



T.C

**KOCAELİ SAęLIK VE TEKNOLOJİ ÜNİVERSİTESİ**  
**BİLGİSAYAR MÜHENDİSLİęİ PROGRAMI**

## **ÖDEV KONUSU**

### **VERİTABANI YÖNETİM SİSTEMİ**

**Hazırlayan**

**Esmanur AYDIN**

**220501005**

**DERS SORUMLUSU**

**Dr. Öğr. Üyesi ERCAN ÖLÇER**

**05/05/2024**

[esm4ydn \(Esmanur AYDIN\) · GitHub](#)

---

## İÇİNDEKİLER

1.	GİRİŞ .....	3
2.	GEREKSİNİM ANALİZİ .....	3
3.	TASARIM .....	6
4.	UYGULAMA.....	10
5.	TEST VE DOĞRULAMA .....	12
6.	KAYNAKÇA .....	14

## 1. GİRİŞ

### Projenin Amacı:

Bu proje, bir gemi şirketinin operasyonel verimliliğini arttırmak ve yönetim süreçlerini otomatikleştirmek amacıyla tasarlanmıştır. Projenin temel amacı, şirketin gemi, sefer, kaptan ve mürettebat yönetimini kolaylaştıracak bir bilgi sistemini geliştirmektir. Sistem, gemilerin farklı seferlere atanması, her sefere uygun kaptan ve mürettebatın belirlenmesi ve sefer sırasında uğranılan limanların takibini sağlayacak şekilde kurgulanmıştır.

Bu yazılım, kullanıcıların seferleri daha etkin planlamasına, gemi ve personel kaynaklarını optimal şekilde kullanmasına ve operasyonel kararları veriye dayalı olarak almasına olanak tanır. Ayrıca, şirketin mevcut ve gelecek seferlerini izlemesi, planlanan seferlere hızlı bir şekilde müdahale etmesi ve gerektiğinde düzenlemeler yapabilmesi için gerekli araçları sağlar.

Sistem, aynı zamanda birden fazla limana uğrayabilen seferler için detaylı rotaların ve zamanlamaların yönetilmesini destekler. Liman ziyaretleri sırasında gerçekleşen tüm faaliyetler kaydedilir ve limanların kullanım frekansına göre analizler yapılabilir. Bu, liman işlemlerinin verimliliğini artırma ve maliyetleri azaltma fırsatı sunar.

Özetle, bu yazılım projesi, gemi şirketinin operasyonel süreçlerini modernize etmeyi, yönetim kararlarını desteklemeyi ve tüm operasyonel verileri merkezi ve erişilebilir bir platformda tutmayı hedeflemektedir. Proje, şirketin karşılaştığı zorluklara çözüm getirerek sektördeki rekabet avantajını artırmayı amaçlamaktadır.

## 2. GEREKSİNİM ANALİZİ

### Arayüz Gereksinimleri:

- Ana Menü: Kullanıcıya gemiler, seferler, kaptanlar, mürettebat ve limanlar gibi ana işlevselliklere hızlı erişim sağlar.
- Detaylı Filtreleme ve Arama: Gemi, sefer ve personel bilgileri üzerinde detaylı filtreleme ve arama yapma imkânı.
- Form Tabanlı Girişler: Gemi ekleme, kaptan atama, sefer planlama gibi işlemler için kullanıcı dostu form tabanlı girişler.

Ödev No:2	Tarih 05.05.2024	3/14
-----------	------------------	------

- 
- Güncelleme ve Silme İşlevleri: Kaydedilmiş bilgiler üzerinde düzenleme ve silme işlevleri kolaylıkla yapılabilmelidir.
  - Raporlama ve İstatistikler: Seferler, liman kullanımları ve operasyonel verimlilik hakkında raporlar ve grafikler sunabilme.

### Donanım Gereksinimleri:

Yazılımın verimli bir şekilde çalışabilmesi için belirli donanım gereksinimleri önerilmektedir. Donanım gereksinimleri aşağıdaki özellikleri kapsamaktadır:

- Sunucu: Yazılımın veritabanı ve arka uç işlemleri için güçlü bir sunucu gerekmektedir. Önerilen minimum konfigürasyon: *4 çekirdek CPU, 16GB RAM, 200GB SSD*.
- İş İstasyonları: Kullanıcıların sistemle etkileşimde bulunacakları bilgisayarlar modern web tarayıcılarıyla uyumlu olmalı ve en az *8GB RAM, i5* veya üzeri *CPU, 256GB SSD* özelliklerine sahip olmalıdır.
- Ağ Bağlantısı: Yazılımın düzgün çalışması için stabil ve hızlı bir internet bağlantısı gerekmektedir. Tüm kullanıcılar ve sunucular arası veri transfer hızı yüksek ve güvenli olmalıdır.

