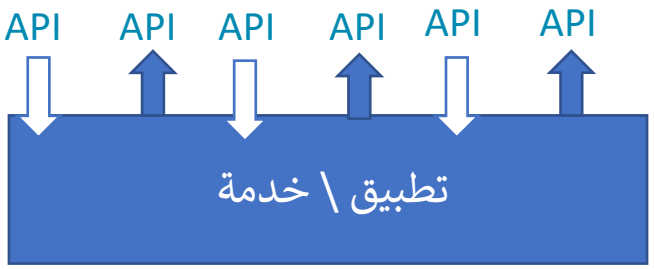
هي آلية تقنية من خلالها يمكن جلب أو إضافة أو تعديل بيانات من التطبيق

ماهي الواجهة البرمجية API
هي آلية تقنية من خلالها يمكن جلب أو إضافة أو تعديل بيانات من التطبيق

تطبيق \ خدمة

ماهي الواجهة البرمجية API

هي آلية تقنية من خلالها يمكن جلب أو إضافة أو تعديل بيانات من التطبيق



الطريقة الحديثة لتطوير أنظمة الأعمال

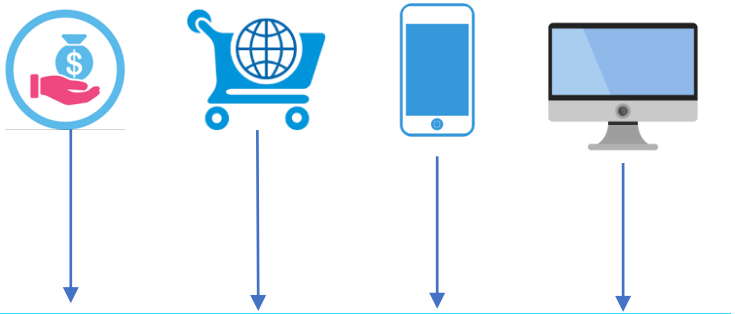
Service-Based, API-Centric

API Gateway

الأصناف	المبيعات	بطاقات الولاء	الشحن
المشتريات	نقاط البيع	إدارة الضرائب	إدارة المخزون
طلبات التوريد	إدارة العروض	إدارة العملاء	إدارة الصلاحيات
الموردين	التسعير	التسوية المالية	التقارير

