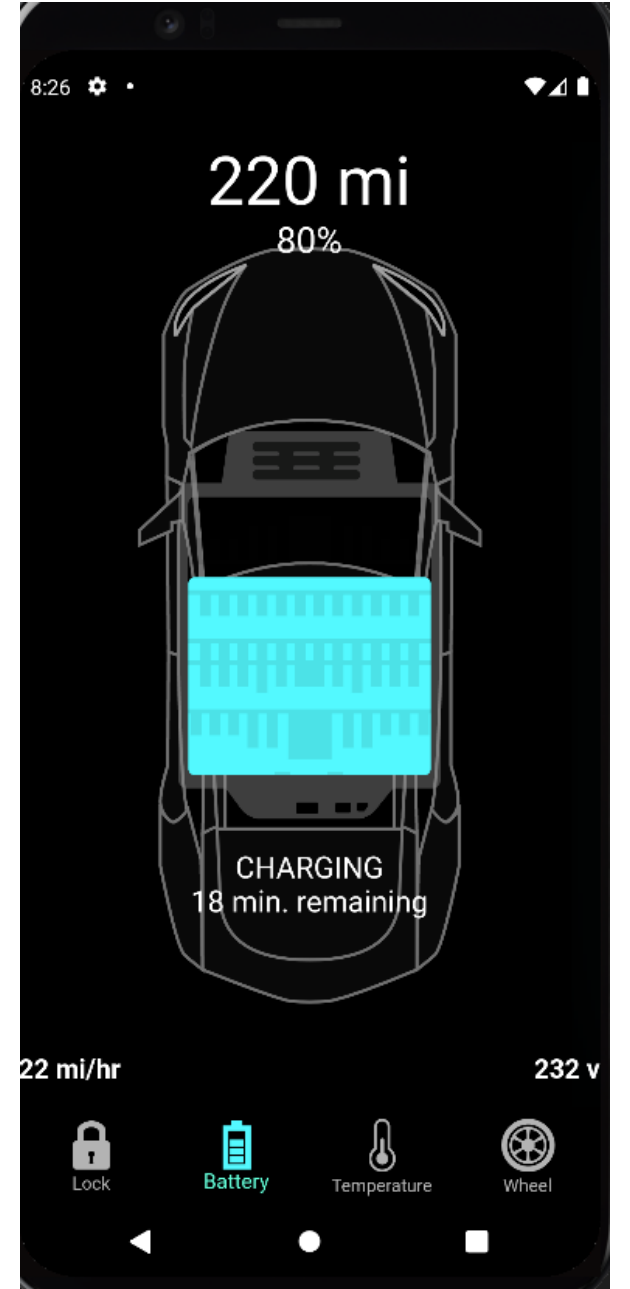
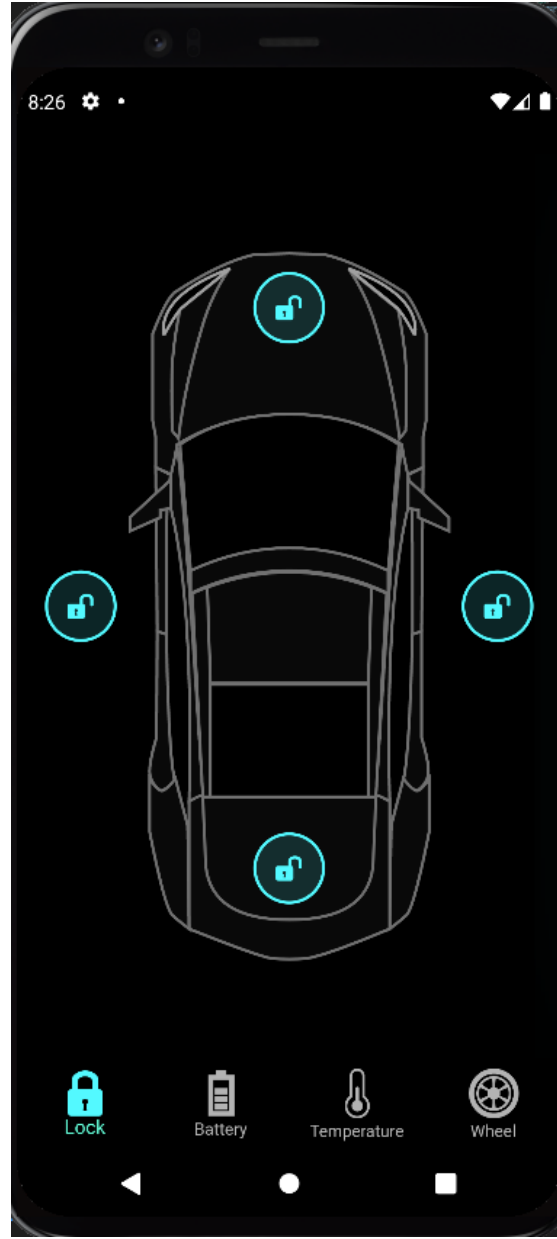
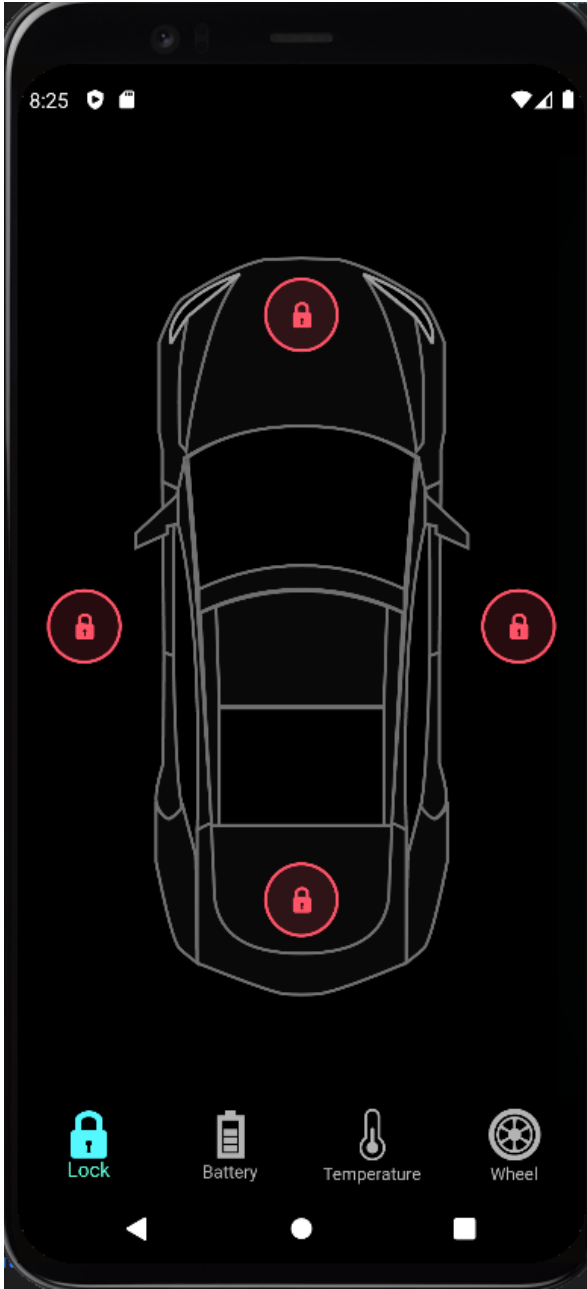


