



CRM : CRM Projesi(Customer Relationship Management-Müşteri İlişkileri Yönetimi)

Genel

Gunumuzde bir cok kurulus verilerini yonetme ve saklama gibi opsiyonlari kismende olsa ucretsiz destekledigi icin Google drive ve bilesenlerini kullanmaktadır. Cesitli organizasyonlar IT alanında calismak isteyen multeci kokenli kisiler icin mentor gorusmesi ve istenilen sartlari saglayanlar icin projelendirme ve mulakat asamalari dahil olmak uzere bir dizi islem yapmakta ve bu islemleri google drive uzerinde gerceklestirmektedir. Adaylari takip edilmesi icin drive uzerinde surekli oturumun acik olmasi , excel metinlerin karisikligi ,istenilen verilere kısa yoldan ulasabilme vs gibi zorluklar nedeni ile isleri daha kolay hale getirebilecek kullanıcı dostu bir uygulama tasarlamak amacıyla asagidaki proje tasarlanmistir.

Kullanici Arayuzu Detaylari **Giris Penceresi**

1-Kullanici Adi ve Sifre

- » google drive ana gmail-hesabi kullanicisi tarafından kaydedilmis kullanıcı adı-sifre sahibi kisilerin erisimine izin verilmeli .

Tercihler

1- Basvurular

- » basvurular butonu kullanıcıyı ilk basvuru penceresine yonlendirmeli

2- Mentor gorusmesi

- » mentor gorusmesi butonu kullanıcıyı mentor penceresine yonlendirmeli

3- Mulakatlar

- » mulakatlar butonu kullanıcıyı mulakatlar penceresine yonlendirmeli

Basvurular

1-Ara

- » text satirina girilen karakterler ile isim soyisimler icinde arama yapabilen bir buton islevi kazandirilmali (orn: 'As' girisinde drive da kayitli as ile baslayan tum isimleri getirebilmeli)

2-Tum Basvurular

- » tum basvurular butonu tiklandiginda driveda kayitli tum basvurular ekrana getirilmeli

3-Mentor Gorusmesi Tanimlananlar

- » Mentor gorusmesi tanimlananlar butonu tiklandiginda , basvuru yaptıktan sonra kendisine mentor gorusmesi tanimlanmis kisiler ekrana getirilmeli

4-Mentor Gorusmesi Tanimlanmayanlar

- » Mentor gorusmesi tanimlanmayanlar butonu tiklandiginda, basvuru yaptıktan sonra kendisine halen mentor atanmamis olan kisiler ekrana getirilmeli

5-Tercihler Ekranina Geri Don

- » Tercihler Ekranina Geri Don butonu tiklandiginda kullanıcı Tercihler ekranina geri donmeli

Mentor

1-Ara

- » text satirina girilen karakterler ile isim soyisimler icinde arama yapabilen bir buton islevi kazandirilmali (orn: 'As' girisinde drive da kayitli as ile baslayan tum isimleri getirebilmeli)

2-Tum Gorusmeler

- » tum gorusmeler butonu tiklandiginda driveda kayitli tum gorusmeler ekrana getirilmeli

3-Coklu sekme

- » bu sekmede secilen tercihe uygun kayitlar ekrana getirilmeli

4-Tercihler Ekranina Geri Don

- » Tercihler Ekranina Geri Don butonu tiklandiginda kullanıcı Tercihler ekranina geri donmeli

Mulakatlar

1-Ara

- » text satirina girilen karakterler ile isim soyisimler icinde arama yapabilen bir buton islevi kazandirilmali (orn: 'As' girisinde drive da kayitli as ile baslayan tum isimleri getirebilmeli)

2-Proje Gonderilmis Olanlar

- » Proje gonderilmis olanlar butonu tiklandiginda driveda kayitli proje gonderilmis adaylar ekrana getirilmeli

3-Projesi Gelmis Olanlar

- » Projesi gelmis olanlar butonu tiklandiginda driveda kayitli projesi gelmis adaylar ekrana getirilmeli

4-Tercihler Ekranina Geri Don

- » Tercihler Ekranina Geri Don butonu tiklandiginda kullanıcı Tercihler ekranina geri donmeli

Proje_Arayuz

Kullanıcı adı ve Şifre giriş kutucukları. Şifre karakterleri yazıldığında • karakterleri görünür.

WFA=HERE

Hoşgeldiniz

Lütfen işlem yapmak için kullanıcı adınızı ve şifrenizi girip Giriş tuşuna basınız.

Kullanıcı adı :

Kullanıcı şifresi :

Sisteme giriş başarısız!

Giriş

Kapat

Giriş tuşuna basıldığında, excel kayıtlı listedeki kullanıcı adı ve şifresini girilen değerlerle karşılaştırıp doğruysa TercihMenu sayfasına giderken yazılanları siler hoşgeldiniz diye seslendirir. Yanlış ise 'Sisteme giriş başarısız ! Kullanıcı adı yada şifre hatalı! yazısını döndürür.

©VIT | HES

TercihMenu

Lütfen işlem yapmak istediğiniz k...

Başvurular

Mentor Görüşmesi

Mülakatlar

Kapat

©VIT | HES

Dropdown menüsü sadece bu sayfada vardır ve listeden seçilen bilgilerle alakalı kişileri listeler

Uyarı yazısı Bulunduğu listeleme olan bütün sayfalarda aynı işlevi görür. Listeleme işlemlerinde, aranan ve bulunan listede kaç kişi olduğu 3 saniye ekranda kalır.