الطريقة الحديثة لتطوير أنظمة الأعمال

Service-Based, API-Centric

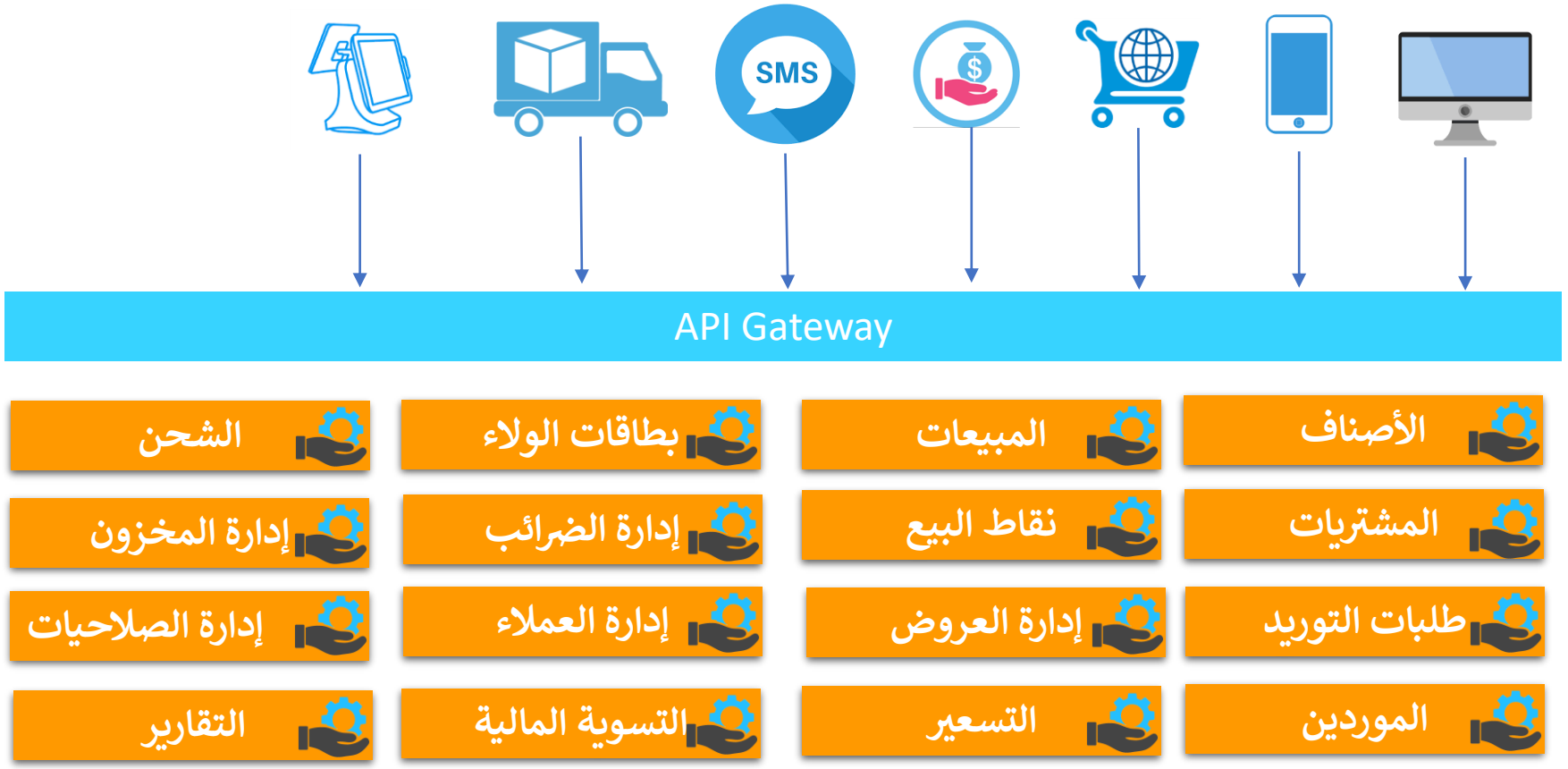


API Gateway

الأصناف	المبيعات	بطاقات الولاء	الشحن
المشتريات	نقاط البيع	إدارة الضرائب	إدارة المخزون
طلبات التوريد	إدارة العروض	إدارة العملاء	إدارة الصلاحيات
الموردين	التسعير	التسوية المالية	التقارير

