

Nesne Tabanlı programlama

Proje Adı: Havayolu Bilgi Sistemi Gerçekleştirimi

Dersin Hocası: SINEM AKYOL

Öğrenci Bilgilerin:

- **ADI:** Ahmed Salih
- **No :** 200290604
- **3.Sınıf**

Proje Tarihi bilgileri:

1. Sınıflar: 10/12 ile 13/12

2. Arayüzler: 13/12 ile 20/12

3. Sınıflar Arayüzlerle Bağlanmak : 20/12 ile 23/12

İÇİNDEKİLER

BÖLÜM 1. GİRİŞ.....	3
1.1. Giriş.....	3
1.2. Sistemin Özellikleri.....	3
1.3. Kullanıcı Roller ve İhtiyaçları.....	3
BÖLÜM 2. Proje Amacı	4
2.1. Amacı	4
2.2. Yönetici	4
2.3. Personel	4
2.4. Yolcu	4
BÖLÜM 3. Kullanılan Teknolojiler.....	6
2.1. Java	6
2.2. Java Frame	6
2.3. Figma (Design)	6
BÖLÜM 4. Proje Geliştirme Süreci.....	6
. I.Sınıflar.....	6
. II. Arayüzler	6
. III.Sınıfların Arayüzlerle Bağlanması.....	6
BÖLÜM 5. Kaynak Kodlar.....	7
5.1. Koltuk Sınıfı	7
5.2. Uçak Sınıfı	8
5.3. Sefer Sınıfı	9
4.3. Uçuş Sınıfı	10
5.3 Şirket Sınıfı	11
5.3 User(Kullanıcı) Sınıfı	12
5.3 Personel Sınıfı	13
5.3 Pilot Sınıfı	14
5.3 Hostese Sınıfı	14
5.3 Yönetici Sınıfı	15
5.3 Personel Sınıfı	15
5.3 HavaYolu Sınıfı	16
5.3 HavaYoluBilgiSistemi Sınıfı (main)	17
5.3create() Metodu	18
BÖLÜM 6. Çıktılar-Sonuç ve Değerlendirmeleri.....	18
5.1. AnaSayfa.....	19
5.2. Yolcu Sayfası	19
5.3. Uçuş Seçme Sayfası	20
5.2. Koltuk Seçme Sayfası	20
5.2. Koltuk Seçme Sayfası	21
5.2. Bilet Sayfası	22
5.2. Personel Sayfası.....	23
5.2. Yönetici Sayfası	24

Giriş

Havacılık sektörü günümüzde giderek karmaşıklılaşan bir yapıya sahipken, bu sektörde faaliyet gösteren havayolu şirketlerinin operasyonlarını düzenlemek ve yönetmek önem arz etmektedir. Havayolu Yönetim Sistemi, bu ihtiyaçları karşılamak ve sektördeki aktörleri daha etkili bir şekilde desteklemek üzere tasarlanmış bir yazılım çözümüdür.

Sistem Özellikleri :

Bu sistem, birden fazla havayolu şirketi için uçuş, personel ve yolcu bilgilerini merkezi bir noktada toplamayı amaçlamaktadır. Her havayolu şirketinin farklı uçak tiplerine, seferlere ve personel ihtiyaçlarına sahip olduğu bir ortamda, sistem bu bilgileri etkili bir şekilde yöneterek operasyonları optimize etmeyi hedefler.

Uçuş bilgileri, rota, uçuş süresi ve sefer numarası gibi sabit verilerle birlikte, dinamik veriler olan personel ve yolcu listelerini içerir. Bilet fiyatlandırma, kullanıcı tipi ve seçilen koltuk tipi gibi faktörlere göre hesaplanır.

Kullanıcı Roller ve İhtiyaçları:

Sistem, yönetici, personel ve yolcu olmak üzere üç farklı kullanıcı rolü ile tasarlanmıştır. Yönetici, sisteme veri ekler, değiştirir ve görüntülerken, personel atamaları yapabilir. Personel, kendisine atanan uçuşları görüntüler ve yolcu bilgilerini takip ederken, yolcu ise bilet alır ve uçuşları sorgular.

Proje Amacı

Bu sistem, havayolu şirketlerinin uçuş, personel ve yolcu bilgilerini etkili bir şekilde takip etmeyi ve yönetmeyi amaçlar.

1. **Yönetici:** Sefer, uçuş, kişisel, çeşitleri gibi bilgilerin sistemin kaydedilmesi, değiştirilmesi, görüntülenmesi ve uçuşlara personelin atanması halinde çalışır.

- değişken bir uçuş uçuşlarının listesi
- Doğum aralığı belirli bir yolcunun uçuşlarının listesi

2. **Personel:** görev aldığı uçuş bilgilerini görüntüler.

- Kendisine atanmış uçuşların listesi

3. **Yolcu:**

Yolcu rolü, belirli kriterlere uygun uçuşları gözden geçirip, istenilen uçuş için bilet alma işlemlerini gerçekleştirir. Bilet alım süreci, yolcuya bilet ücreti gösterildikten sonra basitçe bir düğmeye basılarak uçuşa olan katılımın onaylanması ve koltuk kapasitesinin azaltılması adımlarını içerir.

- Belirli Tarih Aralığında ve Yönde Uçuşlar:
- Belirli bir tarih aralığında ve belirli bir yönde planlanan uçuşların listesini görüntüler.
- Seçilen Koltuk Tipi İçin Belirli Bir Tarih Aralığında ve Yönde Yer Bulunan Uçuşlar:

- Seçilen koltuk tipi için belirli bir tarih aralığında ve belirli bir yönde mevcut uçuşlardan yer bulunanları listeler.
- Bir Hava Yolu Şirketinin Belirli Bir Şehirdeki Havaalanlarına Uçuşlarının Listesi:
- Belirli bir hava yolu şirketinin belirli bir şehirdeki havaalanlarına düzenlediği uçuşların listesini sunar.

Kullanılan Teknolojiler

- Java
- Java Frame
- Figma (Design)

Proje Geliştirme Süreci

I. Sınıflar (10/12 - 13/12):

- İlk olarak, her sınıf için yeni bir dosya ekledik ve ardından her sınıfın değişkenlerini, metotlarını ve oluşturucularını tanımladık. Bu adım, temel sınıfların tasarımını başlattığımız kısım oldu.

II. Arayüzler (13/12 - 20/12):

- Ana sayfa, yolcu, personel ve yönetici sayfalarını oluşturduk. Yolcu arayüzünde sefer arama, bilgi alma , belirli bir tarih aralığında uçuş bulma, uçak koltuk seçme ve bilet alma gibi önemli sayfaları tasarladık. Bu aşamada, kullanıcıların etkileşimde bulunacakları temel arayüzleri hazırladık.

III. Sınıfların Arayüzlerle Bağlanması (20/12 - 23/12):

- Sınıfları oluşturduktan sonra, bu sınıfları arayüzlerle uyumlu hale getirmek için bağlantılar oluşturduk. Yani, sınıfların içindeki metodları ve değişkenleri arayüzlerde nasıl kullanılacağını belirledik. Bu adım, projenin içsel bütünlüğünü sağlamak ve sınıfların işlevselliğini arayüzlerle entegre etmek amacıyla yapılmıştır.