MentorGorusmeSayfasi

Ara

Bu sayfa ile mentor görüşme organizasyonu yapılacaktır.

Tüm Görüşmeler

Lütfen listeden ilgili satırı seçiniz!

'Bulunan Kişi Sayısı : {len(self.df)}'

Görüşme Tarihi	Adı Soyadı	Mentor	IT Bilgisi	Yoğunluk	Yorumlar
Bulunduğu bütün sayfalarda ilgili liste burada görünür. Listeleme yaparken her seferinde eski liste sıfırlanır.					

Kapat

Geri

©VIT | HES





Aratmak istediğiniz kelimeyi yazınız!

Ara

'Aradığınız kişi listede bulunmamaktadır!'

Bulunduğu bütün sayfalarda aynı işlevi görür ve girilen metinle arama yapar. Yanlış metin girildiğinde 'Aradığınız kişi listede bulunmamaktadır!' uyarısı 3 saniye ekranda kalır

Tüm Başvurular

Mentor Görüşmesi Tanımlananlar

Mentor Görüşmesi Tanımlanmayanlar

Zaman Ad Soyad Mail Telefon Posta Kodu Eyalet Su Anki Durum

Bazı işlevler üzerine gelindiğinde uyarı yazıları çıkar.

Kapat

Uygulamadan çıkar!

©VIT | HES

BasvurularSayfasi

Geri

Tercih menüsüne geri döner

Menudeki sayfalara basıldığında ilgili sayfalara gider.

Geri

Bu sayfa ile mülakatların organizasyonu yapılacaktır.

Projesi Gönderilmiş Olanlar

Projesi Gelmiş Olanlar

Ad Soyad Proje Gönderim Tarihi Proje Gelme Tarihi

Sayfalardaki bütün resimler transparant özellikte, yüksek çözünürlük ve düşük dosya boyutundadır.

Kapat

Geri

©VIT | HES

Copyright

Mulakatlar

Geri butonu

Bulunduğu bütün sayfalarda istenen işlevi görür. TercihMenu sayfasında Proje_Arayuz sayfasına diğer sayfalarda TercihMenu sayfasına gider. Bu tuşa basıldığında sayfada yazılı bütün metinler kaybolur ve geri gelindiğinde görünmez.



Satır	Kod	Açıklama
1	<code>import sys</code>	# Sistemle ilgili işlemleri yapmak için kullanılır.
2	<code>from PyQt6.QtWidgets import QApplication, QMainWindow, QWidget, QTableWidgetItem</code>	# PyQt6 widget'ları ve tablo öğelerini içerir.
3	<code>from PyQt6.QtCore import QApplication, QTimer</code>	PyQt6 çekirdek modülü, uygulama işlemleri için kullanılır. QTimer süre modülüdür.
4	<code>from PyQt6.uic import loadUi</code>	# PyQt6 ile UI dosyalarını yüklemek için kullanılır.
5	<code>import gspread</code>	# Google Sheets API'yi kullanmak için kullanılır.
6	<code>import pandas as pd</code>	# Veri analizi ve manipülasyonu için kullanılır.
7	<code>import pygame</code>	# Ses dosyası çaldırmak için kullanılır.



Kod bloğu ile ilgili genel bilgiler

- ▶ Kodun en başında gerekli kütüphaneleri içe aktarıyoruz.
Örneğin terminalden aşağıdaki işlemle pandas kütüphanesini bilgisayara yüklüyoruz.

```
pip install pandas
```

- ▶ Daha sonra kodu içe aktarıyoruz (import). as pd ile kodu yazarken her seferinde pandas yazmaktan kurtulup pd yazıyoruz.

```
import pandas as pd
```

- ▶ **Google Drive** dan dosyaları almak için **gspread** import ettik. gspread JSON Source File (.json) dosyasına erişim sağlamak için kullanılan kütüphanedir. .json dosyası JSON kimlik bilgilerini içerir. Bu kimlik bilgileri, Google Sheets API'ye (yani google drive daki dosyalara) erişim izinlerini sağlar.
- ▶ **Qt designer** ile hazırladığımız arayüzleri (interface) göstermek için **loadUi** kütüphanesini import ettik.

loadUi kullanıldığında qt designer dosyalarını ayrı ayrı oluşturup herbirinde yapacağımız değişiklikleri otomatik .py dosyamızda gösterebiliyoruz. Bu da bize büyük bir işlevsellik kazandırıyor ve .ui dosyasının kod kalabalığından kurtuluyoruz. Öte taraftan kod ile .ui dosyasına müdahale edemiyoruz.

ÖNEMLİ

- ▶ Python kütüphanelerine (Python packages) aşağıdaki linkten ulaşabilirsiniz.

<https://pypi.org>

- ▶ qt designer ücretsiz sürümünü aşağıdaki linkten indirebilirsiniz.

<https://www.qt.io/download>

EXTRA BİLGİ

Aşağıda terminalden yapılabilecek başka işlemler de verilmiştir.