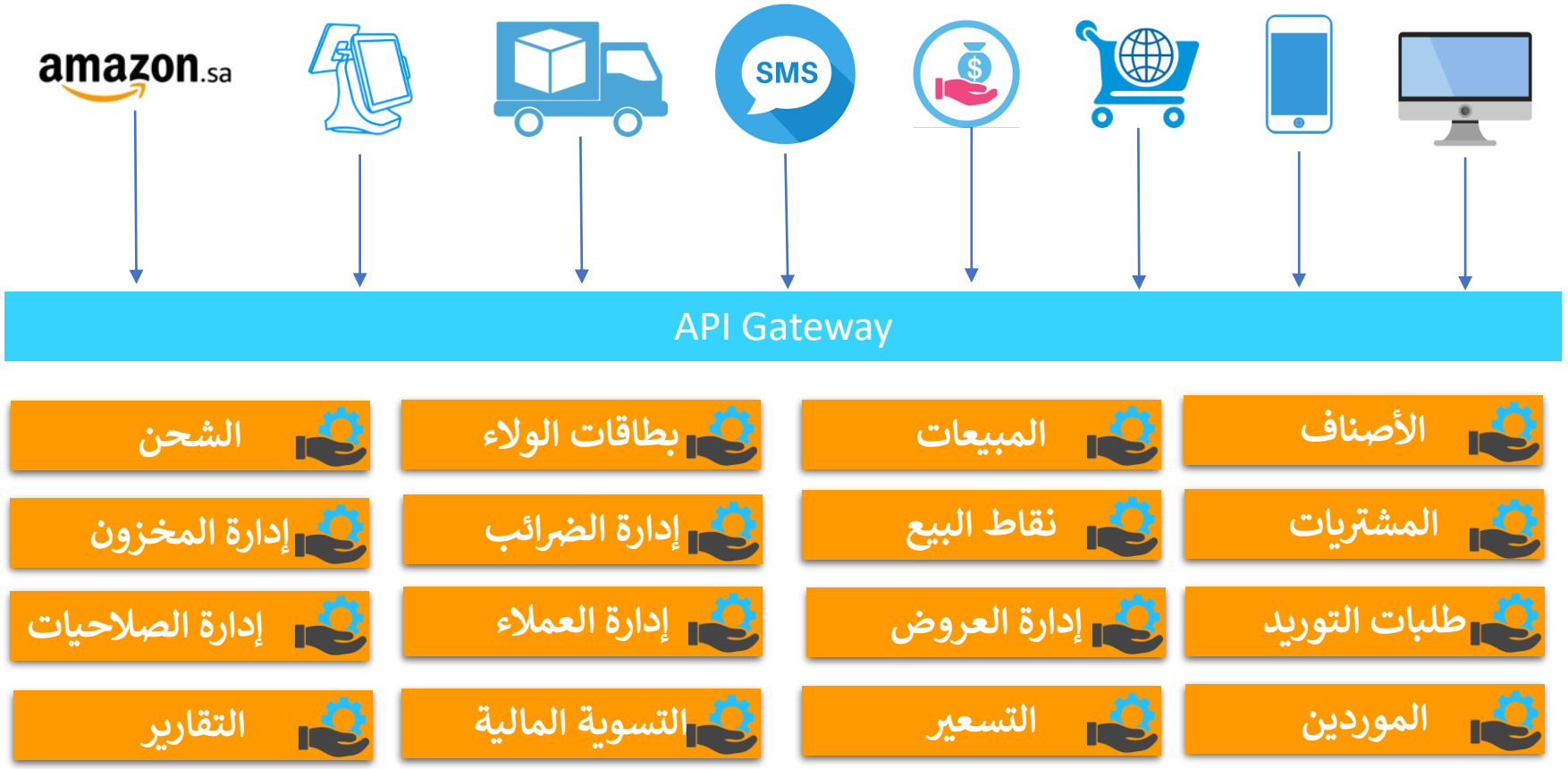
الطريقة الحديثة لتطوير أنظمة الأعمال

Service-Based, API-Centric



الطريقة الحديثة لتطوير أنظمة الأعمال

Service-Based, API-Centric



مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		
صعوبة إضافة مطورين الى الفريق		
يتم نشر النظام ككتلة واحدة Single package deployment		
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		
صعوبة إضافة مطورين الى الفريق		
يتم نشر النظام ككتلة واحدة Single package deployment		
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		
يتم نشر النظام ككتلة واحدة Single package deployment		
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