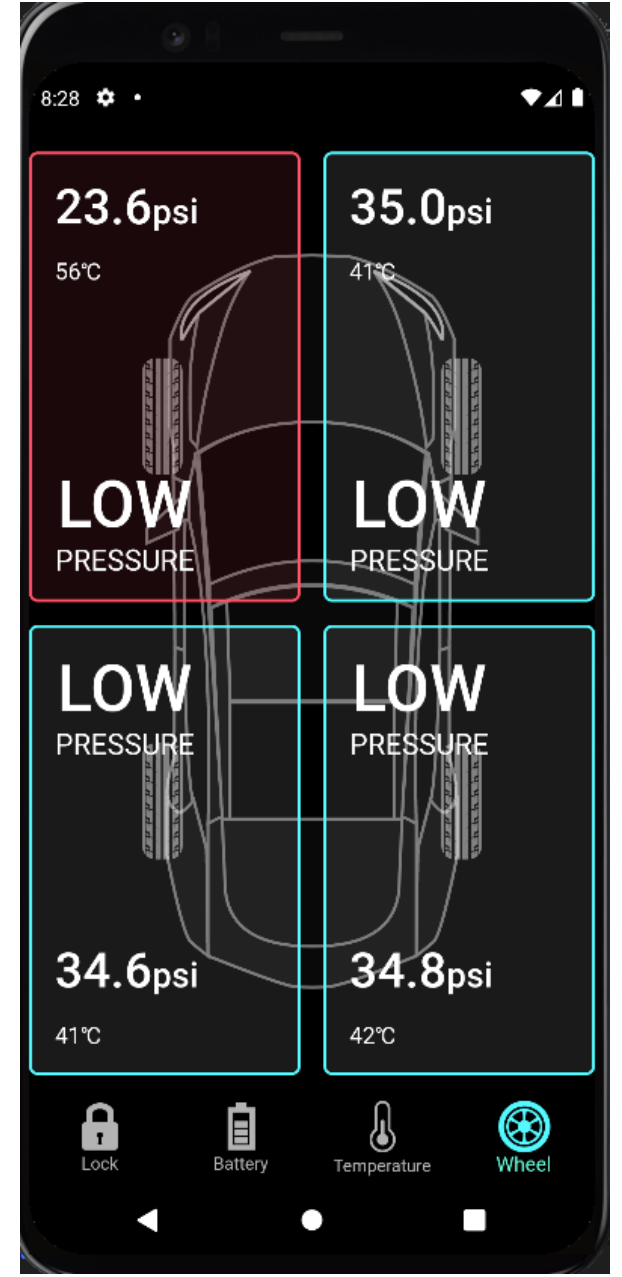
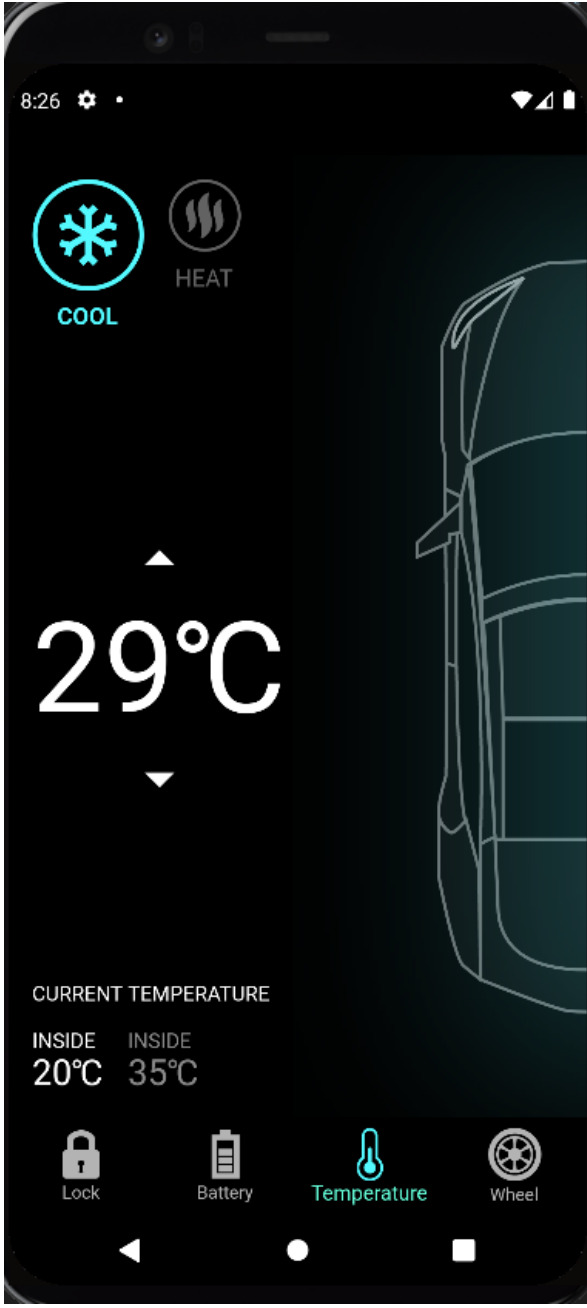