### Fonksiyonel Gereksinimler:

Fonksiyonel gereksinimler, yazılımın ne yapması gerektiğini ve hangi işlevleri yerine getirmesi gerektiğini tanımlar. Bu proje için belirlenen fonksiyonel gereksinimler şunlardır:

- Gemi Yönetimi:
  - Gemi ekleme, silme ve düzenleme işlevleri.
  - Gemi bilgilerini (adı, ağırlığı, inşa yılı, kapasite bilgileri) güncelleme.
  - Gemilerin mevcut durumunu ve özelliklerini listeleme.

Ödev No:2	Tarih 05.05.2024	4/14
-----------	------------------	------

- 
- Sefer Yönetimi:
    - Yeni seferler oluşturma ve mevcut seferleri düzenleme.
    - Seferlere kaptan ve mürettebat atama.
    - Sefer güzergahlarına liman ekleyip çıkarma.
    - Sefer başlangıç ve bitiş tarihlerini belirleme.
    - Her sefer için minimum personel sayısının (*en az iki kaptan ve bir mürettebat*) sağlanmasını kontrol etme.
  - Kaptan ve Mürettebat Yönetimi:
    - Kaptan ve mürettebat için yeni kayıt ekleme, bilgi güncelleme ve silme.
    - Kaptanların lisans bilgilerini ve mürettebatın görev detaylarını yönetme.
    - Personelin seferlere atanma durumunu izleme ve çakışmaları önleme.
  - Liman Yönetimi:
    - Liman bilgilerini ekleme, güncelleme ve silme.
    - Liman kullanım ücretleri ve hizmet detaylarını yönetme.
    - Limanlara ilk ziyaret durumunu (*is\_first\_time*) güncelleme ve takip etme.
  - Raporlama ve İstatistikler:
    - Sefer başarımlar raporumlar, liman kullanım istatistikleri ve personel performans analizleri sunma.
    - Operasyonel verimlilik için gerekli tüm verilerin raporlanması ve görselleştirilmesi.
  - Güvenlik ve Erişim Kontrolü:
    - Kullanıcıların sisteme erişim seviyelerini kontrol etme.
    - Veri güvenliğini sağlamak için yetkilendirme ve kimlik doğrulama mekanizmaları.
    - Kritik işlemler için ek güvenlik önlemleri ve denetim izleri.

Ödev No:2	Tarih 05.05.2024	5/14
-----------	------------------	------

---

### 3. TASARIM

#### Mimari Tasarım:

Bu proje, modüler ve yeniden kullanılabilir bir kod yapısına sahiptir, nesne yönelimli programlama paradigması kullanılarak geliştirilmiştir. Yazılımın ana bileşenleri gemi, sefer, kaptan, mürettebat ve liman yönetimi modüllerini içerir. Her modül, ilgili işlevlerle ilgili sınıflar ve metodlar içerir, bu da sistem üzerinde işlemlerin kolaylıkla yürütülmesini sağlar.

#### Kullanılacak Teknolojiler:

- **Yazılımın Hangi Dilde Yazılacağı:** Bu yazılımın *java* programlama dili kullanılarak yazılması planlanmaktadır. *PostgreSQL* veri tabanı yönetim sistemi tercih edilmiştir.
- **Kullanılacak Harici Kütüphaneler:** JDBC (Java Database Connectivity), veritabanı işlemleri için kullanılır. JDBC, Java uygulamalarının çeşitli veritabanlarıyla etkileşimde bulunmasını sağlar.
- **Java'da Tarih ve Zaman Kütüphaneleri:**
  - `java.time.LocalDate`: Tarih işlemleri için kullanılır, özellikle sefer tarihlerinin yönetimi gibi işlemlerde.
  - `java.time.format.DateTimeFormatter`: Tarih ve zaman verilerinin formatlanması için kullanılır.
  - `java.time.ZoneId`: Zaman dilimi işlemleri için kullanılır, tarih ve zaman verilerinin bölgesel ayarlarla uyumlu hale getirilmesinde yardımcı olur.
- **Veri Yapıları ve Genel Programlama Kütüphaneleri:**
  - `java.math.BigDecimal`: Para birimi hesaplamaları ve diğer hassas değer işlemleri için kullanılır. Özellikle gemi ağırlığı ve liman demirleme ücretleri gibi finansal verilerin doğru bir şekilde işlenmesinde önemlidir.