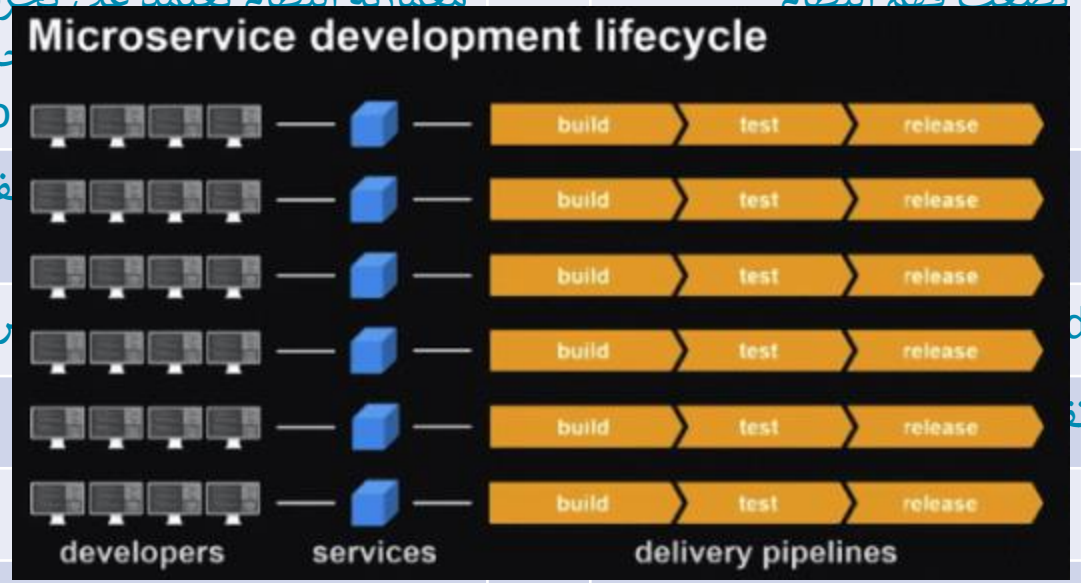
Monolith with no APIs	API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه	حتى مع مرور الوقت وكثرة الاضافات و التعديلات، يضل حجم النظام في
لتشعب و تداخل أجزاء النظام و كبر ح	لى خدمات صغيرة او متوسطة مكونات يضل النظام سهل الفهم
صعوبة إضافة مطورين الى الفريق	اء فرق جديدة لتتولى مكونات او
يتم نشر النظام ككتلة واحدة	yment
مع مرور الوقت، سرعة التطوير على الن	
من الصعب اختبار النظام	
الاعتمادية على فريق التطوير الداخلي	
صعوبة الربط	

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		Service-based API
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		يتم نشر النظام كمكونات صغيرة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم، كل خدمة و المكونات يضل النظام سهل الفهم Mono
صعوبة إضافة مطورين الى الفريق		فريق و إنشاء فرق جديدة لتتولى مكونات او
يتم نشر النظام ككتلة واحدة deployment		رة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تق		
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		



مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		يتم نشر النظام كمكونات صغيرة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		سرعة التطوير عالية حتى مع مرور الوقت وكثرة الإضافات و التعديلات
من الصعب اختبار النظام		
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

