ANKARA UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

(BLM4537-A) IOS İle Mobil Uygulama Geliştirme

Alp Ertunga Elgün

19290238

Ergast Developer API / Ergast F1 Database

Ergast API ticari olmayan projeler için geliştirilmiş, deneysel bir web servisidir. API'i kullanarak motor yarışlarının tarihi verilerini içinde tutan Ergast Database'e kolayca erişebilir, bilgileri elde edebiliriz.

Ergast websitesine buradan ulaşabilirsiniz: http://ergast.com/mrd/

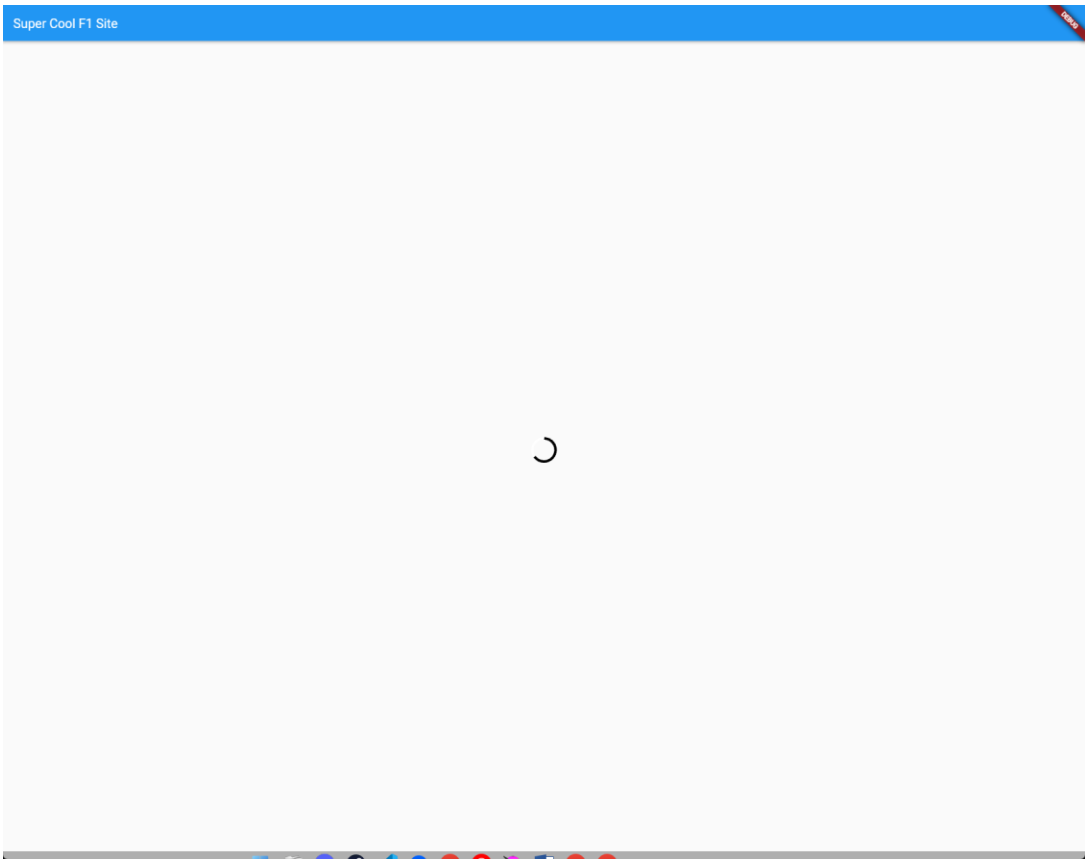
F1 Database Ve Analiz Uygulamasi

Bu web uygulamasının asıl amacı, Ergast Database'inde karmaşık şekillerde bulunan F1 verilerini okunabilir bir şekilde kullanıcıya sunmak.

Ayrıca direk Database'de bulunmayan farklı analiz bilgilerini de hesaplayarak kullanıcıya büyük kolaylıklar sağlamak. Web uygulamaları bu tarz işlemler için biçilmiş kaftan, yeni verileri kullanıcının aygıtına yüklemeye gerek kalmadan database'e ekleyebilir, kolayca güncelleyebiliriz.

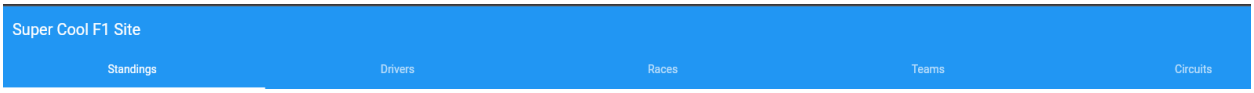
Loading Ekrani

Başlangıçta API'dan verileri çekerken ve bu verileri işlerken kullanıcıya arkada işlemin devam ettiğini belli etmek için bir loading ekranı kullanıyoruz:



Top Navigation Bar

Geliştirdiğimiz web uygulamasında beş adet sayfa bulunuyor. Kullanıcıların bu sayfalar arasında kolayca geçebilmesi için top bar kullandık. Ayrıca sağ ya da sola swipe yaparak da sayfalar arasında kolayca geçiş yapabilirler.



