COMP 205- NESNEYE YÖNELİMLİ PROGRAMLAMA

2020-2021 GÜZ DÖNEMİ

NÜFUS YÖNETİM SİSTEMİ

PROJE TESLİM RAPORU

15 OCAK 2021

Damla Su KARADOĞAN*[1], Tuğberk Onur KURU[1], Kenan GÖZÜAÇIK[1], Tuana AYDOĞAN[1]

Fenerbahçe Üniversitesi, Endüstri Mühendisliği, İstanbul/Türkiye

E-mail: damla.karadogan@stu.fbu.edu.tr

E-mail: tugberk.kuru@stu.fbu.edu.tr

E-mail: kenan.gozuacik@stu.fbu.edu.tr

E-mail: tuana.aydogan@stu.fbu.edu.tr

ÖZET

Bir nüfus yönetim sistemi uygulaması gerçekleştirilecektir. Nüfus yönetim uygulamasının tüm verileri dosyada tutulacak ve güncellemeler dosyaya sürekli senkronize edilecektir.

Anahtar Kelime: Nüfus, Veri tabanı, Kayıt

ABSTRACT

A population management system will be implemented. All news of the Population application will be kept on file and updates will be noted in the file.

Key Words: Population, Database, Registration

I. GİRİŞ

Geliştirilen bu program kullanıcıya komut satırı arayüzü sunarak kullanıcıdan isim, soyisim, kütük ve kan grubu gibi kimlik bilgilerini alıp bunların bir veri tabanında saklanması, kullanıcıdan yeni alınan bilgilerle güncellenebilmesi ve silinmesi işlevlerini sunan geniş çaplı bir nüfus yönetim sistemi programıdır.

II. SİSTEM MİMARİSİ

- Programımızı veri tabanı üzerinde yapacağımız için, geliştirmeye ***sqlite3*** modülünü tanımlayarak başladık.

```
import sqlite3
```

- Sonrasında yeni bir veri tabanı oluşturmak veya mevcut bir veri tabanına bağlanmak için ***connect()*** fonksiyonunu kullandık. Bağlanacağımız veri tabanı mevcut değilse otomatik olarak oluşturulacaktır.
  Yeni oluşturduğumuz veya mevcut olan bir veri tabanına bağlandıktan sonra üzerinde işlem yapabilmek için ***cursor()*** fonksiyonu ile imleç oluşturmamız gerekir. SQL komutlarını oluşturabilmek için imlecimizle beraber ***execute()*** fonksiyonunu kullanmalıyız. Böylelikle tüm SQL komutlarını veri tabanımız üzerinde çalıştırabiliriz. Ayrıca eğer kişiler isimli bir tablomuz yok ise hata mesajı almamak için ***CREATE TABLE kisiler*** komutuna ***IF NOT EXISTS*** ifadesi eklenir ve eğer bu isimli bir tablo yoksa veri tabında oluşturulması sağlanır. Bu şekilde veri tabanımızda kişiler isimli ve 12 sütunlu (kimlik numarası, adı, soyadı…) bir tablo oluşturmuş oluruz.

```
db = sqlite3.connect('kisiler.db')
imlec = db.cursor()
imlec.execute("CREATE TABLE IF NOT EXISTS kisiler(KimlikNo, Adı, Soyadı, Baba_adı, Anne_adı, Doğum_yeri, Medeni_durumu, Kan_grubu,"
              "Kütük_Şehir, Kütük_İlçe, İkametgah_Şehir, İkametgahİlçe)")
```

```
imlec.execute("""SELECT DISTINCT * FROM kisiler""")
veri = imlec.fetchall()
elemansay=0
for i in veri:
    elemansay=elemansay+1
csvdenalma()
kisisayisi=db.total_changes
if elemansay==0 :
    print("Veri tabanında şu an eleman yoktur.")
    if kisisayisi!=0:
        print(f"Veri tabanında, csv dosyasından da içe aktarılan kisilerle birlike {kisisayisi} eleman vardır.")
        dbListele()
else:
    dbListele()
    if kisisayisi==0:
        print(f"Veri tabanında şu an {elemansay} eleman vardır.")
    else:
        print(f"Veri tabanında, csv dosyasından da içe aktarılan kisilerle birlike {kisisayisi} eleman vardır.")
```

