



**YILDIZ TEKNİK ÜNİVERSİTESİ
KİMYA-METALÜRJİ FAKÜLTESİ
MATEMATİK MÜHENDİSLİĞİ BÖLÜMÜ**

Matematik Mühendisliğinde Tasarım Uygulamaları

FLUTTER İLE MOBİL UYGULAMA GELİŞTİRME

Tez Yöneticisi: MERT BAL

ASLI KUŞÇU

19052040

**© Bu tezin bütün hakları Yıldız Teknik Üniversitesi Matematik Mühendisliği
Bölümü'ne aittir.**

İÇİNDEKİLER

ÖNSÖZ	4
ABSTRACT	5
Konunun Önemi ve Amacı	6
Literatür İncelemesi.....	6
Veri Toplama ve Hazırlık	7
Analiz ve Sonuçlar	8
Flutter uygulama çalışma prensibi	11
Flutter veri seti	12
Kullanılan Teknolojiler	14
Uygulama Arayüzü incelemesi.....	14
Uygulama Arayüzleri	18
Kod Kısmı.....	21
Main.dart.....	21
Home_page.dart.....	23
Tourist_details_page.dart	24
Welcome_page.dart.....	26
SONUÇLAR.....	29
KAYNAKLAR	30
ÖZGEÇMİŞ.....	31

ÖNSÖZ

Seyahat, insan hayatında önemli bir yer tutan ve bireyleri farklı kültürlerle buluşturan bir deneyimdir. Teknolojinin hızla ilerlediği günümüzde, mobil uygulamalar seyahat süreçlerini daha kolay, eğlenceli ve etkileşimli hale getirmek için önemli bir araç haline gelmiştir. Bu tez, Flutter framework'ünü kullanarak geliştirilen basit bir seyahat uygulamasını konu almaktadır.

Bu çalışmanın amacı, seyahat severlere yönelik kullanıcı dostu bir mobil uygulama geliştirmek ve seyahat deneyimini daha unutulmaz kılmak için teknolojiyi kullanma potansiyelini ortaya koymaktır. Bu uygulama, seyahat öncesinden başlayarak, rota planlama, yer önerileri, anı paylaşımı gibi bir dizi özelliği içerecek ve kullanıcılarına kişiselleştirilmiş bir seyahat deneyimi sunacaktır.

Tezin bu önsöz bölümü, çalışmanın temel motivasyonunu açıklamakta ve seyahat tutkunlarına yönelik basit bir seyahat uygulaması geliştirme sürecinde yaşanan düşünceleri ve kararları paylaşmaktadır. Tezin genel yapısı, metodolojisi ve hedefleri bu önsözde özetlenmiştir.

Seyahat uygulamaları, insanların dünyayı keşfetme arzusunu destekleyen önemli araçlardan biridir. Bu tez, bu aracın geliştirilmesine yönelik bir adım oluşturarak, seyahat severlere daha etkili ve keyifli bir deneyim sunma potansiyelini göstermeyi amaçlamaktadır.

Bu çalışma sürecinde emeği geçen herkese teşekkür eder, okuyucuların bu tezi ilgiyle incelemesini umarım.

İstanbul, 2023

ABSTRACT

This thesis will focus on a mobile application developed using the Flutter framework and will examine the design, development, and usability processes of a simple travel application. The mobile app aims to streamline users' travel experience by providing travel lovers with essential travel planning and sharing features.

The thesis will begin by first introducing the key features and advantages of the Flutter framework, and then explain in detail the process of designing a simple travel app. The app will include basic features such as user-friendly interface design, travel planning, itinerary creation, hotel booking, and social media sharing.

By emphasizing the challenges and improvements encountered during the development process of the application, the advantages and limitations provided by Flutter will be discussed. This study aims to guide developers who are just starting to develop a simple travel application, especially using Flutter, and to provide a simple solution that focuses on improving the user experience.

The results of the thesis will make a valuable contribution to provide a general understanding of simple mobile application development with Flutter and to show how this framework can be used effectively in travel applications.

Konu: Flutter ile Mobil Uygulama Geliştirme

Matematik Mühendisliği Tasarım Uygulamaları Genel İlerleme:

Bu raporda 5. Haftaya kadar yapılan araştırmalar ve çalışmalar yer almaktadır. Raporda araştırma kısmında kullanılan kaynaklar belirtilmiştir. Raporun son kısmında 9. Hafta Raporuna kadar yapılması planlanan işler belirtilmiştir.

1.Konunun Önemi ve Amacı:

Yapılacak olan seyahat uygulamasının amacı kullanıcılara seyahatlerini planlama, rezervasyon yapma, seyahat deneyimini geliştirme ve gezilerini daha kolay ve keyifli hale getirme konularında yardımcı olmaktır. Seyahat uygulamaları, kullanıcıların seyahatlerini daha düzenli ve keyifli hale getirmelerine yardımcı olurken, seyahat endüstrisindeki hizmet sağlayıcılarına da müşteri sağlama ve pazarlama fırsatları sunar. Yapılacak olan uygulama belirli bir aşamaya kadar getirilecektir.

2.Literatür İncelemesi:

Flutter, Google tarafından geliştirilen açık kaynaklı bir kullanıcı arayüzü (UI) yazılım geliştirme çerçevesidir ve Android, IOS, Web ve diğer platformlar için tek bir kod tabanı kullanarak uygulama geliştirmeyi sağlar. Flutter'ın yaygın olarak kullanıldığı birçok literatür kaynağı ve yazılım geliştirme konularına odaklanan makaleler vardır. Flutter ile mobil uygulama geliştirme alanında literatür incelemesi yaparken aşağıdaki kaynaklar göz önünde bulunduruldu:

Flutter Resmi Web sitesi (Dart ve Flutter için Dokümantasyon):

Flutter Resmi Web sitesi, geliştiricilere güncel ve güvenilir kaynaklar sunarak Flutter framework'ü ile ilgili her türlü bilgiye erişim sağlar. Bu web sitesi, Flutter'ın resmi belgelerini, öğreticilerini, geliştirici rehberlerini ve diğer kaynakları içerir. Bu sayede projeyi oluşturmak, geliştirmek ve sorun gidermek için kapsamlı bir kaynak havuz imkânı sağladı.

2.1. Flutter Eğitim Kursları ve Flutter ile ilgili YouTube Kanalları:

Mitch Koko, Dbestech, Dear Programmer gibi kanallardan eğitim amaçlı yararlanıldı. Bu kanallar, hem Flutter temellerini öğrenmek isteyen başlangıç seviyesindeki kullanıcılar hem de daha deneyimli geliştiriciler için güncel ve değerli içerikler sunmaktadır. Youtube üzerinden HardwareAndro kanalı ile 20 videodan oluşan eğitim setinden destek alındı. Çevrimiçi eğitim platformlarında bulunan video dersler, Flutter'ın uygulama geliştirme süreçlerini görsel olarak açıklar.