Ödev No:2	Tarih 05.05.2024	6/14
-----------	------------------	------

- java.util.List: Java'nın koleksiyon kütüphanelerinden biridir ve nesneleri listeler halinde tutmak için kullanılır. Bu, mürettebat, kaptan listesi veya seferde uğranılan limanların listesini yönetmede kullanılır.
- java.util.ArrayList: List arayüzünü implemente eden bir sınıftır ve dinamik dizi işlevselliği sunar. Bu, sistemde esnek veri yapıları oluşturulmasını sağlar.
- java.util.Scanner: Kullanıcı girdilerini komut satırı üzerinden okumak için kullanılır. Kullanıcıdan alınan verilerin sisteme girilmesinde temel bir araçtır.

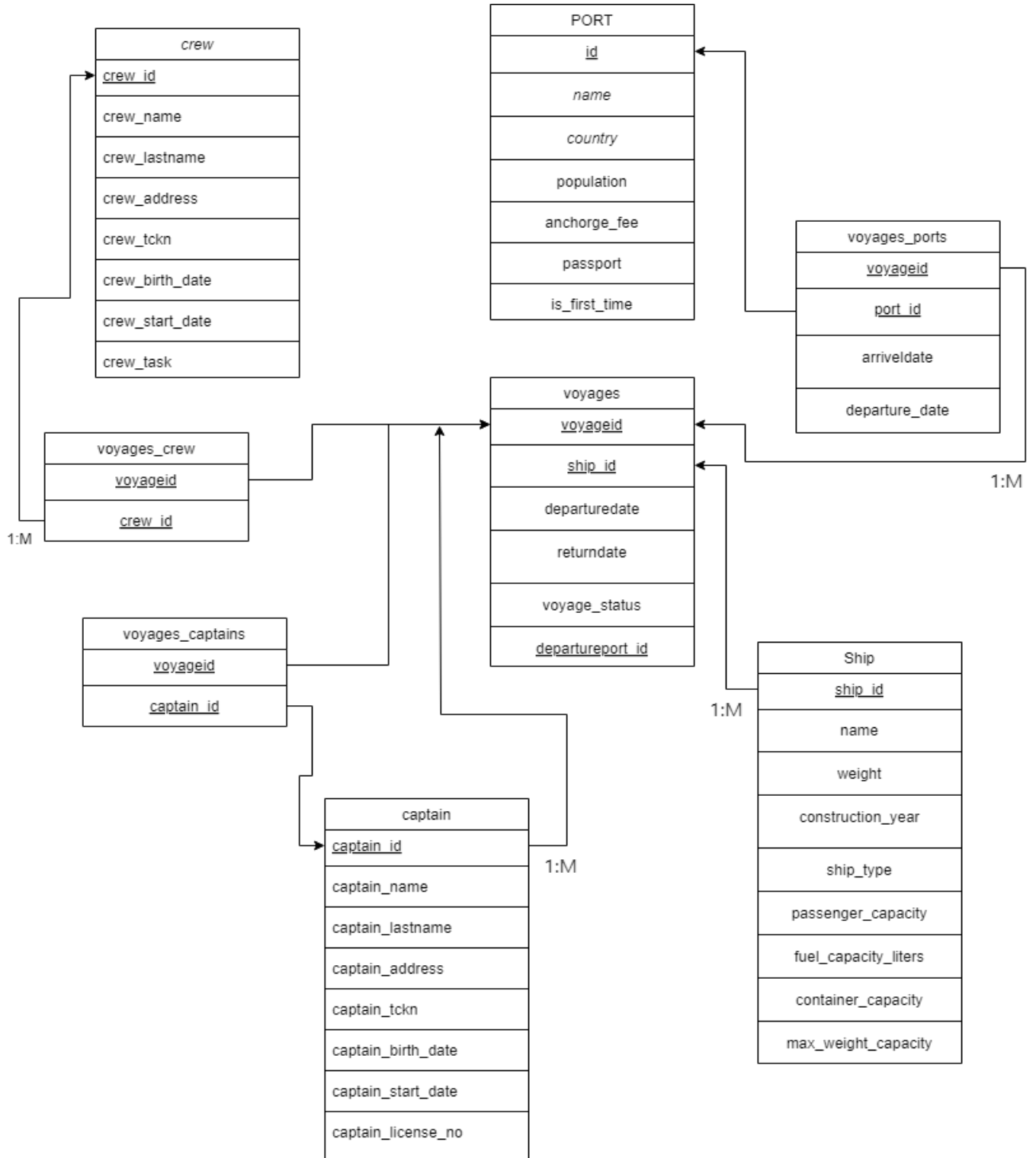
```
• import java.math.BigDecimal;  
import java.sql.*;  
import java.time.LocalDate;  
import java.time.format.DateTimeFormatter;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
import java.util.Scanner;  
import java.time.ZoneId;
```