Paket Güncelleme

pip install --upgrade pandas

Belirli Bir Sürümde Paket Kurma

pip install pandas==1.2.4

Paket Kaldırma

pip uninstall pandas

Kurulu Paketleri Listeleme

pip list

Satır	Kod	Açıklama
	<code>creds = 'key.json'</code>	# 'key.json' adlı JSON kimlik bilgileri dosyasının yolu `creds` değişkenine atanıyor.
	<code>gc=gspread.service_account(filename=creds)</code>	# `gspread` kütüphanesini kullanarak Google Sheets'e bağlanılıyor. Bu adımda, `filename` parametresine `creds` değişkeni verilerek JSON kimlik bilgileri dosyası okunuyor.
	<code>spreadsheet=gc.open('Kullanıcılar')</code>	# 'Kullanıcılar' adlı Google Sheets belgesi açılıyor.
	<code>worksheet= spreadsheet.get_worksheet(0)</code>	# Belgedeki ilk çalışma sayfasına (`worksheet`) erişiliyor.
	<code>all_values = worksheet.get_all_values()</code>	# Çalışma sayfasındaki tüm değerler bir liste olarak alınıyor ve `all_values` adlı değişkene atanıyor.
	<code>del all_values[0]</code>	# Listenin ilk ögesi (başlık satırı) siliniyor.



Kod bloğu ile ilgili genel bilgiler

- ▶ Bu kod parçası, 'Kullanıcılar' adlı Google Sheets belgesinin ilk çalışma sayfasındaki verileri bir liste olarak `all_values` adlı değişken içinde tutuyor.
- ▶ **get fonksiyonu**, bir sözlük (dictionary) üzerinde belirli bir anahtar (key) ile değer (value) çiftine erişim sağlamak için kullanılır. Sözlükler, Python'daki veri tiplerinden biridir ve anahtar-değer çiftlerini içerirler.
- ▶ **del fonksiyonu**, Python'da bir nesneyi (değişkeni, liste ögesini, sözlük anahtarını vb.) silmek için kullanılır.
del kullanırken dikkatli olunmalıdır. Çünkü bu işlem geri alınmaz ve silinen nesne artık kullanılamaz.
- ▶ **creds**, "credentials" (kimlik bilgileri) kelimesinin kısaltmasıdır.
creds = 'key.json' ifadesinde, 'key.json', bu JSON kimlik bilgileri dosyasının yolunu ve adını belirtir. Bu dosya genellikle bir API hizmet hesabının anahtar bilgilerini içerir.
- ▶ **open fonksiyonu**, dosya işlemleri yapmak için Python'da kullanılan bir fonksiyondur. Bu fonksiyon, belirtilen dosyayı açmak ve belirli bir modda (okuma, yazma, ekleme, vb.) kullanmanıza olanak tanır.

ÖNEMLİ

Kodumuzda drive dan 4 farklı excel dosyası çekme işlemi var. 1. si buradaki ilk sayfa kullanıcı adı ve şifresinin çekilmesi, diğerleri de yan sayfadaki 3 ayrı arayüzdeki excellerin çekilmesidir. Hepsinde neredeyse aynı odlar vardır. Bu sayfada anlatılan ilk kodda .json dosyası ilk 2 satırda açılıp okunmuştur.

EXTRA BİLGİ

JSON (JavaScript Object Notation), veri alışverişi için kullanılan bir hafif veri biçimidir. JSON, insanlar tarafından okunabilir ve yazılabilir olmasının yanı sıra bilgisayarlar tarafından da kolayca işlenebilir. Genellikle web servislerinde veri iletimi için kullanılır.

JSON, anahtar-değer çiftleri ve değerlerden oluşan bir veri formatıdır. JSON nesneleri süslü parantezler {} içinde yer alır ve anahtar-değer çiftleri arasında virgül ile ayrılır. Örneğin,

```
{
  "name": "John Doe",
  "age": 30,
  "city": "New York",
  "isStudent": false,
  "grades": [85, 90, 78]
}
```



Drive'dan excel içeriğini çekip listede kaydetme kodları (class'lardan önce 4 farklı yerde)

```
spreadsheet=gc.open('Basvurular')
worksheet3= spreadsheet.get_worksheet(0)
all_values3 = worksheet3.get_all_values()
del all_values3[0]

class BasvurularSayfasi(QMainWindow):
```

Burada Başvurular excel dosyası açılmış,
get_worksheet(0) ile exceldeki ilk sayfaya ulaşılmış,
get_all_values() ile alınan bilgiler all_values3 listesine kaydedilmiş,

all_values3[0] ba;lik satiri del ile silinmiştir. Başlığı silme amacı çui dosyasında başlıkların zaten hazır bulunması ve tekrarın önüne geçilmesidir.

Yukarıdaki işlemlerle alınan excel dosyasından oluşturulan liste BasvurularSayfasi class ı ile aynı adlı arayüzde arama ve listeleme işlemlerinde kullanılacaktır.

```
spreadsheet=gc.open('Mulakatlar')
worksheet1= spreadsheet.get_worksheet(0)
all_values1 = worksheet1.get_all_values()
del all_values1[0]

class Mulakatlar(QMainWindow):
```

Yukarıdaki kodla tamamen aynı olan Mulakatlar için uyarlanmış class (sınıf) kodu

ÖNEMLİ

Drive dan çekilen bilgilerin kaydedildiği listeler (all_values3, all_values1, all_values2) birbirine karışmaması için faklı olarak adlandırılmıştır.

```
spreadsheet=gc.open('Mentor')
worksheet2= spreadsheet.get_worksheet(0)
all_values2 = worksheet2.get_all_values()
del all_values2[0]
for kayit in all_values2:
    kayit[4], kayit[5], kayit[6], kayit[7] = kayit[6], kayit[7], kayit[4], kayit[5]

class MentorGorusmeSayfasi(QMainWindow):
```