2.2. GitHub ve Open Source Projeler:

GitHub üzerinde birçok Flutter projesi bulunmaktadır. Flutter öğrenmek için açık kaynaklı projeler ve mevcut kodlar incelendi. Ayrıca bu projeler, farklı kullanım durumlarını anlaşılmasına ve farklı teknolojilerle etkileşim kurulmasına olanak sağladı. Uygulama mağazalarındaki veya GitHub gibi kaynaklardaki gerçek uygulama örnekleri, Flutter ile nasıl uygulamalar oluşturulduğunu anlaşılmasına yardımcı oldu.

Literatür incelemesi yaparken bu kaynaklar incelenerek, Flutter ile mobil uygulama geliştirmeye yönelik en güncel bilgilere ve kaynaklara erişim sağlandı. İnceleme sürecinin iyi planlanıp literatür incelemesinin geniş bir perspektife sahip ve bilimsel bir temele dayalı olması sağlandı.

3. Veri Toplama ve Hazırlık:

Flutter kullanarak bir seyahat uygulaması geliştirmek için öncelikle veri toplama ve hazırlık aşamasını düşünmek gerekir. Veri toplama ve hazırlık aşaması uygulamanın başarılı ve kullanıcı dostu bir şekilde çalışabilmesi için kritik bir adımdır. Çünkü iyi bir veri altyapısı olmadan kullanıcıların seyahat deneyimini iyileştirmek zor olacaktır. Flutter ile mobil uygulama geliştirirken veri toplama ve hazırlık için izlenen bazı adımlar:

3.1. Geliştirme Ortamını Hazırlama:

-Flutter bilgisayara indirilip kuruldu. Geliştirme ortamı olarak Visual Studio Code tercih edildi.

3.2. API'lar ve Servisler:

-Uygulamanın veriye erişimi için gerekli API'ları ve servisleri belirlendi ve bunlar entegre edildi. Bu, verileri almak ve göndermek için kullanılabilecek bir araç sağlar.

3.3. Veri Modelini Tanımlama:

-Uygulamanın ihtiyaç duyduğu verileri temsil eden veri modelleri oluşturuldu. Bu, veriyi düzenli ve anlamlı bir şekilde saklamak için önemlidir.

3.4. Hata İzleme ve Güncelleme:

-Uygulamanın güvenilir kalmasını sağlamak için düzenli güncellemeler planlandı.

3.5. Veri Görselleştirme:

-Flutter ile veri görselleştirme, Flutter'ı kullanarak verileri grafikler, tablolar veya diğer görsel öğeler aracılığıyla görsel bir şekilde sunma işlemidir. Bu, verilerin daha anlaşılır hale getirilmesine ve kullanıcılara bilgilerin daha kolay bir şekilde anlaşılmasına yardımcı olur. Flutter, farklı veri görselleştirme kütüphaneleri ve widget'lar aracılığıyla bu tür görselleştirmeleri kolayca gerçekleştirmeyi sağlar.

Bu adımlar, seyahat uygulaması için gerekli olan veri altyapısını hazırlamayı sağlamıştır ve Flutter ile mobil uygulama geliştirirken veri toplama ve hazırlık sürecini yönlendirmede yardımcı olmuştur. Bu aşamada, kullanıcı deneyimini iyileştirmek ve verilere güvenli bir şekilde erişmek için önemlidir.

4.Analiz ve Sonuçlar:

Flutter, seyahat uygulamaları gibi mobil uygulamaları geliştirmek için kullanılan popüler bir çerçevedir.

Bir seyahat uygulaması geliştirmek için önce analiz ve tasarım aşamaları gerçekleştirilir ve sonrasında uygulama geliştirilir. Seyahat uygulaması geliştirme sürecinin analiz süreçleri

4.1. İş ve kullanıcı gereksinimlerinin analizi:

- İlk adım, uygulamanın amacı ve hedef kitlesi belirlendi. Hangi tür seyahat uygulaması geliştireceği ve kullanıcıların neleri beklediği anlaşıldı.
- Kullanıcı gereksinimleri, işlevsel gereksinimler ve tasarım beklentileri belirlendi.

4.2. Pazar araştırması:

- Mevcut seyahat uygulamalarının analizi yapıldı ve incelenen uygulamaların güçlü ve zayıf yönleri belirlendi.

4.3. Tasarım:

- Uygulamanın kullanıcı arayüzü (UI) ve kullanıcı deneyimi (UX) tasarımı yapıldı. Bu, uygulamanın kullanıcı dostu ve çekici olmasını sağlamıştır.

4.4. Sonuçlar:

- Bir Flutter ile geliştirilmiş seyahat uygulaması, kullanıcılara seyahat planlama, bilet rezervasyonu, konaklama, harita ve yerel rehberlik gibi çeşitli hizmetler sunabilir.
- Kullanıcılar uygulamayı kolayca kullanabilir, seyahat deneyimlerini iyileştirebilir ve daha fazla hedefe ulaşmalarına yardımcı olabilir.

5.İlerleme ve Planlar:

5.1. Uygulama Geliştirme:

- Flutter kullanarak uygulama geliştirme süreci başlanması planlanmaktadır. Uygulamanın ön yüzü (UI) ve arkası (back-end) geliştirilmeye başlanacaktır.

- Seyahat uygulamasının, hava durumu tahminleri, otel rezervasyonları, uçak bileti satın alma vb. gibi dış verilere erişim sağlaması gerekebilir. Bu nedenle API sistemi kurulması planlanmaktadır.

5.2. Test ve Kalite Kontrol:

- Uygulamanın her yönü test edilmesi ve hataların giderilmesi sağlanacaktır. Mobil uygulama test ve kontrol aşaması, uygulamanın hedeflenen işlevselliği, performansı, güvenliği ve kullanılabilirliği açısından değerlendirildiği kritik bir aşamadır.

Bu kısımda 9.hafta raporu aşağıda belirtilmiştir.

1. Flutter uygulama çalışma prensibi

Flutter, Google tarafından geliştirilen açık kaynaklı bir UI (Kullanıcı Arayüzü) yazılım geliştirme kiti ve SDK'sıdır. Flutter, tek bir kod tabanında hem iOS hem de Android uygulamaları oluşturmanıza olanak tanıyan "geliştir bir, çalıştır her yerde" (write once, run anywhere) yaklaşımını benimser.

Flutter uygulama çalışma prensibi şu adımları içerir:

1.1. Dart Programlama Dili:

- Flutter uygulamaları, Dart adlı bir programlama dilinde yazılır. Dart, hızlı performansa ve geliştiricilere esneklik sağlayan bir dildir. Dart, özellikle Flutter için tasarlanmış ve onunla uyumlu bir şekilde çalışan bir dil olarak öne çıkar.

