# Bus Ticket Automation

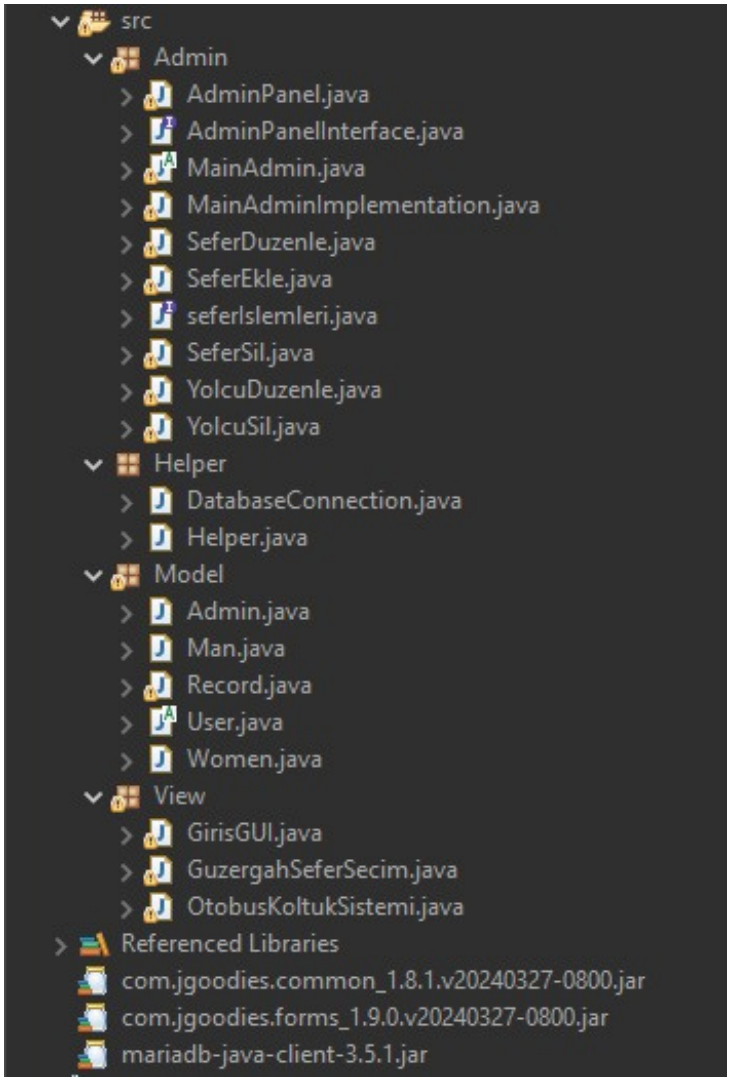
**Project Description:**

The Purpose of the Project is to Design and Implement a Bus Ticket Automation System. This System Aims to Allow Users to View Bus Trips, Make Ticket Reservations and Purchase Tickets.