Kaynak Kodlar

```
1 package havayolubilgisistemi;
2
3 public class Koltuk {
4
5     public String No;
6     public double ucret;
7     public boolean durum = false;
8     public Yolcu yolcu;
9     public String koltuktipi = tipi[1];
10    public static String[] tipi = { "Business", "Standart", "Ekonomik" };
11
12    // uçak tipi göre sıralama
13    public void KoltukUcreti(int sıra,int toplamsıra , double ucret){
14        if(sıra<=(int)toplamsıra*1/3) {
15            koltuktipi=tipi[1];
16            this.ucret=ucret;
17        }
18        else if(sıra<=(int)toplamsıra*2/3 && sıra> (int)toplamsıra*1/3){
19            koltuktipi=tipi[2];
20            this.ucret=ucret-ucret*1/5;
21        }
22        else{
23            koltuktipi=tipi[0];
24            this.ucret=ucret+ucret*1/5;
25        }
26    }
27    // Bilgiler Al ve Koltuk tip Belirle
28    public Koltuk(int sıra,char harf ,int toplamsıra,double ucret) {
29        KoltukUcreti(sıra, toplamsıra, ucret);
30        No=Integer.toString(sıra) + String.valueOf(harf);
31    }
32
33 }
```

• Koltuk Sınıfı:

Her Koltuk için bir Numra , ucret , koltuk durumu dolumu değilmi , koltuğun yolcusu , koltuğun tipi ve her koltuk yerine göre tipi belirlenir .



```
1 package havayolubilgisistemi;
2
3 import java.util.ArrayList;
4
5 public class Ucak {
6
7     private static int ID = 1;
8     private static boolean is = false;
9     public int id;
10    public String name;
11    public Koltuk koltukList[][];
12    public ArrayList<Hostese> hosteseList = new ArrayList<>();
13    public Pilot pilot;
14    public String sirketadi;
15    public String tip;
16    public int harfSayisi;
17    public int sıraSayisi;
18    public double koltukUcreti;
19    public int koltukSayisi;
20
21    // Koltuk sayisi Uçak tip göre sırala
22    public void sayisi(String tip) {
23        this.tip = tip;
24        if (tip == HavaYolu.UcakTipi[0]) {
25            sıraSayisi = 18;
26            harfSayisi = 4;
27        } else if (tip == HavaYolu.UcakTipi[1]) {
28            sıraSayisi = 18;
29            harfSayisi = 6;
30        } else if (tip == HavaYolu.UcakTipi[2]) {
31            sıraSayisi = 30;
32            harfSayisi = 6;
33        }
34    }
35 }
```

• Uçak Sınıfı:

her Uçak için ID ,adı , koltuk listesi , hostesler , pilotu , şirketin adı , uçak tipi (Küçük,orta , büyük) , sıra , harf (uçak tipine göre belirlenir) , koltuk Ücreti ve dolu koltuk sayısı


```

1 // bilgiler al
2 public Ucak(String tip, String name, Pilot pilot, Hostese hostes, String sirketadi) {
3     sayisi(tip);
4     id = ID; ID++;
5     this.name = name;
6     kolutkList = new Koltuk[siraSayisi][harfSayisi];
7     this.pilot = pilot;
8     this.hosteseList.add(hostes);
9     this.sirketadi = sirketadi;
10    for (int i = 1; i < siraSayisi; i++) {
11        for (int j = 0; j < harfSayisi; j++) {
12            kolutkList[i][j] = new Koltuk(i, (char) (j + 65), siraSayisi, koltukUcreti);
13        }
14    }
15    for (Sirket sirket : HavaYolu.Sirketilist) {
16        if (sirket.name.equals(sirketadi)) {
17            sirket.ucakList.add(this);
18            break;
19        }
20    }
21    new Sirket(sirketadi, this);
22    HavaYolu.UcakList.add(this);
23 }
24 // koltuk seç
25 public Boolean Koltuksec(int sıra, int koltuk, Yolcu yolcu) {
26
27     if (kolutkList[sıra][koltuk].durum == false) {
28         koltukSayisi++;
29         yolcu.sıra = sıra;
30         yolcu.harf = koltuk;
31         kolutkList[sıra][koltuk].durum = true;
32         kolutkList[sıra][koltuk].yolcu = yolcu;
33         return true;
34     } else {
35         System.out.println("Koltuk Dolu !!!");
36         return false;
37     }
38 }
39 public void inis() {
40     for (int i = 1; i < kolutkList.length; i++) {
41         for (int j = 0; j < kolutkList[0].length; j++) {
42             kolutkList[i][j].durum = false;
43             kolutkList[i][j].yolcu = null;
44             kolutkList[i][j].ucret = 0;
45         }
46     }
47     koltukSayisi = 0;
48 }
49 }

```

her Uçak için bir iniş ve
koltuk seçme
Methodları sahiptir

- Uçuş Sınıfı:

```
1 public class Ucus extends Sefer {
2
3     private static int ID = 1;
4     public int i;
5     public double ucret;
6     public Ucak ucak;
7     public LocalTime saat;
8
9     // Oluşturucu
10    public Ucus(LocalTime saat, Sefer sefer, Ucak ucak, double ucret) {
11        super(sefer.KM, sefer.nerden, sefer.nereye, sefer.sure, sefer.date);
12        this.ucak = ucak;
13        this.saat = saat;
14        this.ucret = ucret;
15        ucak.koltukUcreti = ucret;
16        i = ID;
17        ID++;
18        for(int i = 1; i < ucak.siraSayisi; i++){ // koltuk Ucret belirle
19            for(int j=0;j<ucak.harfSayisi;j++){
20                ucak.kolutkList[i][j].KoltukUcreti(i, ucak.siraSayisi, ucret);
21            }
22        }
23        ucak.pilot.Ucuslar.add(this);
24        for (Hostese host : ucak.hosteselist) {
25            host.Ucuslar.add(this);
26        }
27        HavaYolu.UcusList.add(this);
28    }
29    // uçuş bilgileri
30    public String UcusBilgileri() {
31        return "Ucus Sirket Adi: " + ucak.sirketadi +
32            "Ucak Adi: " + ucak.name +
33            " Ucus tarihi ve Saati: " + formattedDate +
34            " Ucus Ucreti: " + ucak.koltukUcreti + " TL" +
35            " Ucus Mesafesi: " + KM + " KM" +
36            " Ucus " + nerden + " > " + nereye +
37            " Ucus Suresi:" + sure +
38            " Ucus Koltuk sayisi: " + ucak.kolutkList.length * ucak.kolutkList[0].length +
39            " Ucus bos Koltuk sayisi: " + ((ucak.kolutkList.length * ucak.kolutkList[0].length) - ucak.koltukSayisi);
40    }
41 }
```

her Uçuş için bir Sefer bilgileri , saati , uçağı , ucreti (standart) ve her koltuk için ucrete göre (business , ekonomik) koltukları belirlenir

- **Sefer Sınıfı:**

```
1 import java.time.LocalDateTime;
2 import java.time.LocalTime;
3 import java.time.format.DateTimeFormatter;
4 import java.time.format.DateTimeParseException;
5
6 public class Sefer {
7
8     private static int No = 100;
9     public int no;
10    public float KM;
11    public String nerden;
12    public String nereye;
13    public LocalTime sure;
14    public LocalDateTime date;
15    public String formattedDate;
16    // 1 Oluşturucu
17    public Sefer(float KM, String nerden, String nereye, LocalTime sure, LocalDateTime date) {
18        this.no = No; No++;
19        this.KM = KM;
20        this.nerden = nerden;
21        this.nereye = nereye;
22        this.sure = sure;
23        this.date = date;
24        DateTimeFormatter myFormatObj = DateTimeFormatter.ofPattern("yyyy-MM-dd");
25        formattedDate = date.format(myFormatObj);
26        if (this.getClass() == Sefer.class) {
27            HavaYolu.SeferList.add(this);
28        }
29    }
30    // 2 Oluşturucu
31    public Sefer(float KM, String nerden, String nereye, String sure, String date) {
32        DateTimeFormatter timeFormatter = DateTimeFormatter.ofPattern("HH:mm");
33        LocalTime sure1 = LocalTime.parse(sure, timeFormatter);
34        try {
35            DateTimeFormatter dateFormatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
36            LocalDateTime date1 = LocalDateTime.parse(date, dateFormatter);
37            new Sefer(KM, nerden, nereye, sure1, date1);
38        } catch (DateTimeParseException e) {
39            formattedDate = date;
40            this.no = No;
41            No++;
42            this.KM = KM;
43            this.nerden = nerden;
44            this.nereye = nereye;
45            this.sure = LocalTime.parse(sure, timeFormatter);
46            if (this.getClass() == Sefer.class) {
47                HavaYolu.SeferList.add(this);
48            }
49        }
50    }
51 }
```

her sefer için Numra , KM(mesafe) , kalkış yeri , iniş yeri , sure , tarihi ve iki farklı Oluşturucu methodu farklı tipleri almak için destekleyen bir Oluşturucu