1.2. Widget'lar:

- Widget kelimesi Türkçeye çevrildiğinde “araç” olarak karşımıza çıkmaktadır. Ancak kelime anlamı olarak görsel parçacık veya pencere olarak kullanılabilir. Widget, Android telefonlarında kullanılabilen minik işlevsellik sağlayan araçlardır. Widgetlar, telefonun ana ekranına eklenebilen küçük araçlardır. Bu araçlar sayesinde herhangi bir uygulamaya girmeden ana ekranda bulunan Widget ile veriye anında erişilebilir.
- Widget sayesinde telefon kullanım işlevselliğinizi arttırabilirsiniz. Örneğin, ana ekrana saat Widgeti koyduğunuz zaman ekranınızdan kolay bir şekilde saati takip edebilirsiniz. Bunun dışında Widget kullanan bazı uygulamalar şunlardır: Takvim, hava durumu, haber, pil göstergesi, Facebook Widget, Twitter, instagram ve daha birçok uygulama Widget özelliğini desteklemektedir.

1.3. Stateless ve Stateful Widget'lar:

Flutter'da , durumu değiştiremeyen, yalnızca verileri alıp bunları kullanarak bir arayüz oluşturan bir widget türüdür. Stateless Widget, değişmez bir yapıya sahiptir ve bir kez oluşturulduktan sonra, widgetın durumu veya görünümü değiştirilemez. Bu nedenle, Stateless Widget örneğinin durumu değişmeden aynı kalmalıdır. Stateless Widget sadece bir build metodu içerir. Bu yöntem, widgetın görünümünü oluşturmak için kullanılır ve her çağrıldığında aynı sonucu döndürmelidir. Stateless Widget , genellikle statik veya sabit bir içeriği göstermek veya değişmeyen verileri görüntülemek için kullanılır. Örneğin, bir resim veya bir metin etiketi gibi bir özellikte kullanılabilir.

Flutter'da Stateful widget, widget'ın içindeki durumu (state) değiştirmek için kullanılır. Stateful widget, bir State sınıfı ile eşleştirilir ve widget durumu değiştiğinde State sınıfı güncellenir. Bu sayede, widget'ın içindeki verileri değiştirmek için State sınıfı kullanılabilir.

Stateful widget oluşturmak için öncelikle bir Stateful Widget sınıfı tanımlamak gerekir. Bu sınıfın içinde, widget'ın durumunu tutacak bir State nesnesi oluşturulur. Bu State nesnesi, widget'ın içindeki verileri değiştirmek için kullanılır.

1.4. Hot Reload:

Hot Reload: Hot Reload, kodu anında güncellemek için kullanılan bir özelliktir. Bu özellik sayesinde, uygulamadaki değişiklikler derlenmeden hemen önce yansıtılır. Yani, kodda yapılan değişiklikleri hızlı bir şekilde görüntülemek için kullanılır. Bu sayede, kodda hata bulma süreci hızlandırılır ve uygulamanın gerçek zamanlı olarak nasıl görüneceğini görmek mümkün olur.

Örneğin, uygulamayı çalıştırdığınızda, hot reload özelliği sayesinde kodu anında güncelleyebilirsiniz. Değişiklikleri kaydettiğinizde, Flutter derleyicisi kodu otomatik olarak yeniden yükler ve uygulama hemen güncellenir. Bu sayede, uygulama kodunda yapılan değişiklikleri hemen test edebilirsiniz.

2. Flutter veri seti

"Veri seti" terimi genellikle bir uygulamada kullanılan veri kümesini ifade eder. Flutter uygulamalarında, veri setleri genellikle veri tabanından alınan veya çeşitli API'ler aracılığıyla elde edilen verileri içerir. Veri setleri, uygulama içinde çeşitli

bileşenlere (örneğin, listeler, kartlar, grafikler) veri sağlamak için kullanılır.

Flutter uygulamasında bir veri seti oluştururken şu adımlar takip edilir:

2.1. Veri Modeli Oluşturma:

- Flutter'da bir veri modeli oluşturmak, uygulamanızın belirli bir bileşeni için veriyi temsil eden bir sınıf veya nesne tanımlamak anlamına gelir. Veri modeli, genellikle bir widget tarafından kullanılacak olan veriyi düzenler ve bu veriyi içeren nesne veya sınıfın özelliklerini (property) tanımlar.

2.2. Veri Seti Oluşturma:

- Uygulamanın ana mantığı ile bir veri seti oluşturulur. Bu, örneğin bir dizi kitabı içerebilir. Bu veri seti, uygulamanın başlatılması veya bir ekranın yüklenmesi gibi olaylarla birlikte doldurulabilir.

2.3. Veriyi Görselleştiren Widget'lar Oluşturma:

- Flutter'da, veriyi kullanıcı arayüzünde göstermek için çeşitli widget'lar bulunmaktadır. Örneğin, ListView widget'ı, bir liste içinde veriyi göstermek için kullanılabilir. Her liste ögesi, bir kitabı temsil edebilir.

2.4. Veri Setini Güncelleme:

- Uygulamanın kullanıcı etkileşimleri veya başka olaylar meydana geldiğinde, veri seti güncellenir. Bu, kullanıcının yeni kitaplar eklemesi veya mevcut kitapları silmesi gibi durumları içerebilir.

2.5. State Management (Durum Yönetimi):

- Flutter'da durum yönetimi, uygulama durumu değiştiğinde kullanıcı arayüzünü güncellemenin bir yoludur. State management konsepti, Flutter uygulamalarında veri setlerini güncellemek ve bu güncellemeleri kullanıcı arayüzüne yansıtmak için kullanılır.

Bu adımlar, Flutter uygulamalarında genel bir veri seti yönetimi sürecini açıklar. Her adım, uygulamanın özelliklerine ve gereksinimlerine bağlı olarak özelleştirilebilir.

3. Kullanılan Teknolojiler

Android Studio Projede Android Studio'nun sunduğu Emülatör hizmetinden yararlanılmıştır. Emülatör olarak LG Nexus 5x cihazı android sürümü olarak da Android Pie (9.0) kullanılmıştır.

Visual Studio Code Visual Studio Code, Microsoft tarafından üretilen, Windows, Linux ve Mac işletim sistemlerinde çalışabilen ücretsiz IDE ortamıdır. Versiyon kontrol, güçlü eklentiler, hata ayıklama gibi temel özellikler barındıran bir hızlı kod geliştirme aracıdır. VS Code'un bir diğer çok önemli artısı ve güçlü yönü ise git entegrasyonudur. Eklentiler ile birlikte Visual Basic.NET, Visual C# ve Visual C++ gibi dillerin yanı sıra Node JS, Ruby, Python, Dart gibi birçok programlama dilini desteklemektedir.