Yukarıdaki kodla aynı olan ve Mentor için uyarlanmış class (sınıf) kodu

» Bu kodda bir for döngüsü kullanılmıştır. Bu döngünün kullanılma amacı Drive excelinden alınan değerlerin combobox (Kullanıcının bir seçenek listesinden (dropdown list) bir öge seçmesine izin verir.) ile gösterilirken 2 sütünün arayüzde gösterilmeyecek olmasıdır. Yazdığımız for döngüsü ile 4. ve 5. sütunları excelin son 2 sütunu olan 6 ve 7. sutuna atarak arayüzde göstermemiş ve combobox umuz içinde istenen verilere ulaşmada sorun yaşamamış olduk.

TercihMenu class ından önce bu kod örneklerinin olmaması o arayüzde Drive dan herhangi bir dosyanın çağrılmamış olmasıdır.

Satır	Kod	Açıklama
	<code>class ProjeArayuz(QMainWindow):</code>	# PyQt6 ile arayüz tasarımı için QMainWindow sınıfını kullanan bir sınıf oluşturma
	<code>def __init__(self):</code>	#
	<code>super(ProjeArayuz, self).__init__()</code>	#
	<code>loadUi('Proje_Arayuz.ui', self)</code>	# Proje_Arayuz.ui adlı tasarım dosyasını yükleme
	<code>self.Uyari.clear()</code>	# Uyarı alanını temizleme
	<code>self.Giris_butonu_2.clicked.connect(self.Giris)</code>	# 'Giris' butonuna tıklandığında 'Giris' metodunu çağırma
	<code>self.Kapat_butonu_2.clicked.connect(self.kapatUygulamayi)</code>	# Kapat' butonuna tıklandığında kapatUygulamayi' metodunu çağırma
	<code>self.setWindowFlags(Qt.Core.Qt.WindowType.FramelessWindowHint)</code>	# .ui Frame (dış çerçeve) kapatma



Kod bloğu ile ilgili genel bilgiler

- ▶ Bu kod bloğunda bir sınıf (class) tanımlandı:
- ▶ **clear fonksiyonu**, belirli bir veri yapısındaki (liste, sözlük vb.) tüm elemanları silen bir metoddur.

Bu koddaUyari adlı text alanında görünen yazıyı ekranda görünmez hale getirir.

- ▶ **clicked.connect** PyQt'de, kullanıcı bir düğmeye tıkladığında (clicked) belirli bir slotla (işlev) bağlamak (connect) için kullanılır.

.ui uygulamasında oluşturulan QPushButton nesnesinin basıldığında aktif aktif olmasını ve kodda verilen def (tanıma) uygun olarak çalışmasını sağlar.

```
self.Giris_butonu_2.clicked.connect(self.Giris)
```

Giris_butonu_2 adlı QPushButton türü butona basıldığında Giris adıyla tanımlanmış (def) kodu çalıştırır.

ÖNEMLİ

Python'da girinti (indentation) seviyeleri, kodun yapısını ve hangi kod satırlarının bir bloğa ait olduğunu belirtmek için boşluk veya tab olarak kullanılır. Yukarıdaki kodda son 6 satır def den 1 tab içeride, def satırı da class satırından 1 tab içeridedir.

EXTRA BİLGİ

- ▶ ProjeArayuz adlı sınıf QMainWindow sınıfından türetilir. Yani, ProjeArayuz sınıfı, bir pencere (window) özelliği taşıyan PyQt6'nın QMainWindow sınıfına dayanır.
- ▶ **__init__** (double underscore init double underscore), Python programlama dilinde bir özel metottur ve "constructor" olarak adlandırılır. Bir sınıf (class) tanımlandığında, bu metod otomatik olarak çağrılır. Bu metod, sınıfın bir örneği (instance) oluşturulduğunda çağrılır.
- ▶ **self** parametresi, sınıfın kendisini temsil eden bir referanstır ve diğer tüm metotlarda olduğu gibi bu parametre kullanılarak sınıfın özelliklerine erişilir.
- ▶ `super(ProjeArayuz, self).__init__()` ifadesiyle, üst sınıfın (QMainWindow) constructor'ı başlatılmış olur.
- ▶ Burada ProjeArayuz sınıfı, QMainWindow sınıfının özelliklerini ve davranışlarını miras alır.



class (sınıf) 5 farklı sınıfımız var

```
class TercihMenu(QMainWindow):  
    def __init__(self):  
        super().__init__()   
        loadUi('TercihMenu.ui', self)  
        self.Basvurular_2.clicked.connect(self.basvurular)  
        self.Mentor_Gorusmesi_2.clicked.connect(self.mentorGorusmesi)  
        self.Mulakatlar_2.clicked.connect(self.mulakatlar)  
        self.Kapat_butonu_2.clicked.connect(self.kapatUygulamayi)  
        self.geriButonu.clicked.connect(self.geriGit)  
        self.setWindowFlags(Qt.Core.Qt.WindowType.FramelessWindowHint)
```

