

FLUTTER

Katmanlı Flutter Projesi

34. hafta • Bütünleşik Proje ve Yayınlama

36 hafta, 72 bağımsız çalışan Flutter projesi, gerçek uygulama ekranları, uygulamalı dersler ve öğrenci geri bildiriminden sonra incelenebilen çözüm kaynakları.

Bu kitap güncel ders kaynaklarından otomatik üretilmiştir; özet ya da jenerik şablon değildir.

Öğrenme Haritası

Hafta	Ders	Projeler
34	Katmanlı Flutter Projesi	Mini Kütüphane • Mahalle Paylaşım Ağı

34. Hafta — Katmanlı Flutter Projesi

Bu hafta ne yapacağız?

Tek bir `main.dart` içine sıkıştırılmış örnek yerine sorumlulukları ayrılmış gerçek bir proje kuracağız. Modeli `domain`, veri erişimini `data`, kullanım kurallarını `application`, Flutter ekranını ise sunum katmanında tutacağız. Mini Kütüphane ve Mahalle Paylaşım Ağı aynı katman ilkesini farklı problemler üzerinde uygulayacak.

Kazanımlar

- Katmanların görevlerini açıklama
- Modeli Flutter widget'larından bağımsız yazma
- Depo sözleşmesiyle veri kaynağını soyutlama
- İş kurallarını servis sınıfında toplama
- Bağımlılığı üst katmandan verme
- Arama, güncelleme, boş ve yüklenme durumlarını yönetme
- Sahte/bellek deposuyla bütünleşik widget testi yapma

Önerilen klasör yapısı

```
lib/  
  domain/  
  kitap.dart  
  data/  
    kitap_deposu.dart  
  application/  
    kitap_servisi.dart  
  main.dart
```

- `domain`: Uygulamanın temel kavramları
- `data`: Verinin nereden ve nasıl alındığı
- `application`: Kullanım senaryoları ve iş kuralları
- `main.dart`: Tema, ekran ve kullanıcı etkileşimi

Çözümlü uygulama — Mini Kütüphane

Domain modeli

```
class Kitap {
  Kitap({
    required this.id,
    required this.ad,
    required this.yazar,
    this.okunacak = false,
  });

  final int id;
  final String ad;
  final String yazar;
  bool okunacak;
}
```

Bu sınıf BuildContext, Widget veya veritabanı paketi bilmez.

Depo sözleşmesi

```
abstract class KitapDeposu {
  Future<List<Kitap>> tumunuGetir();
  Future<void> kaydet(Kitap kitap);
}
```

Bugün bellek deposu, daha sonra SQLite veya REST deposu kullanılabilir. Servis katmanı sözleşme değişmediği sürece veri kaynağının ayrıntısını bilmez.

Bellek deposu

```
class BellekKitapDeposu implements KitapDeposu {
  final kitaplar = <Kitap>[/* başlangıç verileri */];

  @override
  Future<List<Kitap>> tumunuGetir() async =>
    List.unmodifiable(kitaplar);

  @override
  Future<void> kaydet(Kitap kitap) async {
    final sıra = kitaplar.indexWhere((k) => k.id == kitap.id);
    if (sıra >= 0) kitaplar[sıra] = kitap;
  }
}
```

Arama iş kuralı

```
Future<List<Kitap>> ara(String sorgu) async {
  final kitaplar = await depo.tumunuGetir();
  final temiz = sorgu.trim().toLowerCase();
  if (temiz.isEmpty) return kitaplar;

  return kitaplar.where(
    (kitap) =>
      kitap.ad.toLowerCase().contains(temiz) ||
      kitap.yazar.toLowerCase().contains(temiz),
  ).toList();
}
```

Arama kuralı TextField içine gömülmediği için bağımsız test edilebilir.