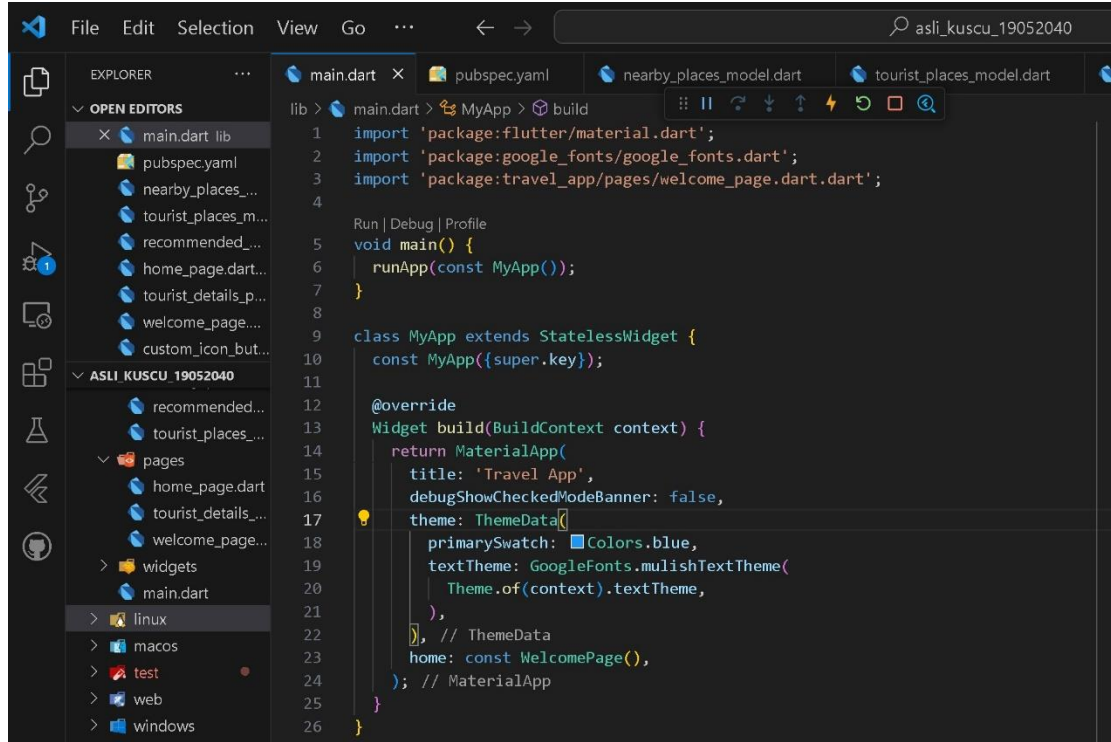
Flutter Flutter, Google tarafından geliştirilen, içinde framework, widget ve diğer araçları barındıran, açık kaynaklı, Cross-platformda uygulama geliştirme yeteneğine sahip bir UI yazılım geliştirme kitidir.

Android, iOS, Windows, Mac, Linux, Google Fuşya ve web için uygulamalar geliştirmek için kullanılır. Stateful Hot Reload özelliği ile uygulamada gerçekleştirdiğimiz değişiklikleri anlık olarak görebilmemizi sağlar.

Dart Genellikle mobil ve web uygulamaları geliştirmek için kullanılan Dart, açık kaynaklı bir dil olarak bilinir. Dart'ın ana kullanım alanlarından biri, Google'ın popüler mobil uygulama geliştirme çerçevesi olan Flutter için kullanılmasıdır.

4. Uygulama Arayüzü

main.dart dosyası incelemesi:



Bu Flutter uygulaması, bir seyahat uygulamasının temel yapısını oluşturan 'MyApp' adlı bir widget içerir. İlgili widget, genel tasarım ve tema ayarlarını tanımlayarak, uygulamanın ana sayfasını başlatmak için 'WelcomePage' widget'ını içerir.

1.1. Main Fonksiyonu:

- main fonksiyonu, uygulamanın başlangıç noktasını belirler.
- runApp fonksiyonu, uygulamanın ana widget'ını (MyApp) içerir.

```
``dart
void main() {
  runApp(const MyApp());
}
...`
```

1.2. MyApp Widget:

- MyApp widget'ı, bir StatelessWidget sınıfıdır ve uygulamanın genel tema ayarlarını içerir.
- MaterialApp widget'ı, uygulama genelinde kullanılacak temel konfigürasyonları içerir.
- debugShowCheckedModeBanner özelliği, hata ayıklama modunda üstteki kırmızı etiketi gizlemek için 'false' olarak ayarlanmıştır.

```

```dart
class MyApp extends StatelessWidget {
 const MyApp({super.key});

 @override
 Widget build(BuildContext context) {
 return MaterialApp(
 title: 'Travel App',
 debugShowCheckedModeBanner: false,
 theme: ThemeData(
 primarySwatch: Colors.blue,
 textTheme: GoogleFonts.mulishTextTheme(
 Theme.of(context).textTheme,
),
),
 home: const WelcomePage(),
);
 }
}
```

```

1.3. Tema Ayarları:

- Tema, primarySwatch özelliği ile mavi renkte bir tema oluşturur.
- GoogleFonts kütüphanesi kullanılarak, uygulama genelinde Mulish fontu kullanılacak şekilde textTheme ayarlanır.

1.4. WelcomePage Widget:

- WelcomePage widget'ı, uygulamanın başlangıç sayfasını içerir.
- Bu sayfa, uygulamanın genel tasarımına ve kullanıcıyı karşılamak için kullanılacak öğelere sahiptir.

```

```dart
home: const WelcomePage(),
```

```

- Not: Kodun başında kullanılan const ifadesi, widget'ın sadece bir kez oluşturulacağını belirtir ve bu, performansı artırabilir.

Bu yapı, uygulamanın başlatılması ve temel konfigürasyonların belirlenmesi için kullanılan temel bir yapıdır. MyApp widget'ı, uygulamanın genel temel tasarımını ve tema ayarlarını içerirken, WelcomePage widget'ı uygulamanın başlangıç sayfasını temsil eder.

5. Analiz ve Sonuçlar:

Flutter, aynı kod tabanını kullanarak iOS ve Android gibi farklı platformlarda çalışabilen çapraz platformlu bir mobil uygulama geliştirme framework'üdür.

Geliştiriciler, ayrı ayrı iOS ve Android için kod yazmak yerine tek bir kod tabanında iş birliği yapabilir, bakım ve güncelleme maliyetlerini düşürebilir.

Hot Reload özelliği, geliştiricilere uygulama geliştirirken anlık geri bildirim alma imkânı sunar.

Geliştiriciler, değişiklikleri hemen görebilir ve kodlarını sürekli test edebilir, bu da geliştirme sürecini hızlandırır.

Flutter, zengin ve özelleştirilebilir widget'lar sunar.

Widget'lar, UI öğelerini oluşturmak ve düzenlemek için kullanılır; bu da geliştiricilere esneklik ve kontrol sağlar.