Bottom Bar

Kullanıcının hangi senenin verilerine bakmak istediğini görmek için ise sayfanın en altına bir “year picker widget’ ekledik. API istekleri ve olmayan bilgilerin hesaplanması sene değiştikçe yapıldığı için kullanıcı yeni bir sene seçtiğinde yüklenme ekranı ile karşılaşıyor.

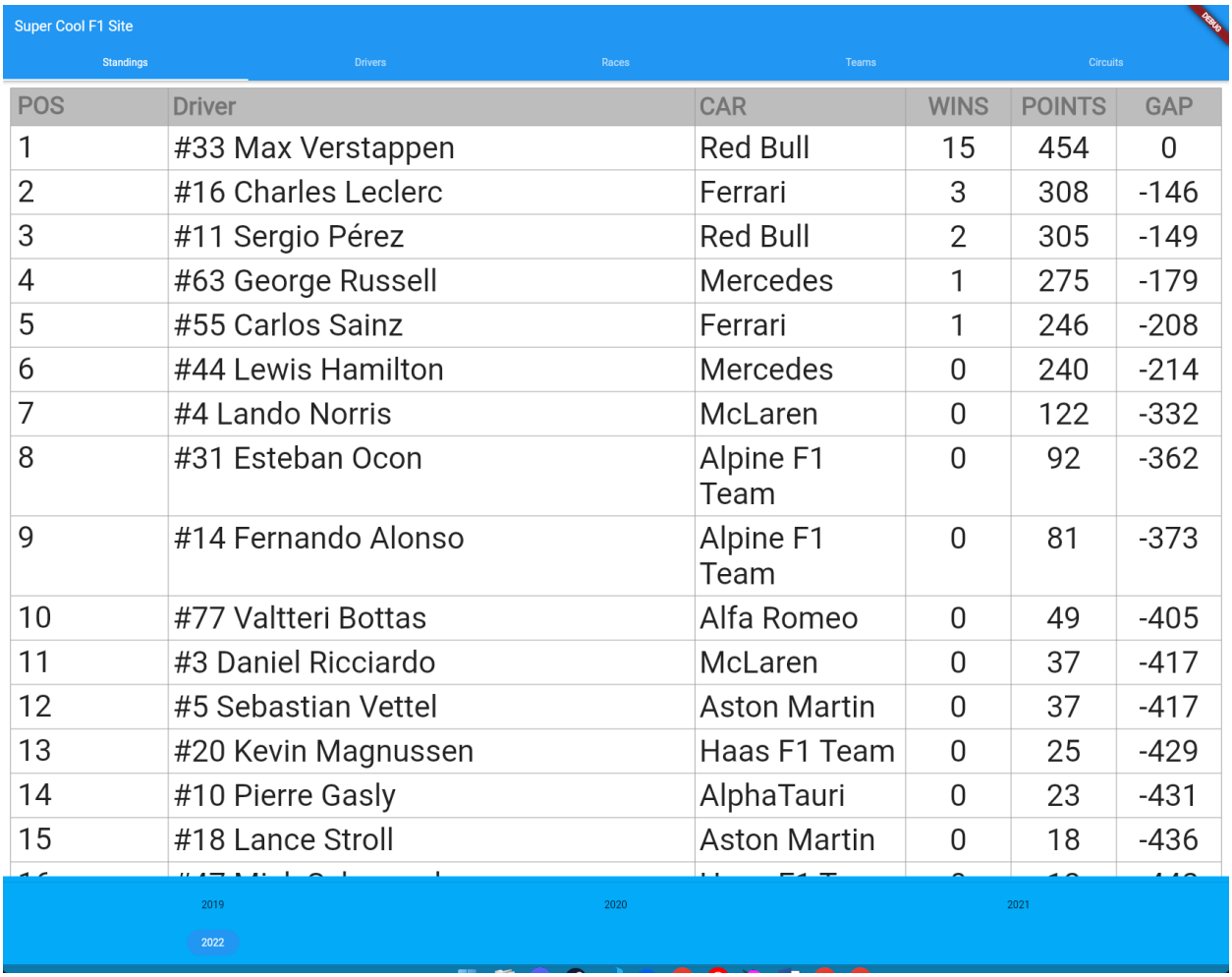
Bar’a sığmayan seneler’e ulaşmak için kullanıcı mouse wheel’i kullanabilir. Ya da telefonda swipe yaparak rahatça ulaşabilir.

