- Clean Architecture ile uygulamanızın her bir katmanı bağımsız olacak şekilde modüler hale gelir, bu da bakım ve test işlemlerini kolaylaştırır.

- Riverpod sayesinde her bir ekranda veya özellikte state yönetimi oldukça esnek ve güçlü bir şekilde yapılabilir.

- Dashboard gibi yönetim paneli özelliklerini de presentation/dashboard katmanına dahil ederek, yönetimsel işlemleri modüler şekilde yapılandırabilirsiniz.

**Dizin Açıklamaları:**

**1. Core:**

* **utils/**: Yardımcı fonksiyonlar ve genel amaçlı araçlar (örneğin, tarih formatlayıcı, validator vb.)
* **network/**: API istemcisi ve HTTP isteklerini yönetir.
* **constants.dart**: Projeye özgü sabitler (API URL'leri, güvenlik anahtarları vb.)
* **errors.dart**: Hata yönetimi ve hata sınıfları (API'den gelen hata kodları gibi).

**2. Data:**

* **models/**: Veritabanı veya API'den alınan verileri temsil eden veri modelleri (örneğin, Product, User, Order).
* **repositories/**: Veritabanı veya API ile iletişim sağlayan sınıflar. Uygulamanın veri katmanını yönetir.
* **datasources/**: Uzak veri kaynakları, yani API ile etkileşim (örneğin, RemoteDataSource).
* **local\_datasources/**: Yerel veri kaynakları, verilerin cihazda saklanması (örneğin, SQLite veya SharedPreferences).

**3. Domain:**

* **entities/**: Temel iş nesneleri (örneğin, Product, User, Order), genellikle sadece veriyi taşıyan sınıflardır.
* **usecases/**: İş mantığını ve uygulama iş akışlarını yöneten sınıflar. Örneğin, AddProductToCartUseCase, FetchProductDetailsUseCase.
* **repositories/**: Veriyi yöneten interface'ler, veri kaynakları ile arayüz sağlar.

**4. Presentation:**

* **screens/**: Her bir ekranın UI'ını temsil eden dosyalar. Örneğin, HomeScreen, CartScreen, ProductDetailScreen vb.
* **widgets/**: Tekrar kullanılabilir UI bileşenleri, örneğin bir ürün kartı, kategori listesi veya özel butonlar.
* **dashboard/**: Yönetim paneline özgü ekranlar ve bileşenler. Ürün, sipariş ve kullanıcı yönetimi gibi bölümler.
* **state/**: Riverpod ile state yönetimi yapılır. Burada UI ile ilgili tüm veriler Riverpod sağlayıcılarıyla yönetilir. Örneğin, ürün listesi veya sepet durumu gibi.

**5. Dashboard:**

* **dashboard\_main.dart**: Yönetim panelinin ana sayfası. Kullanıcıları, siparişleri ve ürünleri yönetme alanlarını barındırır.
* **product\_management.dart**: Ürünlerin yönetildiği ekran. Ürün ekleme, silme ve düzenleme işlemleri yapılır.
* **order\_management.dart**: Siparişlerin yönetildiği ekran. Siparişler üzerine işlem yapma, durumu değiştirme vb.
* **user\_management.dart**: Kullanıcı yönetimi ekranı. Yeni kullanıcılar ekleme, kullanıcı bilgilerini düzenleme gibi işlemler yapılır.

**Riverpod Kullanımı (State Management)**

Riverpod ile **state** yönetimi oldukça esnektir ve her bir ekran ya da özellik için farklı sağlayıcılar kullanarak bağımsız bir yapı oluşturabilirsiniz.

**Bağımlılıklar**

**📦 flutter\_riverpod: ^2.0.0**

* **State management** (durum yönetimi) için kullanılır.
* Provider, StateNotifier, FutureProvider gibi yapılarla veriyi yönetmeni sağlar.
* ProviderScope ile tüm uygulamada geçerlidir.
* Modern, sade ve test edilebilir bir mimari sunar.