Flutter, geniş bir geliştirici topluluğuna sahiptir ve bu topluluk sürekli olarak paylaşılan kaynaklar, paketler ve yardım sağlar.

Bu, geliştiricilere platforma özgü tasarım dillerini uygulama ve uygulamanın doğal görünmesini sağlama olanağı tanır.

Bu özellikler, Flutter'ın popülerliğini artıran ve geliştiricilere esnek, hızlı ve çapraz platformlu uygulama geliştirme deneyimi sunan temel nedenlerden sadece birkaçıdır.

6. İlerleme ve Planlar:

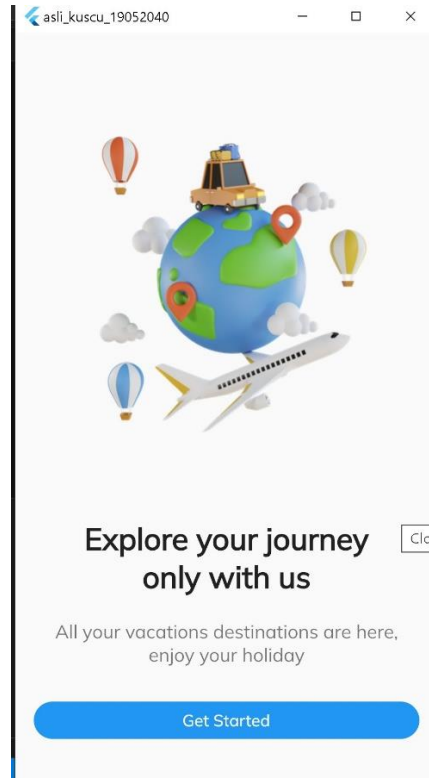
İleride uygulamanın hedeflenen kısma kadar bitirilmesi planlanmaktadır.

Uygulama Arayüzleri

Uygulama Arayüzleri Tasarımı: Basit Seyahat Uygulaması

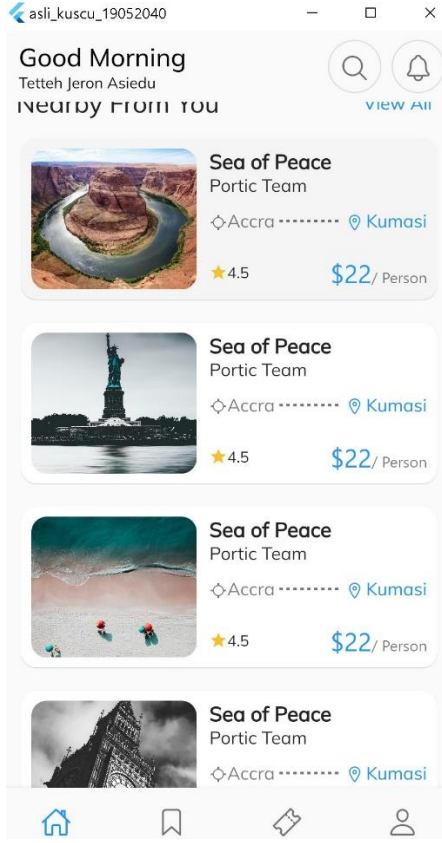
1. Hoşgeldin Ekranı:

Uygulamanın ilk açılışında kullanıcılara hoş geldin mesajıyla karşılanacak bir ekran. Uygulamanın amacını vurgulayan hoş bir görsel ve başlangıç butonu içerecek. Kullanıcı buradan uygulamayı keşfetmeye başlayacak.



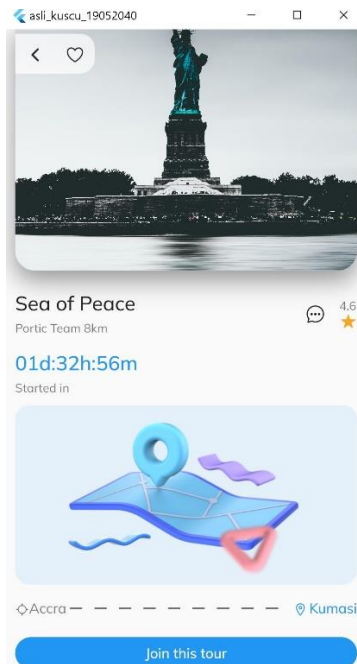
2. Gezilebilecek Yerler Ekranı:

Bu ekran, kullanıcılara ziyaret edilebilecek çeşitli yerleri gösteren bir arayüz sunacak. Her yer, adı, konumu ve kısa bir açıklama ile temsil edilecek. Kullanıcılar bu listeden bir yeri seçerek daha fazla bilgi alabilirler.



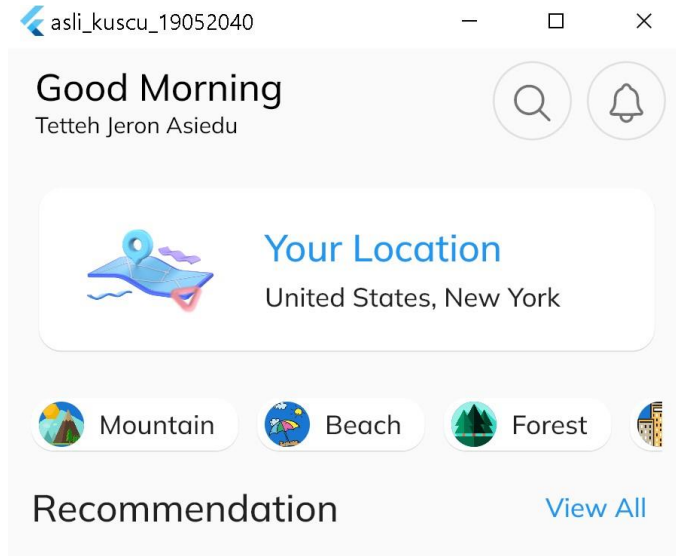
3. Yer Detayları Ekranı:

Seçilen yerin detaylarının yer aldığı ekran. Bu ekran, yerin konumu, puanı, detaylı açıklaması ve çeşitli görseller içerecek. Kullanıcılar burada yer hakkında daha fazla bilgi edinebilirler.



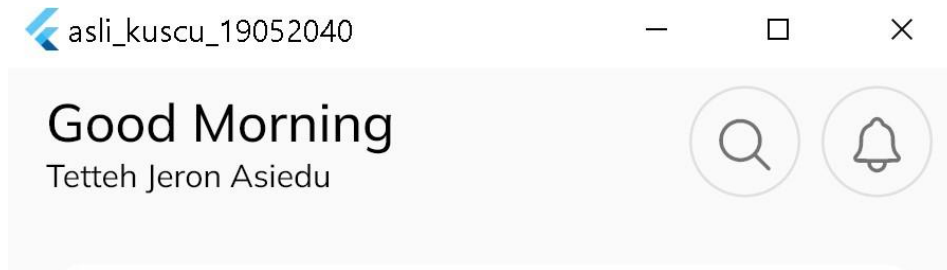
4. Kategori Seçimi Ekranı:

Kullanıcıya "Orman", "Şehir", "Çöl", "Sahil" gibi seçenekler sunan bir ekran. Kullanıcılar bu seçenekler arasından birini seçerek o türdeki yerlere odaklanabilirler.