Standings Sayfasi

Bu sayfada kullanıcı seçilen senenin puan tablosunu görebilir. Yarış pilotlarının isimlerinin sağındaki #1 gibi numaralar pilotların kendi isimlerine adanmış yarış numaralarıdır, bu numaralar eski senelerde olmadığı için her pilotun isminin başında bulunmayabilir.

Gap bilgisi Database'de tutulmadığı için kendimiz hesaplıyoruz. Gap değeri birinci pilot ile diğer pilotların arasındaki puan farkıdır. Geçmişten beri farklar hep negative olarak yazılır.

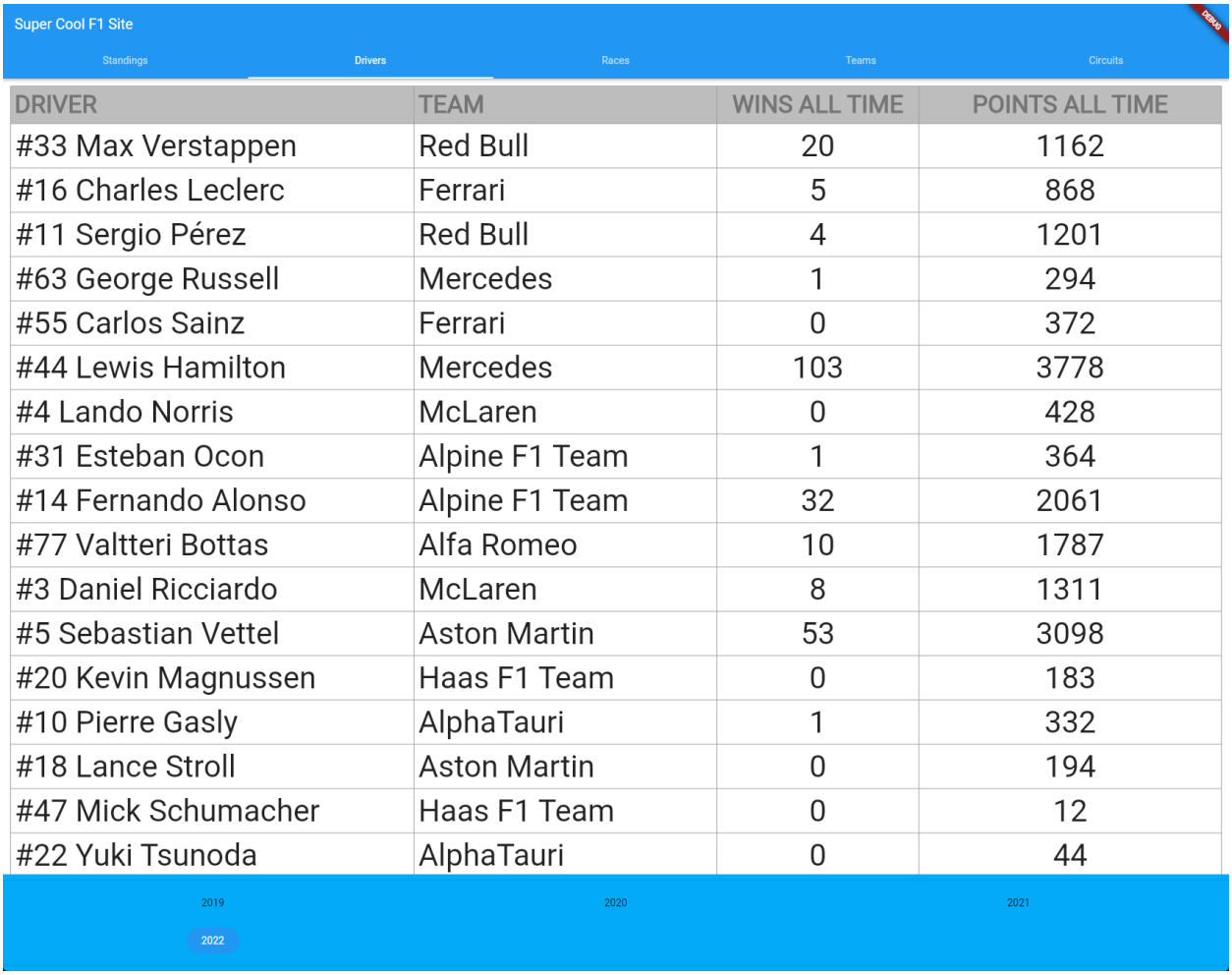
Her sezonda eşit sayıda pilot yarışmıyor. Eğer yarışçı sayısı ekrana sığmiyorsa scroll ya da swipe ile kaydırılabilir.