TercihMenu sınıfı oluşturuldu

TercihMenu adlı tasarım .ui sayfası açıldı

clicked.connect kodlarıyla butonlara basılarak ilgili sayfalara gidildi
son satırla .ui Frame (dış çerçeve) yok edildi

```
class BasvurularSayfasi(QMainWindow):  
    def __init__(self):  
        super().__init__()   
        loadUi('BasvurularSayfasi.ui', self)  
        self.Kapat_butonu_2.clicked.connect(self.kapatUygulamayi)  
        self.geriButonu.clicked.connect(self.geriGit)  
        self.ara_Butonu.clicked.connect(self.Arama)  
        self.TumBasvurular_2.clicked.connect(self.tumBasvurular)  
        self.MentorGorTan_2.clicked.connect(self.MgTamamlanan)  
        self.MentorGorTanMa_4.clicked.connect(self.MgTamamlanmayan)  
        self.setWindowFlags(Qt.Core.Qt.WindowType.FramelessWindowHint)
```

ÖNEMLİ

Bu class kodları neredeyse tamamen birbiriyle aynı.

```
class Mulakatlar(QMainWindow):  
    def __init__(self):  
        super().__init__()   
        loadUi('Mulakatlar.ui', self)  
        self.Cikis_Butonu.clicked.connect(self.kapatUygulamayi)  
        self.geriButonu.clicked.connect(self.geriGit)  
        self.ara_Butonu.clicked.connect(self.Arama)  
        self.ProjeGonderilmis_2.clicked.connect(self.proje1)  
        self.ProjeGelmis_3.clicked.connect(self.proje2)  
        self.setWindowFlags(Qt.Core.Qt.WindowType.FramelessWindowHint)
```

```
class MentorGorusmeSayfasi(QMainWindow):  
    def __init__(self):  
        super().__init__()   
        loadUi('MentorGorusmeSayfasi.ui', self)  
        self.Kapat_butonu_2.clicked.connect(self.kapatUygulamayi)  
        self.geriButonu.clicked.connect(self.geriGit)  
        self.ara_Butonu.clicked.connect(self.Arama)  
        self.TumGorusme_2.clicked.connect(self.TumGorusmeler)  
        self.comboBox.currentIndexChanged.connect(self.update_table)  
        self.setWindowFlags(Qt.Core.Qt.WindowType.FramelessWindowHint)
```




Satır	Kod	Açıklama
	<code>def Giriş(self):</code>	#
	<code>kullaniciadi = self.K_adi_2.text()</code>	#
	<code>sifre = self.K_sifresi_3.text()</code>	#
	<code>#print(all_values)</code>	#
	<code>for row in all_values:</code>	#
	<code>k = row[0]</code>	#
	<code>s = row[1]</code>	#
	<code>if kullaniciadi==k and sifre==s:</code>	#
	<code>#self.hide()</code>	#
	<code>tercih_menu.show()</code>	#
	<code>proje_arayuz.hide()</code>	#
	<code>self.K_adi_2.clear()</code>	#
	<code>self.K_sifresi_3.clear()</code>	#
	<code>self.Uyari.clear()</code>	#
	<code>pygame.mixer.init()</code>	#
	<code>pygame.mixer.music.load("C:/Users/Sami NL/Desktop/vit_odevler/W7a/hosgeldiniz.mp3")</code>	#
	<code>pygame.mixer.music.play()</code>	#
	<code># import time</code>	#
	<code># time.sleep(min(30, clip.seconds()))</code>	#
	<code># clip.stop()</code>	#
	<code>break</code>	#
	<code>elif kullaniciadi!=k or sifre!=s:</code>	#
	<code>self.K_adi_2.clear()</code>	#
	<code>self.K_sifresi_3.clear()</code>	#
	<code>self.Uyari.setText('Sisteme giriş başarısız ! Kullanıcı adı yada sifre hatalı!')</code>	#

Sayfalar arası geçişlerde en uzun kod yazma giriş işlemidir. Çünkü burada kayıtlı kullanıcı adı ve şifresi kullanıcının girdiği değerler ile karşılaştırılıp uygunsa TercihMenu sayfasına geçilecektir.



```
def kapatUygulamayi(self):  
    QCoreApplication.exit()
```

kapatUygulamayi kodu uygulamada 5 kere geçmekte ve her sayfadaki kapat butonuna basıldığında uygulamayı kapatmaktadır.

class TercihMenu(QMainWindow):

```
def mentorGorusmesi(self):  
    mentor_gorusme_sayfasi.show()  
    tercih_menu.hide()
```

```
def basvurular(self):  
    basvurular_sayfasi.show()  
    tercih_menu.hide()
```

```
def mulakatlar(self):  
    mulakatlar.show()  
    tercih_menu.hide()
```

```
def geriGit(self):  
    proje_arayuz.show()  
    tercih_menu.hide()
```

```
def geriGit(self):  
    self.tableWidget.setRowCount(0)  
    self.arama_kutusu.clear()  
    tercih_menu.show()  
    mulakatlar.hide()
```

```
def geriGit(self):  
    self.tableWidget.setRowCount(0)  
    self.arama_kutusu.clear()  
    tercih_menu.show()  
    basvurular_sayfasi.hide()
```

```
def geriGit(self):  
    self.tableWidget.setRowCount(0)  
    self.arama_kutusu.clear()  
    tercih_menu.show()  
    mentor_gorusme_sayfasi.hide()
```