- Şirket Sınıfı:

```
1 package havayolubilgisistemi;
2
3 import java.util.ArrayList;
4
5 public class Sirket {
6
7     private static int ID=1;
8     public int id;
9     public String name;
10    public Personel yoneticisi ;
11    public ArrayList<Ucak> ucakList = new ArrayList<Ucak>();// her Şirketin uçakların listesi verdir
12    public ArrayList<Ucus> ucusList = new ArrayList<Ucus>();// her Şirketin uçuşların listesi verdir
13
14    // set get methodlar
15    public String SirketinAdi() {return name;}
16    public String gettip(){ return "Sirket";}
17    public void ucakEkle(Ucak ucak) { this.ucakList.add(ucak);}
18    public void ucakEkle(ArrayList<Ucak> ucak) { this.ucakList.addAll(ucak);}
19    public void ucusEkle(Ucus ucus) { this.ucusList.add(ucus);}
20    public void setName(String name) {this.name=name;}
21    // Oluşturucu
22    public Sirket(String name,Ucak ucak) {
23        id=ID;
24        ID++;
25        this.name=name;
26        if(ucak!=null) {
27            ucakList.add(ucak);
28        }
29        boolean var=false;
30        for(Sirket sirket: HavaYolu.Sirketilist){
31            if(sirket.name.equals(name)){
32                var=true;
33                break;
34            }
35        }
36        if(!var)
37            HavaYolu.Sirketilist.add(this);
38    }
39
40 }
```

her Şirket için farklı bir ID , adı , yoneticisi , Uçakları , Uçuşları ve set-get Methodlar sahiptir

- **User(Kullanıcı) Sınıfı:**

```
1 package havayolubilgisistemi;
2
3 public abstract class User {
4
5     private static int ID=1; // Her Kulanıcı için farklı bir ID verilmektedir
6     private int id;
7     private String name;
8     public User(String name ) {
9         this.name = name;
10        id=ID;
11        ID++;
12        if(!this.gettip().equals("Yolcu"))
13            HavaYolu.PersonellList.add(this);
14    }
15    public String getName() {return this.name;}
16    public void setName(String name) {this.name=name;}
17    public int getId() {return id;}
18
19    public abstract String gettip();
20
21 }
```

Her bir Kulanıcı için farklı bir Kimlik Numarası , adı , get-set Methodları ve Oluşturucu sahiptir .

• Yolcu Sınıfı:

```
1 package havayolubilgisistemi;
2
3 import java.util.ArrayList;
4
5 public class Yolcu extends User{
6
7     public static String tip[]={"Standart", "VIP"};
8     public String Ytip=tip[0];
9     public int sıra , harf;
10    public ArrayList<Ucus> ucuslar;
11    public ArrayList<String> Koltuklar;
12    public String Koltuk;
13
14    public Yolcu(String name) {
15        super(name);
16        HavaYolu.YolcuList.add(this);
17        this.ucuslar=new ArrayList<>();
18        this.Koltuklar=new ArrayList<>();
19    }
20
21    @Override
22    public String gettip(){
23        return "Yolcu";
24    }
25 }
```

Her bir yolcu için Yolcu tipi , sıra , harf numaraları , uçuşları , aldı koltuk Numarası (Ö:17F) ve Oluşturucu

• Personel Sınıfı:

```
1 package havayolubilgisistemi;
2
3 import java.util.ArrayList;
4
5 public class Personel extends User {
6
7     public ArrayList<Ucus> Ucuslar = new ArrayList<>();
8     public Personel(String name) {
9         super(name);
10    }
11
12    @Override
13    public String gettip() {
14        return "personel";
15    }
16
17 }
```

her bir Personel farklı Görevleri ve Uçuşlar sahiptir .

- **Pilot Sınıfı:**

Pilot Personel aynı Özelliklere sahiptir fakat uçuş yapmak için pilot seçilir .

```
1 package havayolubilgisisistemi;
2
3 public class Pilot extends Personel {
4
5     public Pilot(String name) {
6         super(name);
7     }
8     @Override
9     public String gettip(){
10         return "Pilot";
11     }
12 }
```

- **Hostese Sınıfı:**

Hostese Personel aynı Özelliklere sahiptir fakat uçuş yapmak için uçakta hostestelerde olması lazım.

```
1 package havayolubilgisisistemi;
2
3 public class Hostese extends Personel{
4
5     public Hostese(String name) {
6         super(name);
7     }
8     @Override
9     public String gettip(){
10         return "Hostese";
11     }
12
13 }
```

- **Yönetici Sınıfı:**

Yönetici havaYolunun her bilgileri ulaşabilir, eklemek ve düzenlemek özelliklerine sahiptir .

```
1 package havayolubilgisistemi;
2
3 public class Yonetici extends User {
4
5
6     public HavaYolu havayolu=new HavaYolu();
7     private static Yonetici yonet=new Yonetici("Ahmed Salih");
8
9     public Yonetici(String name) {
10         super(name);
11     }
12     @Override
13     public String gettip(){
14         return "Yonetici";
15     }
16     public static Yonetici getYonet(){return yonet;}
17 }
```


• HavaYolu Sınıfı:

```
1 package havayolubilgisistemi;
2
3 import java.time.LocalDateTime;
4 import java.time.LocalTime;
5 import java.util.ArrayList;
6 import java.util.Scanner;
7
8 public class HavaYolu {
9
10     static String UcakTipi[] = { "Kucuk", "Orta", "Buyuk" };
11     public static String[] Airports = {
12         // region bilgi
13         "Istanbul Airport",
14         "Adana Airport",
15         "Ankara Esenboga Airport",
16         "Gazipasa Airport",
17         "Antalya Airport",
18         "Balikesir Koca Seyit Airport",
19         "Bursa Yenisehir Airport",
20         "Denizli Cardak Airport",
21         "Diyarbakir Airport",
22         "Elazig Airport",
23         "Erzurum Airport",
24         "Eskisehir Hasan Polatkan Airport",
25         "Gaziantep Airport",
26         "Hatay Airport",
27         "Isparta Suleyman Demirel Airport",
28         "Istanbul Sabiha Gokcen Airport",
29         "Izmir Adnan Menderes Airport",
30         "Kars Harakani Airport",
31         "Kayseri Airport",
32         "Kocaeli Cengiz Topel Airport",
33         "Konya Airport",
34         "Zafer Airport",
35
36         "Karaman Airport",
37         "Cukurova Regional Airport",
38         "Hasandagi (Nigde Aksaray) Airport",
39         "Yozgat Airport"
40         // endregion bilgi
41     };
42
43     static ArrayList<Sirket> Sirketilist = new ArrayList<>();
44     static ArrayList<User> Personellist = new ArrayList<>();
45     static ArrayList<Yolcu> YolcuList = new ArrayList<>();
46     static ArrayList<Sefer> SeferList = new ArrayList<>();
47     static ArrayList<Ucak> UcakList = new ArrayList<>();
48     static ArrayList<Ucus> UcusList = new ArrayList<>();
49
50     public void SetSefer(int No, float KM, String nerden, String nereye, LocalTime sure, LocalDateTime date) {
51         SeferList.add(new Sefer(KM, nerden, nereye, sure, date));
52     }
53
54     public void SetSefer(Sefer sefer) {
55         SeferList.add(new Sefer(sefer.KM, sefer.nerden, sefer.nereye, sefer.sure, sefer.date));
56     }
57
58     public void setSirket(Sirket sirket) {
59         Sirketilist.add(sirket);
60     }
61
62     public void UcusEkle(ArrayList<Ucus> ucusList) {
63         UcusList.addAll(ucusList);
64     }
65
66     public void UcusEkle(Ucus ucus) {
67         UcusList.add(ucus);
68     }
69 }
```

Hava Yolu Turkiyede Hava limaları ,
Ucak Tipleri

Hava yolun Şirketleri ,Personelleri ,
Uçakları , Yolcuları, destekledi
seferleri , Uçuşları ve set-get
methodları sahiptir .

