
T.C
KOCAELİ SAĞLIK VE TEKNOLOJİ
ÜNİVERSİTESİ DOĞA BİLİMLERİ VE
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ
PROGRAMI

# VERİ TABANI YÖNETİM SİSTEMLERİ PROJESİ

HAZIRLAYANLAR

ANIL ERDOĞAN – 220501006
GitHub Link: https://github.com/Kwalmm

AKIN TURAN – 220501013
GitHub Link: https://github.com/coldwraith44

DERS SORUMLUSU
Dr. Öğr. Üyesi Ercan ÖLÇER

İÇİNDEKİLER



## 1. ÖZET

Verilen istek doğrultusunda kullanıcının şirketine ait gemiler ile gerçekleştiren seferlerin bir veri tabanında tutulmasına dair bir projedir. Gemiler seferlere çıkmaktadır ve gemilerin içinde kaptanlar ve mürettebatlar bulunmaktadır. Gemiler limana giderek seferlerini bitirirler.

## 2. GİRİŞ

İhtiyaç doğrultusunda bunların python yazılım dilinde nasıl tasarlandığına göz atacağız.

## 3. YÖNTEM

Bir veri tabanı sistemi oluşturduk ve bu sistemi bir tasarım-menü kütüphanesi olan “tkinter” ile entegre ettik. Sınıfları oluşturduk;

## 3.1 Sınıfların tasarlanması aşaması

### 3.1.1 Gemi sınıfı
Gemilerin bir seri numarası, adı, ağırlığı ve üretim yılı bulunmaktadır. Bu sınıfın 3 alt sınıfı da bulunmaktadır. Bunlar: yolcu gemisi, petrol tankerleri ve konteyner gemileridir. Sırasıyla bu gemiler için ek olarak: yolcu kapasitesi, litre cinsinden petrol kapasitesi ve konteyner sayısı kapasitesi verileri bulunmaktadır.

### 3.1.2 Seferler sınıfı
Sefer sınıfında sefere ait özel bir ID, çıkan geminin ID'si, gemide ki kaptanlar, gemide ki mürettebatlar, geminin yola çıkış tarihi, geminin dönüş tarihi ve çıkılan liman verileri yer almaktadır.

### 3.1.3 Limanlar sınıfı
Limanlar sınıfında sefere çıkan geminin varacağı limanın adı, limanın bulunduğu ülke, limanda ki bekleyen gemiler, limanın nüfusu, pasaport kontrol durumu ve demirleme ücreti verileri yer almaktadır

### 3.1.4 Kaptanlar sınıfı
Kaptanlar sınıfında kaptanın no'su, kaptanın adı, kaptanın soyadı, kaptanın adresi, kaptanın vatandaşlık durumu, kaptanın doğum tarihi, kaptanın işe giriş tarihi, kaptanın lisansı ve kaptana ait olan lisansın numarası verileri yer almaktadır.

### 3.1.5 Mürettebat sınıfı
Mürettebat sınıfında mürettebatın no'su, mürettebatın adı, mürettebatın soyadı, mürettebatın adresi, mürettebatın vatandaşlık durumu, mürettebatın doğum tarihi, mürettebatın işe giriş tarihi ve mürettebatın görevi verileri yer almaktadır.

## 3.2 Veri tabanı yönetiminin oluşum aşaması

### 3.2.1 Veri tabanının temelini kurma aşaması

Öncelikle pyodbc kütüphanesini kodun içine aktarıldı. Bu kütüphane açık veri tabanı bağlantısı için gereklidir. Ardından “pyodbc.connect()” fonksiyonu ile driver, server, database ve trusted_connection değerleri atandı. Bu aşamada SQL veri tabanı resmen oluşmuş bukunmaktadır. Bu kod bloğunun ardından bağlantı üstünde işlem yapmak için bir “cursor” oluşturuldu.

### 3.2.2 Her tablonun veri tabanında oluşturulması aşaması

Tabloların nasıl oluşturulduğuna dair koddan bir örneğe göz atabiliriz. Mesela gemi tablosunun oluşum aşamasını inceleyelim,

-> Geminin adını string olarak bir değişkene atadık.

-> Ardından INFORMATION_SCHEMA.TABLES adlı bir veri tabanı tablosunda gemi_tablo stringinin var olup olmadığını kontrol eder. Eğer varsa; Count(*) fonksiyonu 1, yoksa 0 değerini döndürecektir.

-> cursor.execute() fonksiyonu oluşturulan SQL sorgusunu çalıştırmak için bulunmaktadır.

-> cursor.fetchone() fonksiyonu ise oluşturulan sorgu sonucunu ele alıp bir değişkene atar.