5. Arama ve Bildirim Butonu:

Kullanıcılara belirli bir yer veya kategori araması yapma imkanı sağlayan arama ekranı. Arama sonuçları, kullanıcının girdiği kriterlere uygun olarak listelenecek.



Bu basit seyahat uygulamasının arayüzleri, kullanıcıların uygulama içinde kolayca gezinmelerini ve istedikleri yerlere hızlıca ulaşmalarını sağlayacak şekilde tasarlanmıştır. Arayüzler, kullanıcı dostu ve görsel olarak çekici olacak şekilde özenle oluşturulmuştur.

Kod Kısmı:

Seyahat uygulaması geliştirmek için Flutter kullanarak basit bir mobil uygulama tasarlanacaktır. Aşağıda, uygulamanın temel yapıları hakkında açıklamalar bulunmaktadır.

Main.dart

Bu Flutter uygulaması, temel bir seyahat uygulamasının başlangıcını oluşturan iki ana bileşeni içermektedir.

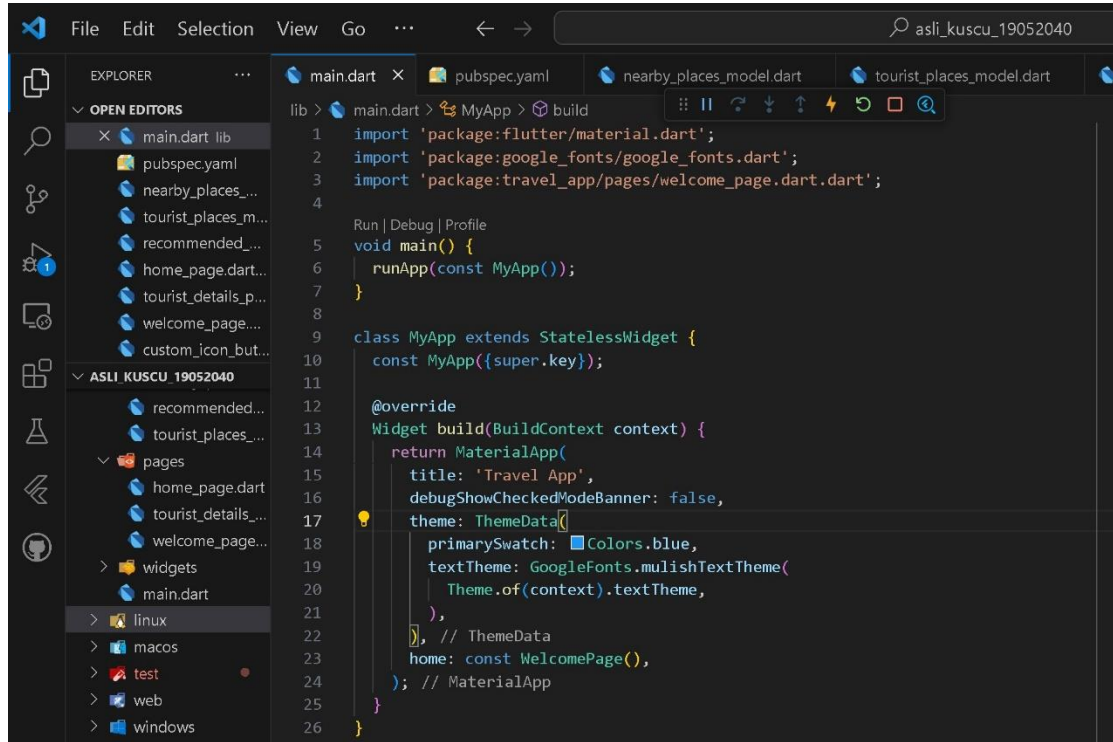
1. Main Fonksiyonu:

- `main()` fonksiyonu, uygulamanın başlatılmasını sağlar.
- `runApp` fonksiyonu, `MyApp` widget'ını başlatarak uygulamanın ana widget'ını belirler.
- Uygulama başladığında, başlığı "Travel App" olan ve hata ayıklama modu banner'ını gizleyen bir `MaterialApp` kullanılır.
- Uygulama teması, genel renk şemasını belirlemek ve Google Fonts kütüphanesinden `Mulish` fontunu kullanmak üzere konfigüre edilir.
- Ana sayfa olarak, hoş geldiniz sayfasının bulunduğu `WelcomePage` widget'ı atanmıştır.

2. MyApp Widget'ı:

- `MyApp` widget'ı, `StatelessWidget` sınıfından türetilmiştir.
- Temel olarak, uygulamanın genel temalarını belirlemek için kullanılır.
- `GoogleFonts` paketi kullanılarak uygulama genelinde kullanılacak metin temaları `Mulish` fontuyla tanımlanır.
- `MaterialApp` içinde, uygulama teması, başlık, hata ayıklama modu ve ana sayfa belirtilmiştir.
- Ana sayfa olarak, uygulama başladığında gösterilecek olan `WelcomePage` widget'ı atanmıştır.

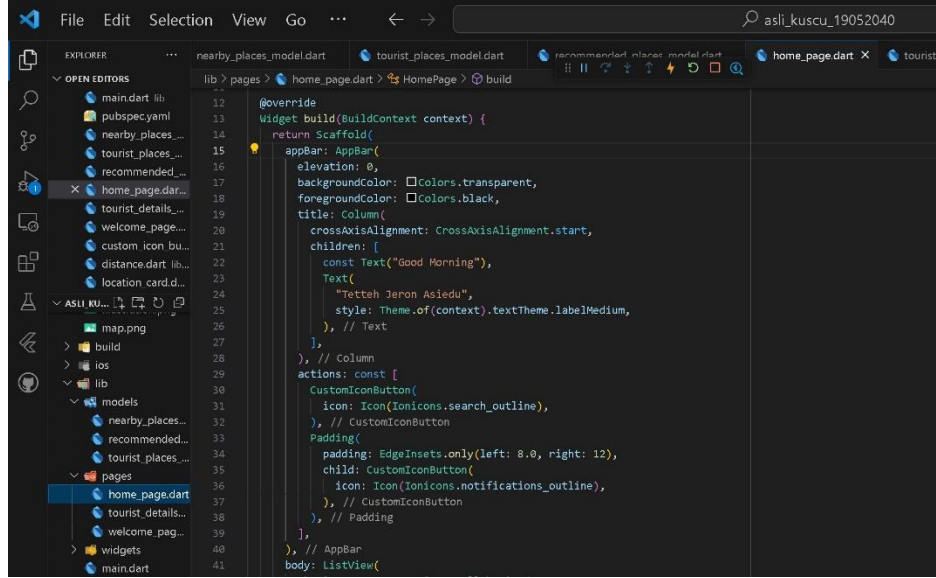
Özetle, bu kod parçası, temel bir Flutter uygulamasının başlangıcını oluşturan `MyApp` widget'ını içerir. Uygulama, "Travel App" başlığına ve hoş geldiniz sayfasına sahiptir. ThemeData kullanılarak genel temalar belirlenmiş ve Google Fonts ile yazı stili özelleştirmeleri yapılmıştır.



```
lib > main.dart > MyApp > build
1 import 'package:flutter/material.dart';
2 import 'package:google_fonts/google_fonts.dart';
3 import 'package:travel_app/pages/welcome_page.dart.dart';
4
5 void main() {
6   runApp(const MyApp());
7 }
8
9 class MyApp extends StatelessWidget {
10   const MyApp({super.key});
11
12   @override
13   Widget build(BuildContext context) {
14     return MaterialApp(
15       title: 'Travel App',
16       debugShowCheckedModeBanner: false,
17       theme: ThemeData(
18         primarySwatch: Colors.blue,
19         textTheme: GoogleFonts.mulishTextTheme(
20           Theme.of(context).textTheme,
21         ),
22       ), // ThemeData
23       home: const WelcomePage(),
24     ); // MaterialApp
25   }
26 }
```