Ödev 7

Açıklama	
Created	@December 12, 2021 3:37 PM
Link	

Uygulamanın Youtube Linki : <https://www.youtube.com/watch?v=1BO0zSS-84E>

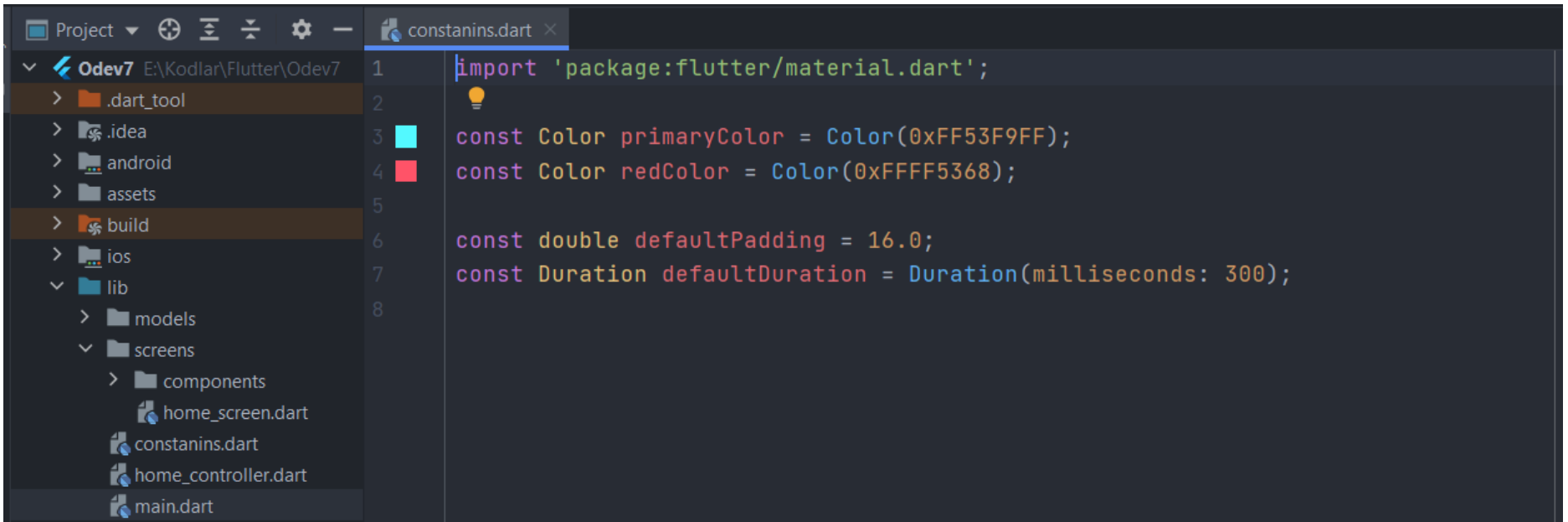




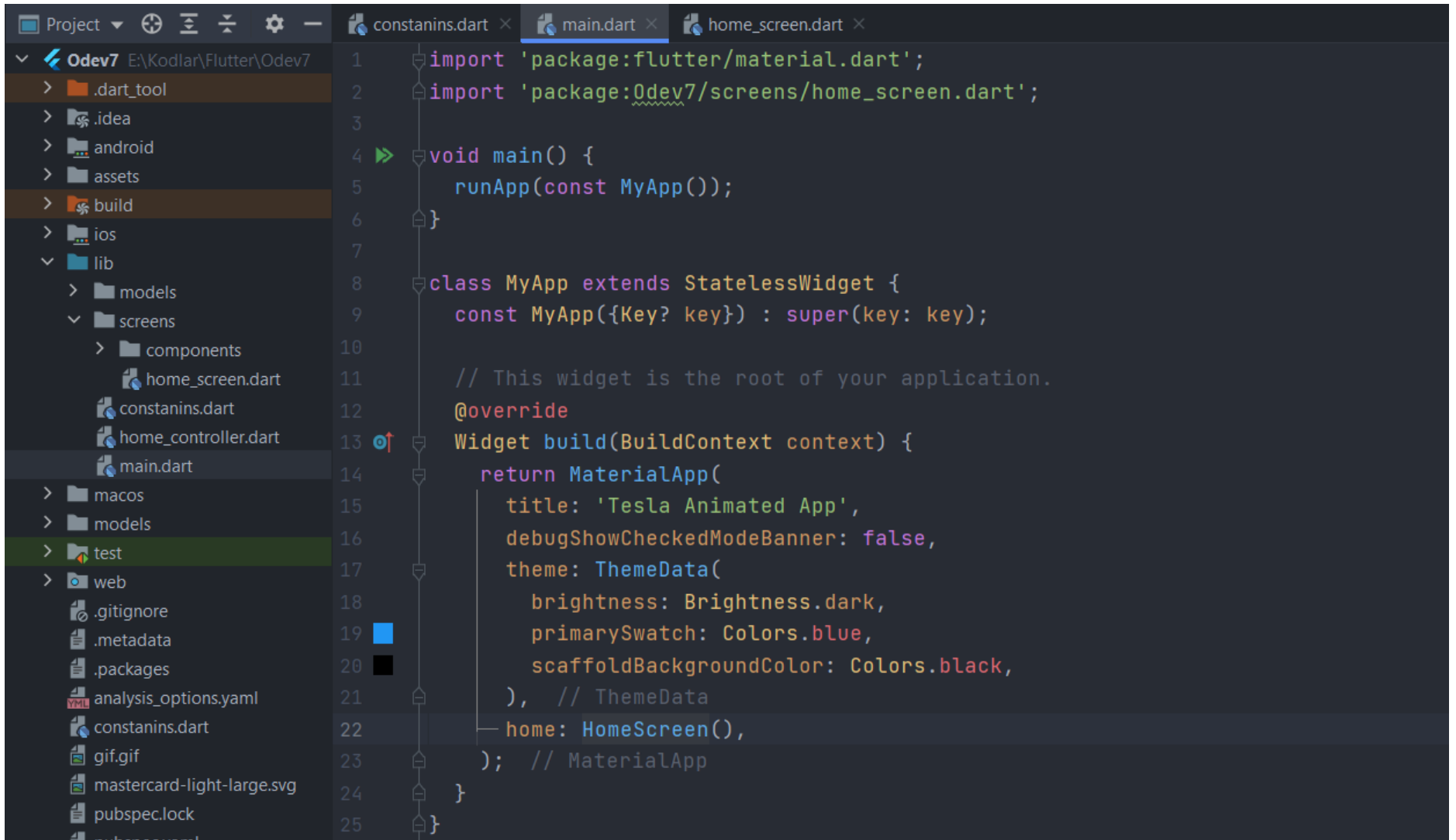
Uygulamamız main.dart dosyasından başlamaktadır.

Uygulama içerisinde kullanacağımız sabitler constanins.dart dosyası içerisinde bulunmaktadır.

constanins.dart



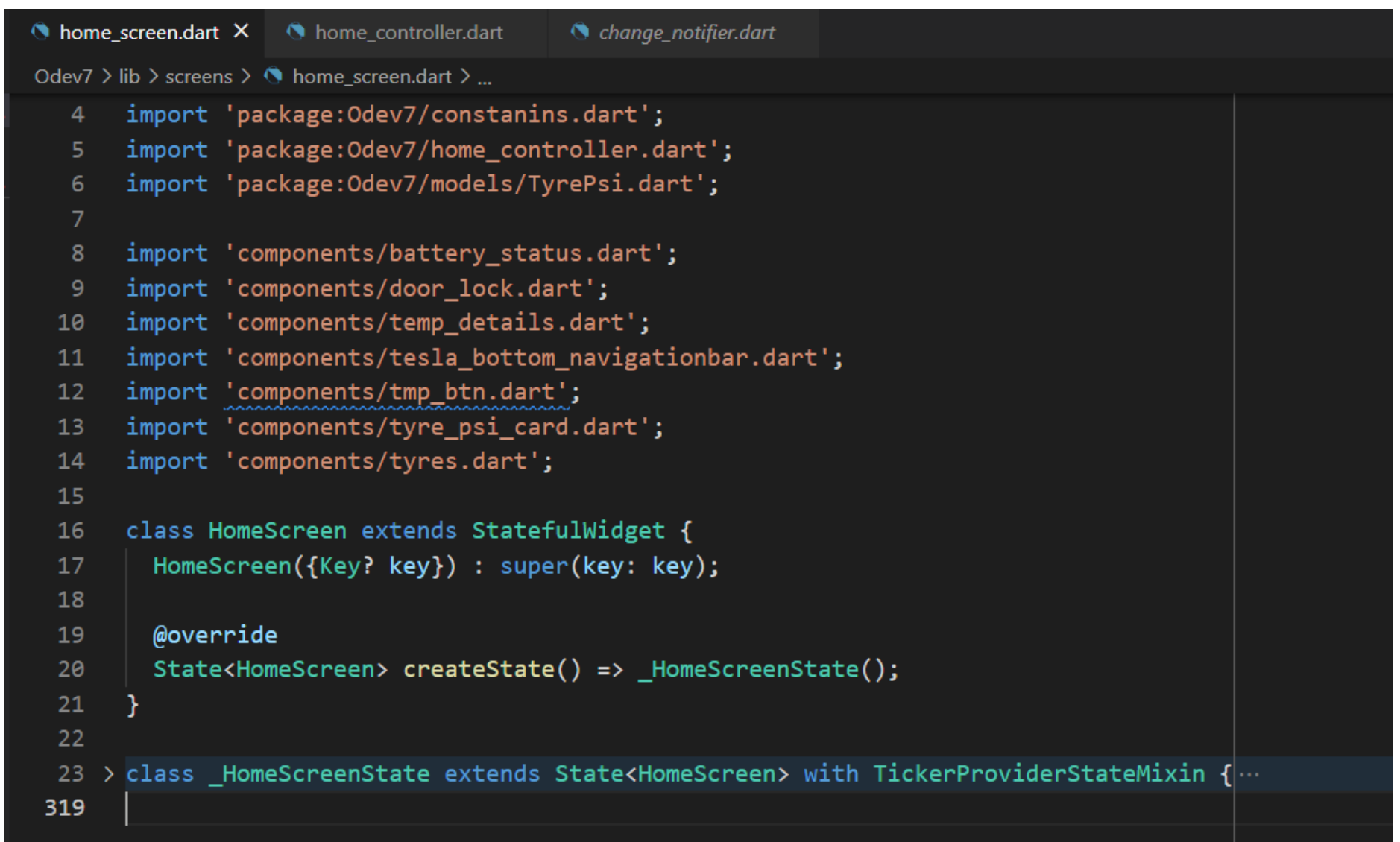
main.dart



```
1 import 'package:flutter/material.dart';
2 import 'package:Odev7/screens/home_screen.dart';
3
4 void main() {
5   runApp(const MyApp());
6 }
7
8 class MyApp extends StatelessWidget {
9   const MyApp({Key? key}) : super(key: key);
10
11   // This widget is the root of your application.
12   @override
13   Widget build(BuildContext context) {
14     return MaterialApp(
15       title: 'Tesla Animated App',
16       debugShowCheckedModeBanner: false,
17       theme: ThemeData(
18         brightness: Brightness.dark,
19         primarySwatch: Colors.blue,
20         scaffoldBackgroundColor: Colors.black,
21       ), // ThemeData
22       home: HomeScreen(),
23     ); // MaterialApp
24   }
25 }
```

MaterialApp widget kullanılmaktadır. home parametresine HomeScreen verilmiştir. HomeScreen kendi yazdığımız sınıf.

home_screen.dart



```
home_screen.dart X home_controller.dart change_notifier.dart
Odev7 > lib > screens > home_screen.dart > ...
4 import 'package:Odev7/constanins.dart';
5 import 'package:Odev7/home_controller.dart';
6 import 'package:Odev7/models/TyrePsi.dart';
7
8 import 'components/battery_status.dart';
9 import 'components/door_lock.dart';
10 import 'components/temp_details.dart';
11 import 'components/tesla_bottom_navigationbar.dart';
12 import 'components/tmp_btn.dart';
13 import 'components/tyre_psi_card.dart';
14 import 'components/tyres.dart';
15
16 class HomeScreen extends StatefulWidget {
17   HomeScreen({Key? key}) : super(key: key);
18
19   @override
20   State<HomeScreen> createState() => _HomeScreenState();
21 }
22
23 > class _HomeScreenState extends State<HomeScreen> with TickerProviderStateMixin { ...
319
```

HomeScreen sınıfını içerir. StatefulWidget'tan miras alır.

```

23 class _HomeScreenState extends State<HomeScreen> with TickerProviderStateMixin {
24   final HomeController _controller = HomeController();
25
26   late AnimationController _batteryAnimationController;
27   late Animation<double> _animationBattery;
28   late Animation<double> _animationBatteryStatus;
29
30   late AnimationController _tempAnimationController;
31   late Animation<double> _animationCarShift;
32   late Animation<double> _animationTempShowInfo;
33   late Animation<double> _animationCoolGlow;
34
35   late AnimationController _tyreAnimationController;
36   // We want to animate each tyre one by one
37   late Animation<double> _animationTyre1Psi;
38   late Animation<double> _animationTyre2Psi;
39   late Animation<double> _animationTyre3Psi;
40   late Animation<double> _animationTyre4Psi;
41
42   late List<Animation<double>> _tyreAnimations;

```

burada daha sonrasında işimize yarayacak olan değişkenler tanımlandı. late verilerek çalışma zamanında tipinin belli olması sağlandı. _controller nesnesi HomeController tipinde yaratıldı.