- Tablomuzu ve veri tabanımızda oluşturduktan sonra; uygulama çalıştırıldığında ekrana veri tabanının içinde kaç tane eleman olduğunu kullanıcıya gösterebilmek amacıyla, veri tabanı üzerinde işlem yapabilmek için ***cursor()*** fonksiyonunu atadığımız imleç ile beraber SQL komutlarını gerçekleştirebilmek için ***execute()*** fonksiyonunu çağırıp parantez içerisine ***SELECT DISTINCT * FROM kişiler*** yazarak kisiler isimli tablonun içerisindeki tüm verileri ***SELECT*** ile seçtik. ***DISTINCT*** ifadesi, tablonun belli kolonlarında tekrar eden verilerin bir adet olacak şekilde çekilmesine olanak sağlar. Veri tabanımızdaki elemanları seçtikten sonra kayıtlı olan elemanları ***fetchall()*** fonksiyonu ile tüm verilerimizi ***veri*** isimli değişkenimizin içine attık. Bir döngü kurarak veri tabanımızdaki tüm veriler için ***elemansay*** isimli değişkenimizin değerini 1 arttırdık.
  Veri tabanımızın içindeki elemanları okuduktan sonra ***csvdenalma()*** fonksiyonunu çağrılır. ***try...except...*** bloğu kullanılarak '***kisiler.csv***' isimli bir dosya yok ise programın hata vermesin engeller ve ekrana dosyanın oluşturulduğu bilgisini yazdırır. Dosya var ise içeriği okuduk ve elemanları ***kisiListe*** isimli bir listenin içine ekledik. Sonrasında ise dosyadan okuyup listeye eklediğimiz kişileri bir for döngüsünün içerisinde ***INSERT INTO kisiler VALUES*** komutuyla birlikte dosyanın içerisindeki elemanları ***kisiListe*** yardımıyla veri tabanımızın içine eklemiş olduk.

```
def csvdenalma():
    kisiListe = []
    try:
        dosya = open("kisiler.csv", "r")
        for prsn in dosya:
            kisiListe.append(prsn.strip().split(";"))
        dosya.close()
    except:
        print("Dosya oluşturuldu")

    for i in kisiListe:
        imlec.execute("""INSERT INTO kisiler VALUES ('{}', '{}', '{}', '{}', '{}', '{}', '{}', '{}',
        '{}', '{}', '{}', '{}')""".format(i[0], i[1], i[2], i[3], i[4], i[5], i[6], i[7], i[8], i[9], i[10], i[11]))
```

Dosya içerisindeki elemanları da veri tabanına ekledikten sonra ***kisisayisi*** değişkenin içine veri tabanımızın içerisindeki bütün elemanların(değişikliğin) sayısını atadık. Eğer veri tabanımızın içinde hiç eleman yoksa veri tabanının boş olduğu bilgisi yazdırılacaktır. Ama csv dosyasının içerisini okuduktan sonra eğer kişi sayımız sıfırdan farklıysa artık dosyanın içerisindeki veriler de veri tabanına geçmiş olacağı için güncel veri tabanında bulunan eleman sayısı ekrana bastırılacaktır ve veri tabanımızda bulunan bütün kişiler ***dbListele()*** fonksiyonuyla ekrana yazdırılacaktır. Bu fonksiyonda kişiler isime tablomuzda bulunan sütunların ismi ekranda daha güzel durması için her iki sütun başlığının arasına ***ljust(10)*** yazılarak içerideki sayı kadar boşluk konulur. Sonrasında da yine bütün veri tabanımızı