Home_page.dart

Bu Flutter widget'i, seyahat uygulamasının ana sayfasını temsil eder ve aşağıdaki özellikleri içerir:



1. AppBar (Üst Çubuk):

- Sayfa, bir AppBar içerir. AppBar, yükseltme efektsiz (elevation: 0) ve şeffaf (transparent) olarak ayarlanmıştır.
- Başlık, "Good Morning" metni ve kullanıcının adını içeren bir sütun widget'ını içerir.
- Sağ üst köşede, arama ve bildirim simgelerini içeren özel ikon düğmeleri bulunur.

2. Body (Ana İçerik):

- Sayfa içeriği, bir ListView içinde düzenlenmiştir ve BouncingScrollPhysics ile kaydırma efekti ekler.
- İlk olarak, konum bilgisini gösteren bir LocationCard widget'ı bulunur.
- Ardından, turistik yerleri listeleyen bir TouristPlaces widget'ı bulunur.
- Kategorilerin altında, "Recommendation" ve "Nearby From You" başlıklarıyla ayrılmış olan önerilen ve yakındaki yerleri listelemek için RecommendedPlaces ve NearbyPlaces widget'ları bulunur.

3. BottomNavigationBar (Alt Menü Çubuğu):

- Sayfanın altında, genel olarak seyahat uygulamasında yaygın olarak bulunan bir BottomNavigationBar bulunur.
- BottomNavigationBar, dört ana sekme içerir: "Home" (Ana Sayfa), "Bookmark" (Yer İşareti), "Ticket" (Bilet) ve "Profile" (Profil).
- Seçilen ve seçilmemiş öğelerin etiketleri görüntülenmez (showSelectedLabels: false, showUnselectedLabels: false).

Bu HomePage widget'ı, seyahat uygulamasının ana sayfasını gösterir ve kullanıcıya konum, turistik yerler, önerilen ve yakındaki yerler gibi içeriklere kolay erişim sağlar. Aynı zamanda, kullanıcı dostu bir tasarıma sahip olan BottomNavigationBar, uygulamanın farklı bölümlerine gezinmeyi kolaylaştırır.

Tourist_details_page.dart

Bu Flutter widget'i, bir turistik yerin detaylarını gösteren bir sayfa tasarımını içermektedir. Sayfa, turistik yerin bir resmi, başlığı, uzaklığı, değerlendirmesi, başlangıç zamanı, harita görüntüsü ve katılım butonu içerir. İlgili özellikler ve aksiyonlar şu şekilde özetlenebilir:

1. Resim ve Başlık:

- Sayfa, turistik yerin bir resmi ile başlar. Resim, konaklama ve kullanıcı deneyimini daha çekici kılmak için görsel bir öğedir.
- Başlık, turistik yerin adını temsil eder ve büyük bir başlık stili kullanılarak vurgulanır.

2. Navigasyon ve Favori Ekleme:

- Sayfanın üst kısmında, kullanıcıyı önceki sayfaya (gezilebilecek yerler listesine) geri götüren bir geri tuşu bulunmaktadır.
- Aynı zamanda sayfanın üst kısmında bir kalp ikonu da yer alır, ancak bu ikonun işlevi boştur ve şu anda bir aksiyon gerçekleştirmez.

3. Puan ve Mesafe Bilgileri:

- Başlık altında, turistik yerin puanı ve Portic Team'e olan uzaklığı gibi bilgiler yer almaktadır.
- Puan, bir yıldız ikonu ile birlikte renklendirilmiş ve 4.6 olarak belirtilmiştir.

```
Column(  
  mainAxisAlignment: MainAxisAlignment.min,  
  children: [  
    Text(  
      "4.6",  
      style: Theme.of(context).textTheme.bodySmall,  
    ), // Text  
    Icon(  
      Icons.star,  
      color: Colors.yellow[800],  
      size: 15,  
    ) // Icon  
  ],  
) // Column  
],  
) // Row
```

4. Chat Butonu:

- Sağ üst köşede bir konuşma balonu ikonu, kullanıcıya tur hakkında daha fazla bilgi almak veya başkalarıyla iletişim kurmak için bir sohbet özelliği ekleyebilecek bir potansiyel aksiyonu temsil eder.

5. Harita Görüntüsü:

- Sayfa, turistik yerin konumunu gösteren bir harita görüntüsü içerir. Bu, kullanıcının turun nerede gerçekleştiğini görmesine yardımcı olabilir.

6. Uzaklık Widget'ı:

- Sayfa, "Distance" adlı özel bir widget içerir. Bu widget, turistik yerin uzaklığını daha ayrıntılı bir şekilde gösterir.

7. Katılım Butonu:

- En altta, kullanıcının turu katılmasını sağlayan bir "Join this tour" adlı yükseltilmiş düğme bulunmaktadır.

```
Text(
  "Started in",
  style: Theme.of(context).textTheme.bodySmall,
) // Text
],
), // Column
],
), // Row
const SizedBox(height: 10),
Container(
  height: 180,
  width: double.maxFinite,
  decoration: BoxDecoration(
    borderRadius: BorderRadius.circular(15),
    color: Theme.of(context).colorScheme.secondary.withOpacity(0.1),
    image: const DecorationImage(
      image: AssetImage('assets/map.png'),
      fit: BoxFit.cover,
    ), // DecorationImage
  ), // BoxDecoration
), // Container
const SizedBox(height: 15),
const Distance(),
const SizedBox(height: 20),
ElevatedButton(
  onPressed: () {},
  style: ElevatedButton.styleFrom(
    elevation: 0,
    shape: const StadiumBorder(),
    padding: const EdgeInsets.symmetric(
      vertical: 15,
      horizontal: 8.0,
    ), // EdgeInsets.symmetric
  ),
  child: const Text("Join this tour"),
) // ElevatedButton
],
), // ListView
), // SafeArea
); // Scaffold
```

Bu sayfa, kullanıcıya turistik bir yer hakkında detaylı bilgiler sunmak ve katılım için bir düğme sağlamak amacıyla tasarlanmıştır.