POS	Driver	CAR	WINS	POINTS	GAP
1	#33 Max Verstappen	Red Bull	15	454	0
2	#16 Charles Leclerc	Ferrari	3	308	-146
3	#11 Sergio Pérez	Red Bull	2	305	-149
4	#63 George Russell	Mercedes	1	275	-179
5	#55 Carlos Sainz	Ferrari	1	246	-208
6	#44 Lewis Hamilton	Mercedes	0	240	-214
7	#4 Lando Norris	McLaren	0	122	-332
8	#31 Esteban Ocon	Alpine F1 Team	0	92	-362
9	#14 Fernando Alonso	Alpine F1 Team	0	81	-373
10	#77 Valtteri Bottas	Alfa Romeo	0	49	-405
11	#3 Daniel Ricciardo	McLaren	0	37	-417
12	#5 Sebastian Vettel	Aston Martin	0	37	-417
13	#20 Kevin Magnussen	Haas F1 Team	0	25	-429
14	#10 Pierre Gasly	AlphaTauri	0	23	-431
15	#18 Lance Stroll	Aston Martin	0	18	-436

Drivers Sayfasi

İkinci sayfada kullanıcı, seçilen sezon yarışmış pilotların detaylı bilgilerine ulaşabilir. Yarış hayatı boyunca kazandığı galibiyet ve puan sayısına da buradan ulaşabilir. Bu bilgılar Database'e direk olarak tutulmadığı için API'dan pilotun yarıştığı her sezon için taker teker yarış kazanma ve aldığı puan bilgilerini istiyoruz. Sonra bu değerleri topluyoruz.



DRIVER	TEAM	WINS ALL TIME	POINTS ALL TIME
#33 Max Verstappen	Red Bull	20	1162
#16 Charles Leclerc	Ferrari	5	868
#11 Sergio Pérez	Red Bull	4	1201
#63 George Russell	Mercedes	1	294
#55 Carlos Sainz	Ferrari	0	372
#44 Lewis Hamilton	Mercedes	103	3778
#4 Lando Norris	McLaren	0	428
#31 Esteban Ocon	Alpine F1 Team	1	364
#14 Fernando Alonso	Alpine F1 Team	32	2061
#77 Valtteri Bottas	Alfa Romeo	10	1787
#3 Daniel Ricciardo	McLaren	8	1311
#5 Sebastian Vettel	Aston Martin	53	3098
#20 Kevin Magnussen	Haas F1 Team	0	183
#10 Pierre Gasly	AlphaTauri	1	332
#18 Lance Stroll	Aston Martin	0	194
#47 Mick Schumacher	Haas F1 Team	0	12
#22 Yuki Tsunoda	AlphaTauri	0	44

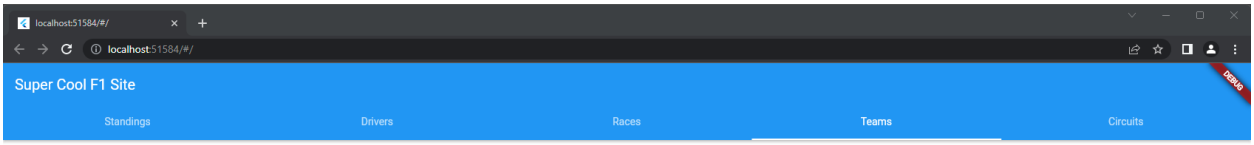
Races Sayfasi

Bu sayfada kullanıcı, seçilen sezoda yarışılan tüm yarışlara ulaşabilir, yarışın ismi ve yarışın yapıldığı pist ismine, yarışın kazananına, pole pozisyonunda yarışa kimin başladığına kolayca ulaşabilir.

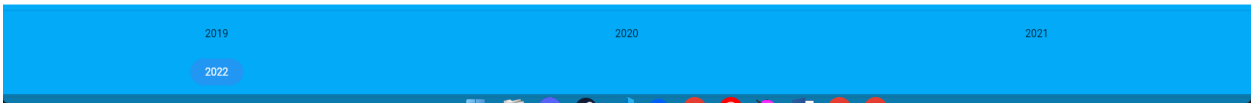
Yine bilgiler sayfaya sığmıyor ise scroll ya da swipe yaparak kaydırabilir.

Teams Sayfasi

Bu sayfada kullanıcı, seçilen sezonda yarışmış olan araba markalarının kaç puan aldıklarını ve kaç yarış kazandığını görebilir.