Bu sınıfı görelim öncelikle :

```

Odev7 > lib > home_controller.dart > ...
1  import 'package:flutter/material.dart';
2
3  class HomeController extends ChangeNotifier {
4    int selectedBottomTab = 0;
5
6    // panel değişimini yakalar
7    void onBottomNavigationTabChange(int index) { ...
11
12    // tüm kilit şeklini tutan değişkenler. başlangıçta true değer verildi
13    bool isRightDoorLock = true;
14    bool isLeftDoorLock = true;
15    bool isBonnetLock = true;
16    bool isTrunkLock = true;
17
18    // sağ kapı kilit şeklini değiştir
19    void updateRightDoorLock() { ...
23
24    // sol kapı kilit şeklini değiştir
25    void updateLeftDoorLock() { ...
29
30    // kaput kilit şeklini değiştir
31    void updateBonnetDoorLock() { ...
35
36    // bagaj kapı kilit şeklini değiştir
37    void updateTrunkDoorLock() { ...
41
42    bool isCoolSelected = true;
43
44    // cool - heat değişimini yapar
45    void updateCoolSelectedTab() { ...
49
50    bool isShowTyre = false;
51    // lastik bölümü
52    void showTyreController(int index) { ...
67
68    bool isShowTyreStatus = false;
69    // lastik durumu
70    void tyreStatusController(int index) { ...
81
82    }

```

Uygulamadaki durum değişimlerinin mantık kısmıdır. Kapı kilidlerini , sıcaklık bölümünü ve lastik bölümünü kontrol eder.

```
home_screen.dart  home_controller.dart X  tyre_psi_card.dart

Odev7 > lib > home_controller.dart > ...
1  import 'package:flutter/material.dart';
2
3  class HomeController extends ChangeNotifier {
4    int selectedBottomTab = 0;
5
6    // panel değişimini yakalar
7    void onBottomNavigationTabChange(int index) {
8      selectedBottomTab = index;
9      notifyListeners();
10   }
11
12   // tüm kilit şeklini tutan değişkenler. başlangıçta true değer verildi
13   bool isRightDoorLock = true;
14   bool isLeftDoorLock = true;
15   bool isBonnetLock = true;
16   bool isTrunkLock = true;
17
18   // sağ kapı kilit şeklini değiştir
19   void updateRightDoorLock() {
20     isRightDoorLock = !isRightDoorLock;
21     notifyListeners();
22   }
23
24   // sol kapı kilit şeklini değiştir
25   void updateLeftDoorLock() {
26     isLeftDoorLock = !isLeftDoorLock;
27     notifyListeners();
28   }
29
30   // kaput kilit şeklini değiştir
31   void updateBonnetDoorLock() {
32     isBonnetLock = !isBonnetLock;
33     notifyListeners();
34   }
35
36   // bagaj kapı kilit şeklini değiştir
37   void updateTrunkDoorLock() {
38     isTrunkLock = !isTrunkLock;
39     notifyListeners();
40   }
41 }
```

onBottomNavigationTabChange ile sekme değişimi alınır. ...DoorLock metodları ile kapıların kilit durumları (bool değişkenleri) değiştirilir.

```
home_screen.dart  home_controller.dart X  tyre_psi_card.dart  batter
Odev7 > lib > home_controller.dart > ...
40 }
41
42 bool isCoolSelected = true;
43
44 // cool - heat deęiřimini yapar
45 void updateCoolSelectedTab() {
46   isCoolSelected = !isCoolSelected;
47   notifyListeners();
48 }
49
50 bool isShowTyre = false;
51 // lastik bۆlümü
52 void showTyreController(int index) {
53   if (selectedBottomTab != 3 && index == 3) {
54     // lastikler sonra gelicek
55     Future.delayed(
56       Duration(milliseconds: 400),
57       () {
58         isShowTyre = true;
59         notifyListeners();
60       },
61     ); // Future.delayed
62   } else {
63     isShowTyre = false;
64     notifyListeners();
65   }
66 }
67
68 bool isShowTyreStatus = false;
69 // lastik durumu
70 void tyreStatusController(int index) {
71   if (selectedBottomTab != 3 && index == 3) {
72     isShowTyreStatus = true;
73     notifyListeners();
74   } else {
75     Future.delayed(Duration(milliseconds: 400), () {
76       isShowTyreStatus = false;
77       notifyListeners();
78     }); // Future.delayed
79   }
80 }
81 }
```

uptadeCoolSelectedTab metodu ile soęuk-sıcak deęiřimi seęilir. isShowTyre ve isShowTyreStatus metodları ise lastik bۆlümünün ve durumunun mantıęını ayarlar.

řimdi HomeScreen'e devam edelim


```
home_screen.dart X tmp_btn.dart home_controller.dart tyre_psi_card.dart batter

Odev7 > lib > screens > home_screen.dart > _HomeScreenState > build
42 late List<AnimationController> _tyreAnimations,
43 // batarya animasyonları
44 void setupBatteryAnimation() {
45   _batteryAnimationController = AnimationController(
46     vsync: this,
47     duration: Duration(milliseconds: 600),
48   ); // AnimationController
49
50   _animationBattery = CurvedAnimation(
51     parent: _batteryAnimationController,
52     // Here the animation end on 0.5
53     // it ends on 300 milliseconds
54     curve: Interval(0.0, 0.5),
55   ); // CurvedAnimation
56
57   _animationBatteryStatus = CurvedAnimation(
58     parent: _batteryAnimationController,
59     // After a delay we start this animation
60     // after 60 milliseconds delay it start
61     // so it start at 360 and end on 600 milliseconds
62     curve: Interval(0.6, 1),
63   ); // CurvedAnimation
64 }
65
```

setupBatteryAnimation metodu batarya bölümünün animasyonlarını oluşturur.Önceden tanımladığımız değişkenler burada tanımlandırlar. Nasıl animasyon yapacakları Interval yardımıyla belirlendi.

```
home_screen.dart X tmp_btn.dart home_controller.dart tyre_psi_card.dart battery_status.dart
Odev7 > lib > screens > home_screen.dart > _HomeScreenState > build

66 // sıcaklık animasyonları
67 void setupTempAnimation() {
68   _tempAnimationController = AnimationController(
69     vsync: this,
70     duration: Duration(milliseconds: 1500),
71   ); // AnimationController
72   _animationCarShift = CurvedAnimation(
73     parent: _tempAnimationController,
74     curve: Interval(0.2, 0.4),
75   ); // CurvedAnimation
76   _animationTempShowInfo = CurvedAnimation(
77     parent: _tempAnimationController,
78     curve: Interval(0.45, 0.65),
79   ); // CurvedAnimation
80   _animationCoolGlow = CurvedAnimation(
81     parent: _tempAnimationController,
82     curve: Interval(0.7, 1),
83   ); // CurvedAnimation
84 }
85
86 // teker animasyonları
87 void setupTyreAnimation() {
88   _tyreAnimationController = AnimationController(
89     vsync: this, duration: Duration(milliseconds: 1200)); // AnimationController
90   _animationTyre1Psi = CurvedAnimation(
91     parent: _tyreAnimationController, curve: Interval(0.34, 0.5)); // CurvedAnimation
92   _animationTyre2Psi = CurvedAnimation(
93     parent: _tyreAnimationController, curve: Interval(0.5, 0.66)); // CurvedAnimation
94   _animationTyre3Psi = CurvedAnimation(
95     parent: _tyreAnimationController, curve: Interval(0.66, 0.82)); // CurvedAnimation
96   _animationTyre4Psi = CurvedAnimation(
97     parent: _tyreAnimationController, curve: Interval(0.82, 1)); // CurvedAnimation
98   _tyreAnimations = [
99     _animationTyre1Psi,
100     _animationTyre2Psi,
101     _animationTyre3Psi,
102     _animationTyre4Psi,
103   ];
104 }
105
```

setupTempAnimation metodu sıcaklık bölümü için animasyonları tanımlar. Başlangıçta deklare ettiğimiz değişkenler kullanılır. Araç resminin kaydırılması için bir tane , panel geçişi için bir tane ve soğuk sıcak animasyonu için birer tane olmak üzere toplam 4 tane animasyon tanımlanmıştır.