- **Diğer Teknolojilerle İlgili Açıklama:** *PostgreSQL* veritabanı yönetim sistemi. *PostgreSQL*, açık kaynaklı, güçlü ve ölçeklenebilir bir veritabanı yönetim sistemidir.

### Kullanıcı arayüzü tasarımı:

- **Arayüz Gereksinimleri:** Kullanıcı arayüzü, komut satırı üzerinden etkileşimli olarak tasarlanmıştır. Kullanıcılar, çeşitli menüler aracılığıyla işlemlerini seçip, gerekli bilgileri girerek sistemle etkileşimde bulunurlar.
- **Uygulamanın nasıl çalıştırılacağı ile ilgili açıklama:** Uygulamayı çalıştırmak için Java'nın yüklü olduğu bir bilgisayar gereklidir. Terminal veya komut istemcisinde ilgili Java dosyası çalıştırıldıktan sonra, kullanıcılar gerekli işlemleri komut satırı üzerinden yapabilirler.

Ödev No:2	Tarih 05.05.2024	7/14
-----------	------------------	------







## Veri tabanındaki gemileri listeliyor;

```
2. Seferler
3. Kaptanlar
4. Mürettebat
5. Limanlar
0. Çıkış
Seçiminizi yapın: 1
----- Gemiler Menüsü -----
1. Gemileri Listele
2. Gemi Ekle
3. Gemi Sil
4. Gemi Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: 1
Veritabanına başarıyla bağlandı!
Gemi ID: 101001, Gemi Adı: 1997 KALKAN, Gemi Ağırlığı: 5500, Yapım Yılı: 2021, Gemi Tipi: Yolcu, Y.Gemi Kapasite: 500, Petrol Kapasite:null, Konteyner kapasite: 0, Max
konteyner Ağırlığı: null
Gemi ID: 101002, Gemi Adı: DERVA KAPTAN, Gemi Ağırlığı: 5000, Yapım Yılı: 2020, Gemi Tipi: Yolcu, Y.Gemi Kapasite: 200, Petrol Kapasite:null, Konteyner kapasite: 0, Max
konteyner Ağırlığı: null
Gemi ID: 101003, Gemi Adı: Kaptan 02, Gemi Ağırlığı: 300, Yapım Yılı: 2003, Gemi Tipi: Yolcu, Y.Gemi Kapasite: 600, Petrol Kapasite:null, Konteyner kapasite: 0, Max
konteyner Ağırlığı: null
Gemiler listelendi.
----- Gemiler Menüsü -----
1. Gemileri Listele
2. Gemi Ekle
3. Gemi Sil
4. Gemi Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: |
```

```
--Gezgin Gemi Şirketi--
----- Ana Menü -----
1. Gemiler
2. Seferler
3. Kaptanlar
4. Mürettebat
5. Limanlar
0. Çıkış
Seçiminizi yapın: 2
----- SEFERLER Menüsü -----
1. Seferleri Listele
2. Sefer Ekle
3. Sefer Sil
4. Sefer Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: 4
----Sefer Düzenleme----

Düzenlenecek Sefer ID'sini girin:
```