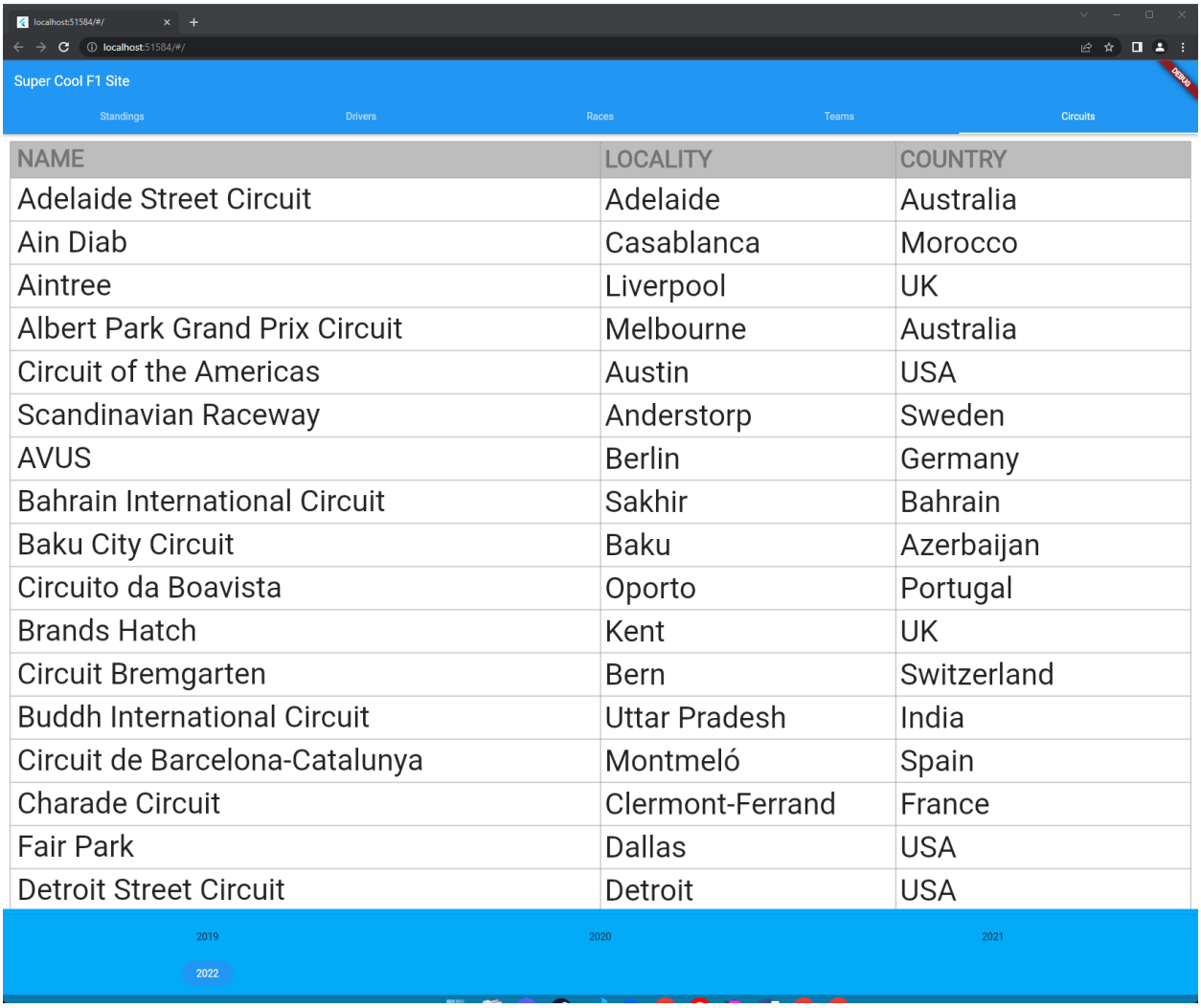


NAME	NATIONALITY	WINS	POINTS
Red Bull	Austrian	17	759
Ferrari	Italian	4	554
Mercedes	German	1	515
Alpine F1 Team	French	0	173
McLaren	British	0	159
Alfa Romeo	Swiss	0	55
Aston Martin	British	0	55
Haas F1 Team	American	0	37
AlphaTauri	Italian	0	35
Williams	British	0	8



Circuits Sayfasi

Son sayafada ise hangi pistlerde hangi yarışların yapıldığını, pistlerin hangi şehirde ve ülkede olduğunu görebilir.



NAME	LOCALITY	COUNTRY
Adelaide Street Circuit	Adelaide	Australia
Ain Diab	Casablanca	Morocco
Aintree	Liverpool	UK
Albert Park Grand Prix Circuit	Melbourne	Australia
Circuit of the Americas	Austin	USA
Scandinavian Raceway	Anderstorp	Sweden
AVUS	Berlin	Germany
Bahrain International Circuit	Sakhir	Bahrain
Baku City Circuit	Baku	Azerbaijan
Circuito da Boavista	Oporto	Portugal
Brands Hatch	Kent	UK
Circuit Bremgarten	Bern	Switzerland
Buddh International Circuit	Uttar Pradesh	India
Circuit de Barcelona-Catalunya	Montmeló	Spain
Charade Circuit	Clermont-Ferrand	France
Fair Park	Dallas	USA
Detroit Street Circuit	Detroit	USA

Home Page State

Bu state'de API'ın URL'sini alıyoruz. URL'i seçilen sezona göre modifiye ediyoruz, API kullanarak dolduracağımız listeleri yaratıyoruz ve en son olarak fetchPosts fonksiyonunda get komutu ile API'a isteği gönderip dönüş olarak aldığımız XML dosyasını parse ediyoruz.