-> Atanan değere sahip olan değişken bir if koşuluna kontrole girer. Bu kontrol sonucu bir True ya da False değeri dönecektir. True değeri dönerse, Count(*) fonksiyonu “1” çıktısını vermiştir ve bunun anlamı tablonun zaten var olmasıdır. Bu koşul kodda bir etki yaratmayacaktır ve pass komutu ile atlanacaktır

-> Fakat ilgili tablo yoksa gemi_tablo tablosunun oluşumu başlayacaktır. Bu esnada cursor.execute() fonksiyonu bu tablo için sırasıyla:

-> Tablonun adı oluşturulur.

-> “seri_no” adlı bir sütun oluşur. Veri tipi VARCHAR’dır. Maximum 255 karakter olabilir. Bu sütundaki her eleman bağımsız olmalıdır. bunun sebebi PRIMARY KEY ile bu sütunun benzersiz hale getirilmesidir.

-> “adi” adlı bir sütun oluşturulur. Veri tipi VARCHAR’dır. Maximum 250 karakter olabilir. NOT NULL ibaresi bu sütunun hiç bir şekilde boş bırakılamayacağını belirtir. Bu her geminin bir adı olması gerektiğini belirtir.

-> “agirlik” adlı bir sütun oluşturulur. Veri tipi DECIMAL’dır. DECIMAL(10, 2) ibaresinde yer alan 10 bu sayının maximum 10 basamak -lı olabileceğini belirtir. 2 sayısı ise bu basamakların 2’sinin ondalıklı olabileceğini söyler. NOT NULL ibaresi bu sütunun hiç bir şekilde boş bırakılamayacağını belirtir. Her gemi mutlak suretle bir ağırlığa sahip olmalıdır.

-> “yapim_yili” adlı bir sütun oluşturulur. Veri tipi INT’dir. Yani bu sütundaki satırlar bir tam sayı olmalıdır. NOT NULL ibaresi bu satırların boş bırakılamayacağını belirtir.

Tabloların hepsi bu şekilde bu formatta oluşurlar. Bazı tablolar bünyesine diğer tablolardan öge barındıracağından yabancı anahtarlar da kullanırlar. Son olarak göz atılacak ise:

-> db.commit() ibaresi değişiklikleri kalıcı hale getirir. “commit” metodu çağırılmaz ise değişiklikler geçici olacaktır.

-> cursor.close() ibaresi veri tabanı üzerinde işlem yapmak için kullanılan “cursor” imlecini kapatır. İmleç kapatıldığında artık bu kaynak kullanılamaz.

-> db.close() ibaresi ile veri tabanı bağlantısı da kapatılmış olur. Veri tabanı bağlantısı sunucuyla iletişim kurmaya yarar.

## 3.3 Menünün tasarım aşaması

Veri tabanını oluşturduktan sonra geriye artık menü ile entegrasyonu kalıyor kullanıcının veri tabanında yapacağı işlemleri somutlaştırmak için tkinter kütüphanesi ile bir menü tasarlandı. Sırasıyla yapılan adımlar:

-> tkinter kütüphanesi kodun içine aktarıldı.

-> 5 Adet buton tasarlandı. Bu butonlar genel olarak class'larda yer alan ögeler ile ilgili işlem yapmamıza yardımcı olacak butonlardır. Bunlar: Veri ekleme, veri düzenleme, veri silme ve veri görüntülemedir.

-> Verilerimiz genel olarak şöyle sıralanabilir:

-> Gemiler
  -> Konteyner gemisi
  -> Petrol gemisi
  -> Yolcu gemisi

-> Seferler

-> Limanlar

-> Kaptanlar

-> Mürettebatlar

-> Veriler üzerinde işlem yapan fonksiyonlar önce classları oluşturulan elemanlar hakkında yapılabilecek görevleri sunar. Daha sonrasında bu ögeler ile işlemler yapılır. Bunlar: Veri ekleme, veri düzenleme, veri silme ve veri görüntülemedir. Gemiler için 3 farklı gemi türü daha bulunduğundan, gemiler de kendi içinde gruba ayrılır ve onlar için 3 ayrı buton daha aktif hale gelecektir.

-> Butonları ekleme işlemi: Butonlar eklenmeden önce kaldir_mevcut_butonlar() fonksiyonu ekrandaki mevcut butonları kaldırır yeni eklenecek butonlara zemin ayırır. kaldir_mevcut_butonlar_2() fonksiyonu ise gemi içerisinde ki ara gemiler ile ilgili yapılacak işlemler esnasında kullanılır. Aynı görevi farklı zamanda gerçekleştirir.