- **HavaYoluBilgiSistemi Sınıfı(main):**

Uygulama Başlatırken önce biraz bilgiler oluşturmak gerekir.

```
1 package havayolubilgisistemi;
2
3 public class HavaYoluBilgiSistemi {
4
5     public static void main(String[] args) {
6
7         HavaYolu.create();
8
9         Baslangis basla =new Baslangis();
10        basla.setVisible(true);
11
12    }
13 }
```

- **create() Methodu :**

Uygulama açılırken bilgiler oluşturur

```
1 public static void create() {
2
3     LocalDateTime saat1 = LocalDateTime.now();
4     LocalDateTime saat = LocalDateTime.of(saat1.getHour(), saat1.getMinute());
5     LocalDateTime date = LocalDateTime.now();
6     Pilot pilot = new Pilot("Ahmed");
7     Hostese host = new Hostese("Fatma");
8     Ucak ucak = new Ucak(HavaYolu.UcakTipi[2], "AirJet", pilot, host, "AnadoluJet");
9     Yolcu yolcu = new Yolcu("Yusuf Demir");
10    ucak.kolutkList[2][2].yolcu=yolcu;
11    ucak.kolutkList[2][2].durum=true;
12    ucak.kolutkList[3][3].durum=true;
13    yolcu.Koltuk=ucak.kolutkList[2][2].No;
14    for(int j=1;j<ucak.siraSayisi;j++){
15        for(int k=0;k<ucak.harfSayisi;k++){
16            if(k>=2 && k<5)
17                ucak.kolutkList[j][k].durum=true;
18        }
19    }
20    int i = 100;
21    for (String airport : HavaYolu.Airports) {
22        i++;
23        date = date.plusDays(1);
24        Sefer sefer = new Sefer(i, airport, HavaYolu.Airports[0], saat, date);
25
26        new Ucus(saat, sefer, ucak,400);
27        new Sirket("AirPort" + Integer.toString(i), ucak);
28        new Ucak(HavaYolu.UcakTipi[1], "AnadoluJet", pilot, host, "AirPort");
29        new Yolcu(airport);
30
31    }
32 }
```

AnaSayfa

Ana Sayfada 3 Seçenek sahibtir .

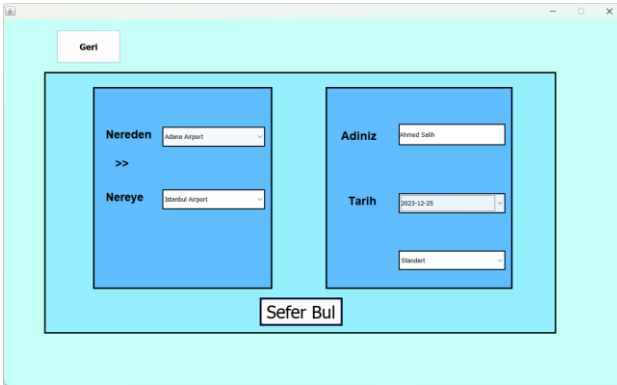


```
1 private void YolcuButtonActionPerformed(java.awt.event.ActionEvent evt) {  
2     // Yolcu Butonu  
3     Home home=new Home();  
4     home.setVisible(true);  
5     setVisible(false);  
6 }  
7  
8 private void PersonelButtonActionPerformed(java.awt.event.ActionEvent evt) {  
9     // Personel Butonu  
10    PersonelBilgileri home=new PersonelBilgileri();  
11    home.setVisible(true);  
12    setVisible(false);  
13 }  
14  
15 private void YoneticibuttonActionPerformed(java.awt.event.ActionEvent evt) {  
16     // Yoneticibutton Butonu  
17    Yoneticipage home=new Yoneticipage(Yoneticibutton.getYonet());  
18    home.setVisible(true);  
19    setVisible(false);  
20 }
```

Yolcu Sayfası

Yolcu sayfası Sefer bulmak için nereden nereye gideceni ve belirli bir tarihte bilgiler istenir aynı sayfada adı ve Yolcunu tipi Girmek zorunda

```
1 private void GeriButtonActionPerformed(java.awt.event.ActionEvent evt) {  
2     Baslangic basla= new Baslangic();  
3     basla.setVisible(true);  
4     setVisible(false);  
5 }  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100
```

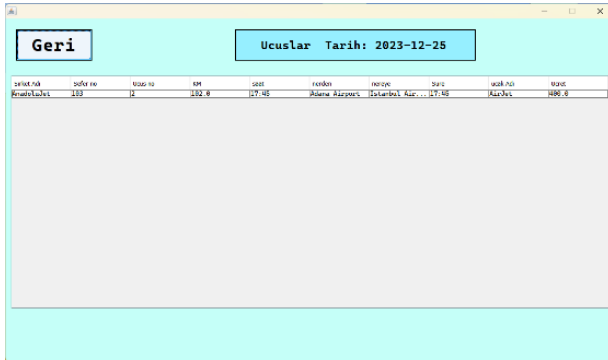


```
1 public Home() {  
2     initComponents();  
3     try {  
4         UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());  
5     } catch (ClassNotFoundException | ClassNotFoundException | InstantiationException | IllegalAccessException e) {  
6         e.printStackTrace();  
7     }  
8     Color backgroundColor = new Color(255, 255, 255);  
9     getContentPane().setBackground(backgroundColor);  
10    setTitle("Sky blue aeroplane");  
11    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
12    setSize(400, 300);  
13    setLocationRelativeTo(null);  
14    Nereden.addActionListener(new ActionListener() {  
15        @Override  
16        public void actionPerformed(ActionEvent e) {  
17            Nereye.setEnabled(true);  
18            for (String str : Home.Airports) {  
19                if(str.equals(Nereden.getSelectedText()))  
20                    Nereye.setSelected(str);  
21            }  
22        }  
23    });  
24 }  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100
```

```
1 private void gonder(String nereden,String nereye,String tarih,Yolcu yolcu){  
2     Object[] args = new Object[] { "Sefer",nereye,tarih,yolcu};  
3     ucus.setVisible(true);  
4     setVisible(false);  
5 }  
6  
7 private void formatactActionPerformed(java.awt.event.ActionEvent evt) {  
8     for (String str : Home.Airports) { // Nereden Bana Nereye Gideceğim diye sorar  
9         Nereden.setSelected(str);  
10    }  
11    Yolcu.setEnabled(true);  
12    Yolcu.setSelected("Yolcu"); // Yolcu tipi (Standart , VIP)  
13    Yolcu.setSelected("Yolcu");  
14    Nereye.setEnabled(true);  
15    Nereye.setSelected("Nereye");  
16    Tarih.setEnabled(true);  
17    Tarih.setSelected("Tarih");  
18 }  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100
```

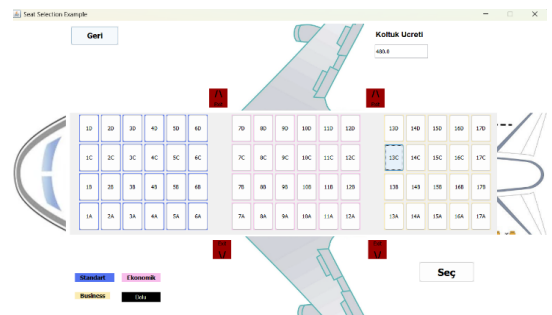
Uçuş Seçme Sayfası

Yolcu Sayfasında seçilen sefere göre uçuşlar gösterir , istedi uçuş basabilir ve o uçuşun uçak koltukları gösterir

[illegible]