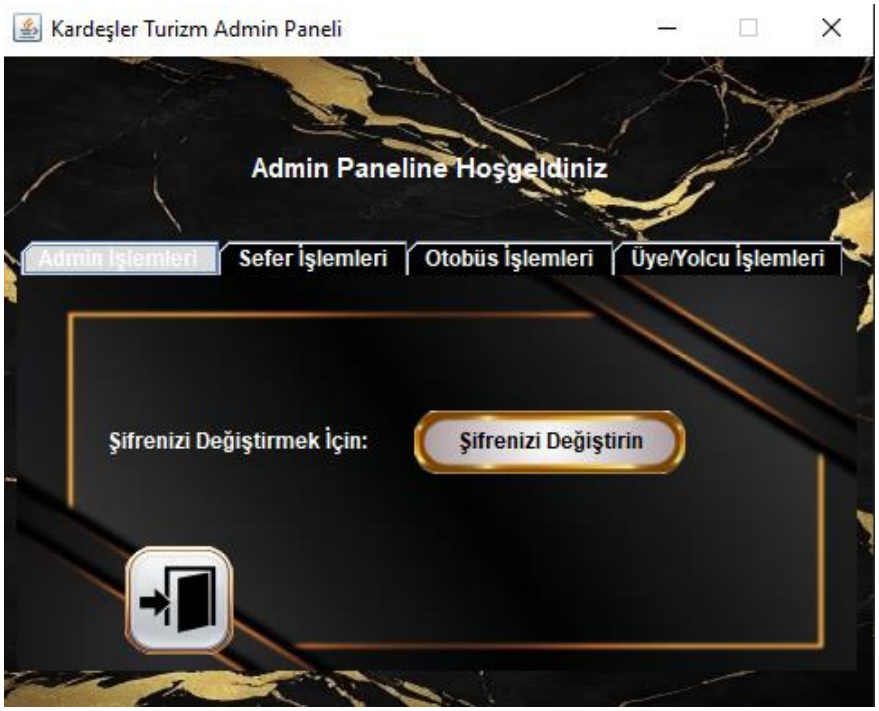
There is also an Admin Panel in the Project. This Admin Panel includes Admin
Password Transactions, Add/Delete Driver, Add/Delete Bus, Add/Edit/Delete Trip, Edit/Delete Passenger and Delete Member Transactions. In this way, System Management

Can be Performed Easily and Necessary Updates Can be Made Quickly.

## Software  Architecture:



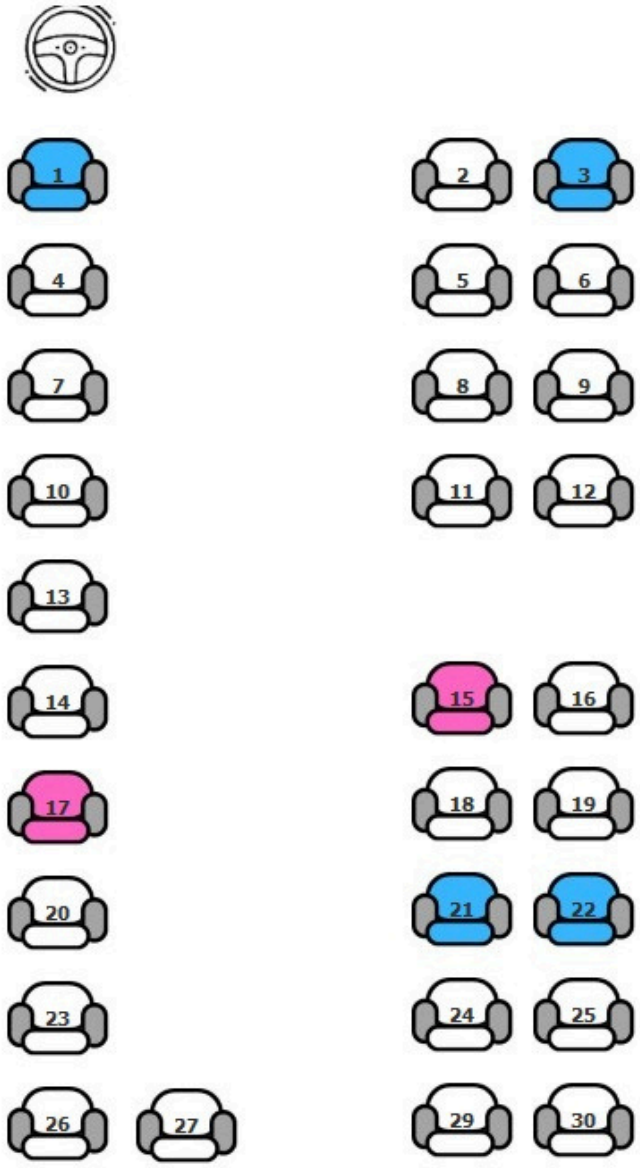
There are 4 packages in our project: **Admin**, **Helper**, **Model** and **View**. **Admin** Package includes Admin Panel and related operations. Database Connection and **Helper** Classes are included in Helper Package. **Model** Package includes **Admin**, **Man**, **Record**, **User** and **Women** Classes and Variables of These Classes. **View** Package includes Login Operations, Travel Route Selection Panel and Bus Seat Selection Panel.




