Proje Raporu

Üniversite: Erciyes Üniversitesi
Fakülte: Mühendislik Fakültesi
Bölüm: Bilgisayar Mühendisliği
Ders: Mobile Application Development
Danışman: Dr. Öğr. Üyesi Fehim Köylü
Hazırlayan: Ahmet Göker
Öğrenci No: 1030510380
Proje Adı: Flutter ile Local Storage Not Uygulaması
Teslim Türü: Proje Ödevi

1. Giriş

Bu proje, Erciyes Üniversitesi Bilgisayar Mühendisliği bölümü kapsamında aldığım "Mobile Application Development" dersi için hazırladığım bir çalışmadır. Amacım, Flutter kullanarak internet bağlantısına ihtiyaç duymadan not yazılabilen, düzenlenebilen ve saklanabilen bir mobil uygulama yapmaktı. Uygulamayı hem mobil cihazlarda hem de web tarayıcılarında çalışacak şekilde tasarladım.

2. Projenin Amacı ve Neden Yaptım?

Günümüzde insanlar hızlıca fikirlerini ya da hatırlamak istedikleri şeyleri kaydedebilecekleri dijital araçlara ihtiyaç duyuyor. Bu ihtiyacı karşılamak adına, sade, anlaşılır ve kullanımı kolay bir not uygulaması yapmak istedim. Uygulamanın temelinde; not oluşturmak, düzenlemek, silmek ve kalıcı olarak saklamak gibi işlevler yer alıyor. Flutter'ı tercih etmemin sebebi, tek bir kod tabanı ile hem Android hem iOS platformlarına hitap edebilmemdi.

3. Hangi Teknolojileri Kullandım?

Projeyi Flutter framework'ü ile geliştirdim çünkü "bir kez yaz, her yerde çalıştır" yaklaşımı hem zamandan kazandırıyor hem de geliştirme sürecini sadeleştiriyor. Kodlama dili olarak Dart kullandım; çünkü hem modern yapısı hem de asenkron işlemler için sunduğu destek oldukça güçlü.

Veri saklama konusunda shared_preferences paketinden yararlandım. Bu sayede kullanıcı uygulamayı kapatsa bile notlar cihazın hafızasında saklanıyor. Web versiyonu için de bu paket tarayıcının localStorage API’sini kullanıyor.

Tasarımı, Google’ın Material Design kurallarına uygun olacak şekilde oluşturdum. Böylece uygulama farklı cihaz boyutlarında bile şık ve kullanıcı dostu görünmeyi başardı.

4. Uygulamanın Yapısı (Mimari)

Uygulamayı Flutter'ın widget yapısına göre parçalara ayırarak geliştirdim. Her ekran kendi görevini yerine getiriyor. State yönetiminde StatefulWidget tercih ettim. Kullanıcının girdiği notlar önce bellekte tutuluyor, ardından shared_preferences kullanılarak kalıcı hale getiriliyor.

Uygulama 3 temel bileşenden oluşuyor:

- **NotesApp**: Uygulamanın başlangıç noktası. Tema ve yönlendirme burada yapılır.
- **NotesHomePage**: Notları listelediğim ve düzenleme işlemlerini yaptığım ana ekran.
- **NoteEditorPage**: Yeni not yazmak ya da var olanı düzenlemek için açtığım sayfa.

Bu yapı sayesinde kodlarımı hem kolay okudum hem de gerektiğinde değişiklik yapmam oldukça pratik hale geldi.

5. Veri Modeli

Notları sistemli bir şekilde saklamak için her notun başlık, içerik, tarih ve benzersiz bir kimlik bilgisi olacak şekilde bir veri yapısı oluşturdum.

Note adını verdiğim Dart sınıfında hem veri oluşturma hem de saklama işlemlerini yöneten fromJson ve toJson metodlarını tanımladım. Bu sayede verileri JSON formatına kolayca çevirdim ya da tekrar uygulamaya aldım.

6. Verilerin Saklanması ve Yüklenmesi

Kullandığım shared_preferences paketi sayesinde notları cihaz hafızasında tutabiliyorum. Uygulama kapanıp tekrar açıldığında, bu veriler JSON formatından okunup tekrar kullanılabilir hale geliyor.

Tüm bu işlemleri async/await ile yazdım. Böylece uygulama arayüzü donmadan arka planda veri işlemleri sorunsuz şekilde tamamlandı. Ayrıca oluşabilecek hataları da kullanıcıya bildirmek için try-catch yapısını kullandım.

Örnek kod:

```dart
Future<void> _saveNotes() async { // Notları kaydeden fonksiyon
  final prefs = await SharedPreferences.getInstance(); // SharedPreferences örneği alınıyor
  final String encodedNotes = jsonEncode(_notes.map((note) => note.toJson()).toList()); // Notlar JSON string'e dönüştürülüyor
  await prefs.setString(_storageKey, encodedNotes); // JSON string, local storage'a kaydediliyor
}
```

Yükleme işlemi:

```dart
Future<void> _loadNotes() async { // Notları yükleyen fonksiyon
  final prefs = await SharedPreferences.getInstance(); // SharedPreferences örneği alınıyor
  setState(() {
    _isLoading = true; // Yükleniyor durumu başlatılıyor
  });

  try {
    final String? notesJson = prefs.getString(_storageKey); // JSON verisi alınır
    if (notesJson != null) {
      final List<dynamic> decoded = jsonDecode(notesJson); // JSON çözülür
      final loadedNotes = decoded.map((item) => Note.fromJson(item)).toList(); // Note nesnelerine dönüştürülür

      setState(() {
        _notes = loadedNotes; // Not listesi güncellenir
      });
    }
  } catch (e) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Notlar yüklenirken bir hata oluştu: $e')), // Hata mesajı gösterilir
    );
  } finally {
    setState(() {
      _isLoading = false; // Yükleniyor durumu kapatılır
    });
  }
}
```

7. Karşılaştığım Zorluklar ve Nasıl Aştım?

- İlk olarak, asenkron veri işlemleri sırasında uygulamanın takılmaması için async/await yapısını dikkatli kullandım.
- DateTime nesnesi doğrudan JSON'a çevrilemediği için önce toIso8601String() ile string'e dönüştürdüm, sonra DateTime.parse() ile tekrar kullandım.
- Bazı durumlarda, arayüzden silinmiş widget'lara erişim hatası alabiliyordum. Bunu if (mounted) ile kontrol ederek çözdüm.

```
if (mounted) { // Widget hala aktif mi kontrol edilir
  ScaffoldMessenger.of(context).showSnackBar(
    const SnackBar(content: Text('Not silindi')), // Kullanıcıya bilgi verilir
  );
}
```

8. Genel Değerlendirme

Bu projeyle birlikte Flutter'da mobil uygulama geliştirme, verileri yerel olarak saklama ve kullanıcı arayüzü tasarlama gibi birçok konuda deneyim kazandım. Uygulama oldukça sade, işlevsel ve kullanımı rahat bir hale geldi.

İlerleyen dönemlerde uygulamaya etiketleme, arama ve bulut senkronizasyonu gibi gelişmiş özellikler de eklemeyi planlıyorum. Tek kodla çok platformda çalışabilen bir yapı kurduğum için ilerisi için sağlam bir temel atmış oldum.

Kaynaklar

1. Flutter Documentation: https://flutter.dev/docs
2. Dart Language: https://dart.dev
3. shared_preferences Plugin: https://pub.dev/packages/shared_preferences
4. Material Design: https://material.io/design