Kod bloğu ile ilgili genel bilgiler

- ▶ tercihMenu deki butonlarına basıldığında halihazırdaki sayfayı gizleyip (hide) gideceği sayfayı göstermektedir. (show)
- ▶ Bu kodlar butona basma komutuyla beraber aşağıdaki gibi tek satırda da yazılabilir.

```
self.Basvurular_2.clicked.connect(lambda: (basvurular_sayfasi.show(), tercih_menu.hide()))
```



Kod bloğu ile ilgili genel bilgiler

- ▶ geriGit kodu uygulamada 4 kere geçmekte ve her sayfadaki geri butonuna basıldığında halihazırdaki sayfayı gizleyip (hide) gideceği sayfayı göstermektedir. (show)
- ▶ **clear fonksiyonu**, arama_kutusu adlı text alanında görünen yazıyı siler.
- ▶ QTableWidgetItem üzerindeki satır sayısını sıfırlamak için kullanılır. Bu satır, mevcut tablodaki tüm satırları temizleyerek tabloyu sıfırlar, yani boş hale getirir.

```
self.tableWidget.setRowCount(0)
```

Satır	Kod	Açıklama
	<code>def Arama(self):</code>	# Arama butonuna tıklandığında
	<code>if self.arama_kutusu.text()!="":</code>	# Eğer arama kutusunun metni boş değilse, yani kullanıcı bir şeyler girmişse
	<code>results= []</code>	# Arama sonuçlarını için boş liste
	<code>ara= self.arama_kutusu.text()</code>	# Kullanıcının girdiği metni alır.
	<code>for kayıt in all_values3:</code>	#
	<code>if ara.lower() in kayıt[1].lower():</code>	# Girilen metni içeren kayıtları excel 2. sütunda filtreleme ve result listesine ekleme yapar
	<code>results.append(kayıt)</code>	#
	<code>if not results:</code>	# Eğer sonuçlar listesi boşsa
	<code>self.arama_negatif.setText('Aradığınız kişi listede bulunmamaktadır!')</code>	# Sonuç olmayınca metni döner
	<code>QTimer.singleShot(3000, lambda: self.arama_negatif.clear())</code>	# Üst satırdaki yazının ekranda 3 saniye kalmasını sağlar
	<code>self.df = pd.DataFrame(results)</code>	# Sonuçları DataFrame(df) e dönüştürüp pandas (pd) tabloya ekleme
	<code>self.tableWidget.setRowCount(0)</code>	# Tablonun satır sayısını sıfırlar.
	<code>for row_index, (index, row) in enumerate(self.df.iterrows()):</code>	#
	<code>self.tableWidget.insertRow(row_index)</code>	#
	<code>for col_index, col_value in enumerate(row):</code>	#
	<code>item = QTableWidgetItem(str(col_value))</code>	#
	<code>self.tableWidget.setItem(row_index, col_index, item)</code>	#
	<code>self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')</code>	# Listede kaç kişi varsa ekrana yazar
	<code>QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())</code>	# Üst satırdaki yazının ekranda 3 saniye kalmasını sağlar



Kod bloğu ile ilgili genel bilgiler

- ▶ **lower()**, bir dize (string) ifadesinin tüm karakterlerini küçük harfe dönüştüren bir dize metodudur.
- ▶ **append()**, bir liste sonuna yeni bir öge (eleman) eklemek için kullanılır.
- ▶ **setText()** metodu QLabel, QLineEdit, QPlainTextEdit gibi PyQt6 metin öğeleri üzerinde kullanılır. Öğenin görünen metni güncellenir ve ekranda yeni metin görüntülenir.

ÖNEMLİ

`if ara.lower() in kayıt[1].lower():`
in kullanma nedenimiz aradığımız excel 1. sütundaki metindeki herhangi bir harfi bile arayabilmektir. Eğer

`if ara.lower() == kayıt[1].lower():`
yazsa idik excel kutuğundaki metni tam olarak yazmamız gerekirdi.



Text girerek arama fonksiyonları

```
def Arama(self):
    if self.arama_kutusu.text()!="":
        results= []
        ara= self.arama_kutusu.text()
        for kayıt in all_values1:
            if ara.lower() in kayıt[0].lower():
                results.append(kayıt)
        if not results:
            self.arama_negatif.setText('Aradığınız kişi listede bulunmamaktadır!')
            QTimer.singleShot(3000, lambda: self.arama_negatif.clear())
        self.df = pd.DataFrame(results)
        self.tableWidget.setRowCount(0)
        for row_index, (index, row) in enumerate(self.df.iterrows()):
            self.tableWidget.insertRow(row_index)
            for col_index, col_value in enumerate(row):
                item = QTableWidgetItem(str(col_value))
                self.tableWidget.setItem(row_index, col_index, item)
            self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
            QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class Mulakatlar(QMainWindow):
sınıfından miras alan arama
diğer aramalardan farkı miras aldığı sınıf
kullandığı excel dosyası (all_values1)
kullandığı excel dosyası sütunu (kayıt[0])

EXTRA BİLGİ

f{len(self.df)}

Bu kod parçası bir veri çerçevesi (DataFrame)
olan self.df içindeki satır sayısını verir.

len() listenin uzunluğunu (length) ifade eder.