Our Admin Panel, which has a modern and stylish design, is enriched with tables such as **Admin Operations, Trip Operations, Bus Operations** and **Member/Passenger Operations.** These tables ensure that management processes are carried out in the easiest and fastest way and take the user experience to the next level.

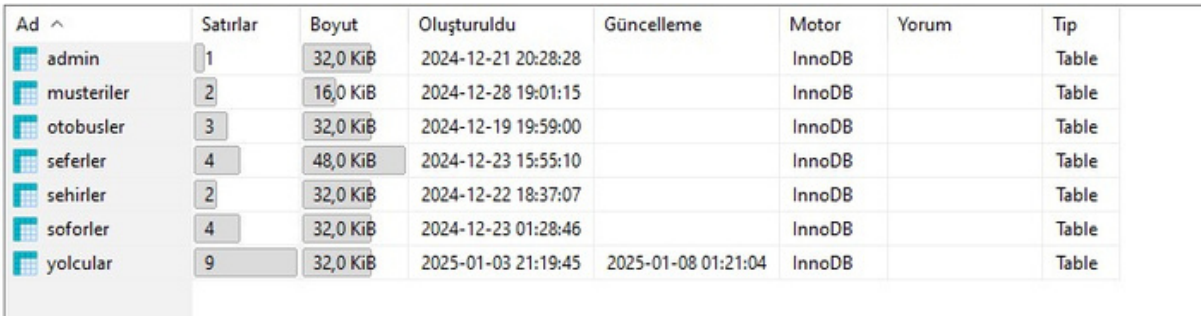

We have a user-friendly login panel designed with the inspiration of simplicity and plainness. Thanks to this panel, users can easily complete their registration processes and log in to the system. At the same time, there is also a stylish and functional admin panel login area reserved for administrators.

In our trip selection screen, which we designed with elegance and simplicity in mind, users can easily select the trip they want from the trips they see by specifying their trip date and route and continue their transactions quickly.




Thanks to the seat designs prepared by us, it has become clear to see which passenger is sitting in which seat and the status of the empty seats. This design aims to facilitate the seat selection of users and to increase the functionality of the system.

## Database Architecture:



| Ad | Satırlar | Boyut | Oluşturuldu | Güncelleme | Motor | Yorum | Tip |
|---|---|---|---|---|---|---|---|
| admin | 1 | 32,0 KiB | 2024-12-21 20:28:28 | | InnoDB | | Table |
| musteriler | 2 | 16,0 KiB | 2024-12-28 19:01:15 | | InnoDB | | Table |
| otobusler | 3 | 32,0 KiB | 2024-12-19 19:59:00 | | InnoDB | | Table |
| seferler | 4 | 48,0 KiB | 2024-12-23 15:55:10 | | InnoDB | | Table |
| sehirler | 2 | 32,0 KiB | 2024-12-22 18:37:07 | | InnoDB | | Table |
| soforler | 4 | 32,0 KiB | 2024-12-23 01:28:46 | | InnoDB | | Table |
| yolcular | 9 | 32,0 KiB | 2025-01-03 21:19:45 | 2025-01-08 01:21:04 | InnoDB | | Table |

There are 7 tables in total in our project database. Admin information is kept in the **"admin"** table, customer information is kept in the **"customers"** table, bus information is kept in the **"buses"** table, trip content and information is kept in the **"trips"** table, routes are kept in the **"cities"** table, driver information and content is kept in the **"drivers"** table, passengers who will join the trip and necessary information is kept in the **"passengers"** table.