setupTyreAnimation metodu tekerlerin durumunu gösteren bölüm için animasyonları oluşturur. Geçiş için bir tane ve her teker için bir tane olmak üzere 5 tane animasyon vardır.


```
105
106 // etkinleştir
107 @override
108 void initState() {
109     setupBatteryAnimation();
110     setupTempAnimation();
111     setupTyreAnimation();
112
113     super.initState();
114 }
115
116 // pasifle
117 @override
118 void dispose() {
119     _batteryAnimationController.dispose();
120     _tempAnimationController.dispose();
121     _tyreAnimationController.dispose();
122     super.dispose();
123 }
124
```

initState metodu animasyon oluşturan metodların çalışmasını sağlar. dispose ise durdurur.

```

125
126 @override
127 Widget build(BuildContext context) {
128     return AnimatedBuilder(
129         animation: Listenable.merge([
130             // kontrol edilecek , uygulanacak
131             _controller, // !
132             _batteryAnimationController,
133             _tempAnimationController,
134             _tyreAnimationController,
135         ]), // Listenable.merge
136         builder: (context, _) {
137             return Scaffold(
138                 bottomNavigationBar: TeslaBottomNavigationBar(
139                     onTap: (index) {
140                         // seçili olan yere göre animasyonlar ayarlanır.
141                         if (index == 1) // batarya durumu
142                             _batteryAnimationController.forward();
143                         else if (_controller.selectedBottomTab == 1 && index != 1)
144                             _batteryAnimationController.reverse(from: 0.7);
145
146                         if (index == 2) // sıcaklık durumu
147                             _tempAnimationController.forward();
148                         else if (_controller.selectedBottomTab == 2 && index != 2)
149                             _tempAnimationController.reverse(from: 0.4);
150
151                         if (index == 3) // teker durumu
152                             _tyreAnimationController.forward();
153                         else if (_controller.selectedBottomTab == 3 && index != 3)
154                             _tyreAnimationController.reverse();
155
156                         _controller.showTyreController(index);
157                         _controller.tyreStatusController(index);
158                         // Make sure you call it before [onBottomNavigationTabChange]
159                         _controller.onBottomNavigationTabChange(index);
160                     },
161                     selectedTab: _controller.selectedBottomTab,
162                 ), // TeslaBottomNavigationBar

```

build metodu AnimatedBuilder döndürür. Bu sınıfın animation parametresi , animasyonları kontrol eden değişkenleri alır. (_controller da dahil)

builder parametresi bir Scaffold döndürür. Scaffold'ın bottomNavigationBar parametresine kendi yazdığımız TeslaBottomNavigationBar sınıfı almıştır. Bu sınıfı görelim.

```
home_screen.dart tesla_bottom_navigationbar.dart X tmp_btn.dart home_controller.dart
Odev7 > lib > screens > components > tesla_bottom_navigationbar.dart > TeslaBottomNavigationBar
1 import 'package:flutter/material.dart';
2 import 'package:flutter_svg/flutter_svg.dart';
3
4 import '../constanins.dart';
5
6 class TeslaBottomNavigationBar extends StatelessWidget {
7   const TeslaBottomNavigationBar({
8     Key? key,
9     required this.selectedTab,
10    required this.onTap,
11  }) : super(key: key);
12
13   final int selectedTab;
14   final ValueChanged<int> onTap;
15
16   @override
17   Widget build(BuildContext context) {
18     return BottomNavigationBar(
19       onTap: onTap,
20       currentIndex: selectedTab,
21       type: BottomNavigationBarType.fixed,
22       backgroundColor: Colors.black,
23       items: List.generate(
24         // liste oluřturuldu
25         navIconSrc.length,
26         (index) => BottomNavigationBarItem(
27           icon: SvgPicture.asset(
28             navIconSrc[index], // İkonları listeden alır
29             color: index == selectedTab // seili sekmenin ...
30               ? primaryColor // rengi primarycolor
31               : Colors.white70, // diğerkleri beyaz
32           ), // SvgPicture.asset
33           label: ikonAdlari[index], // ikonların altına yazılacak yazı
34         ), // BottomNavigationBarItem
35       ), // List.generate
36     ); // BottomNavigationBar
37   }
38 }
39
40 List<String> navIconSrc = [
41   "assets/icons/Lock.svg",
42   "assets/icons/Charge.svg",
43   "assets/icons/Temp.svg",
44   "assets/icons/Tyre.svg",
45 ];
46
47 List<String> ikonAdlari = [
48   "Lock",
49   "Battery",
50   "Temperature",
51   "Wheel",
52 ];
53
```

Bu sınıf StatelessWidget'tır. selectedTab ve onTap durumunu dışarıdan almalıdır. Bu sınıf build metodunda bir widget'tan yararlanılır. Bu widget BottomNavigationBar'dır. Arka plan siyah olarak seçilmiştir. currentIndex parametresi ise dışarıdan alınan selectedTab'dır. items parametresinde List.generate ile bir liste oluşturulmuştur. Bu liste alt kısımdaki panelleri oluşturacaktır. icon bilgisini sınıf dışından (aşağıdan) navIconSrc listesinden alır ve bunları SvgPicture.asset ile yerleştirir. Rengini ise seçili olanın primaryColor diğerlerinin ise beyaz (white70) olmasını sağlar. (Fazladan label etiketine ikonAdlari listesi verilmiştir. Her bir ikonun altına ilgili stringi koyacaktır.)

Şimdi devam edelim . if else-if blokları ile batarya , sıcaklık ve teker durumunun animasyonlarını çalıştıracak ya da geri alacak. Bunu da seçili sekmeye ve indexe göre yapacak.

```

163     body: SafeArea(
164       child: LayoutBuilder(
165         builder: (context, constraints) {
166           return Stack(
167             alignment: Alignment.center,
168             children: [
169               // Let's fixed it
170 >             SizedBox( // SizedBox ...
174               // Nothing really chnage, let's fix that
175 >             Positioned( // Positioned ...
189               // Kapı durumunu deęiřtiren bölüm
190 >             AnimatedPositioned( // AnimatedPositioned ...
204 >             AnimatedPositioned( // AnimatedPositioned ...
218 >             AnimatedPositioned( // AnimatedPositioned ...
232 >             AnimatedPositioned( // AnimatedPositioned ...
246             // Battery
247 >             Opacity( // Opacity ...
254
255 >             Positioned( // Positioned ...
264             // Temp
265 >             Positioned( // Positioned ...
274             // We also need to animate the glow
275 >             Positioned( // Positioned ...
292
293             // Tyre
294             if (_controller.isShowTyre) ...tyres(constraints),
295             if (_controller.isShowTyreStatus)
296               // Let's add the animation
297
298 >             GridView.builder( // GridView.builder ...
317           ],
318         ); // Stack
319       },
320     ), // LayoutBuilder
321   ), // SafeArea
322 ); // Scaffold
323 }); // AnimatedBuilder
324 }
325 }
326

```

body bölümünde SafeArea widget'ı kullanılmış (işletim sistemi interface'lerinden kaçınmak için) . child'ında LayoutBuilder widget'ı kullanılmış. Bu widget ise Stack widget'ı döndürür. Bu Stack'in children'ına 11 adet widget verilmiştir. Tek tek bunlara bakalım.

```

169           // Let's fixed it
170           SizedBox(
171             height: constraints.maxHeight,
172             width: constraints.maxWidth,
173           ), // SizedBox