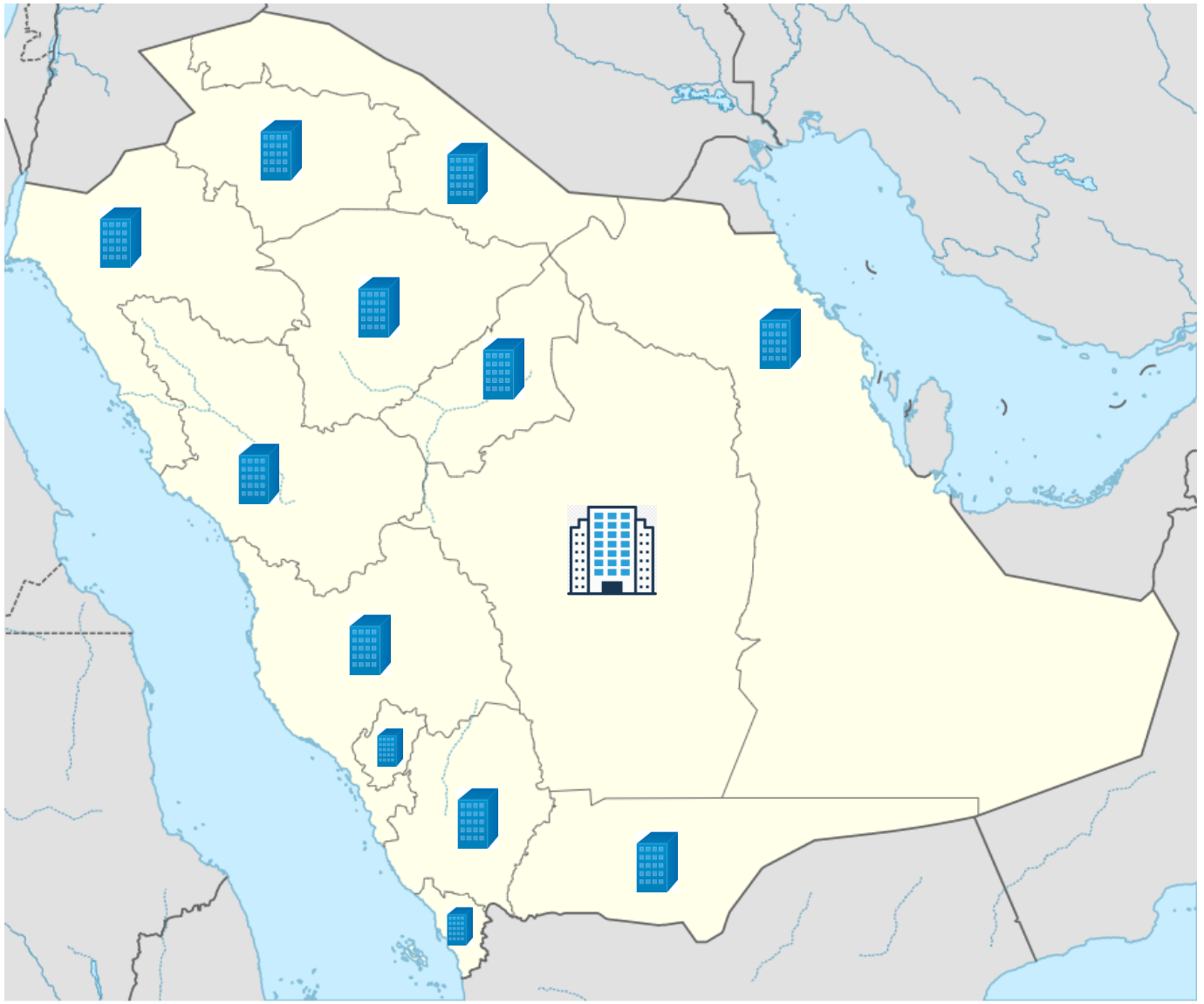
Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		يتم نشر النظام كمكونات صغيرة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		سرعة التطوير عالية حتى مع مرور الوقت وكثرة الإضافات و التعديلات
من الصعب اختبار النظام		من السهل اختبار النظام
الاعتمادية على فريق التطوير الداخلي		
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		يتم نشر النظام كمكونات صغيرة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		سرعة التطوير عالية حتى مع مرور الوقت وكثرة الإضافات و التعديلات
من الصعب اختبار النظام		من السهل اختبار النظام
الاعتمادية على فريق التطوير الداخلي		يمكن بسهولة التعاقد مع شركاء و شركات لتطوير أجزاء من النظام
صعوبة الربط		