Sütunlar: Ekle Sil Yukarı Aşağı

| # | Ad | Veritipi | Uzunluk/Ayar | İmzasız | NULL'a izin ver | Sıfırdoldur | Varsayılan | Yorum | Collation |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ID | INT | 11 | ☐ | ☐ | ☐ | AUTO_INCREMENT | | |
| 2 | tc_kimlik | VARCHAR | 50 | ☐ | ☐ | ☐ | '' | | utf8mb4_uca1400_ai_ci |
| 3 | isim | VARCHAR | 50 | ☐ | ☐ | ☐ | '' | | utf8mb4_uca1400_ai_ci |
| 4 | soyisim | VARCHAR | 50 | ☐ | ☐ | ☐ | '' | | utf8mb4_uca1400_ai_ci |
| 5 | koltuk_durumu | ENUM | 'alindi','bos' | ☐ | ☐ | ☐ | Varsayılan yok | | utf8mb4_uca1400_ai_ci |
| 6 | koltuk_cinsiyet | ENUM | 'Erkek','Kadin' | ☐ | ☐ | ☐ | Varsayılan yok | | utf8mb4_uca1400_ai_ci |
| 7 | koltuk_numara | INT | 11 | ☐ | ☐ | ☐ | Varsayılan yok | | |
| 8 | sefer_id | INT | 11 | ☐ | ☐ | ☐ | Varsayılan yok | | |

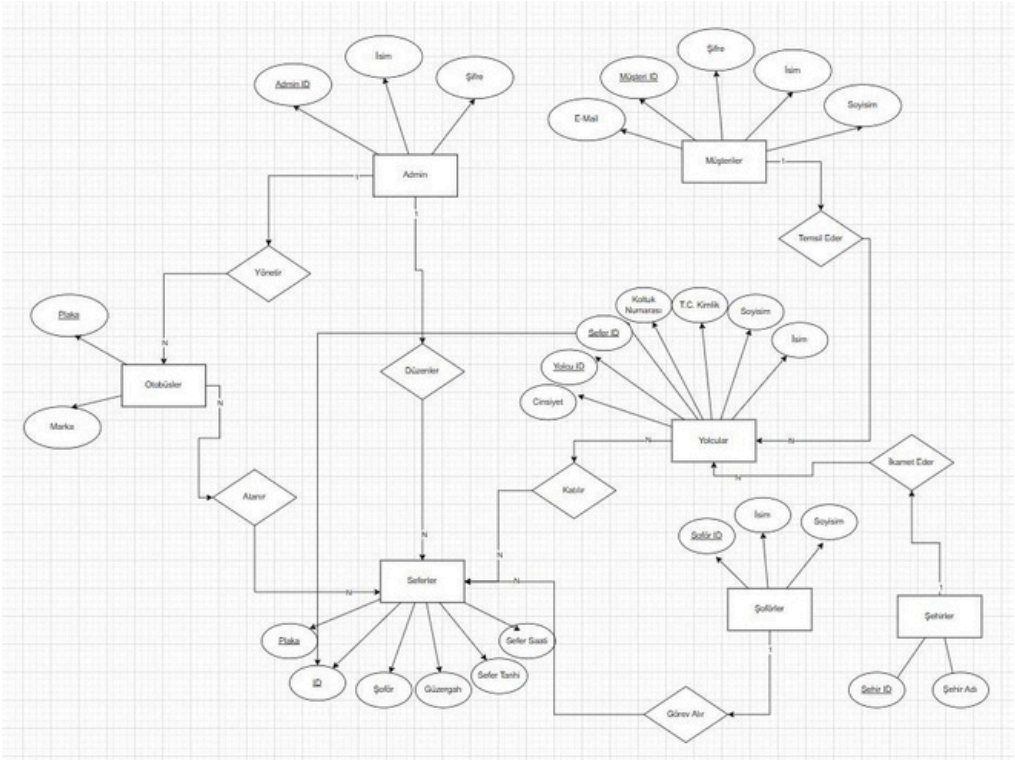
seferyönetim.yolcular: 8 satır mevcut (exact) Sonraki Hepsini gö:

| # | ID | tc_kimlik | isim | soyisim | koltuk_durumu | koltuk_cinsiyet | koltuk_numara | sefer_id |
|---|---|---|---|---|---|---|---|---|
| 1 | 16 | 12345678901 | Mehmet | [redacted] | alindi | Erkek | 3 | 3 |
| 2 | 22 | 12312312321 | Ali Adem | [redacted] | alindi | Erkek | 3 | 1 |
| 3 | 24 | 12312333322 | abbbb | cccc | alindi | Kadin | 7 | 1 |
| 4 | 26 | 1231212 | Ayşe | [redacted] | alindi | Kadin | 22 | 1 |
| 5 | 27 | 9898998 | Ali | [redacted] | alindi | Erkek | 21 | 1 |
| 6 | 28 | 6876887878 | Duygu | [redacted] | alindi | Kadin | 17 | 1 |
| 7 | 29 | 312312321 | Ahmet | [redacted] | alindi | Erkek | 12 | 3 |
| 8 | 30 | 123123123 | Ayşe | [redacted] | alindi | Kadin | 9 | 3 |

Let's give an example from our database tables **"passengers"** table. The Turkish ID Number, Names, Surnames, Gender, Seat Numbers and Trip IDs of Passengers Who Purchase Tickets are Recorded. In addition, **"seat_status"** is automatically marked as **'taken'**.
At the same time, the **'trip_id'** column in the **"passengers"** table and the **'ID'** column in the **"trips"** table are connected to each other and there is a different ID for each trip.

A different seat layout is opened for each ID depending on the **BusSeatSystem.Java** file.

## E-R Diagram:




The diagram includes Admin, Customers, Passengers, Buses, Trips, Drivers and Cities. Each Element is Defined by Its Own Properties and Relationships with Other Elements. The Diagram Helps Us See How the Elements in the System Are Interconnected.

## Some Extra Features:

```
private static void butonDegistir(JButton button) {

    button.setHorizontalTextPosition(SwingConstants.CENTER); // Yazıyı merkeze hizala
    button.setVerticalTextPosition(SwingConstants.CENTER); // Yazıyı merkeze hizala
    button.setForeground(Color.BLACK); // Yazıyı Siyah yap
    button.setFont(new Font("Arial", Font.BOLD, 12)); // Yazı tipini ayarla
    button.setOpaque(false); // Şeffaflık
    button.setContentAreaFilled(false); // Dolgu olmadan fotoğraf kullan
    button.setFocusPainted(false); // Tıklama efekti kaldır
    button.setBorderPainted(false); // Kenarlıkları kaldır
    button.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR)); // El imleci

}
```

A function called **"buttonDegistir"** has been designed to improve the appearance and functionality of the buttons. This function allows adding photos, fonts, text colors, effects, borders, mouse cursors and other visual features to all buttons included in the project in a way that will improve the user experience. Thus, both the visual and interaction features of the buttons have been made more user-friendly.

```
//Yan koltuğun cinsiyetinin çelişip çelişmediğini kontrol eden fonksiyon
private boolean cinsiyet_kontrol(String koltukNo, String gender) {
 int koltukNumarasi = Integer.parseInt(koltukNo);
 if (yanKoltuklar.containsKey(koltukNumarasi)) {
     int yanKoltukNo = yanKoltuklar.get(koltukNumarasi);
     if (koltukCinsiyetMap.containsKey(yanKoltukNo)) {
         // Eğer yan koltuk zaten dolu ve cinsiyet farklıysa
         return !koltukCinsiyetMap.get(yanKoltukNo).equals(gender);
     }
 }
 return false;
}
```

During the seat selection phase, there is a **"gender_control"** function that prevents a passenger of the opposite sex from sitting next to an existing female passenger or a current male passenger. This function increases the comfort and safety of passengers by ensuring that the seating arrangement complies with the standards.

```
// Geçerli e-posta adresi kontrolü
if (!email.matches("^[A-Za-z0-9+_.-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}$")) {
    JOptionPane.showMessageDialog(null, "Lütfen geçerli bir e-posta adresi girin.", "Hata", JOptionPane.ERROR_MESSAGE);
    return;
}
```