```

İlk olarak bir SizedBox. tüm ekranı kaplayan bir kutu oluşturduk. (max)

```

175           Positioned(
176             left:
177               constraints.maxWidth / 2 * _animationCarShift.value,
178             height: constraints.maxHeight,
179             width: constraints.maxWidth,
180             child: Padding(
181               padding: EdgeInsets.symmetric(
182 >                 vertical: constraints.maxHeight * 0.1), // EdgeInsets.symmetric
183             child: SvgPicture.asset(
184               "assets/icons/Car.svg",
185               width: double.infinity,
186             ), // SvgPicture.asset
187           ), // Padding
188         ), // Positioned

```

Ardından Positioned widget'ı ile bir SvgPicture.asset widget'ı gerekli yere yerleştirilmiş. Boyut ve yükseklik bilgisi ayarlanmış ve simetrik olması sağlanmıştır. animationCarShift değeri ile ise araba ikonunun kayması için gerekli işlem yapılmış.

```
190 AnimatedPositioned(  
191   duration: defaultDuration,  
192   right: _controller.selectedBottomTab == 0  
193     ? constraints.maxWidth * 0.05  
194     : constraints.maxWidth / 2,  
195   child: AnimatedOpacity(  
196     duration: defaultDuration,  
197     opacity: _controller.selectedBottomTab == 0 ? 1 : 0,  
198     child: DoorLock(  
199       isLock: _controller.isRightDoorLock,  
200       press: _controller.updateRightDoorLock,  
201     ), // DoorLock  
202   ), // AnimatedOpacity  
203 ), // AnimatedPositioned
```

```
204 AnimatedPositioned(  
205   duration: defaultDuration,  
206   left: _controller.selectedBottomTab == 0  
207     ? constraints.maxWidth * 0.05  
208     : constraints.maxWidth / 2,  
209   child: AnimatedOpacity(  
210     duration: defaultDuration,  
211     opacity: _controller.selectedBottomTab == 0 ? 1 : 0,  
212     child: DoorLock(  
213       isLock: _controller.isLeftDoorLock,  
214       press: _controller.updateLeftDoorLock,  
215     ), // DoorLock  
216   ), // AnimatedOpacity  
217 ), // AnimatedPositioned
```

```
218 AnimatedPositioned(  
219   duration: defaultDuration,  
220   top: _controller.selectedBottomTab == 0  
221     ? constraints.maxHeight * 0.13  
222     : constraints.maxHeight / 2,  
223   child: AnimatedOpacity(  
224     duration: defaultDuration,  
225     opacity: _controller.selectedBottomTab == 0 ? 1 : 0,  
226     child: DoorLock(  
227       isLock: _controller.isBonnetLock,  
228       press: _controller.updateBonnetDoorLock,  
229     ), // DoorLock  
230   ), // AnimatedOpacity  
231 ), // AnimatedPositioned
```

```

232  AnimatedPositioned(
233    duration: defaultDuration,
234    bottom: _controller.selectedBottomTab == 0
235      ? constraints.maxHeight * 0.17
236      : constraints.maxHeight / 2,
237    child: AnimatedOpacity(
238      duration: defaultDuration,
239      opacity: _controller.selectedBottomTab == 0 ? 1 : 0,
240      child: DoorLock(
241        isLock: _controller.isTrunkLock,
242        press: _controller.updateTrunkDoorLock,
243      ), // DoorLock
244    ), // AnimatedOpacity
245  ), // AnimatedPositioned

```

Ardından AnimatedPositioned widget'ları yardımıyla kapı kilit bölümünün düzenlemesi yapılmış. Sırasıyla sağ ve sol kapılar ile kaput ve bagaj kısımlarının kilit animasyonları oluşturulmuş. Herbirinde ekranın neresinde duracakları ve animasyon değişiminde nereye gidecekleri ayarlanmış. En içte ise DoorLock sınıfı kullanılmış. Bu sınıf yine kapı kilitleri için kendi yazdığımız sınıftır.

door_lock.dart

```

1  import 'package:flutter/material.dart';
2  import 'package:flutter_svg/flutter_svg.dart';
3
4  import '../constanins.dart';
5
6  // kapı ekranında kilit animasyonunu yapan sınıf
7  class DoorLock extends StatelessWidget {
8    final VoidCallback press;
9    final bool isLock;
10
11    const DoorLock({Key? key, required this.press, required this.isLock,}) : super(key: key);
12
13    @override
14    Widget build(BuildContext context) {
15      return GestureDetector(
16        onTap: press,
17        child: AnimatedSwitcher(
18          duration: defaultDuration,
19          switchInCurve: Curves.easeInOutBack,
20          transitionBuilder: (child, animation) => ScaleTransition(
21            scale: animation,
22            child: child,
23          ), // ScaleTransition
24          child: isLock ? SvgPicture.asset(
25            "assets/icons/door_lock.svg",
26            key: ValueKey("lock"), // farklı oldukları anlaşılması için key verildi
27          ) // SvgPicture.asset
28            : SvgPicture.asset(
29              "assets/icons/door_unlock.svg",
30              key: ValueKey("unlock"), // farklı oldukları anlaşılması için key verildi
31            ), // SvgPicture.asset
32        ), // AnimatedSwitcher
33      ); // GestureDetector
34    }
35  }

```

StatelessWidget'tır. GestureDetector döndürür. Dışarıdan aldığı isLock bool değişkenine göre lock veya unlock animasyonunu yerine getirir.


```

247     Opacity(
248       opacity: _animationBattery.value,
249       child: SvgPicture.asset(
250         "assets/icons/Battery.svg",
251         width: constraints.maxWidth * 0.45,
252       ), // SvgPicture.asset
253     ), // Opacity
254

```

Opacity widget'ı ile batarya bölümündeki animasyon düzenlenir.

```

255     Positioned(
256       top: 50 * (1 - _animationBatteryStatus.value),
257       height: constraints.maxHeight,
258       width: constraints.maxWidth,
259       child: Opacity(
260         opacity: _animationBatteryStatus.value,
261         child: BatteryStatus(constraints: constraints),
262       ), // Opacity
263     ), // Positioned
264     // Temp
265     Positioned(
266       height: constraints.maxHeight,
267       width: constraints.maxWidth,
268       top: 60 * (1 - _animationTempShowInfo.value),
269       child: Opacity(
270         opacity: _animationTempShowInfo.value,
271         child: TempDetails(controller: _controller),
272       ), // Opacity
273     ), // Positioned

```

Üstteki Positioned widget'ı ile batarya panelinin ekranda yeri , alttaki Positioned widget'ı ile sıcaklık panelinin ekrandaki yeri ayarlanır. Buralarda kullanılan sınıfları görelim :

battery_status.dart

```
home_screen.dart  main.dart  tmp_btn.dart  home_controller.dart  tyre_psi_card.dart  battery_status.dart  temp_details.dart
Odev7 > lib > screens > components > battery_status.dart > BatteryStatus
1 import 'package:flutter/material.dart';
2
3 import '../constanins.dart';
4
5 class BatteryStatus extends StatelessWidget {
6   const BatteryStatus({
7     Key? key,
8     required this.constraints,
9   }) : super(key: key);
10
11   final BoxConstraints constraints;
12
13   @override
14   Widget build(BuildContext context) {
15     return Column(
16       children: [
17         Text(
18           "220 mi",
19           style: Theme.of(context)
20             .textTheme
21             .headline3!
22             .copyWith(color: Colors.white),
23         ), // Text
24         Text(
25           "80%",
26           style: TextStyle(fontSize: 24),
27         ), // Text
28         Spacer(),
29         Text(
30           "Charging".toUpperCase(),
31           style: TextStyle(fontSize: 20),
32         ), // Text
33         Text(
34           "18 min. remaining",
35           style: TextStyle(fontSize: 20),
36         ), // Text
37         SizedBox(height: constraints.maxHeight * 0.14),
38         DefaultTextStyle(
39           style: TextStyle(
40             fontSize: 18,
41             fontWeight: FontWeight.bold,
42           ), // TextStyle
43           child: Row(
44             mainAxisAlignment: MainAxisAlignment.spaceBetween,
45             children: [
46               Text("22 mi/hr"),
47               Text("232 v"),
48             ],
49           ), // Row
50         ), // DefaultTextStyle
51         const SizedBox(height: defaultPadding),
52       ],
53     ); // Column
54   }
55 }
56
```

batter_status dosyası BatteryStatus sınıfına sahiptir. Batarya bölümünün düzenini yapar. Build metodu içerisinde Column widgeti kullanıldı. Widget'ın 8 adet child'ı vardır. İki child'ı üstte Text ve ardından boşluk bırakmak için Spacer . Altında yine iki adet Text widget. Altına bir SizedBox ile boşluk. Ve son olarak