Main.dart:

```
class _MyHomePageState extends State<MyHomePage> {
  String year = "2022";
  String url = "https://ergast.com/api/f1/2022/DriverStandings";

  var _driverList = [];
  var _raceList = [];
  var _constructorList = [];
  var _pointsList = [];
  var _winsList = [];
  var _circuitsList = [];
  var _constructorStandings = [];

  var _isLoaded = false;
  var _selectedDate = DateTime(2022);

  void fetchPosts() async {
    try {
      var response = await get(Uri.parse(url));
      var parsed = XmlDocument.parse(response.body);

      final driverList = parsed
          .findAllElements('Driver')
          .map((xmlElement) => Driver.fromXmlElement(xmlElement))
          .toList();

      final constructorList = parsed
          .findAllElements('Constructor')
          .map((xmlElement) => Constructor.fromXmlElement(xmlElement))
          .toList();
```

Data Classlari

XML dosyasındaki tag ve elementleri kolayca bulabilmek için “XML.dart” package’ini kullanıyoruz.

FindAllElements fonksiyonunun döndürdüğü iteratable’ı, xml’den çekdiğimiz bilgiler ile doldurmak istediğimiz class’ın constructor’ı ile maplıyoruz. Bu şekilde her class için ayrı olarak parse kodu yapmamıza gerek kalmıyor.

Ornek olarak driver.dart class’i:

```dart
class Driver {
  final int permanentNumber;
  final String givenName;
  final String familyName;
  final String dateOfBirth;
  final String nationality;
  final String driverId;
  int totalWin;
  int totalPoint;

  Driver(
      this.permanentNumber,
      this.givenName,
      this.familyName,
      this.dateOfBirth,
      this.nationality,
      this.driverId,
      this.totalWin,
      this.totalPoint);

  factory Driver.fromXmlElement(XmlElement xmlElement) => Driver(
      int.parse(Utils.firstOrNull(xmlElement.findElements('PermanentNumber'))),
      Utils.firstOrNull(xmlElement.findElements('GivenName')),
      Utils.firstOrNull(xmlElement.findElements('FamilyName')),
      Utils.firstOrNull(xmlElement.findElements('DateOfBirth')),
      Utils.firstOrNull(xmlElement.findElements('Nationality')),
      Utils.emptyIfNull(xmlElement.getAttribute('driverId')),
      0,
      0);
```

Bunun gibi constructor.dart, circuit.dart, race.dart, standings.dart classlarimiz mevcut.

Static Utils Fonksiyonlari

Database'de ya da API'da oluşabilecek herhangi bir hatanın uygulamayı bug'a sokmaması için xml'den çektiğimiz bilgilerin doğruluğundan emin olmalıyız. Utils class'ının içindeki static fonksiyonlar bunu sağlıyor:

```dart
import 'package:xml/xml.dart';

class Utils {
  static String firstOrNull(Iterable<XmlElement> iterable) {
    if (iterable.isEmpty) return "-1";
    return iterable.single.text;
  }

  static XmlElement findAndfirst(XmlElement iterable, String elementName) {
    return iterable.findElements(elementName).first;
  }

  static String emptyIfNull(String? str) {
    if (str == null) return "";
    return str;
  }
}
```

Dynamic Hesaplar

fetchPost fonksiyonunda sadace APİ'dan data almakla kalmıyor, ilk aldığımız bu datalardan dynamic olarak oluşturduğumuz yeni API call'ları ile kullanıcının analiz için kullanabileceği yeni bilgiler üretiyoruz.

Örnek olarak bir pilot'un bugüne kadar katıldığı her sezonda toplam kaç puan aldığını ve kaç yarış kazandığını hesaplamak için aşağıdaki code'u kullanıyoruz:

Main.cs, fetchPost fonksiyonu:

```
await Future(
  () => {
    driverList.forEach((element) async {
      String driverId = element.driverId;
      String url =
          "http://ergast.com/api/f1/drivers/$driverId/driverStandings";

      var response = await get(Uri.parse(url));
      var parsed = XmlDocument.parse(response.body);
      var itar = parsed.findAllElements('DriverStanding');
      for (var xmlElement in itar) {
        element.totalWin +=
            int.parse(Utils.emptyIfNull(xmlElement.getAttribute('wins')));
        element.totalPoint += int.parse(
            Utils.emptyIfNull(xmlElement.getAttribute('points')));
      }
      setState(() {
        _driverList = driverList;
      });
    })
  },
); // Future
```

Bu ekstra call'lari async yaparak hesaplamalarin yapilmasini ana thred'den ayiriyoruz. Bu sayede kullanici arka tarafta islemler yapilmasina ragmen uygulama ile etkilesimde kalmaya devam edebiliyor.