SELECT ile seçerek bir döngü içerisinde her eleman bilgileriyle birlikte ekrana yazdırılmış olur.

```
def dbListele():
    print('KİMLİK NO'.ljust(10), '  ADI'.ljust(10), 'SOYADI'.ljust(10), 'BABA ADI'.ljust(10), 'ANNE ADI'.ljust(11),
          'DOGUM YERI'.ljust(11), 'MEDENI DURUM'.ljust(15), 'KAN GRUBU'.ljust(13), 'KUTUK SEHRI'.ljust(15),
          'KUTUK ILCE'.ljust(15),'IKAMETGAH SEHRİ'.ljust(20),'IKAMETGAH ILCE'.ljust(10))
    print('--------- ---------  ---------  --------- -----------  -----------  ------------   ---------- '
          '    -----------     -----------     ---------------     ---------------')
    imlec.execute("""SELECT DISTINCT * FROM kisiler""")
    for veri in imlec:
        print(veri[0].ljust(10),veri[1].ljust(10),veri[2].ljust(10),veri[3].ljust(10),veri[4].ljust(11),
              veri[5].ljust(11),veri[6].ljust(15),veri[7].ljust(13),veri[8].ljust(15),veri[9].ljust(15),
              veri[10].ljust(20),veri[11].ljust(10))
```

Ama eğer ki programımız ilk başladığında veri tabanımız boş değilse o zaman, öncelikle ***dbListele()*** ile veri tabanımız yazdırılır ve eğer okuduğumuz dosyanın içerisinde bir veri yoksa sadece veri tabanımızın içerisinde kaç adet eleman olduğu; dosyanın içerisinde de eleman var ise içe aktarılanla beraber kaç adet eleman olduğu sayısı ekrana yazdırılmış olur.

```
while True:
    print("Lütfen yapmak istediğiniz işlemi seçiniz \n"
          "0. Çıkış \n"
          "1. Kişi Eklme\n"
          "2. Kişi Listeleme\n"
          "3. Kişi Arama\Sorgulama\n"
          "4. Kişi Güncelleme\n"
          "5. Kişi Silme\n")
    islev = input("Seçiminiz: ")
    if islev not in ["0", "1", "2", "3", "4", "5"]:
        print("Hatalı giriş !!!")
    elif islev == "0":
        print("Çıkış seçildi")
        db.close()
        exit(0)
```

- Veri tabanı okunup, csv uzantılı dosya da okunduktan ve veri tabanına aktarıldıktan sonra program kullanıcının erişimine açılır. ***while True*** döngüsüne alarak kullanıcı 0'ı tuşlamadığı sürece yani ***False*** olmadığı sürece program sürekli olarak kendini tekrar edecek ve kullanıcının seçtiği işlevi yerine getirecektir.

- Kullanıcının yanlışlıkla seçenek dışı bir rakam yazmasının sonucu programın hata vermemesi için eğer girilen sayı seçenekler arasında yoksa kullanıcıya hatalı giriş uyarısı çıktıktan sonra program kendini tekrar döndürür.
- Kullanıcı programda işlemlerini bitirdikten sonra 0 rakamını tuşlayarak programın çalışmasını bitirebilir. Ekrana "çıkış seçildi" uyarısı yazdırılır, veri tabanı bağlantısını sonlandırmak için ***close()*** fonksiyonu çağrılır ve sonrasında da ***exit(0)*** ile program sonlandırılır.

```
elif islev == "1":
    ekleme()
    csvkayit()
```

- 1 numaralı seçenek seçildiğinde kişi ekleme işlemini gerçekleştirebilmek için ***ekleme()*** fonksiyonun gidilir.

- ***ekleme()*** fonksiyonunda, eklenmek istenen kişinin veri tabanında kayıtlı olup olmadığını kontrol etmek için kimlik numarası bilgisi istenir. ***SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'*** komutu ile girilen kimlik numaralı bir kişinin olup olmadığı araştırılır. ***fetchall()*** fonksiyonuyla seçilen veriler değişkenin içerisine aktarılır. Bir for döngüsüyle ilgili kimlik numaralı kişi ***sorgu*** isimli listenin içinde tutulur. ***sorgu*** listesinin eleman sayısı sıfırdan farklıysa eklenmek istenen kimlik numaralı kişi veri tabanında mevcut olduğu bilgisi ekrana yazdırılır ve fonksiyonun içinden çıkılır, ama ***sorgu*** listesinin