Koltuk Seçme Sayfası

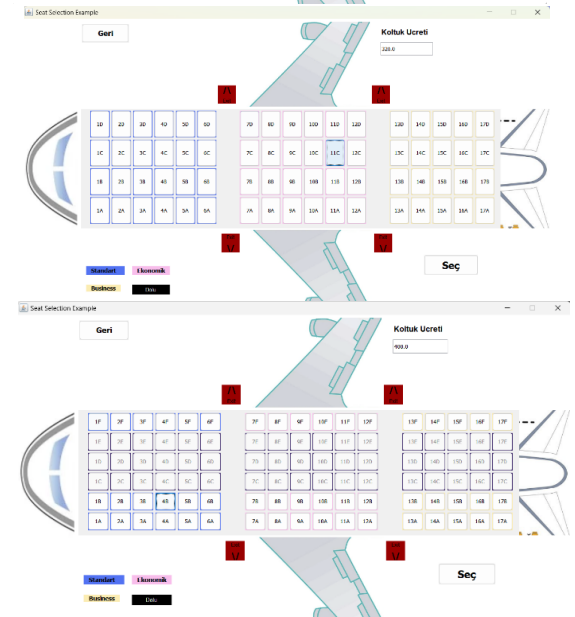
KotukSeçme Sayfası bir uçuş ve Yolcu alır o uçuşun uçak bilgilere göre gösterir ve aldı yolcunu tipine göre Koltuk ücreti gösterir aynı zamanda seçti koltuğuna göre ücret değişir



```

1 public class KoltukSec extends javax.swing.JFrame {
2
3     private JPanel seatPanel;
4     private ButtonGroup buttonGroup;
5     private Ucus ucus = null;
6     private Yolcu yol = null;
7
8     public KoltukSec(Ucus ucus, Yolcu yolcu) {
9         initComponents();
10        try {
11            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
12        } catch (UnsupportedLookAndFeelException | ClassNotFoundException | InstantiationException
13                | IllegalAccessException e) {
14            e.printStackTrace();
15        }
16        this.ucus = ucus;
17        this.yol = yolcu;
18        Color backgroundColor = new Color(197, 255, 248);
19        getContentPane().setBackgroundColor(backgroundColor);
20        Olustur();
21        jLabel1.setBackground(new Color(248, 189, 235));
22        SecButon1.setEnabled(false);
23    }

```



```

1 private void dolu(String koltuk) {
2     int row = 0, kol = 0, cul = 0;
3     if (koltuk.length() == 2) {
4         row = Character.getNumericValue(koltuk.charAt(0));
5         kol = koltuk.charAt(1) - 65;
6     } else if (koltuk.length() == 3) {
7         row = Character.getNumericValue(koltuk.charAt(0)) * 10;
8         cul = Character.getNumericValue(koltuk.charAt(1));
9         kol = koltuk.charAt(2) - 65;
10    }
11    ucus.ucak.kolutkList[row + cul][kol].durum = true;
12    double ucret = ucus.ucak.kolutkList[row + cul][kol].ucret;
13    if (yol.Ytip == Yolcu.tip[1]) {
14        ucret -= ucret * 1 / 5;
15    }
16    UcretTextField.setText(Double.toString(ucret));
17 }

```

Seat Selection Interface (Koltuk Seçim Ekranı)

Buttons: **Geri**, **Koltuk Ücreti** (input field), **Seç**

Seat Selection Tables:

2F	23F	44F	2F	2F	18F	18F
18	50	60	60	70	80	80
19	51	61	61	71	81	81
20	52	62	62	72	82	82
21	53	63	63	73	83	83
22	54	64	64	74	84	84
23	55	65	65	75	85	85
24	56	66	66	76	86	86
25	57	67	67	77	87	87
26	58	68	68	78	88	88
27	59	69	69	79	89	89
28	60	70	70	80	90	90
29	61	71	71	81	91	91
30	62	72	72	82	92	92
31	63	73	73	83	93	93
32	64	74	74	84	94	94
33	65	75	75	85	95	95
34	66	76	76	86	96	96
35	67	77	77	87	97	97
36	68	78	78	88	98	98
37	69	79	79	89	99	99
38	70	80	80	90	100	100
39	71	81	81	91	101	101
40	72	82	82	92	102	102
41	73	83	83	93	103	103
42	74	84	84	94	104	104
43	75	85	85	95	105	105
44	76	86	86	96	106	106
45	77	87	87	97	107	107
46	78	88	88	98	108	108
47	79	89	89	99	109	109
48	80	90	90	100	110	110
49	81	91	91	101	111	111
50	82	92	92	102	112	112
51	83	93	93	103	113	113
52	84	94	94	104	114	114
53	85	95	95	105	115	115
54	86	96	96	106	116	116
55	87	97	97	107	117	117
56	88	98	98	108	118	118
57	89	99	99	109	119	119
58	90	100	100	110	120	120
59	91	101	101	111	121	121
60	92	102	102	112	122	122
61	93	103	103	113	123	123
62	94	104	104	114	124	124
63	95	105	105	115	125	125
64	96	106	106	116	126	126
65	97	107	107	117	127	127
66	98	108	108	118	128	128
67	99	109	109	119	129	129
68	100	110	110	120	130	130
69	101	111	111	121	131	131
70	102	112	112	122	132	132
71	103	113	113	123	133	133
72	104	114	114	124	134	134
73	105	115	115	125	135	135
74	106	116	116	126	136	136
75	107	117	117	127	137	137
76	108	118	118	128	138	138
77	109	119	119	129	139	139
78	110	120	120	130	140	140
79	111	121	121	131	141	141
80	112	122	122	132	142	142
81	113	123	123	133	143	143
82	114	124	124	134	144	144
83	115	125	125	135	145	145
84	116	126	126	136	146	146
85	117	127	127	137	147	1

Sevi Selection Example

Geri

Kolluk Ücreti

1P	2P	3P	4P	5P	6P	7P	8P	9P	10P	11P	12P	13P	14P	15P	16P	17P	18P	19P	20P	11P	12P	13P	14P	15P	16P	17P	18P	19P	20P	21P	22P	23P	24P	25P	26P	27P	28P	29P	30P
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25					

{ 21 }

Bilet Sayfası

Bilet seçildi uçuş bilgilere göre yazdırır ve ve
biletin kim olduna adını yazdırır


Ana Saha

Biletiniz Hazır İyi Yolculuklar

AnadoluJetboarding Pass | Biniş Kartı

AnadoluJet

TK 166S



DATE | TARİH

2023-12-25

Mesafe

102.0 KM

GATE | KAPI

311

TIME | SAATİ

17:46

SEAT | KOLTUK

3C

Istanbul Airport

Adi : Ahmed Salih

Nerden : Adana Airport

Nereye : Istanbul Airport

Ucak Adi : AirJet

06	Yozgat Airport	Standart	13C
06	Ahmed Salih	VIP	

```
1 public class Bilet extends javax.swing.JFrame {
2
3     public Bilet(){initComponents();}
4
5     public Bilet(Ucus ucus , Yolcu yol) {
6         initComponents();
7         try {
8             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
9         } catch (UnsupportedLookAndFeelException | ClassNotFoundException | InstantiationException | IllegalAccessException e) {
10             e.printStackTrace();
11         }
12         // Bilet Bilgileri doldur
13         yolcuadilabel.setText(yol.koltuk);
14         yolcuadilabel.setText(yol.koltuk);
15         ucakadilabel.setText(ucus.ucak.name);
16         sirketlabel.setText(ucus.ucak.sirketadi);
17         sirketlabel.setText(ucus.ucak.sirketadi);
18         saatlabel.setText(ucus.saat.toString());
19         nereyelabel.setText(ucus.nereye);
20         nereyelabel2.setText(ucus.nereye);
21         nerdenlabel.setText(ucus.nerden);
22         tarihlabel.setText(ucus.formattedDate);
23         kmlabel.setText(Double.toString(ucus.km));
24         ImageIcon icon = createImageIcon("qr.png");
25         jlabel6 = new JLabel(icon);
26         getContentPane().add(jlabel6);
27
28     }
29     private ImageIcon createImageIcon(String path) {
30         URL imgurl = getClass().getResource(path);
31         if (imgurl != null) {
32             return new ImageIcon(imgurl);
33         } else {
34             System.err.println("Could not find file: " + path);
35             return null;
36         }
37     }
38 }
```

Personel Sayfası

Personel adına göre sorgulama yapılmaktadır ve o adına görevi ne olursa olsun tüm görevleri gösterir