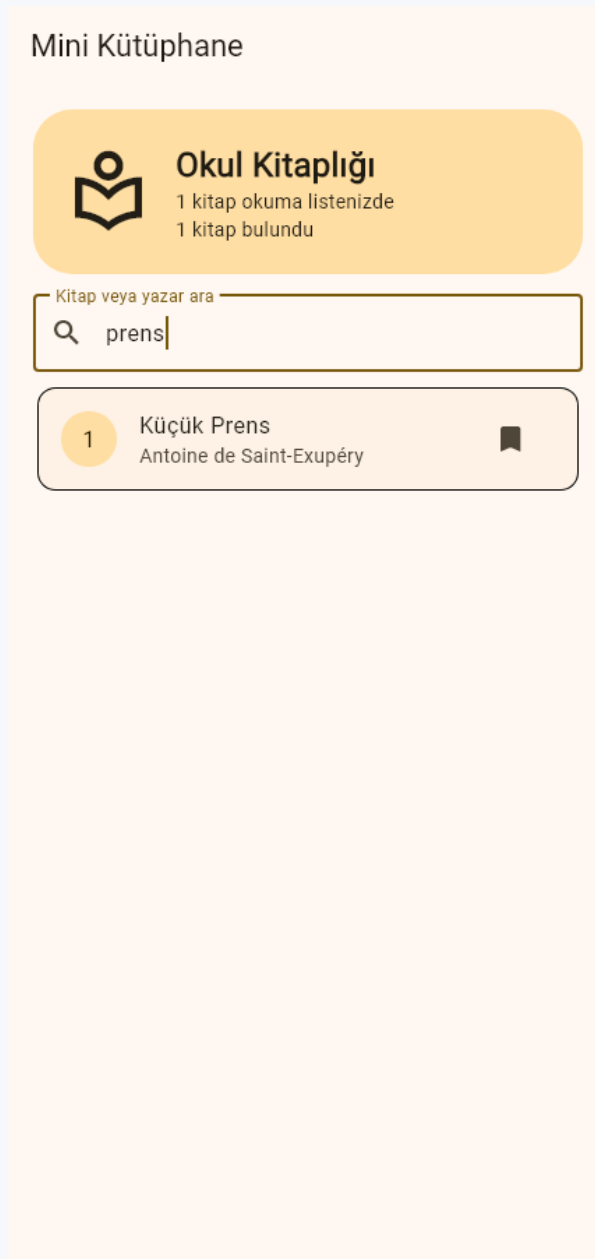
Bağımlılık verme

```
void main() => runApp(  
  KutuphaneApp(  
    servis: KitapServisi(BellekKitapDeposu()),  
  ),  
);
```

Nesneler uygulamanın girişinde birleştirilir. Widget kendi deposunu gizlice oluşturmaz.

Yükleniyor ve boş durum

```
if (yukleniyor) const LinearProgressIndicator();  
  
if (!yukleniyor && kitaplar.isEmpty)  
  const Card(  
    child: Text('Farkl■ bir arama deneyin'),  
  );
```



Mini Kütüphane gerçek Flutter ekranı

Test “prens” aramasının yalnız Küçük Prens'i döndürdüğünü, okuma listesi düğmesinin servis/depo katmanlarından geçerek durumu güncellediğini doğruladı.

Özgün uygulama — Mahalle Paylaşım Ağı

İlan modeli ve depo sözleşmesi

```
class İlan {
  İlan({
    required this.id,
    required this.esya,
    required this.mahalle,
    required this.sahip,
    this.ayrildi = false,
  });

  final int id;
  final String esya;
  final String mahalle;
  final String sahip;
  bool ayrildi;
}
```

```
abstract class İlanDeposu {
  Future<List<İlan>> listele();
  Future<void> ekle(İlan ilan);
  Future<void> guncelle(İlan ilan);
}
```

Serviste doğrulama ve kimlik üretme

```
if ([esya, mahalle, sahip].any((v) => v.trim().isEmpty)) {
  throw ArgumentError('Bütün alanlar zorunludur');
}

final mevcut = await depo.listele();
final id = mevcut.fold<int>(
  0,
  (enBuyuk, i) => i.id > enBuyuk ? i.id : enBuyuk,
) +
1;
```

Formun boş olup olmadığı yalnız arayüzde değil, kullanım senaryosunu yöneten serviste de korunur.