eleman sayısı sıfıra eşit ise eklenmek istenen kişinin adı, soyadı, baba adı vs. bilgileri de sırasıyla girilerek ***INSERT INTO kisiler VALUES*** komutuyla, kisiler isimli veri tabanındaki

```
def ekleme():
    kimlikno = input("Eklemek istediğiniz kişinin kimlik numarasını giriniz:")
    imlec.execute("""SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'""".format(kimlikno))
    veri = imlec.fetchall()
    sorgu = []
    for i in veri:
        sorgu.append(i)
    if len(sorgu) == 0:
        ad = input("Eklemek istediğiniz kişinin ADINI giriniz:")
        soyad = input("Eklemek istediğiniz kişinin SOYADINI giriniz:")
        babaAdi = input("Eklemek istediğiniz kişinin BABA ADINI giriniz:")
        anneAdi = input("Eklemek istediğiniz kişinin ANNE ADINI giriniz:")
        dogumYeri = input("Eklemek istediğiniz kişinin DOGUM YERİNİ giriniz:")
        medeniDurum = input("Eklemek istediğiniz kişinin MEDENİ DURUMUNU giriniz:")
        kanGrubu = input("Eklemek istediğiniz kişinin KAN GRUBUNU giriniz:")
        kutukSehir = input("Eklemek istediğiniz kişinin KUTUK SEHRİNİ giriniz:")
        kutukIlce = input("Eklemek istediğiniz kişinin KUTUK ILCESI giriniz:")
        ikametgahSehir = input("Eklemek istediğiniz kişinin IKAMETGAH SEHRİNİ giriniz:")
        ikametgahIlce = input("Eklemek istediğiniz kişinin IKAMETGAH ILCESI giriniz:")

        imlec.execute("""INSERT INTO kisiler VALUES ('{}', '{}', '{}', '{}', '{}',
                    '{}', '{}', '{}', '{}', '{}', '{}', '{}')""".format(kimlikno, ad, soyad, babaAdi, anneAdi,
                                                                        dogumYeri,medeniDurum, kanGrubu, kutukSehir, kutukIlce,
                                                                        ikametgahSehir, ikametgahIlce))
        print(f"{kimlikno} kimlik numaralı {ad} kisi veri tabanına eklendi.")
        db.commit()
    else:
        print("Bu kişi veri tabanında mevcut.")
```

tabloya ilgili sütunların altına eklenmiş olur. Veri tabanına ekleme işlemi bittikten sonrada bilgilendirme mesajı kişinin eklendiği bilgisini ekrana yazdırır. Yapılan değişikliğin veri tabanında da uygulanabilmesi için ***db.commit()*** fonksiyonunu çalıştırırız.

```
def csvkayit():
    imlec.execute("""SELECT DISTINCT * FROM kisiler""")
    kisiListe = []
    for i in imlec:
        kisiListe.append(i)
    dosya=open("kisiler.csv","w")
    kaydet=[]
    for insan in kisiListe:
        kaydet.append(";".join(insan) + "\n")
    dosya.writelines(kaydet)
    dosya.close()
```

- Her ekleme, güncelleme, silme gibi veri tabanında değişiklik yaratan işlemden sonra ***csvkayit()*** fonksiyonuna gidilir. Tüm veriler öncelikle ***SELECT*** komutuyla seçilir ve ***kisiListe*** isimli listeye eklenir. Dosya yeniden yazılmak üzere "write" işleviyle açılır. ***kaydet*** isimli yeni bir liste daha oluşturulur ve ***kisiListe*** isimli listeye eklenen veriler csv formatına uygun bir şekilde ***kaydet*** listesine eklenir. ***writelines*** metodu ile dosyaya liste tipinde kaydedilen veriler yazılır ve ***close()*** fonksiyonu ile dosya kapatılır. Bu şekilde veri tabanında yapılan her değişiklik sonucu dosyada da aynı güncellemeler gerçekleşir.

```
    elif islev == "2":
        dbListele()
    elif islev == "3":
        arama()
```