Ödev No:2	Tarih 05.05.2024	10/14
-----------	------------------	-------

## Veri tabanındaki Limanları listeliyor ve siliyor;

```
Limani No: 2 Liman Adı: İzmir Limanı, Liman Ülkesi: Türkiye
Stop 'Main' Ctrl+F2
Liman ID: 4, Liman Adı: Port of Singapore, Liman Ülkesi: Singapur
Limanlar listelendi.
----- Liman Menüsü -----
1. Limanları Listele
2. Liman Ekle
3. Liman Sil
4. Liman Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: 3
Veritabanına başarıyla bağlandı!
! ! İlk Önce Listeyi Kontrol Edip ID Bilgisini Alın ! !
Silincek Liman ID girin:
4
Liman başarıyla silindi.
----- Liman Menüsü -----
1. Limanları Listele
2. Liman Ekle
3. Liman Sil
4. Liman Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: 1
Veritabanına başarıyla bağlandı!
Liman ID: 1, Liman Adı: Körfez, Liman Ülkesi: Türkiye
Liman ID: 2, Liman Adı: İzmir Limanı, Liman Ülkesi: Türkiye
Liman ID: 3, Liman Adı: Port of Singapore, Liman Ülkesi: Singapur
Limanlar listelendi.
----- Liman Menüsü -----
```

## 4.3 Karşılaşılan zorluklar ve çözüm yöntemleri

```
String url = "jdbc:postgresql://localhost:5432/odv2";
String user = "postgres";
String password = "1234";

try {
    Connection connection = DriverManager.getConnection(url, user,
password);
    System.out.println("Veritabanına başarıyla bağlandı!");

    Statement statement = connection.createStatement();
    ResultSet resultSet = statement.executeQuery("SELECT * FROM
\"PORT\"");

    while (resultSet.next()) {
        System.out.println("Name: " + resultSet.getString("name") + ",
Country: "
+ resultSet.getString("country"));
    }
}
```

Port tablosunun iki sütunundaki verileri görüntüleme için test ettiğimde tablonun adıyla alakalı hata aldım. Tablolarda isimlerin büyük küçük harf duyarlılığı fazla olduğu için tablo isminin başına ve sonuna \" ekledim.

Ödev No:2	Tarih 05.05.2024	11/14
-----------	------------------	-------

#### 4.4 Proje isterlerine göre eksik yönler

Uygulama tipi bir arayüz yapmaya zamanım kalmadığı için eksikliğim bu oldu. Genel olarak istenilen birçok şeyi yapabilmek için elimden geleni yaptım. Veri tabanı kullanımı sınıf metotlarından daha kolay olsa da sınıf nesnelerini de kullanmak için uygun yolları denedim.

### 5. TEST VE DOĞRULAMA

#### GEMİ VERİLERİNİ VERİ TABANINA EKLEME:

```
Main.java Voyage.java Ship.java PassengerShip.java ContainerShip.java PetrolTankerShip.java
3 public class Main {
4     public static void main(String[] args) {
5         // Örnek bir yolcu gemisi
6         PassengerShip passengerShip = new PassengerShip();
7         passengerShip.setShipId(10101);
8         passengerShip.setName("1997 KALKAN");
9         passengerShip.setWeight(5500); // Örnek ağırlık değeri
10        passengerShip.setConstructionYear(2021);
11        passengerShip.setPassengerCapacity(500); // Yolcu kapasitesi
12
13        // Veritabanına bağlantı bilgileri
14        String url = "jdbc:postgresql://localhost:5432/odv2";
15        String user = "postgres";
16        String password = "1234";
17
18        try {
19            Connection connection = DriverManager.getConnection(url, user, password);
20            System.out.println("Veritabanına başarıyla bağlandı!");
21
22            // PreparedStatement kullanarak gemi bilgisi ekleme;
23            String sql = "INSERT INTO Ship (ship_id, name, weight, construction_year, ship_type, passenger_capacity) " +
24                "VALUES (?, ?, ?, ?, ?, ?)";
25            PreparedStatement preparedStatement = connection.prepareStatement(sql);
26            preparedStatement.setInt(1, passengerShip.getShipId());
27            preparedStatement.setString(2, passengerShip.getName());
28            preparedStatement.setDouble(3, passengerShip.getWeight());
29            preparedStatement.setInt(4, passengerShip.getConstructionYear());
30            preparedStatement.setString(5, "Yolcu"); // Ship_type değeri
31            preparedStatement.setInt(6, passengerShip.getPassengerCapacity());
```

1.