Geri

Adınızı giriniz

Sorgula

Titre 1	Titre 2	Titre 3	Titre 4

[illegible]

```

1  public class PersonelBilgileri extends javax.swing.JFrame {
2
3      private String adi;
4
5      public PersonelBilgileri() {
6          initComponents();
7          color backgroundColor = new Color(197, 255, 248);
8          getContentPane().setBackground(backgroundColor);
9
10         SorButton.setEnabled(false);
11         AdiField.addActionListener(new ActionListener() {
12             @Override
13             public void actionPerformed(ActionEvent e) {
14                 adi = AdiField.getText().trim();
15                 if (ladi.equals("") && ladi.isEmpty()) {
16                     SorButton.setEnabled(true);
17                 }
18             }
19         });
20
21     }

```

```

1 private void NormalLoadActionPerformed(java.awt.event.ActionEvent evt) {
2     // TODO add your handling code here:
3     Personal p;
4     for (Person person : ModelValues.personModel) {
5         if (person instanceof Personal & person.getName().equals(Adi)) {
6             p = (Personal) person;
7             System.out.println(person.getName());
8             break;
9         } else {
10             continue;
11         }
12     }
13     if(p==null){
14         DefaultTableModel tableModel = new DefaultTableModel();
15         tableModel.addColumn("first Ad");
16         tableModel.addColumn("two");
17         tableModel.addColumn("three");
18         tableModel.addColumn("four");
19         tableModel.addColumn("five");
20         tableModel.addColumn("six");
21         tableModel.addColumn("seven");
22         tableModel.addColumn("eight");
23         tableModel.addColumn("nine");
24         tableModel.addColumn("ten");
25         for (User user : prc.getUser()) {
26             tableModel.addRow(new Object[] {user.userName, user.firstName, user.lastName, user.formattedDate, user.salary, user.senior, user.hobby, user.country, user.userName});
27         }
28         jTable1.setModel(tableModel);
29     }
30 }
31 private void AdiFieldActionPerformed(java.awt.event.ActionEvent evt) {
32     adi = AdiField.getText().trim();
33 }
34 // New Buttons
35 private void GetInitiationActionPerformed(java.awt.event.ActionEvent evt) {
36     ba1a1ng1 ba1a = new ba1a1ng1();
37     ba1a.setVisible(true);
38     setVisible(false);
39 }

```

Yönetici Sayfası

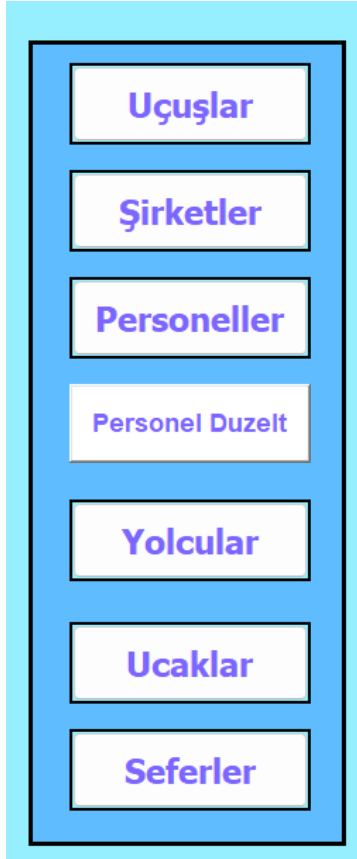
ilk olarak Yönetici şifresi sayfanın çalışmak için

```
1 public void Giriş() {
2     SirketButton.setEnabled(false);
3     PersonellerBtn.setEnabled(false);
4     SirketButton.setEnabled(false);
5     SirketlerBtn.setEnabled(false);
6     PilotButton.setEnabled(false);
7     HosteseButton.setEnabled(false);
8     SeferBtn.setEnabled(false);
9     SeferButton.setEnabled(false);
10    UcakButton.setEnabled(false);
11    UcusButton.setEnabled(false);
12    UcuslarBtn.setEnabled(false);
13    YolcuBtn.setEnabled(false);
14    UcakBtn1.setEnabled(false);
15    Userbutton.setEnabled(false);
16 }
17
18 public void Gizle() {
19     SaatBox.setEnabled(false);
20     UcretTxt.setEnabled(false);
21     AdiTxt.setEnabled(false);
22     KmTxt.setEnabled(false);
23     NeredenCBox.setEnabled(false);
24     NereyeCBox.setEnabled(false);
25     SaatTxt.setEnabled(false);
26     SureTxt.setEnabled(false);
27     TarihTxt.setEnabled(false);
28     SeferCBox.setEnabled(false);
29     UcakBox.setEnabled(false);
30     label3.setEnabled(false);
31     NLabel4.setEnabled(false);
32     label5.setEnabled(false);
33     label6KM.setEnabled(false);
34     NLabel7.setEnabled(false);
35     label8S.setEnabled(false);
36     label9T.setEnabled(false);
37     label10.setEnabled(false);
38     label11.setEnabled(false);
39     SeferGizBtn.setEnabled(false);
40     UcusGizBtn.setEnabled(false);
41     adiGizBtn.setEnabled(false);
42 }
```

```
1 public class Yoneticipage extends javax.swing.JFrame {
2
3     private String adi;
4     private static int id = 0;
5     private String mesaj;
6
7     public Yoneticipage(Yoneticipage yonet) {
8         initComponents();
9         try {
10             UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
11         } catch (UnsupportedLookAndFeelException | ClassNotFoundException | InstantiationException | IllegalAccessException e) {
12             e.printStackTrace();
13         }
14         color backgroundColor = new Color(150, 230, 255);
15         getContentPane().setBackground(backgroundColor);
16         if (Yoneticipage.getYonet().getId() != id) {
17             Giriş();
18         }
19         Girişbutton.setEnabled(false);
20         PassTextfield.addActionListener(new ActionListener() {
21             @Override
22             public void actionPerformed(ActionEvent e) { // Giriş Butonu basmak için Sifre Yazmak Kerekir
23                 Girişbutton.setEnabled(true);
24             }
25         });
26         Girişbutton.addActionListener(new ActionListener() {
27             @Override
28             public void actionPerformed(ActionEvent e) {
29                 id = Integer.parseInt(PassTextfield.getText()); // Sifre Durumu Değişimi
30                 if (id != 0) {
31                     if (Yoneticipage.getYonet().getId() == id) {
32                         en();
33                     } else {
34                         mesaj = "Sifre Yanlış !";
35                     }
36                 } else {
37                     Giriş();
38                 }
39             }
40         });
41     }
42 }
```

```
1 private void en() {
2     SirketButton.setEnabled(true);
3     PersonellerBtn.setEnabled(true);
4     SirketButton.setEnabled(true);
5     SirketlerBtn.setEnabled(true);
6     PilotButton.setEnabled(true);
7     HosteseButton.setEnabled(true);
8     SeferBtn.setEnabled(true);
9     SeferButton.setEnabled(true);
10    UcakButton.setEnabled(true);
11    UcusButton.setEnabled(true);
12    UcuslarBtn.setEnabled(true);
13    YolcuBtn.setEnabled(true);
14    UcakBtn1.setEnabled(true);
15    Userbutton.setEnabled(true);
16    PassTextfield.setText("");
17    System.out.println("Welcome");
18 }
```