In the registration panel, a condition-based security measure has been taken to create a valid e-mail address. Thanks to this measure, e-mail addresses that do not contain certain characters or are invalid will not be saved in the system. Thus, the use of correct and valid e-mail addresses is ensured.

```
private void koltuklariYukle() {
    koltukComboBox.removeAllItems(); // Mevcut öğeleri temizler
    String selectedSeferId = (String) seferComboBox.getSelectedItem(); // Seçilen sefer_id'yi al

    if (selectedSeferId == null) {
        return; // Sefer seçilmediyse çık
    }

    try (Connection connect = DatabaseConnection.getConnection()) {
        // Sefer_id'ye göre alınmış koltukları al
        String sql = "SELECT koltuk_numara FROM yolcular WHERE sefer_id = ?";
        PreparedStatement state = connect.prepareStatement(sql);
        state.setString(1, selectedSeferId); // Sefer_id parametresini ayarla
        ResultSet rs = state.executeQuery();

        // Alınan koltukları bir listeye ekle
        boolean[] doluKoltuklar = new boolean[30];
        while (rs.next()) {
            int koltukNumara = rs.getInt("koltuk_numara");
            if (koltukNumara > 0 && koltukNumara <= 30) {
                doluKoltuklar[koltukNumara - 1] = true; // Koltuk numarasını işaretle
            }
        }

        // Boş koltukları combobox'a ekle
        for (int i = 0; i < 30; i++) {
            if (!doluKoltuklar[i]) {
                koltukComboBox.addItem(String.valueOf(i + 1)); // Boş koltuk numaralarını ekle
            }
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
```

The **"seatsUpload"** function in the **"passengerDuzenle.java"** file finds the relevant trip in the database by taking the **'ID'** value on the **"seferComboBox"**. Then, it adds the seats marked as **'taken'** to a list and adds only the **'empty'** seats to the **"seatComboBox"** with the help of a loop. This process ensures that the passenger's information is successfully edited.

## Proje Özeti Ve Sonucu :

Bu Proje, Harran Üniversitesi Bilgisayar Mühendisliği Bölümü'nün "Java İle Programlama" Dersi Kapsamında Geliştirilen Bir Otobüs Bilet Otomasyonu Sistemini Sunmaktadır. Sistem, Kullanıcıların Otobüs Seferlerini Görüntüleyip Bilet Rezervasyonu Yapmalarını ve Satın Almalarını Sağlamaktadır. Ayrıca, Bir Admin Paneli Aracılığıyla Yöneticilerin Şifre İşlemleri, Otobüs, Sefer, Şoför ve Yolcu Yönetim İşlemlerini Kolaylıkla Gerçekleştirmelerini Mümkün Kılmaktadır. Proje, Kullanıcı Dostu Bir Arayüz ve Verimli Yönetim Araçlarıyla Donatılmış Olup, Modern ve Şık Tasarımlarla Kullanıcı Deneyimini En Üst Düzeye Çıkarmayı Hedeflemektedir. Veritabanı Yapısı ve E-R Diyagramı, Sistemdeki Tüm Öğeler Arasındaki İlişkileri Görsel Olarak Sunarak İşleyişin Daha Anlaşılır Olmasını Sağlamaktadır. Ayrıca, Çeşitli Güvenlik ve İşlevsellik Özellikleri, Kullanıcıların Doğru ve Güvenli İşlemler Yapmalarını Güvence Altına Almaktadır. Sonuç Olarak, Proje Başarılı Bir Şekilde Otobüs Bilet Otomasyonu Sürecini Dijitalleştirip, Verimli Bir Yönetim ve Kullanıcı Deneyimi Sunarak Amacına Ulaşmıştır.