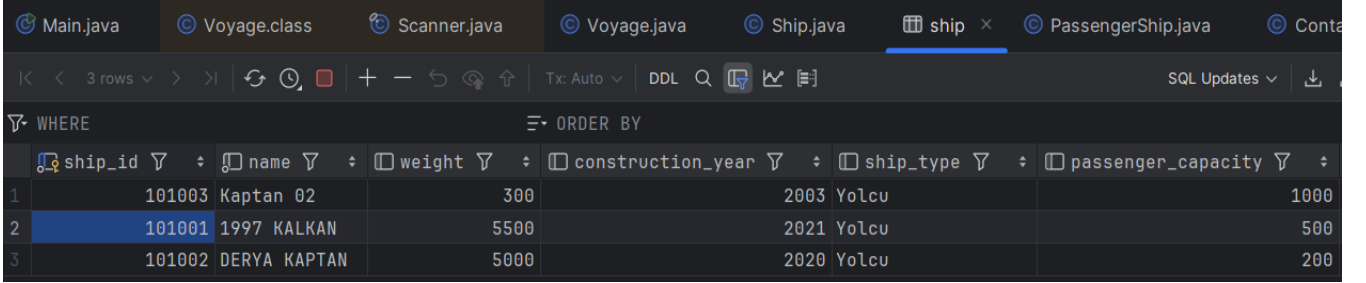
ship_id	name	weight	construction_year	ship_type	passenger_capacity	fuel_capacity_liters	co
10102	Firat	5000	2020	Yolcu	200	<null>	
10101	1997 KALKAN	5500	2021	Yolcu	500	<null>	

Ödev No:2

Tarih 05.05.2024

12/14

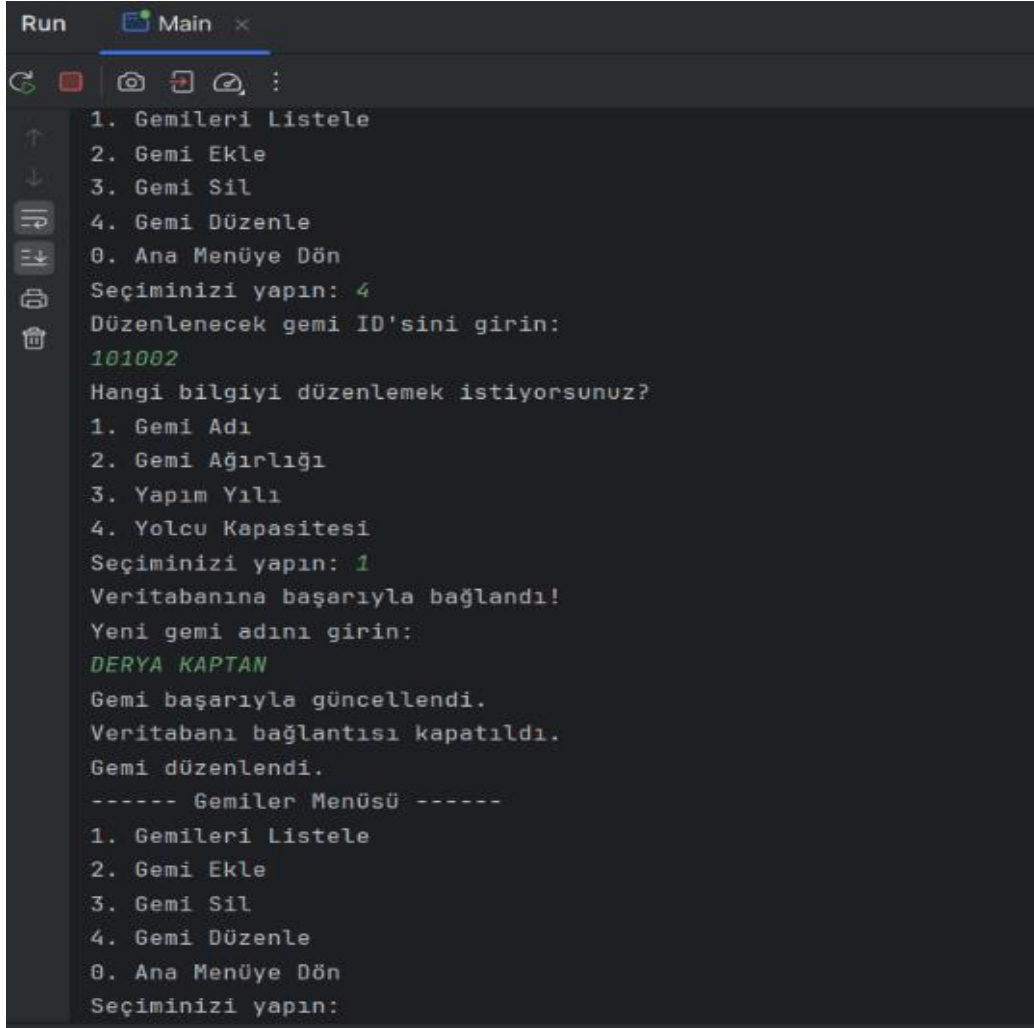
2.