مقارنة بين الطريقة التقليدية و الطريقة الحديثة للتطوير

Monolith with no APIs		API-based services
النظام يبدأ صغيراً، لكن مع مرور الوقت يكبر حجم النظام بشكل كبير جداً و يصعب التطوير و التعديل عليه		حتى مع مرور الوقت وكثرة الإضافات و التعديلات، يضل حجم النظام في النطاق المقبول
لتشعب و تداخل أجزاء النظام و كبر حجمة، يصعب فهم النظام		معمارية النظام تعتمد على تجزئة النظام الى خدمات صغيرة او متوسطة الحجم لذلك مهما تعددت الخدمات و المكونات يضل النظام سهل الفهم مقارنةً بال Monolith application
صعوبة إضافة مطورين الى الفريق		من السهل إضافة أعضاء الى الفريق و إنشاء فرق جديدة لتتولى مكونات او خدمات معينة.
يتم نشر النظام ككتلة واحدة Single package deployment		يتم نشر النظام كمكونات صغيرة او متوسطة الحجم، كل خدمة على حده
مع مرور الوقت، سرعة التطوير على النظام تقل و المشاكل تكثر		سرعة التطوير عالية حتى مع مرور الوقت وكثرة الإضافات و التعديلات
من الصعب اختبار النظام		من السهل اختبار النظام
الاعتمادية على فريق التطوير الداخلي		يمكن بسهولة التعاقد مع شركاء و شركات لتطوير أجزاء من النظام
صعوبة الربط		سهولة الربط



التحديات

- أكثر من 80% من الخدمات تقدم عن طريق الفروع. حيث ان الحظور الشخصي للمكتب هو القناة الرئيسية للحصول على الخدمات
- طوابر طويلة و معدل وقت الانتظار للحصول على الخدمة 30 دقيقة.
- 20. % من الخدمات تتطلب تجهيز أوراق و إثباتات من عدة جهات حكومية
- 15% من الخدمات تحتاج عمل إداري مكتبي و مخاطبات لجهات خارجية
  - معدل إنجاز هذه الخدمات 25 يوم
  - نقص أي اوراق يتسبب في رفض خدمة المستفيد و طلب تكلمة الأوراق والرجوع مرة أخرى
- المخاطبات الإدارية بين الأفرع يستغرق أسابيع فضلاً عن حالات ضياع لبعض المعاملات
- بعض مباني الأفرغ قديمة (مواقف، أثاث، مكاتب)وغير مناسبة لإستقبال المستفيدين
- يوجد تحدي في إدارة الموارد البشرية و تأهيلهم
- نسبة رضى العملاء عن الخدمات المقدمة أقل من 30%