Welcome_page.dart

Bu Flutter widget'i, bir karşılama sayfasını temsil etmektedir ve uygulamanın ana sayfasına yönlendiren bir "Get Started" butonu içermektedir.

1. Ara yüz Tasarımı:

- Sayfa, bir resim, başlık, alt başlık ve bir buton içeren basit bir arayüz tasarımına sahiptir.

- Resim, "assets/illustration.png" yolundan alınan bir görseli temsil eder.

- Başlık, "Explore your journey only with us" şeklinde büyük puntolarla vurgulanmıştır ve iki satırı kapsayacak şekilde tasarlanmıştır.

- Alt başlık, "All your vacations destinations are here, enjoy your holiday" şeklinde gri renk tonunda ve biraz daha küçük puntolarla gösterilmiştir.

```
ib > pages > welcome_page.dart.dart > WelcomePage
1 import 'package:flutter/material.dart';
2 import 'package:travel_app/pages/home_page.dart';
3
4 class WelcomePage extends StatelessWidget {
5   const WelcomePage({Key? key}) : super(key: key);
6
7   @override
8   Widget build(BuildContext context) {
9     return Scaffold(
10      body: SafeArea(
11        child: Center(
12          child: Padding(
13            padding: const EdgeInsets.all(16),
14            child: Column(
15              children: [
16                const Spacer(),
17                Image.asset(
18                  'assets/illustration.png',
19                ), // Image.asset
20                const SizedBox(height: 40),
21                const Text(
22                  "Explore your journey \nonly with us",
23                  textAlign: TextAlign.center,
24                  style: TextStyle(
25                    fontSize: 26,
26                    fontWeight: FontWeight.bold,
27                  ), // TextStyle
28                ), // Text
29                const SizedBox(height: 20),
30                const Text(
31                  "All your vacations destinations are here,\nenjoy your holiday",
32                  textAlign: TextAlign.center,
33                  style: TextStyle(
34                    color: Colors.black54,
35                    fontSize: 16,
36                  ), // TextStyle
37                ), // Text
38              ],
39            ),
40          ),
41        ),
42      ),
43    );
44  }
```

2. Yönlendirme Butonu:

- Sayfa altında "Get Started" adında bir buton bulunmaktadır.
- Bu butona tıklanması durumunda, kullanıcıyı `HomePage` widget'ına yönlendirmek için `Navigator` kullanılmıştır.
- `Navigator.pushReplacement` metodu, geçmişteki sayfaları temizler ve belirtilen sayfaya yönlendirir.

3. Stil ve Düzenleme:

- Widget, `SafeArea` ve `Padding` kullanılarak içeriği çerçeveler ve kenarlardan bir miktar boşluk bırakacak şekilde düzenlenmiştir.

- Butonun şekli `StadiumBorder` ile yuvarlatılmış bir formda ve gölgelendirme olmadan tasarlanmıştır.

- Butonun içeriği "Get Started" yazısı ile temsil edilmiştir.

Bu sayfa, kullanıcıyı uygulamayı keşfetmeye ve ana içeriğe geçmeye teşvik etmek için basit, görsel açıdan çekici bir tasarıma sahiptir.

SONUÇLAR

Bu çalışma, Flutter framework'ünü kullanarak basit bir seyahat uygulaması geliştirmeyi amaçlamış ve bu süreçte çeşitli önemli sonuçlara ulaşılmıştır.

Geliştirilen seyahat uygulaması, kullanıcı dostu arayüz tasarımıyla dikkat çekmektedir. Flutter'ın esnek arayüz yetenekleri sayesinde, uygulama hem estetik hem de kullanıcılar için sezgisel bir deneyim sunmaktadır.

Seyahat planlama ve rehberlik özellikleri, kullanıcıların seyahat öncesi ve sırasında ihtiyaç duydukları bilgilere kolay erişim sağlamaktadır. Rota oluşturma, otel rezervasyonu gibi işlevler, kullanıcıların seyahat deneyimini planlama ve yönetme konusunda etkili bir yardımcıdır.

Kişiselleştirilmiş deneyim, uygulamanın kullanıcının tercihlerini anlamak ve buna göre önerilerde bulunmak konusundaki başarısını göstermektedir. Bu, kullanıcıların daha önceki seyahat deneyimlerinden elde edilen verileri kullanarak, onlara özel öneriler sunma açısından önemlidir.

Bu çalışmanın genelinde elde edilen sonuçlar, Flutter kullanarak basit bir seyahat uygulaması geliştirmenin teknik olarak başarılı ve kullanıcı deneyimi odaklı bir süreç olduğunu ortaya koymaktadır. Geliştirilen uygulama, seyahatle ilgili temel ihtiyaçlara pratik çözümler sunarak kullanıcılara güçlü bir mobil deneyim sunmaktadır.

KAYNAKLAR

- [1] <https://pub.dev/>
- [2] <https://docs.flutter.dev/codelabs>
- [3] <https://halilozel1903.medium.com/dart-programlama-dili-eeafb64ad300>
- [4] <https://www.youtube.com/playlist?list=PLixrf2q8roU23XGwz3Km7sQZFTdB996iG>
- [5] <https://www.youtube.com/@createdbykoko/videos>
- [6] <https://www.udemy.com/course/learn-flutter-beginners-course/>
- [7] <https://www.udemy.com/course/sifirdan-flutter-ile-android-ve-ios-apps-development/>
- [8] <https://github.com/>
- [9] <https://www.cenuta.com/blog/widget-nedir-ve-ne-ise-yarar-telefonlarda-widget-nasil-eklenir-ve-kaldirilir/>
- [10] <https://www.mucahitakin.com/blog/2023/04/16/stateless-widget/>

ÖZGEÇMİŞ

Aslı Kuşçu, 2000 İstanbul doğumludur. İlk ve orta öğrenimini Atışalanı İlköğretim Okulu'nda, lise eğitimini ise Füsun Yönder Anadolu Lisesi'nde tamamlamıştır. Yıldız Teknik Üniversitesi'nde bir yıl İngilizce hazırlık okuduktan sonra Matematik Mühendisliği Bölümü'ne başlamıştır. 2023 yılında Ecodation şirketinde ilk stajını tamamlamıştır. Şu an Yıldız Teknik Üniversitesi Matematik Mühendisliği 4. sınıf öğrencisidir.