	ship_id	name	weight	construction_year	ship_type	passenger_capacity
1	101003	Kaptan 02	300	2003	Yolcu	1000
2	101001	1997 KALKAN	5500	2021	Yolcu	500
3	101002	DERYA KAPTAN	5000	2020	Yolcu	200

ÖR: (1. ve 2. Görsel)

Burada gemi adı Fırat isminden DERYA KAPTAN ismine çeviriliyor;



```
Run Main x
1. Gemileri Listele
2. Gemi Ekle
3. Gemi Sil
4. Gemi Düzenle
0. Ana Menüye Dön
Seçiminizi yapın: 4
Düzenlenecek gemi ID'sini girin:
101002
Hangi bilgiyi düzenlemek istiyorsunuz?
1. Gemi Adı
2. Gemi Ağırlığı
3. Yapım Yılı
4. Yolcu Kapasitesi
Seçiminizi yapın: 1
Veritabanına başarıyla bağlandı!
Yeni gemi adını girin:
DERYA KAPTAN
Gemi başarıyla güncellendi.
Veritabanı bağlantısı kapatıldı.
Gemi düzenlendi.
----- Gemiler Menüsü -----
1. Gemileri Listele
2. Gemi Ekle
3. Gemi Sil
4. Gemi Düzenle
0. Ana Menüye Dön
Seçiminizi yapın:
```

---

## 6. KAYNAKÇA

- ❖ <https://chat.openai.com>
- ❖ <https://www.canva.com/design/DAGAcdmB1v4/NUeL-2TUUU3wfhq9rygEpA/edit>
- ❖ [https://tubitak-bilgem-yte.github.io/pg-gelistirici/docs/01-veri-operator-tipleri/veri\\_tipleri/](https://tubitak-bilgem-yte.github.io/pg-gelistirici/docs/01-veri-operator-tipleri/veri_tipleri/)
- ❖ <https://www.yusufsezer.com.tr/java-postgresql/>
- ❖ <https://www.youtube.com/watch?v=dkn9NOChFdQ>
- ❖ <https://www.youtube.com/watch?v=dkn9NOChFdQ>
- ❖ <https://stackoverflow.com/questions/58057490/how-to-trigger-java-process-when-a-postgresql-table-has-been-updated-in-the-data>
- ❖ [https://www.cybertec-postgresql.com/en/postgresql-how-to-write-a-trigger/?gad\\_source=1&gclid=CjwKCAjw3NyxBhBmEiwAyofDYauOGeYtqSFnsU75ovRPoVHyUujL9pcuJEMr-PpEewMnA019WHP1VhoCShYQAvD\\_BwE](https://www.cybertec-postgresql.com/en/postgresql-how-to-write-a-trigger/?gad_source=1&gclid=CjwKCAjw3NyxBhBmEiwAyofDYauOGeYtqSFnsU75ovRPoVHyUujL9pcuJEMr-PpEewMnA019WHP1VhoCShYQAvD_BwE)
- ❖ <https://www.postgresql.org/docs/current/sql-createtrigger.html>
- ❖ <https://onurkul.com.tr/sql-tablo-iliskilendirme>
- ❖ <https://www.edrawsoft.com/article/use-case-diagram-examples.html>
- ❖ <https://app.diagrams.net/>

Ödev No:2	Tarih 05.05.2024	14/14
-----------	------------------	-------