-> Girdi alma işlemi: entry_alma_modulu(x) fonksiyonu içindeki x tıklanılan butona göre string atanan bir değişkendir. Atanan string bu fonksiyon içerisinde uygun olan kod bloğu ile eşleşir ve doğru fonksiyon çalıştırılır.

-> Veri ekleme işlemi: Uygun buton veri_ekle() fonksiyonu ile seçilir ve ardından butonun verdiği çıktı entry_alma_modulu(x) fonksiyonuna aktarılır hangi butona tıklandığı kontrol edilir ve tk.Toplevel(root) yardımıyla seçilen ögenin adıyla bir ekleme başlığı oluşturulur. Eklenen veri bir yolcu gemisi, petrol tankeri, konteyner gemisi, liman, sefer, mürettebat ya da kaptan olabilir. Bu veri fonksiyonların ilişki kurması sonucu bulunur ve başlıklar ya da info kutularının string çıktıları ona göre dağıtılmış olur.

-> Uygun başlık oluşturuldu.

-> İlgili veriye ait sütunların adlarına dair veri alımı yapılır. Mesela geminin seri numarası gibi. Sütun NOT NULL ile başta SQL dilinde işaretlenmişse ozaman o sütunun satırları boş bırakılamaz.

-> Her sütun için sırayla tk.label bize mesaj yoluyla sütunun satırlarına girdi istenir. Pady ilgili widgetin kaç birim yukarı ya da aşağı olacağını bize burada söyler.

-> tk.entry ile kullanıcıdan bir girdi alınır aynı şekilde bu girdi için de pady kullanılabilir.

-> İşlemler bittikten sonra elde edilen veriler kullanılmaya hazır hale gelir. En son bu veriler entry_alma_moduku(x) tarafında return ile döndürülür ve veri ekleyici fonksiyona iletilir. Bitiris() fonksiyonu ile tekrardan uygulama yeniden açılmış hale getirilir

-> Veri düzenleme işlemi:

-> tk.Toplevel ile ilgil işlemin başlığı belirtilir.

-> Düzenleme yapılacak olan veriyi tanımlayan bir PRIMARY KEY yardımıyla veri bulunur.

-> tk.entry ile kullanıcıdan düzenlenecek verinin yerine konulacak veri alınır. Düzenlenecek olan verinin varlığı da aranır. Hata oluşumu engellenir.

-> İlgili çıktı tüm aşamalar bitince gerekli fonksiyona iletilir.

-> Veri silme işlemi:

-> Silinme işlemini belirten başlık oluşturulur.

-> tk.label ile soru başlığı oluşturulur.

-> tk.entry ile silinmesi istenen veriye ait PRIMARY KEY istenir.

-> Silinecek olan verinin varlığı kontrol edilir. Bütün işlemler başarılıysa işlemde gerçekleştirilir.

-> Veri görüntüleme işlemi:

-> Uygun başlık tk.Toplevel(root) ve giris_penceresi.title ile oluşturulur.

-> Görüntülenecek olan veriye ait bir PRIMARY KEY girdisi istenir.

-> Veri bulunur. Burada bitiris() fonksiyonu çağırılmaz çünkü amaç veri görüntüleme yapmaktır.

### 3.3.1 Menünün temeli ve butonların varlığı aşaması

Menünün temelini oluşturan bazı fonksiyonlar kullanıldı bunlar tkinterin temel fonksiyonlarıdır.

-> tk.Tk() Tkinterde pencere oluşturmamızı sağlar.

-> root.title() ile uygulamanın adı oluşur.

-> root.geometry() ile pencerenin x, y boyutları ayarlanır.

-> intro.png dosyası, tk.PhotoImage ile intro_image değişkenine atanır.

-> tk.Label yardımıyla intro_image adlı görüntüyle bir etiket oluşturulur.

-> Bu etiket intro_label.place ile yerleştirilir. Reldwith=1 ve relheight=1 olması bu görüntünün ekranı kaplayacağını ifade eder.

-> root.after(3000, intro_bitir) fonksiyonu, 3000 milisalisede bir intro_bitir fonksiyonunu çalıştırır. intro_bitir fonksiyonuda ilgili resmi tekrarlatır

-> Başka bir arka plan resmide arka plana dahil edilir.

-> Bu resim background_label değişkenine tk.Label() yardımıyla atanır.

-> background_label.place() ile resim ekranı kaplayacak biçimde yerleştirilir.

-> Son olarak root.mainloop() Tkinter döngüsünün ana döngüsünü başlatır. Penceredeki tüm olayları işler.

## 4. KAYNAKÇA


**>> Chat GPT AI**
**>> Bing.com**
**>> Bing AI**
**>> Google chrome**
**>> www.programiz.com**