Loading Sayfasina Girme ve Cikma

En son tüm işlemler bittiğinde çağırdığımız setState fonksiyonunda, ilk başta oluşturup sayfalar'da kullandığımız listeler yerine ile yeni elde ettiğimiz listeleri atıyoruz.

Tablo oluştururken ilk satır'daki başlıkları oluşturmak için bu listelerin 0. Index'ine default value'ler insert ediyoruz. Ve en son olarak da _isLoaded'i true yapıyoruz. Bu sayede loading ekranından çıkıyoruz.

Main.dart, fetchPost fonksiyonun en sonu:

```
await Future(() => setState(() {
      _driverList = driverList;

      _constructorList = constructorList;
      _constructorList.insert(0, Constructor("Name", "Name", []));

      _pointsList = points;
      _pointsList.insert(0, "0");

      _winsList = wins;
      _winsList.insert(0, "0");

      _raceList = races;
      _raceList.insert(0, Race("0", "0", "0", "0", "0"));

      _circuitsList = circuitList;
      _circuitsList.insert(0, Circuit("name", "loc", "count"));

      _constructorStandings = constructorStandings;
      _constructorStandings.insert(
          0, ConstructorStanding("-1s", "-1s", "name", "nat"));

      _isLoaded = true;
    })); // Future
```

Build fonksiyonu

Build fonksiyonunda basit olarak sayfa düzenini oluşturuyoruz. Sayfaların nasıl gözükeceğini pages.dart class'ında static olarak ayarlıyoruz.

Return yaparken kontrol ettiğimiz _isLoaded değişkeni ile loading ekranını mı yoksa ana ekranı mı açmalıyız karar veriyoruz.

Main.dart:

```
@override
Widget build(BuildContext context) {
  Pages.fontSizeBig = MediaQuery.of(context).size.width * 0.02;
  Pages.fontSizeSmall = MediaQuery.of(context).size.width * 0.025;
  return _isLoaded
      // If loaded
      ? MaterialApp(
          theme: ThemeData.from(
              colorScheme: ColorScheme.fromSwatch(
                  primarySwatch: Colors.blue,
                  cardColor: Colors.white,
                  accentColor: Colors.white,
                  primaryColorDark: Colors.white,
                  backgroundColor: Colors.white,
                  errorColor: Colors.white)), // ColorScheme.fromSwatch // ThemeData.from
          home: DefaultTabController(
              length: 5,
              child: Scaffold(
                  appBar: Pages.topBar(widget.title),
                  body: TabBarView(children: [
                    Pages.standingsPage(_driverList, _constructorList,
                        _winsList, _pointsList),
                    Pages.driversPage(_driverList, _constructorList),
                    Pages.racesPage(_raceList),
                    Pages.teamsPage(_constructorStandings),
                    Pages.circuitsPage(_circuitsList)
                  ]), // TabBarView
                  bottomNavigationBar: Pages.yearPicker(_selectedDate,
                      yearInput, MediaQuery.of(context).size.height)))) // Scaffold // DefaultTabController // MaterialApp
      // Still loading
      : Pages.loadingApp(widget.title);
```

En yukarıda atama yaptığımız iki static değer pages class'ında bulunuyor. Bu static değerler ile sayfanın boyutuna göre gap, padding, font size gibi değerleri değiştirerek sayfayı farklı ekran çozunurlüklerine uygun hale getiriyoruz.

Year Picker

Kullanıcının istediği seneyi seçmesi için YearPiçker widget'ini kullanıyoruz. Yeni bir yıl seçilirse, yearInput fonksiyonunu çağırarak sayfanın yenilenmesini sağlıyoruz:

Pages.dart:

```
static Stack yearPicker(DateTime _selectedDate, Function(DateTime) yearInput,
    double contextHeight) {
  return Stack(
    children: [
      Container(
        color: Color.fromARGB(255, 2, 170, 248),
        height: contextHeight * 0.1,
        child: YearPicker(
          firstDate: DateTime.utc(1950),
          initialDate: DateTime.utc(2022),
          lastDate: DateTime.utc(2022),
          selectedDate: _selectedDate,
          onChanged: yearInput,
        ), // YearPicker
      ), // Container
    ],
  ); // Stack
```

Main.dart:

```
void yearInput(DateTime input) {
  setState(() {
    _isLoaded = false;
    _selectedDate = input;
  });

  year = input.year.toString();
  url = "https://ergast.com/api/f1/$year/DriverStandings";

  fetchPosts();
}
```

Tablolar

Tabloları hazırlamak için List<TableRow>.generate kullanıyoruz. Bu sayede listenin büyüklüğüne bakmadan tablo oluşturabiliyoruz. Index'İ kullanarak API'dan get komutuyla oluşturduğumuz listelerdeki doğru bilgilere ulaşabiliyoruz.

