

# دليل المطور (العربية)

## (1) لمحة عامة

تطبيق DeliveryApp هو تطبيق أحادي الطبقات (Layered Monolith) مبني على .NET 9 وإطار ABP مع Entity Framework Core لقاعدة بيانات SQL Server. يوفر واجهات ويب (MVC/Razor) ومضيف HTTP API، إضافة إلى مشاريع Blazor. يعتمد على OpenIddict للمصادقة و Stripe للدفع وإرسال رسائل البريد عبر SendPulse والإشعارات الفورية عبر Firebase.

التقنيات الأساسية:

- NET 9 / ASP.NET Core / ABP 9.x. (هوية، أذونات، إعدادات، إدارة مستأجرين، OpenIddict)
- EF Core + SQL Server
- واجهات: MVC/Razor + Blazor
- المصادقة: OpenIddict (OIDC/OAuth2) + تفويض بالأدوار
- التكاملات: Stripe, SendPulse, Firebase
- السجلات: Serilog (ملفات + وحدة التحكم)

## (2) بنية الحل

- DeliveryApp.Domain و DeliveryApp.Domain.Shared: الكيانات، الثوابت
- DeliveryApp.Application و DeliveryApp.Application.Contracts: خدمات التطبيق وواجهاتها و DTOs
- DeliveryApp.EntityFrameworkCore: سياق قاعدة البيانات وتعيينات EF Core
- DeliveryApp.Web: تطبيق MVC/Razor واللوحه الإدارية
- DeliveryApp.HttpApi + DeliveryApp.HttpApi.Host: واجهات REST والمضيف
- مشاريع Blazor: DeliveryApp.Blazor.\*
- الاختبارات: ضمن /test

## (3) نموذج المجال (مختصر)

الجدور المجمعة (Aggregate Roots): Restaurant, MealCategory, Order, SpecialOffer, Advertisement, AdRequest, SystemSetting, ChatSession, ChatMessage وغيرها.

كيانات المدفوعات: PaymentTransaction, StripeCustomer, ConnectedAccount, FinancialTransaction, RestaurantPayout, CODTransaction للدفع عند التسليم.

## (4) خدمات التطبيق

موجودة ضمن src/DeliveryApp.Application/Services مثل: RestaurantAppService, OrderAppService, MealCategoryAppService, StripePaymentService, CODService, AdminUserAppService,

DeliveryPerformanceService, FirebaseNotificationService,  
.SendPulseEmailNotifier

## (5) الواجهات والتحكم بالوصول

- مسيّرات API ضمن DeliveryApp.HttpApi.Host و DeliveryApp.HttpApi
- مسيّرات الويب ضمن DeliveryApp.Web
- التفويض عبر الأدوار: admin, manager, restaurant\_owner, delivery, customer، مع استخدام [AllowAnonymous] حسب الحاجة.

## (6) الإعدادات والأسرار

- ملفات appsettings.\*.json لكل مشروع (بيئات مختلفة)
- ملف env. example: نموذجي
- يجب عدم تخزين أسرار حقيقية داخل المستودع؛ استخدم متغيرات البيئة أو مدير الأسرار.

## (7) البناء والتشغيل (محلي)

المتطلبات: .NET 9, Node 20.

- تثبيت مكتبات الواجهة (عند الحاجة):

```
abp install-libs
```

- إنشاء/تحديث قاعدة البيانات:

```
cd src/DeliveryApp.DbMigrator  
(dotnet run)
```

- تشغيل تطبيق الويب:

```
cd src/DeliveryApp.Web  
(dotnet run)
```

- تشغيل مضيف الـ API:

```
cd src/DeliveryApp.HttpApi.Host  
(dotnet run)
```

## (8) الأمن

- OpenIddict لإصدار الرموز وعملاء OIDC
- شهادات توقيع/تشفير في الإنتاج
- تفويض على مستوى مسارات المسيّرات عبر [Authorize(Roles = "...")]

## (9) التكاملات

- Stripe: معالجة الدفع والنوايا والسحوبات والتحويلات وحسابات متصلة
- SendPulse: إرسال البريد (باستخدام SendPulse:ClientId/ClientSecret)
- Firebase: إرسال الإشعارات الفورية عبر FirebaseNotificationService

## (10) النشر

راجع ملفات: DEPLOYMENT\_GUIDE.md, SMARTERASP\_DEPLOYMENT\_GUIDE.md, NETWORK\_ACCESS\_SETUP.md

## (11) ملاحظات مهمة

- قم بتدوير جميع الأسرار الموجودة في appsettings.\*.json ونقلها إلى متغيرات البيئة.
  - فعّل HTTPS، CORS المناسب، وسجلات المراجعة.
- لرؤية مخططات التدفق والمواصفات التفصيلية، راجع المستند: system-specification.ar.md