- 2 numaralı seçenek seçildiğinde veri tabanı listeleme işlemini gerçekleştirebilmek için ***dbListele()*** fonksiyonuna gidilir.
- 3 numaralı seçenek seçildiğinde kişi arama işlemini gerçekleştirebilmek için ***arama()*** fonksiyonuna gidilir.

```
def arama():
    kimlikno=input("Bulmak istediğiniz/aradığınız kişinin kimlik numarasını giriniz:")
    imlec.execute("""SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'""".format(kimlikno))
    veri = imlec.fetchall()
    sorgu=[]
    for i in veri:
        sorgu.append(i)
        yazdirma(i)
    if len(sorgu)==0:
        print("Aradığınız kişi veri tabanımızda bulunamadı.")
        ekle=input("Kişiyi eklemek ister misiniz?(evet/hayır)")
        if ekle.lower()=="evet":
            ekleme()
            csvkayit()
```

- Kullanıcı bulmak istediği kişinin kimlik numarasını girer. ***SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'*** komutu ile girilen kimlik numaralı kişinin nerede olduğu araştırılır. Bulunan veri ***fetchall()*** ile değişkene atılır ve kişi for döngüsü içinde ***sorgu*** isimli listeye eklenir ve bulunan kişinin bilgilerinin yazdırılması için ***yazdirma(i)*** isimli fonksiyona kişinin verileri gönderilir.

```
def yazdirma(arg):
    print('KİMLİK NO'.ljust(10), '  ADI'.ljust(10), 'SOYADI'.ljust(10), 'BABA ADI'.ljust(10), 'ANNE ADI'.ljust(11),
          'DOGUM YERI'.ljust(11), 'MEDENI DURUM'.ljust(15),'KAN GRUBU'.ljust(13), 'KUTUK SEHRI'.ljust(15),
          'KUTUK ILCE'.ljust(15), 'IKAMETGAH SEHRİ'.ljust(20), 'IKAMETGAH ILCE'.ljust(10))
    print('--------- ---------  ---------  --------- -----------  -----------  ------------   ---------- '
          '    -----------     -----------     ---------------     ---------------')
    print(arg[0].ljust(10), arg[1].ljust(10),arg[2].ljust(10), arg[3].ljust(10),arg[4].ljust(11),
          arg[5].ljust(11),arg[6].ljust(15), arg[7].ljust(13),arg[8].ljust(15),
          arg[9].ljust(15),arg[10].ljust(20), arg[11].ljust(10))
```

- ***yazdirma(i)*** isimli fonksiyonun ***dbListele()*** isimli fonksiyondan farkı; ***dbListele()*** fonksiyonunda tüm veri tabanı ***SELECT*** komutuyla seçildiği için tüm veri tabanı yazdırılıyor iken, ***yazdirma(i)*** fonksiyonunda sadece bulunması istenen kişinin bilgileri ekrana çıktı olarak verilmektedir.

- ***sorgu*** listesinin eleman sayısı sıfır ise aranan kişi veri tabanında bulunamamıştır. Kullanıcı isterse arattığı kimlik numaralı kişiyi veri tabanına ekleyebilir. Bunun için "evet" yazması yeterlidir. Eklemek istemiyor ise "hayır" yazarak ana menüye dönebilmektedir.

```
elif islev == "4":
    guncelleme()
    csvkayit()
```

- 4 numaralı seçenek seçildiğinde kişi bilgisi güncelleme işlemini gerçekleştirebilmek için ***guncelleme()*** fonksiyonuna gidilir. Güncelleme işlemi sonrasında da değişikliğin dosyaya da kaydedilmesi için ***csvkayit()*** fonksiyonuna gidilir.

```python
def guncelleme():
    kimlikno=input("Güncellemek istediğiniz kişinin kimlik numarasını giriniz:")
    set=input("Hangi bilgiyi güncelleyeceksiniz\n1.Kimlik Numarası\t2.Adı\t3.Soyadı\t4.Baba Adı\n"
              "5.Anne Adı\t6.Dogum Yeri\t7.Medeni Durum\t8.Kan Grubu\n"
              "9.Kütük Sehri\t10.Kütük İlçe\t11.İkametgah Şehir\t12.İkametgah İlçe\n->Seçiminiz::")
    if set=="1":set="KimlikNo"
    elif set == "2": set = "Adı"
    elif set == "3": set = "Soyadı"
    elif set == "4": set = "Baba_adı"
    elif set == "5": set = "Anne_adı"
    elif set == "6": set = "Doğum_yeri"
    elif set == "7": set = "Medeni_durumu"
    elif set == "8": set = "Kan_grubu"
    elif set == "9": set = "Kütük_Şehir"
    elif set == "10": set = "Kütük_İlçe"
    elif set == "11": set = "İkametgah_Şehir"
    elif set == "12": set = "İkametgahİlçe"
    komut = "UPDATE kisiler SET {} = ? WHERE kimlikno = ?".format(set)
    imlec.execute("""SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'""".format(kimlikno))
    veri = imlec.fetchall()
    sorgu=[]
    for search in veri:
        sorgu.append(search)
    if len(sorgu) == 0:
        print("Güncellemek istediğiniz kişi veri tabanımızda bulunamadı.")
    else:
        yeniverigirisi = input(f"Yeni {set} bilgisini giriniz:")
        imlec.execute(komut,(yeniverigirisi, kimlikno))
        db.commit()
        dbListele()
```