```
def Arama(self):
    if self.arama_kutusu.text()!="":
        results= []
        ara= self.arama_kutusu.text()
        for kayıt in all_values2:
            if ara.lower() in kayıt[1].lower():
                results.append(kayıt)
        if not results:
            self.arama_negatif.setText('Aradığınız kişi listede bulunmamaktadır!')
            QTimer.singleShot(3000, lambda: self.arama_negatif.clear())
        self.df = pd.DataFrame(results)
        self.tableWidget.setRowCount(0)
        for row_index, (index, row) in enumerate(self.df.iterrows()):
            self.tableWidget.insertRow(row_index)
            for col_index, col_value in enumerate(row):
                item = QTableWidgetItem(str(col_value))
                self.tableWidget.setItem(row_index, col_index, item)
            self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
            QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class MentorGorusmeSayfasi(QMainWindow):
sınıfından miras alan arama
diğer aramalardan farkı miras aldığı sınıf
kullandığı excel dosyası (all_values2)
kullandığı excel dosyası sütunu (kayıt[1])

ÖNEMLİ

for döngüsü, bir iterable (tekrarlanabilir)
nesnenin her elemanı üzerinde dolaşmak
için kullanılan bir kontrol yapılarından biridir.

for eleman in iterable:

eleman: Her bir döngü iterasyonunda, ite-
rable (tekrarlanabilir) nesnenin bir elemanı-
nı temsil eden değişken adıdır.

iterable: Tekrarlanabilir bir nesnedir, örneğin
liste, demet (tuple), dize (string), küme (set),
sözlük (dictionary), vb.

Döngü her iterasyonunda eleman değişkeni,
iterable nesnenin bir sonraki elemanını alır ve
döngü içinde belirtilen işlemler gerçekleştiri-
lir. Iterable nesne üzerinde eleman kalmadı-
ğında döngü sona erer.

Butona basarak listeleme fonksiyonları

```
def TumGorusrmeler(self):
    self.df = pd.DataFrame(all_values2)
    self.tableWidget.setRowCount(0)
    for row_index, (index, row) in enumerate(self.df.iterrows()):
        self.tableWidget.insertRow(row_index)
        for col_index, col_value in enumerate(row):
            item = QTableWidgetItem(str(col_value))
            if item:
                self.tableWidget.setItem(row_index, col_index, item)
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

```
def MgTamamlanan(self):
    results= []
    for kayıt in all_values3:
        for x in all_values2:
            if kayıt[1] == x[1]:
                results.append(kayıt)
    self.df = pd.DataFrame(results)
    self.tableWidget.setRowCount(0)
    for row_index, (index, row) in enumerate(self.df.iterrows()):
        self.tableWidget.insertRow(row_index)
        for col_index, col_value in enumerate(row):
            item = QTableWidgetItem(str(col_value))
            if item:
                self.tableWidget.setItem(row_index, col_index, item)
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

```
def MgTamamlanmayan(self):
    results= []
    x= []
    for kayıt in all_values2:
        x.append(kayıt[1])
    for kayıt in all_values3:
        if not (kayıt[1] in x):
            results.append(kayıt)
        break
    self.df = pd.DataFrame(results)
    self.tableWidget.setRowCount(0)
    for row_index, (index, row) in enumerate(self.df.iterrows()):
        self.tableWidget.insertRow(row_index)
        for col_index, col_value in enumerate(row):
            item = QTableWidgetItem(str(col_value))
            if item:
                self.tableWidget.setItem(row_index, col_index, item)
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class BasvurularSayfasi(QMainWindow):
sınıfından miras alan listeleme
arama özelliğinin 2. kısmı ile aynı. Çünkü
sadece belirli bir exceldaki bilgilerin tamamını
alıp ekranda listeliyor.
diğer aramalardan farkı miras aldığı sınıf
kullandığı excel dosyası (all_values2)



Butona basarak listeleme fonksiyonları

```
def proje1(self):
    results= []
    for kayıt in all_values1:
        if kayıt[1] == ' ':
            results.append(kayıt)
    self.df = pd.DataFrame(results)
    self.tableWidget.setRowCount(0)
    for row_index, (index, row) in enumerate(self.df.iterrows()):
        self.tableWidget.insertRow(row_index)
        for col_index, col_value in enumerate(row):
            item = QTableWidgetItem(str(col_value))
            if item:
                self.tableWidget.setItem(row_index, col_index, item)
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class Mulakatlar(QMainWindow):
sınıfından miras alan listeleme
arama özelliğinin 2. kısmı ile aynı. Çünkü
sadece belirli bir exceldeki bilgilerin tama-
mını alıp ekranda listeliyor.
diğer aramalardan farkı miras aldığı sınıf
kullandığı excel dosyası (all_values1)
if kayıt[1] == ' ': koduyla 1. sütunu boş sa-
tırları süzüyor.

```
def proje2(self):
    results= []
    x= []
    for kayıt in all_values1:
        if kayıt[2] == ' ':
            results.append(kayıt)
    self.df = pd.DataFrame(results)
    self.tableWidget.setRowCount(0)
    for row_index, (index, row) in enumerate(self.df.iterrows()):
        self.tableWidget.insertRow(row_index)
        for col_index, col_value in enumerate(row):
            item = QTableWidgetItem(str(col_value))
            if item:
                self.tableWidget.setItem(row_index, col_index, item)
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class Mulakatlar(QMainWindow):
sınıfından miras alan listeleme
arama özelliğinin 2. kısmı ile aynı. Çünkü
sadece belirli bir exceldeki bilgilerin tama-
mını alıp ekranda listeliyor.
diğer aramalardan farkı miras aldığı sınıf
kullandığı excel dosyası (all_values1)
if kayıt[1] == ' ': koduyla 2. sütunu boş sa-
tırları süzüyor.

Butona basarak listeleme fonksiyonları