Ayırma işlemi

```
Future<void> ayir(İlan ilan) async {
  ilan.ayrildi = !ilan.ayrildi;
  await depo.guncelle(ilan);
}
```

Mahalle Paylaşım Ağı

**3 paylaşım aktif**
Paylaşım ilanı yayınlandı

Paylaşılacak eşya

Mahalle

Çınar

Adınız

Ece

İlanı yayınla

 Roman seti
Çınar Mahallesi • Ayşe

Ayr

 Futbol topu
Bahçe Mahallesi • Can

Ayr

 Satranç takımı
Çınar Mahallesi • Ece

Ayrıldı

Mahalle Paylaşım Ağı gerçek Flutter ekranı

Test “Satranç takımı” ilanını gerçek formdan ekledi, listenin üç kayda çıktığını doğruladı ve yeni ilanı “Ayrıldı” durumuna geçirdi.

Katmanlar arası bağımlılık

```
Arayüz → Servis → Depo sözleşmesi ← Bellek/SQLite/REST deposu
      ↓
      Model
```

İç katmanlar Flutter ekranını bilmez. Veri kaynağı, depo sözleşmesini uygular. Bu yapı veri kaynağını değiştirmeyi ve testte sahte depo kullanmayı kolaylaştırır.

Sık yapılan hatalar

- Her sınıf için gereksiz katman oluşturmak
- Model içine BuildContext koymak

- SQL veya HTTP kodunu widget içinde çalıştırmak
- Servisin doğrudan belirli veritabanına bağımlı olması
- Depodan değiştirilebilir listeyi kontrolsüz döndürmek
- Hata ve boş durumlarını göstermemek
- Testte üretim veritabanına veya internete bağlanmak

Üç uygulama görevi

Kolay — Alışveriş listesi

Ürün modeli, depo sözleşmesi, bellek deposu ve ekleme/tamamlama servisi oluşturun.

Orta — Atölye araç takibi

Araçları arama, ödünç verme ve iade etme iş kurallarını servis katmanına ayırın. Gecikmiş araçları ayrı filtreleyin.

İleri — Çevrimdışı duyuru uygulaması

Yerel ve uzak depo uygulamaları yazın. Servis, bağlantı yokken yerel veriyi göstermeli ve bekleyen değişiklikleri daha sonra eşitlemelidir.

Kontrol listesi

- Model Flutter'dan bağımsız mı?
- Veri erişimi depo sözleşmesi arkasında mı?
- Arama ve doğrulama iş kuralları serviste mi?
- Widget bağımlılıkları dışarıdan alıyor mu?
- Yükleniyor, boş ve hata durumları var mı?
- Veri kaynağı değiştiğinde arayüz kodu korunuyor mu?
- Test gerçek ağ/veritabanı yerine kontrollü depo kullanıyor mu?
- Her proje kendi başına çalışıyor mu?

Geri bildirim ve çözüm erişimi

Öğrenci katmanlı tasarımını gönderdikten sonra İpucunu göster, hangi sorumluluğun hangi katmana ait olduğunu açıklayan yönlendirme sunar. Örnek çözümü göster, gerçek model, depo, servis ve Flutter ekran dosyalarını ayrı ayrı açar. İndirme bağlantısı her projenin bütün klasör yapısını içerir.

Kaynak projeler

- Konu uygulaması: 09 - Bütünleşik Proje ve Yayınlama/Uygulamalar/Uygulama-34-Mini Kütüphane
- Özgün uygulama: 09 - Bütünleşik Proje ve Yayınlama/Uygulamalar/Ozgün-34-Mahalle Paylaşım Ailesi