DefaultTextStyle widget'ı ise ekranda alt kısımda bulunan yeri oluşturur. Row widget'ı yardımıyla iki adet Text basar ve en alta ise bir boşluk oluşturur.

temp_details.dart

```
home_screen.dart  main.dart  tmp_btn.dart  home_controller.dart  tyre_psi_card.dart  battery_status.dart  temp_details.dart X
Odev7 > lib > screens > components > temp_details.dart > ...
1  import 'package:flutter/material.dart';
2
3  import '../constanins.dart';
4  import '../home_controller.dart';
5  import 'tmp_btn.dart';
6
7  // This is what we want
8  class TempDetails extends StatelessWidget {
9    const TempDetails({Key? key, required HomeController controller,}) : _controller = controller, super(key: key);
10
11    final HomeController _controller;
12
13    @override
14    Widget build(BuildContext context) {
15      return Padding(
16        padding: const EdgeInsets.all(defaultPadding),
17        child: Column(
18          crossAxisAlignment: CrossAxisAlignment.start,
19          children: [
20 >      SizedBox( // SizedBox ...
41      Spacer(),
42 >      Column( // Column ...
60      Spacer(),
61      Text("CURRENT TEMPERATURE"),
62 >      const SizedBox(height: defaultPadding),
63 >      Row( // Row ...
96      ],
97      ), // Column
98    ); // Padding
99  }
100 }
101
```

TempDetails sınıfı sıcaklık bölümü için ekranın düzenini oluşturur. build metodu Padding döndürür. child widget'a Column widget verilmiştir. Column'un child'larına bakalım

```

19  children: [
20    SizedBox(
21      height: 120,
22      child: Row(
23        children: [
24          TempBtn(
25            isActive: _controller.isCoolSelected,
26            svgSrc: "assets/icons/coolShape.svg",
27            title: "Cool",
28            press: _controller.updateCoolSelectedTab,
29          ), // TempBtn
30          const SizedBox(width: defaultPadding),
31          TempBtn(
32            isActive: !_controller.isCoolSelected,
33            svgSrc: "assets/icons/heatShape.svg",
34            title: "Heat",
35            activeColor: redColor,
36            press: _controller.updateCoolSelectedTab,
37          ), // TempBtn
38        ],
39      ), // Row
40    ), // SizedBox
41    Spacer(),
42    Column(
43      children: [
44        IconButton(
45          padding: EdgeInsets.zero,
46          onPressed: () {},
47          icon: Icon(Icons.arrow_drop_up, size: 48),
48        ), // IconButton
49        Text(
50          "29" + "\u2103",
51          style: TextStyle(fontSize: 86),
52        ), // Text
53        IconButton(
54          padding: EdgeInsets.zero,
55          onPressed: () {},
56          icon: Icon(Icons.arrow_drop_down, size: 48),
57        ), // IconButton
58      ],
59    ), // Column

```

Üstte bir SizedBox içinde Row widget kullanıldı. Row widget'ın 2 adet child'ı var. İkisi de TempBtn sınıfı (kendi yazdığımız). Bu sınıfı görelim.

tmp_btn.dart

```
home_screen.dart  main.dart  tmp_btn.dart  home_controller.dart  tyre_psi_card.dart  battery_status.dart  temp_details.dart
Odev7 > lib > screens > components > tmp_btn.dart > ...
1  import 'package:flutter/material.dart';
2  import 'package:flutter_svg/flutter_svg.dart';
3  import '../constanins.dart';
4
5  class TempBtn extends StatelessWidget {
6    const TempBtn({Key? key,required this.svgSrc,required this.title,this.isActive = false, required this.press,this.activeColor = primaryColor,}) : super(key: key);
7
8    final String svgSrc, title;
9    final bool isActive;
10   final VoidCallback press;
11   final Color activeColor;
12
13   @override
14   Widget build(BuildContext context) {
15     return GestureDetector(
16       onTap: press,
17       child: Column(
18         children: [
19           AnimatedContainer(
20             duration: Duration(milliseconds: 200),
21             curve: Curves.easeInOutBack,
22             height: isActive ? 76 : 50,
23             width: isActive ? 76 : 50,
24             child: SvgPicture.asset(
25               svgSrc,
26               color: isActive ? activeColor : Colors.white38,
27             ), // SvgPicture.asset
28           ), // AnimatedContainer
29           const SizedBox(height: defaultPadding / 2),
30           AnimatedDefaultTextStyle(
31             duration: Duration(milliseconds: 200),
32             style: TextStyle(
33               fontSize: 16,
34               color: isActive ? activeColor : Colors.white38,
35               fontWeight: isActive ? FontWeight.bold : FontWeight.normal,
36             ), // TextStyle
37             child: Text(
38               title.toUpperCase(),
39             ), // Text
40           ), // AnimatedDefaultTextStyle
41         ],
42       ), // Column
43     ); // GestureDetector
44   }
45
46
```

TempBtn sınıfı dışarıdan aldığı parametrelerle Cool ve Heat butonlarını oluşturan sınıftır. AnimatedContainer sınıfı kullanılır. Animasyon özelliği verilir. Aktif veya pasif olma durumuna göre renk , boyut gibi özellikleri belirlenir. Tıklanabilir olması için GestureDetector widget'ı kullanılmıştır. Text düzenlemesi aktif olma durumuna (üstüne basma) göre değiştirilmiştir (kalın , büyük harfler , renk).

Şimdi temp_details 'e devam edebiliriz. Butonlar yerleştirildikten sonra Spacer ile boşluk bırakılmış. Ardından bir Column widget' içine IconButton , Text ve IconButton widget'ları eklenmiş (42 - 59) . Text ekranda gözüken santigrat derece ifadesini yazdırırken butonlar artırma ve eksiltme işaretlerini koyarlar. Fakat onpressed parametresine birşey verilmediği için sayıyı değiştirmezler.

```

59     ), // Column
60     Spacer(),
61     Text("CURRENT TEMPERATURE"),
62     const SizedBox(height: defaultPadding),
63     Row(
64       children: [
65         Column(
66           crossAxisAlignment: CrossAxisAlignment.start,
67           children: [
68             Text(
69               "Inside".toUpperCase(),
70             ), // Text
71             Text(
72               "20" + "\u2103",
73               style: Theme.of(context).textTheme.headline5,
74             ) // Text
75           ],
76         ), // Column
77         const SizedBox(width: defaultPadding),
78         Column(
79           crossAxisAlignment: CrossAxisAlignment.start,
80           children: [
81             Text(
82               "Inside".toUpperCase(),
83               style: TextStyle(color: Colors.white54),
84             ), // Text
85             Text(
86               "35" + "\u2103",
87               style: Theme.of(context)
88                 .textTheme
89                 .headline5!
90                 .copyWith(color: Colors.white54),
91             ) // Text
92           ],
93         ), // Column
94       ],
95     ) // Row
96   ],
97 ), // Column
98 ); // Padding
99 }

```

Spacer ile boşluk bırakıldı. Text widget'ı ile yazı yazıldı ve boşluk koyuldu. Ardından Row widget'ı içerisine sırasıyla bir Column var. İçerisinde iki adet Text widget var. Ardından bir SizedBox widget'ı. Altta ise Column içinde yine iki adet Text widget var.


```

275     Positioned(
276       right: -180 * (1 - _animationCoolGlow.value),
277       child: AnimatedSwitcher(
278         duration: defaultDuration,
279         child: _controller.isCoolSelected
280           ? Image.asset(
281             "assets/images/Cool_glow_2.png",
282             key: UniqueKey(),
283             width: 200,
284           ) // Image.asset
285           : Image.asset(
286             "assets/images/Hot_glow_4.png",
287             key: UniqueKey(),
288             width: 200,
289           ), // Image.asset
290       ), // AnimatedSwitcher
291     ), // Positioned
292