**📦 dio: ^5.0.0**

* HTTP istekleri (GET, POST, PUT vs.) yapmak için kullanılır.
* interceptor, timeout, cancel token gibi gelişmiş özellikler içerir.
* REST API'lerle çalışmak için en güçlü Flutter istemcilerindendir.

**📦 freezed: ^2.3.2**

* **Immutable veri modelleri** ve **union/sealed class** yapıları için kullanılır.
* Özellikle model sınıflarında copyWith, equality, toString() gibi şeyleri otomatik üretir.
* Kod tekrarını azaltır, performansı artırır.

**📦 json\_serializable: ^6.6.1**

* JSON verisini Dart sınıflarına kolayca çevirmeni sağlar (fromJson, toJson).
* API'den gelen veriyi hızlıca modele döndürmek için @JsonSerializable() anotasyonu ile çalışır.
* freezed ile çok sık birlikte kullanılır.

**📦 shared\_preferences: ^2.0.15**

* Küçük veri parçalarını (token, tema tercihi, giriş bilgisi vs.) cihazda kalıcı olarak saklar.
* key-value şeklinde çalışır. Örneğin: prefs.setString('token', 'abc123');

**📦 flutter\_hooks: ^0.18.5+1**

* Flutter'da **React Hooks** benzeri bir yapı sağlar.
* useState, useEffect, useMemoized gibi hook’larla widget lifecycle yönetimini sadeleştirir.
* StatefulWidget yazmadan state yönetimi yapabilirsin.

**📦 hooks\_riverpod: ^2.0.1**

* **Riverpod + Flutter Hooks** uyumlu çalışması için kullanılır.
* HookConsumerWidget gibi widget’larla hem Riverpod hem Hooks özelliklerini birlikte kullanmanı sağlar.
* Kodun daha kısa ve okunabilir olur.

| **ENDPOİNTLER**  **🛒 1. Ürünler (Products)** |  |  |
| --- | --- | --- |
| /products | GET | Tüm ürünleri getir |
| /products/{id} | GET | Belirli bir ürünü getir |
| /products | POST | Yeni ürün ekle *(admin)* |
| /products/{id} | PUT | Ürünü güncelle *(admin)* |
| /products/{id} | DELETE | Ürünü sil *(admin)* |

**📦 2. Kategoriler (Categories)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /categories | GET | Tüm kategorileri getir |
| /categories | POST | Yeni kategori ekle *(admin)* |
| /categories/{id} | PUT | Kategori güncelle *(admin)* |
| /categories/{id} | DELETE | Kategori sil *(admin)* |

**👤 3. Kullanıcı (Auth & Profile)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /auth/register | POST | Kayıt ol |
| /auth/login | POST | Giriş yap |
| /auth/logout | POST | Oturumu kapat |
| /user/profile | GET | Profil bilgisi getir |
| /user/profile | PUT | Profil güncelle |

**🛍️ 4. Sepet (Cart)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /cart | GET | Kullanıcının sepetini getir |
| /cart | POST | Sepete ürün ekle |
| /cart/{productId} | DELETE | Sepetten ürün çıkar |

**💳 5. Siparişler (Orders)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /orders | GET | Kullanıcının siparişleri |
| /orders | POST | Yeni sipariş oluştur |
| /orders/{id} | GET | Sipariş detayı |
| /orders/{id}/status | PUT | Sipariş durumunu güncelle *(admin)* |

**🧾 6. Ödeme (Payment)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /payment/checkout | POST | Ödeme işlemini başlat |
| /payment/status | GET | Ödeme durumu sorgula |

**📊 7. Admin Panel (Yönetimsel)**

| **Endpoint** | **Yöntem** | **Açıklama** |
| --- | --- | --- |
| /admin/stats | GET | Satış istatistikleri |
| /admin/users | GET | Tüm kullanıcıları getir |
| /admin/orders | GET | Tüm siparişleri getir |
| /admin/dashboard | GET | Genel özet (toplam satış, ürün, kullanıcı vs.) |