Yönetici istedi bilgiler görebilir ve düzeltebilir



```
1 private void PersonellerBtnActionPerformed(java.awt.event.ActionEvent evt) {  
2     Ucuslar ucus = new Ucuslar("Personel");  
3     ucus.setVisible(true);  
4     setVisible(false);  
5 }  
6 private void SirketlerBtnActionPerformed(java.awt.event.ActionEvent evt) {  
7     Ucuslar ucus = new Ucuslar("Sirket");  
8     ucus.setVisible(true);  
9     setVisible(false);  
10 }  
11 private void YolcuBtnActionPerformed(java.awt.event.ActionEvent evt) {  
12     Ucuslar ucus = new Ucuslar("Yolcu");  
13     ucus.setVisible(true);  
14     setVisible(false);  
15 }  
16 private void SeferBtnActionPerformed(java.awt.event.ActionEvent evt) {  
17     Ucuslar ucus = new Ucuslar("Sefer");  
18     ucus.setVisible(true);  
19     setVisible(false);  
20 }  
21 private void UcakBtnActionPerformed(java.awt.event.ActionEvent evt) {  
22     Ucuslar ucus = new Ucuslar("Ucak");  
23     ucus.setVisible(true);  
24     setVisible(false);  
25 }  
26 private void UcuslarBtnActionPerformed(java.awt.event.ActionEvent evt) {  
27     Ucuslar ucus = new Ucuslar("Ucus");  
28     ucus.setVisible(true);  
29     setVisible(false);  
30 }  
31 private void UserbuttonActionPerformed(java.awt.event.ActionEvent evt) {  
32     Ucuslar ucus = new Ucuslar("User");  
33     ucus.setVisible(true);  
34     setVisible(false);  
35 }
```

```

1 public Ucuslar(String tip) {
2     tip = tip;
3     initComponents();
4     try {
5         UManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
6     } catch (ClassNotFoundException | ClassNotFoundException | InstantiationException | IllegalAccessException e) {
7         e.printStackTrace();
8     }
9     setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
10    setTitle("Uçuşlar");
11    getContentPane().setBackground(new Color(192, 203, 213));
12    getContentView().setBackground(new Color(192, 203, 213));
13    this.setSize(1000, 600);
14    Ucuslar.tip = tip;
15    DefaultTableModel tableModel = new DefaultTableModel();
16
17    if (tip.equals("Ucuş")) {
18        tableModel.addColumn("No");
19        tableModel.addColumn("Tarih");
20        tableModel.addColumn("Kod");
21        tableModel.addColumn("Tarih");
22        tableModel.addColumn("Nerden");
23        tableModel.addColumn("Nereye");
24        tableModel.addColumn("Sure");
25        tableModel.addColumn("Durum");
26        tableModel.addColumn("Geri");
27
28        for (Ucus ucus : HavaYolu.UcusList) {
29            tableModel.addRow(new Object[]{ucus.no, ucus.kod, ucus.tarih, ucus.nerden, ucus.nereye, ucus.sure, ucus.durum, ucus.geri});
30        }
31        UcuslarTable.setModel(tableModel);
32        UcuslarTable.getSelectionModel().addListSelectionListener(new ListSelectionListener() {
33            @Override
34            public void valueChanged(ListSelectionEvent e) {
35                if (e.getValueIsAdjusting()) {
36                    int selectedRow = UcuslarTable.getSelectedRow();
37                    if (selectedRow != -1) {
38                        for (Ucus ucus : HavaYolu.UcusList) {
39                            if (ucus.no == Integer.parseInt(UcuslarTable.getValueAt(selectedRow, 0).toString())) {
40                                Duzelt ucus = new Duzelt(ucus);
41                                ucus.setVisible(true);
42                                ucus.setVisible(false);
43                            }
44                        }
45                    }
46                }
47            }
48        });
49    }
50    } else if (tip.equals("Sefer")) {
51        tableModel.addColumn("No");
52        tableModel.addColumn("Kod");
53        tableModel.addColumn("Tarih");
54        tableModel.addColumn("Nerden");
55        tableModel.addColumn("Nereye");
56        tableModel.addColumn("Sure");
57
58        for (Sefer sefer : HavaYolu.SeferList) {
59            tableModel.addRow(new Object[]{sefer.no, sefer.kod, sefer.tarih, sefer.nerden, sefer.nereye, sefer.sure});
60        }
61        UcuslarTable.setModel(tableModel);
62        UcuslarTable.getSelectionModel().addListSelectionListener(new ListSelectionListener() {
63            @Override
64            public void valueChanged(ListSelectionEvent e) {
65                if (e.getValueIsAdjusting()) {
66                    int selectedRow = UcuslarTable.getSelectedRow();
67                    if (selectedRow != -1) {
68                        Sefer sefer = HavaYolu.SeferList.get(Integer.parseInt(UcuslarTable.getValueAt(selectedRow, 0).toString()) - 101);
69                        Duzelt ucus = new Duzelt(sefer);
70                        ucus.setVisible(true);
71                        ucus.setVisible(false);
72                    }
73                }
74            }
75        });
76    }
77    } else if (tip.equals("Personel")) {
78        tableModel.addColumn("ID");
79        tableModel.addColumn("Adi");
80        tableModel.addColumn("Gorevi");
81
82        for (User user : HavaYolu.PersonelList) {
83            tableModel.addRow(new Object[]{user.getId(), user.getAdi(), user.getGorevi()});
84        }
85        UcuslarTable.getSelectionModel().addListSelectionListener(new ListSelectionListener() {
86            @Override
87            public void valueChanged(ListSelectionEvent e) {
88                if (e.getValueIsAdjusting()) {
89                    int selectedRow = UcuslarTable.getSelectedRow();
90                    if (selectedRow != -1) {
91                        for (User per : HavaYolu.PersonelList) {
92                            if (per.getId() == Integer.parseInt(UcuslarTable.getValueAt(selectedRow, 0).toString())) {
93                                if (per.getGorevi().equals("Yoneticisi")) {
94                                    continue;
95                                } else {
96                                    Ucuslar ucus = new Ucuslar((Personel) per);
97                                    ucus.setVisible(true);
98                                    ucus.setVisible(false);
99                                }
100                            }
101                        }
102                    }
103                }
104            }
105        });
106        UcuslarTable.setModel(tableModel);
107    }
108 }

```

Yönetici seçti tip Uçuşlar sayfası
tip gönderir ve o tipi göre
bilgileri gösterir.

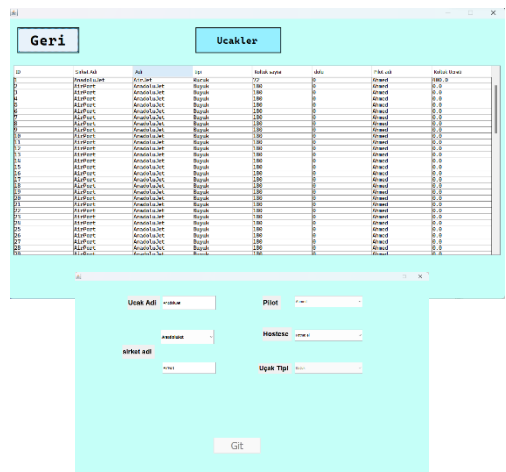
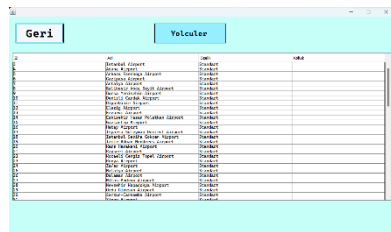
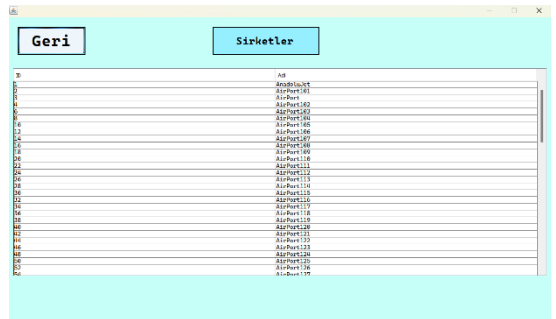
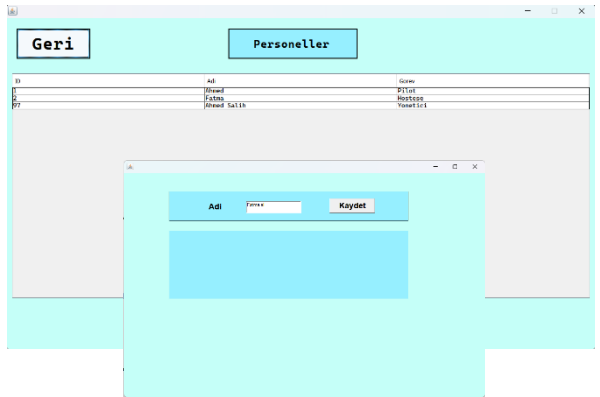
No	Tarih	Kod	Nerden	Nereye	Sure	Durum	Geri
101	2023-12-24	101	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	101
102	2023-12-25	102	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	102
103	2023-12-26	103	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	103
104	2023-12-27	104	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	104
105	2023-12-28	105	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	105
106	2023-12-29	106	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	106
107	2023-12-30	107	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	107
108	2023-12-31	108	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	108
109	2024-01-01	109	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	109
110	2024-01-02	110	Atatürk Hava	Atatürk Hava	15:00	Uçuşta	110