```
def TumGorusmeler(self):  
    self.df = pd.DataFrame(all_values2)  
    self.tableWidget.setRowCount(0)  
    for row_index, (index, row) in enumerate(self.df.iterrows()):  
        self.tableWidget.insertRow(row_index)  
        for col_index, col_value in enumerate(row):  
            item = QTableWidgetItem(str(col_value))  
            if item:  
                self.tableWidget.setItem(row_index, col_index, item)  
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')  
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class MentorGorusmeSayfasi(QMainWindow):
sınıfından miras alan listeleme
arama özelliğinin 2. kısmı ile aynı. Çünkü sadece belirli bir exceldeki bilgilerin tamamını alıp ekranda listeliyor.
diğer aramalardan farkı miras aldığı sınıf kullandığı excel dosyası (all_values2)

comboBox kullanarak listeleme

```
def update_table(self):  
    results= []  
    for kayıt in all_values1:  
        if self.comboBox.currentText() == kayıt[6]:  
            results.append(kayıt)  
    self.df = pd.DataFrame(results)  
    self.tableWidget.setRowCount(0)  
    for row_index, (index, row) in enumerate(self.df.iterrows()):  
        self.tableWidget.insertRow(row_index)  
        for col_index, col_value in enumerate(row):  
            item = QTableWidgetItem(str(col_value))  
            if item:  
                self.tableWidget.setItem(row_index, col_index, item)  
    self.liste_kisi_sayisi.setText(f'Bulunan Kişi Sayısı : {len(self.df)}')  
    QTimer.singleShot(3000, lambda: self.liste_kisi_sayisi.clear())
```

class Mulakatlar(QMainWindow):
sınıfından miras alan listeleme
arama özelliğinin 2. kısmı ile aynı. Çünkü sadece belirli bir exceldeki bilgilerin tamamını alıp ekranda listeliyor.
diğer aramalardan farkı miras aldığı sınıf kullandığı excel dosyası (all_values1)
comboBox.currentText() == kayıt[6]:
kodu listedeki yazı ile excelin 6. sütunundaki eşleyen yazıları bulur.

ÖNEMLİ

QComboBox sınıfı, PyQt kütüphanesinde, kullanıcıya bir dizi seçenek sunan bir açılır liste oluşturmanıza olanak tanır.

comboBox terimi, genellikle GUI (Grafiksel Kullanıcı Arayüzü) programlamasında kullanılan bir bileşeni ifade eder.

```
if self.comboBox.currentText()
```

Arama işlevindeki **.Text** yerine comboBoxlarda **.currentText** işlevi vardır. Halihazırda ki textten işlem yapar.



```
app = QApplication(sys.argv)
proje_arayuz = ProjeArayuz()
tercih_menu = TercihMenu()
basvurular_sayfasi = BasvurularSayfasi()
mulakatlar = Mulakatlar()
mentor_gorusme_sayfasi = MentorGorusmeSayfasi()

proje_arayuz.show()
sys.exit(app.exec())
```



Kod bloğu ile ilgili genel bilgiler

- ▶ Bu kod bloğu, PyQt ile bir masaüstü uygulamasının başlatılmasını ve pencere yönetimini sağlar.

Her bir satırın açıklaması:

- ▶ `QApplication(sys.argv):`
PyQt uygulamasını başlatır. `QApplication` sınıfı, PyQt uygulamalarının temelini oluşturur ve bu uygulamayı çalıştırmak için gereken kaynakları yönetir. `sys.argv`, komut satırı argümanlarını alır.
- ▶ `proje_arayuz = ProjeArayuz():`
`ProjeArayuz` adlı bir sınıf örneği oluşturur. Bu, uygulamanın ana penceresini temsil eder.
- ▶ `tercih_menu = TercihMenu():` `TercihMenu`
adlı bir sınıf örneği oluşturur. Bu, tercih menüsünü temsil eder.
- ▶ `basvurular_sayfasi = BasvurularSayfasi():`
`BasvurularSayfasi` adlı bir sınıf örneği oluşturur. Bu, başvurular sayfasını temsil eder.
- ▶ `mulakatlar = Mulakatlar():`
`Mulakatlar` adlı bir sınıf örneği oluşturur. Bu, mulakatlar sayfasını temsil eder.
- ▶ `mentor_gorusme_sayfasi = MentorGorusmeSayfasi():`
`MentorGorusmeSayfasi` adlı bir sınıf örneği oluşturur. Bu, mentor görüşmeleri sayfasını temsil eder.
- ▶ `proje_arayuz.show():`
Ana pencereyi görünür hale getirir. Bu, uygulama başladığında ana pencerenin görüntülenmesini sağlar.
- ▶ `sys.exit(app.exec()):`
Uygulamayı başlatır ve `sys.exit` ile programın doğru bir şekilde sona ermesini sağlar. `app.exec()` PyQt uygulamasını çalıştırır ve kullanıcı tarafından pencere kapatıldığında bu işlev tamamlanır.