- Güncellenecek kişinin kimlik numarası kullanıcıdan istenir. Sonrasında ekrana hangi bilgiyi güncellemek istediğine dair seçenekler çıkar. Burada if ifadesini kullanmamızın sebebi güncellenmek istenilen bilginin sütun başlığının, kullanıcı tarafından bilinmediği için programdan hata almamaktır. Kullanıcı hangi bilgiyi güncellemek istiyor ise ilgili sayıyı yazar ve program kullanıcı için gerekli sütuna gider. ***SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'*** komutu ile girilen kimlik numaralı kişinin nerede olduğu araştırılır. Bulunan veriler for döngüsüyle ***sorgu*** isimli fonksiyonun içine eklenir. Eğer ***sorgu*** listesinin eleman sayısı sıfır ise kişinin veri tabanında kayıtlı olmadığı anlamına gelir. Uyarı mesajı ekrana bastırılır ve fonksiyondan çıkılır. Ama fonksiyonun eleman sayısı sıfırdan farklı ise güncellenecek olan bilginin yeni değeri girilir. Bilgi girildikten sonra yukarıda tanımlanmış olan ***UPDATE kisiler SET {} = ? WHERE kimlikno = ?*** komutu ile girilen bilgiler ***yeniverigirisi*** ile birleştirilir. Yapılan değişikliğin veri tabanında da uygulanabilmesi için ***commit()*** fonksiyonunu çalıştırırız. Sonrasında da güncel veri tabanını kullanıcının da görebilmesi için ***dbListele()*** fonksiyonuyla veri tabanı yazdırılır.

```python
elif islev == "5":
    silme()
    csvkayit()
```

- 5 numaralı seçenek seçildiğinde kişi silme işlemini gerçekleştirebilmek için ***silme()*** fonksiyonuna gidilir. Silme işlemi sonrasında da değişikliğin dosyaya da kaydedilmesi için ***csvkayit()*** fonksiyonuna gidilir.

```
def silme():
    komut = "DELETE FROM kisiler WHERE kimlikno = ?"
    kimlikno = input("Silmek istediğiniz kişinin kimlik numarasını giriniz:")
    imlec.execute("""SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'""".format(kimlikno))
    veri = imlec.fetchall()
    sorgu = []
    for i in veri:
        sorgu.append(i)
    if len(sorgu)==0:
        print("Silmek istediğiniz kişi veri tabanımızda bulunamadı.")
    else:
        imlec.execute(komut, (kimlikno,))
        db.commit()
        print("Yeni veri tabanı;")
        dbListele()
```

- Silmek istenilen kişinin kimlik numarası kullanıcıdan istenir. ***SELECT DISTINCT * FROM kisiler WHERE kimlikno = '{}'*** komutu ile girilen kimlik numaralı kişinin nerede olduğu araştırılır. Bulunan veriler for döngüsüyle ***sorgu*** isimli fonksiyonun içine eklenir. Eğer ***sorgu*** listesinin eleman sayısı sıfır ise kişinin veri tabanında kayıtlı olmadığı anlamına gelir. Uyarı mesajı ekrana bastırılır ve fonksiyondan çıkılır. Ama fonksiyonun eleman sayısı sıfırdan farklı ise ***DELETE FROM kisiler WHERE kimlikno = ?*** komutuna, girilen kimlik numarası atanır ve kişi veri tabanından silinmiş olunur. Yapılan değişikliğin veri tabanında da uygulanabilmesi için ***commit()*** fonksiyonunu çalıştırırız. Sonrasında da güncel veri tabanını kullanıcının da görebilmesi için ***dbListele()*** fonksiyonuyla veri tabanı yazdırılır.