No	Tarih	Kod	Nerden	Nereye	Sure
101	2023-12-24	101	Atatürk Hava	Atatürk Hava	15:00
102	2023-12-25	102	Atatürk Hava	Atatürk Hava	15:00
103	2023-12-26	103	Atatürk Hava	Atatürk Hava	15:00
104	2023-12-27	104	Atatürk Hava	Atatürk Hava	15:00
105	2023-12-28	105	Atatürk Hava	Atatürk Hava	15:00
106	2023-12-29	106	Atatürk Hava	Atatürk Hava	15:00
107	2023-12-30	107	Atatürk Hava	Atatürk Hava	15:00
108	2023-12-31	108	Atatürk Hava	Atatürk Hava	15:00
109	2024-01-01	109	Atatürk Hava	Atatürk Hava	15:00
110	2024-01-02	110	Atatürk Hava	Atatürk Hava	15:00

ID	Adı	Görevi
1	Yılmaz	Yönetici
2	Yılmaz	Yönetici
3	Yılmaz	Yönetici

Tabloda seçti personel bilgilere göre görevleri gösterir

Personel düzelt sayfası seçti personele göre bilgileri düzeltebilir



Istedi uçak bilgileri düzeltebilir

Ekle

Pilot

Hostese

Şirket

Uçak

Kaydet

Adı

```
1 private void PilotButtonActionPerformed(java.awt.event.ActionEvent evt) {
2     Gizle(); // Pilot Ekle
3     Label3.setEnabled(true);
4     AdiTxt.setEnabled(true);
5     adiGitBtn.setEnabled(true);
6     adiGitBtn.addActionListener(new ActionListener() {
7         @Override
8         public void actionPerformed(ActionEvent e) {
9             String adi1 = AdiTxt.getText();
10            if (!adi1.equals("") && !adi1.isEmpty()) {
11                new Pilot(AdiTxt.getText().trim());
12                AdiTxt.setText("");
13                adi = "Personel";
14            } else {
15                adi = "Personel";
16                new Personel(adi);
17            }
18        }
19    });
20 }
```

```
1 private void HosteseButtonActionPerformed(java.awt.event.ActionEvent evt) {
2     Gizle();
3     Label3.setEnabled(true);
4     AdiTxt.setEnabled(true);
5     adiGitBtn.setEnabled(true);
6     adiGitBtn.addActionListener(new ActionListener() {
7         @Override
8         public void actionPerformed(ActionEvent e) {
9             String adi1 = AdiTxt.getText();
10            if (!adi1.equals("") && !adi1.isEmpty()) {
11                new Hostese(AdiTxt.getText().trim());
12                adi = "Personel";
13                AdiTxt.setText("");
14            }
15        }
16    });
17 }
```

```
1 private void SirketButtonActionPerformed(java.awt.event.ActionEvent evt) {
2     Gizle();
3     Label3.setEnabled(true);
4     AdiTxt.setEnabled(true);
5     adiGitBtn.setEnabled(true);
6     adiGitBtn.addActionListener(new ActionListener() {
7         @Override
8         public void actionPerformed(ActionEvent e) {
9             String adi1 = AdiTxt.getText();
10            if (!adi1.equals("") && !adi1.isEmpty()) {
11                new Sirket(AdiTxt.getText().trim(), null);
12                adi = "Sirket";
13                AdiTxt.setText("");
14            }
15        }
16    });
17 }
```

ekleme yapmak için önce
istedi tipi seçerek ondan
sonra bilgileri girir

```
1 private void adiGitBtnActionPerformed(java.awt.event.ActionEvent evt) {
2     if (adi.equals("Personel")) {
3         Ucuslar home = new Ucuslar(adi);
4         home.setVisible(true);
5         setVisible(false);
6     } else if (adi.equals("Sirket")) {
7         System.out.println(AdiTxt.getText());
8         Ucuslar home = new Ucuslar(adi);
9         home.setVisible(true);
10        setVisible(false);
11    } else {
12        adi = "Personel";
13        new Personel(adi);
14    }
15 }
```

```
1 private void UcakButtonActionPerformed(java.awt.event.ActionEvent evt) {
2     Gizle();
3     UcakPage home = new UcakPage();
4     home.setVisible(true);
5 }
```

Uçuş Ekle

00:30

Saat

Ücret

Uçak

AirJet

Kaydet

Sefer

Ankara Esenboga Airport > İstanbul Airport

Sefer Ekle

Kaydet

Nereden

Ankara Esenboga Airport

Nereye

İstanbul Airport

KM

103.0

Tarih

2023-12-27

Süre

19:26

Sefer eklemek için ister var olan sefer düzeltir yade ekleyebilir

Uçuş eklemek için var olan sefer seçilebilir yada eklene bilir ve uçuşun hangi uçağa olacağını seçilebilir

```
private void SeferGitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    Sefer sef = new Sefer(Float.parseFloat(KmTxt.getText()), NeredenCBox.getSelectedIndex().toString(),
        NereyeCBox.getSelectedIndex().toString(), SureTxt.getText(), TarihTxt.getText());
    HavaYolu.SeferList.add(sef);
    Ucuslar ucus = new Ucuslar("Sefer");
    ucus.setVisible(true);
    setVisible(false);
}
```

```
private void SeferCBoxActionPerformed(java.awt.event.ActionEvent evt) {
    SeferCBox.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent a) {
            int selectedIndex = SeferCBox.getSelectedIndex();
            System.out.println("Selected Index: " + selectedIndex);
            if ((HavaYolu.SeferList.isEmpty() && selectedIndex >= 0 && selectedIndex < HavaYolu.SeferList.size()) {
                Sefer sefer = HavaYolu.SeferList.get(selectedIndex);
                NeredenCBox.setSelectedIndex(sefer.nerden);
                NereyeCBox.setSelectedIndex(sefer.nereye);
                HavaYolu.SeferList.add(sefer);
                SeferCBox.setSelectedIndex(sefer.nereye);
                SureTxt.setText(sefer.sure.toString());
                TarihTxt.setText(sefer.formattedDate);
                KmTxt.setText(Double.toString(sefer.KM));
            } else {
                System.out.println("Invalid selection or SeferList is empty.");
            }
        }
    });
}
```

```
private void UcusBtnActionPerformed(java.awt.event.ActionEvent evt) {
    G114();
    SaatBox.setEnabled(true);
    NeredenCBox.setEnabled(true);
    NereyeCBox.setEnabled(true);
    SureTxt.setEnabled(true);
    TarihTxt.setEnabled(true);
    KmTxt.setEnabled(true);
    N1label14.setEnabled(true);
    N1label17.setEnabled(true);
    Label100.setEnabled(true);
    Label107.setEnabled(true);
    Label105.setEnabled(true);
    SaatTxt.setEnabled(true);
    UcureTxt.setEnabled(true);
    Label111.setEnabled(true);
    Label110.setEnabled(true);
    Label15.setEnabled(true);
    SeferCBox.setEnabled(true);
    UcusGitBtn.setEnabled(true);
    UcakBox.setEnabled(true);
}
```

```
private void UcusGitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    LocalTime saat = LocalTime.parse(SaatTxt.getText());
    Sefer sefer = HavaYolu.SeferList.get(SeferCBox.getSelectedIndex());
    Ucak ucak = HavaYolu.UcakList.get(UcakBox.getSelectedIndex());
    double koltuk = Double.parseDouble(UcretTxt.getText());
    new Ucus(saat, sefer, ucak, koltuk);
    Ucuslar ucus = new Ucuslar("Ucus");
    ucus.setVisible(true);
    setVisible(false);
}
```