```

Sonraki Positioned widget'ı ile sıcaklık panelindeki hot - cool animasyon değişimi ayarı yapılır.

```

293     // Tyre
294     if (_controller.isShowTyre) ...tyres(constraints),
295     if (_controller.isShowTyreStatus)
296       // Let's add the animation
297
298     GridView.builder(
299       itemCount: 4,
300       physics: NeverScrollableScrollPhysics(),
301       gridDelegate:
302         SliverGridDelegateWithFixedCrossAxisCount(
303           crossAxisCount: 2,
304           mainAxisSpacing: defaultPadding,
305           crossAxisSpacing: defaultPadding,
306           childAspectRatio:
307             constraints.maxWidth / constraints.maxHeight,
308         ), // SliverGridDelegateWithFixedCrossAxisCount
309       itemBuilder: (context, index) => ScaleTransition(
310         scale: _tyreAnimations[index],
311         child: TyrePsiCard(
312           isBottomTwoTyre: index > 1,
313           tyrePsi: demoPsiList[index],
314         ), // TyrePsiCard
315       ), // ScaleTransition
316     ) // GridView.builder
317   ],
318 ); // Stack
319 },
320 ), // LayoutBuilder
321 ), // SafeArea
322 ); // Scaffold
323 }); // AnimatedBuilder
324 }
325 }

```

Üstteki if blokları lastiklerin olduğu bölüm. tyres listesini kullanır. Bu liste ekranda (arka planda) 4 adet lastik resmi yerleştirir. Altta GridView.builder widget'ı ile lastik bölümü bilgileri demoPsiList listesinden alındı. Bu liste TyrePsi dosyası içerisindedir. TyrePsiCard ise ekrana kartları oluşturacak olan sınıftır. itemBuilder ile her nesne ekrana yerleştirilir.

child içindeki TyrePsiCard ise kendi yazdığımız bir sınıftır.

tyres.dart

```
Odev7 > lib > screens > components > tyres.dart > tyres
1 import 'package:flutter/material.dart';
2 import 'package:flutter_svg/flutter_svg.dart';
3
4 List<Widget> tyres(BoxConstraints constraints) {
5   return [
6     Positioned(
7       left: constraints.maxWidth * 0.2,
8       top: constraints.maxHeight * 0.22,
9       child: SvgPicture.asset("assets/icons/FL_Tyre.svg"),
10    ), // Positioned
11    Positioned(
12      right: constraints.maxWidth * 0.2,
13      top: constraints.maxHeight * 0.22,
14      child: SvgPicture.asset("assets/icons/FL_Tyre.svg"),
15    ), // Positioned
16    Positioned(
17      right: constraints.maxWidth * 0.2,
18      top: constraints.maxHeight * 0.63,
19      child: SvgPicture.asset("assets/icons/FL_Tyre.svg"),
20    ), // Positioned
21    Positioned(
22      left: constraints.maxWidth * 0.2,
23      top: constraints.maxHeight * 0.63,
24      child: SvgPicture.asset("assets/icons/FL_Tyre.svg"),
25    ), // Positioned
26  ];
27 }
28
```

TyrePsi.dart

```
home_screen.dart • TyrePsi.dart X tmp_btn.dart home_controller.dart tyre_psi_card.dart
Odev7 > lib > models > TyrePsi.dart > demoPsiList
1 class TyrePsi {
2   final double psi;
3   final int temp;
4   final bool isLowPressure;
5
6   TyrePsi({required this.psi, required this.temp, required this.isLowPressure});
7 }
8
9 final List<TyrePsi> demoPsiList = [
10   TyrePsi(psi: 23.6, temp: 56, isLowPressure: true),
11   TyrePsi(psi: 35.0, temp: 41, isLowPressure: false),
12   TyrePsi(psi: 34.6, temp: 41, {required bool isLowPressure}),
13   TyrePsi(psi: 34.8, temp: 42, isLowPressure: false),
14 ];
15
```

TyrePsi sınıfı TyrePsi.dart dosyası içerisinde. Bu dosyada demoPsiList listesi vardır ve buradaki sınıftan türetilen nesneleri tutar.

tyre_psi_card.dart

```
home_screen.dart  tyre_psi_card.dart  •
Odev7 > lib > screens > components > tyre_psi_card.dart > TyrePsiCard > build
1  import 'package:flutter/material.dart';
2  import 'package:Odev7/models/TyrePsi.dart';
3  import '../constanins.dart';
4
5  class TyrePsiCard extends StatelessWidget {
6    const TyrePsiCard({Key? key,required this.isBottomTwoTyre,required this.tyrePsi,}) : super(key: key);
7
8    final bool isBottomTwoTyre;
9    final TyrePsi tyrePsi;
10
11    @override
12    Widget build(BuildContext context) {
13      return Container(
14        padding: EdgeInsets.all(defaultPadding),
15        decoration: BoxDecoration(
16          color:
17            tyrePsi.isLowPressure ? redColor.withOpacity(0.1) : Colors.white10,
18          border: Border.all(
19            color: tyrePsi.isLowPressure ? redColor : primaryColor, width: 2), // Border.all
20          borderRadius: BorderRadius.all(Radius.circular(6)),
21        ), // BoxDecoration
22        child: isBottomTwoTyre ? Column(
23          crossAxisAlignment: CrossAxisAlignment.start,
24          children: [
25            lowPressureText(context),
26            Spacer(),
27            psiText(context, psi: tyrePsi.psi.toString()),
28            const SizedBox(height: defaultPadding),
29            Text("${tyrePsi.temp}\u2103",style: TextStyle(fontSize: 16)),
30          ],
31        ) // Column
32        : Column(
33          crossAxisAlignment: CrossAxisAlignment.start,
34          children: [
35            psiText(context, psi: tyrePsi.psi.toString()),
36            const SizedBox(height: defaultPadding),
37            Text("${tyrePsi.temp}\u2103",style: TextStyle(fontSize: 16)),
38            Spacer(),
39            lowPressureText(context),
40          ],
41        ), // Column
42      ); // Container
43    }
44
45 > Column lowPressureText(BuildContext context) { ...
62
63 > Text psiText(BuildContext context, {required String psi}) { ...
80
81 }
```

TyrePsiCard sınıfını içerir. TyrePsi sınıfından yararlanarak ekrana lastik panelini oluşturur. build metodunda Container döndürür. padding değeri constanins.dart dosyasından alınmmıştır. BoxDecoration widget'ı ile dekorasyon bilgileri verilmiştir. tyrePsi nesnesinin isLowPressure bilgisine göre renk ayarlanır. HomeScreen'den verilen isBottomTwoTyre'a göre farklı yapılandırma yapar.

Eğer nesneler alttaysa (isBottomTwoTyre = true) üstteki Column çalışır. lowPressureText , spacer , psiText , sizedBox ve son olarak Text yazdırır.

Eğer nesneler alltaysa alttaki Column çalışır. psiText , SizedBox , Text , Spacer , lowPressureText yazdırır.

```

58 Column lowPressureText(BuildContext context) {
59   return Column(
60     children: [
61       Text(
62         "Low".toUpperCase, BuildContext context
63         style: Theme.of(context)
64         .textTheme
65         .headline3!
66         .copyWith(color: Colors.white, fontWeight: FontWeight.w600),
67       ), // Text
68       Text(
69         "Pressure".toUpperCase(),
70         style: TextStyle(fontSize: 20),
71       ), // Text
72     ],
73   ); // Column
74 }
75
76 Text psiText(BuildContext context, {required String psi}) {
77   return Text.rich(
78     TextSpan(
79       text: psi,
80       style: Theme.of(context).textTheme.headline4!.copyWith(
81         fontWeight: FontWeight.w600,
82         color: Colors.white,
83       ),
84     children: [
85       TextSpan(
86         text: "psi",
87         style: TextStyle(fontSize: 24),
88       ) // TextSpan
89     ],
90   ), // TextSpan
91   ); // Text.rich
92 }
93 }
94

```

lowPressureText Column döndürür ve ilk child'ı olarak büyük harfle Low yazdırır. İkinci child ise Pressure yazdırır.

psiText ise önce psi değerini alır ve yazdırır. Ardından psi text'i yazar.

Böylelikle ekranda simetri sağlanmış olur.