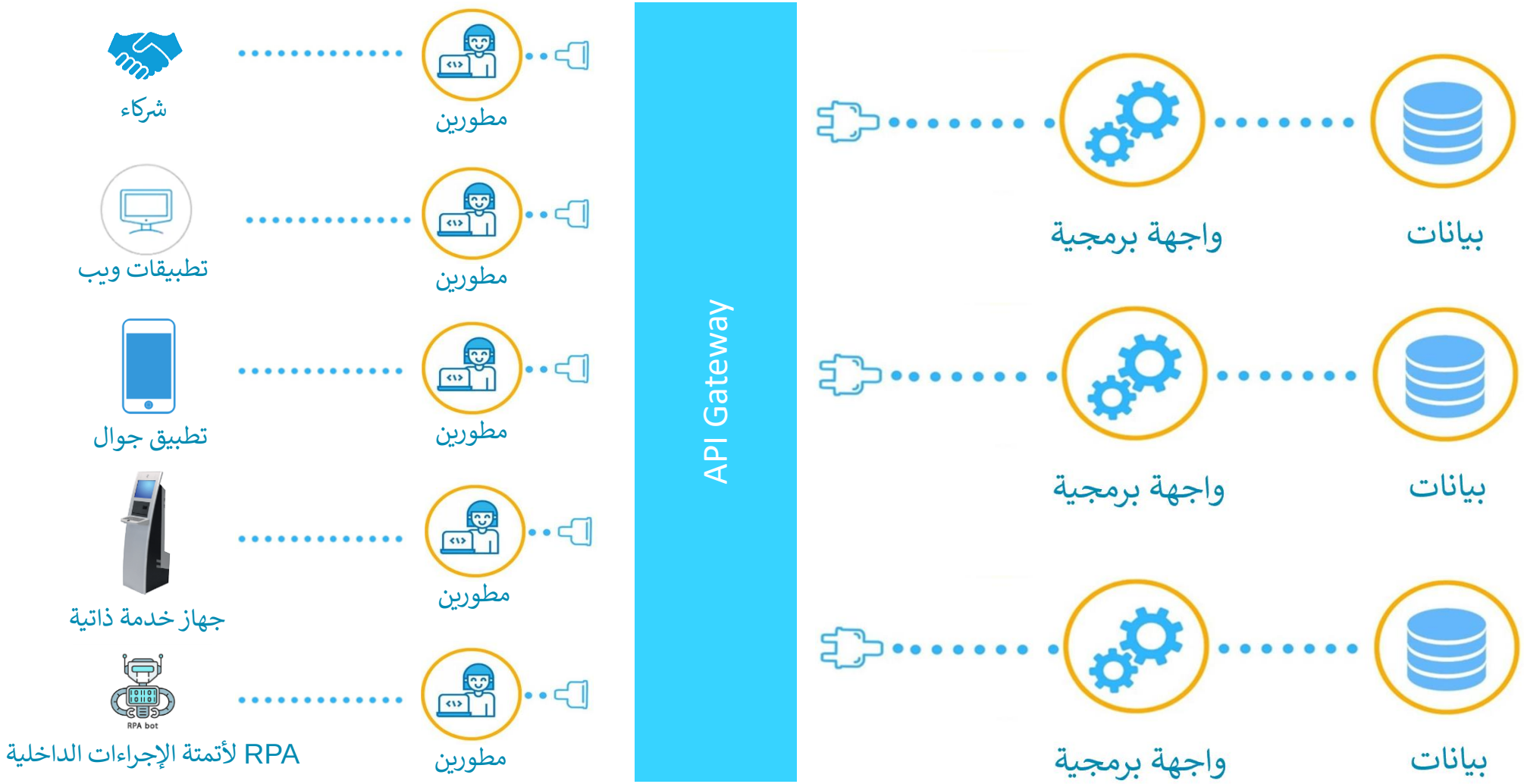
الطموح

- تقديم كل الخدمات بلا إستثناء عن طريق قنوات مختلفة لكي يختار المستفيد القناة المناسبة لة:
  - الموقع على الشبكة العنكبوتية
  - تطبيق الجوال
  - الشركاء
  - أجهزة الخدمة الذاتية
- الربط مع الجهات الحكومية ذات العلاقة لجلب المعلومات للتخلص من الحاجة لطلب أوراق و إثباتات من المستفيدين
- أتمتة الإجراءات الداخلية بأكبر قدر ممكن
- إعادة هندسة الاجراءات لتتناسب مع التحول الرقمي

آلية حل التحديات

- تحويل كل الأنظمة الرئيسية الى API Enabled system
- بناء شراكات مع شركات لتقيم الخدمات بالنيابة
  - الشركاء يوفرون هذه الخدمات عن طريق مكاتب تابعة لهم في المولات الشهيرة بالمملكة خلال الفترات الصباحية و المسائية
- إعادة هندسة الإجراءات لتتناسب مع أتمتة الإجراءات و الخدمات.
- بناء بوابة و تطبيق إلكتروني و التعامل معهما على أنهما فروع المستقبل
- تطوير جهاز خدمة ذاتية مزود ببصمة و شاشة تلفزيونيه لتقديم الخدمات التي تستوجب حضور المستفيد بنفسية
- الإستفادة بأكبر قدر ممكن من تقنية الـ RPA لأتمتة الأعمال الداخلية.

آلية حل التحديات



آلية حل التحديات

نموذج عمل حديث

طرق جديدة للتواصل للوصول للمستفيدين

منتجات جديدة

كفاءة أعلى في الأداء

؟

شكراً لكم..

م. بندر ابراهيم الصانع
Bandar@balsanie.com