Developer Guide

1. Overview

DeliveryApp is a layered monolith built with ABP Framework on .NET 9 and Entity Framework Core. It exposes both MVC/Razor endpoints (`DeliveryApp.Web`) and a modular HTTP API host (`DeliveryApp.HttpApi.Host`). The domain is organized around restaurants, menus, orders, payments, ads, chat, and notifications.

Key stacks:

- .NET 9, ASP.NET Core, ABP Framework 9.x (Identity, OpenIddict, Tenant/Setting/Permission modules)
- EF Core (SQL Server) with a single DbContext `DeliveryAppDbContext`
- Frontends: ABP MVC/Razor and Blazor (server + tiered variants)
- Auth: OpenIddict (first-party OAuth2/OIDC), role-based authorization
- Integrations: Stripe (payments), SendPulse (email), Firebase (push notifications)
- Observability: Serilog to file and console

2. Solution Structure

- `DeliveryApp.Domain` and `DeliveryApp.Domain.Shared`: Entities, enums, constants
- `DeliveryApp.Application` and `DeliveryApp.Application.Contracts`: Application services and DTOs
- `DeliveryApp.EntityFrameworkCore`: EF Core DbContext and mappings
- `DeliveryApp.Web`: MVC/Razor application (admin + web controllers)
- `DeliveryApp.HttpApi` + `DeliveryApp.HttpApi.Host`: REST APIs and host
- `DeliveryApp.DbMigrator`: Console for migrations and seeding
- Blazor Apps: `DeliveryApp.Blazor.*` projects
- Tests: under `test/`

3. Domain Model (selected)

Aggregate roots (FullAuditedAggregateRoot):

- Restaurant, MealCategory, SpecialOffer, Advertisement, SystemSetting, AdRequest
- ChatSession, ChatMessage (Guid)

Supporting entities:

- PaymentTransaction, StripeCustomer, ConnectedAccount, FinancialTransaction, RestaurantPayout
- Review, FavoriteRestaurant, Address, Order, OrderItem, DeliveryStatus, UserPreferences, NotificationSettings, Location, CODTransaction

DbContext: `DeliveryApp.EntityFrameworkCore/EntityFrameworkCore/DeliveryAppDbContext.cs` defines DbSets and Fluent mappings, including discriminator for `IdentityUser/AppUser` and relationship configurations.

4. Application Services

Located in `DeliveryApp.Application/Services`:

- Core: RestaurantAppService, OrderAppService, MealCategoryAppService, CustomerAppService, DeliveryPersonAppService
- Admin/Reports: AdminRestaurantAppService, AdminUserAppService, DeliveryPerformance(Admin)Service, RestaurantReportService
- Payments: StripePaymentService, CODService, FinancialManagementService
- Auth/Mobile: AuthService, MobileAuthService, UserAppService
- Notifications: SendPulseEmailNotifier, FirebaseNotificationService, Restaurant(Notification/OwnerNotification)Service, NotificationAnalyticsService, SignalRNotifier
- Scheduling/Offers: OfferSchedulingService, SpecialOfferAppService, DeliveryFeeCalculationService

These inherit `ApplicationService` and declare interfaces in `Application.Contracts`.

5. HTTP API and Web Controllers

- API controllers under `src/DeliveryApp.HttpApi/Controllers` and `src/DeliveryApp.HttpApi.Host/Controllers`
- Web controllers under `src/DeliveryApp.Web/Controllers`
- Authorization attributes use roles: admin, manager, restaurant_owner, delivery, customer. Anonymous endpoints are explicitly marked `[AllowAnonymous]`.

6. Authentication & Authorization

- OpenIddict configured via appsettings and ABP modules; clients defined in `appsettings.json` with `ClientId`, `ClientSecret`, and `RootUrl`.
- Role-based guards via `[Authorize(Roles = "...")]`.
- Certificates: use production signing/encryption certs per README guidance.

7. Configuration

Environment-specific `appsettings.*.json` are present for Web, HttpApi.Host, DbMigrator, and Blazor. Also `.env` style file `env.example` documents environment variables for connection string, JWT, URLs, Redis, OAuth, SendPulse, Firebase, and ASPNETCORE_ENVIRONMENT.

Sensitive values must be provided via secrets or environment variables. Do not commit real secrets.

8. Build & Run

Prereqs: .NET 9 SDK, Node 20+

- Install UI libs (if needed):

```
abp install-libs
```

- Create/update database:

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

- Run Web (MVC/Razor):

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

- Run API Host:

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

Ports: in Development, apps bind to `http://localhost:5000`, `https://localhost:5001` unless overridden. In deployment, IIS/web.config sets bindings.

9. Database & Migrations

- `DeliveryAppDbContext` includes Identity and Tenant modules via `ReplaceDbContext` to enable cross-module queries.
- Run `DeliveryApp.DbMigrator` to apply migrations and seed data.
- SQL helpers and fix scripts are provided at repository root (e.g., `final_migration.sql`, `complete_openiddict_fix.sql`).

10. Integrations

- Stripe: `StripePaymentService`, entities in `PaymentEntities.cs`. Configure keys via environment or secrets.
- SendPulse: `SendPulseEmailNotifier` uses `SendPulse:ClientId/ClientSecret`; FromName/FromEmail defaulted.
- Firebase: `FirebaseNotificationService` reads `Firebase:ServiceAccountPath` and `Firebase:ProjectId`.

11. Security Notes

- Do not commit real secrets to `appsettings.*.json`. Use `appsettings.secrets.json` or environment variables.
- Rotate `StringEncryption:DefaultPassPhrase`, JWT secrets, and OpenIddict `ClientSecret`.
- Enforce HTTPS and set proper CORS in production.

12. Testing

Tests live under `test/` with ABP test base support. Scripts exist to test auth endpoints.

13. Deployment

- See `DEPLOYMENT_GUIDE.md`, `SMARTERASP_DEPLOYMENT_GUIDE.md`, `NETWORK_ACCESS_SETUP.md`.
- For IIS/Windows deployment scripts: various `deploy_*.ps1` and fix scripts are included.

14. Troubleshooting

Guides available: performance, memory leaks, localization, delivery fee system, notification system, security implementation, environment configuration, OpenIddict fixes, discriminator migration.

15. Roadmap Hints (Dev)

- Consolidate secrets to KeyVault/Secret Manager
- Add health checks and liveness/readiness probes
- CI/CD with migrations and smoke tests
- Improve domain validation and repository boundaries