III. KULLANILAN YAZILIM

Proje kapsamında 3 araç kullanılmıştır.

- ➔ Visual Studio Community
- ➔ Python Pycharm
- ➔ Sqlite3 modülü

IV. SONUÇLAR

Proje sonucunda SQLite3 modülünü ve komutlarıyla veri tabanına yeni eleman ekleme, eleman silme, arama, güncelleme ve listeleme gibi komutları gerçekleştirmeyi ve bunu python diliyle bir arada kullanmayı öğrendik.

Aynı zamanda da SQLite modülüyle veri tabanında yaptığımız işlemleri sonrasında csv dosyamızın içine ekleme ve program başlangıcında dosyada var olan verileri de veri tabanına geçirme işlemlerini gerçekleştirdik.

PROJE EKİBİ

Damla Su KARADOĞAN, 11.02.2001 yılında doğdu. 2019 yılında Özel Envar Anadolu Lisesinden mezun oldu. Şu anda Fenerbahçe Üniversitesinde Endüstri Mühendisliği bölümünde lisans eğitimi almakta. C , Python dillerine ve AutoCAD'e hakim, başlangıç seviyesinde verilog biliyor. Programlama ve sanatla ilgileniyor. Öğrenci numarası; 190302016.

Tuğberk Onur KURU 11.05.2001 yılında Sakarya'da doğdu. 2019 Yılında Teksen Anadolu Lisesinden mezun oldu. Şu anda Fenerbahçe Üniversitesinde Endüstri Mühendisliği bölümünde lisans eğitimi almakta. C, Python ve AutoCAD dilinde yetkin giriş seviyesinde PHP biliyor. Güncel teknoloji, Yüzücülük ve Satrançla ilgileniyor. Öğrenci numarası; 190302002.

Kenan GÖZÜAÇIK 14.03.2001 yılında Rize'de doğdu. 2019 yılında Pazar Fen Lisesinden mezun oldu. Şu anda Fenerbahçe Üniversitesinde Endüstri Mühendisliği bölümünde lisans eğitimi almakta. C, Python ve Autocad eğitimleri aldı. Tarih, siyaset, ekonomi alanlarında araştırmalar yapmakta. Amatör olarak dağ tırmanışı kamp ile ilgileniyor. 2012 yılından beri özel bir şirketin bayiliğinde dönemlik çalışıyor. Öğrenci numarası; 190302009.

Tuana Aydoğan 17.07.2000 yılında Ankara'da doğdu. 2018 yılında Zonguldak Fen Lisesinden mezun oldu. Şu anda Fenerbahçe Üniversitesinde Endüstri mühendisliği bölümünde lisans eğitimi almakta. C ve Python diline hakim, AutoCAD biliyor. Ekonomi, borsa ve sporla ilgileniyor. Öğrenci numarası; 190302030.

REFERANS DOSYALAR

Sunu:

nüfusyönetimsistemi_proje.pptx

CV:

cv.pdf

Github: https://github.com/damlasu/NufusYonetimSistemi

Youtube: https://www.youtube.com/watch?v=z92TmiOhR3Q&feature=youtu.be

KAYNAKLAR

Kaynakça

(2021, Ocak 6). https://derslik.kerteriz.net/python-dersleri/veritabani-islemleri/python-sqlite-veritabani adresinden alındı

(2021, Ocak 6). https://www.youtube.com/watch?v=LLL4hTa6FDA adresinden alındı

Başer, M. (2019). Veritabanı. *Python* (s. 283). içinde İstanbul: dikeyeksen.

Levent, V. E. (2021, Ocak 6). *Nüfus Yönetim Sistemi*. levent.tc: http://levent.tc/files/courses/object_oriented_design/projects/project4/BLM205_nufus_yonetim.pdf adresinden alındı