Pages.dart:

```
static SingleChildScrollView driversPage(
    List _driverList, List _constructorList) {
  return SingleChildScrollView(
    child: Padding(
      padding: EdgeInsets.all(10),
      child: Table(
        columnWidths: {
          0: FlexColumnWidth(0.4),
          1: FlexColumnWidth(0.3),
          2: FlexColumnWidth(0.2),
          3: FlexColumnWidth(0.3),
        },
        border: TableBorder.all(color: Colors.grey),
        children: List<TableRow>.generate(_driverList.length, (index) {
          final driver = _driverList[index];
          final constructor = _constructorList[index];
          final win = driver.totalWin;
          final point = driver.totalPoint;
```

Başlıkları ilk satıra yerleştirmek için setState fonksiyonun sonunda listelere sıfırıncı index'e default value insert etmiştik, bunu kullanarak eğer ındex sıfır ise başlıkları yazabiliyoruz.

Basliklarin yazilmasi:

```
if (index == 0) {
  return TableRow(
    decoration: BoxDecoration(color: Colors.grey[400]),
    children: [
      Padding(
          padding: EdgeInsets.all(6.0),
          child: Text(
            "DRIVER",
            textAlign: TextAlign.left,
            style: TextStyle(
                fontWeight: FontWeight.bold,
                color: Colors.grey[600],
                fontSize: fontSizeBig), // TextStyle
          )), // Text // Padding
      Padding(
          padding: EdgeInsets.all(6.0),
          child: Text(
            "TEAM",
            textAlign: TextAlign.left,
            style: TextStyle(
                fontWeight: FontWeight.bold,
                color: Colors.grey[600],
                fontSize: fontSizeBig), // TextStyle
          )), // Text // Padding
      Padding(
          padding: EdgeInsets.all(6.0),
          child: Text(
            "WINS ALL TIME",
            textAlign: TextAlign.center,
            style: TextStyle(
                fontWeight: FontWeight.bold,
                color: Colors.grey[600],
                fontSize: fontSizeBig), // TextStyle
          )), // Text // Padding
      Padding(
          padding: EdgeInsets.all(6.0),
          child: Text(
            "POINTS ALL TIME",
            textAlign: TextAlign.center,
            style: TextStyle(
```

Devamında ise her row için istediğimiz bilgileri dolduruyoruz:

```
return TableRow(
  children: [
    Padding(
        padding: EdgeInsets.all(6.0),
        child: Text(
          driver.GetNumber() +
              " " +
              driver.givenName +
              " " +
              driver.familyName,
          textAlign: TextAlign.left,
          style: TextStyle(fontSize: fontSizeSmall),
        )), // Text // Padding
    Padding(
        padding: EdgeInsets.all(6.0),
        child: Text(
          constructor.name,
          textAlign: TextAlign.left,
          style: TextStyle(fontSize: fontSizeSmall),
        )), // Text // Padding
    Padding(
        padding: EdgeInsets.all(6.0),
        child: Text(
          win.toStringAsFixed(0),
          textAlign: TextAlign.center,
          style: TextStyle(fontSize: fontSizeSmall),
        )), // Text // Padding
    Padding(
        padding: EdgeInsets.all(6.0),
        child: Text(
          point.toStringAsFixed(0),
          textAlign: TextAlign.center,
          style: TextStyle(fontSize: fontSizeSmall),
        )), // Text // Padding
  ],
```

AndroidManifest Internet Erisim Izini

Eğer telefona apk olarak uygulamayı indirirsek, android işletim sisteminden internet kullanmak için izin istememiz gerekiyor. Bu izini AndroidManifest.xml'den isteyebiliyoruz:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.flutter_f1">
     <uses-permission android:name="android.permission.INTERNET"/>
```

Dikkat Edilmesi Gerekenler

OdevBuilds klasöründen web build'ine ve apk dosyasına ulaşabilirsiniz. Malesef herhangi bir apple ürününe ulaşamadığım için IOS build'i alamadım.

Eğer uygulama yüklenme ekranında takılı kalırsa internetinizin olduğuna emin olunuz. Eğer internetiniz varsa http://ergast.com/mrd/ web sitesinden API'in durumunu kontrol ediniz. Bazen yoğunluktan dolayı çökebiliyor. Böyle bir durumda 1-2 saat sonra tekrar deniyebilirsiniz.