**T.C.**

**DOKUZ EYLÜL ÜNİVERSİTESİ**

**İKTİSADİ VE İDARİ BİLİMLER FAKÜLTESİ**

**YÖNETİM BİLİŞİM SİSTEMLERİ BÖLÜMÜ**

**PROGRAMLAMA 2 DERSİ**

**“AL-SAT ROBOTU”**

**FİNAL ÖDEVİ**

**HAZIRLAYAN**

**BERFİN AY**

**DERS SORUMLUSU**

**DOÇ. DR. CAN AYDIN**

1. **Yazılım Tanımı**

Programın Adı: Al-Sat Robotu

Programlama Dilleri: Python

İşletim Sistemi: Microsoft Windows

Gerekli Kütüphaneler: bs4, pyqt5, mysql-connector, pandas, sklearn, lxml

Kaynak Kod Satır Sayısı: 222

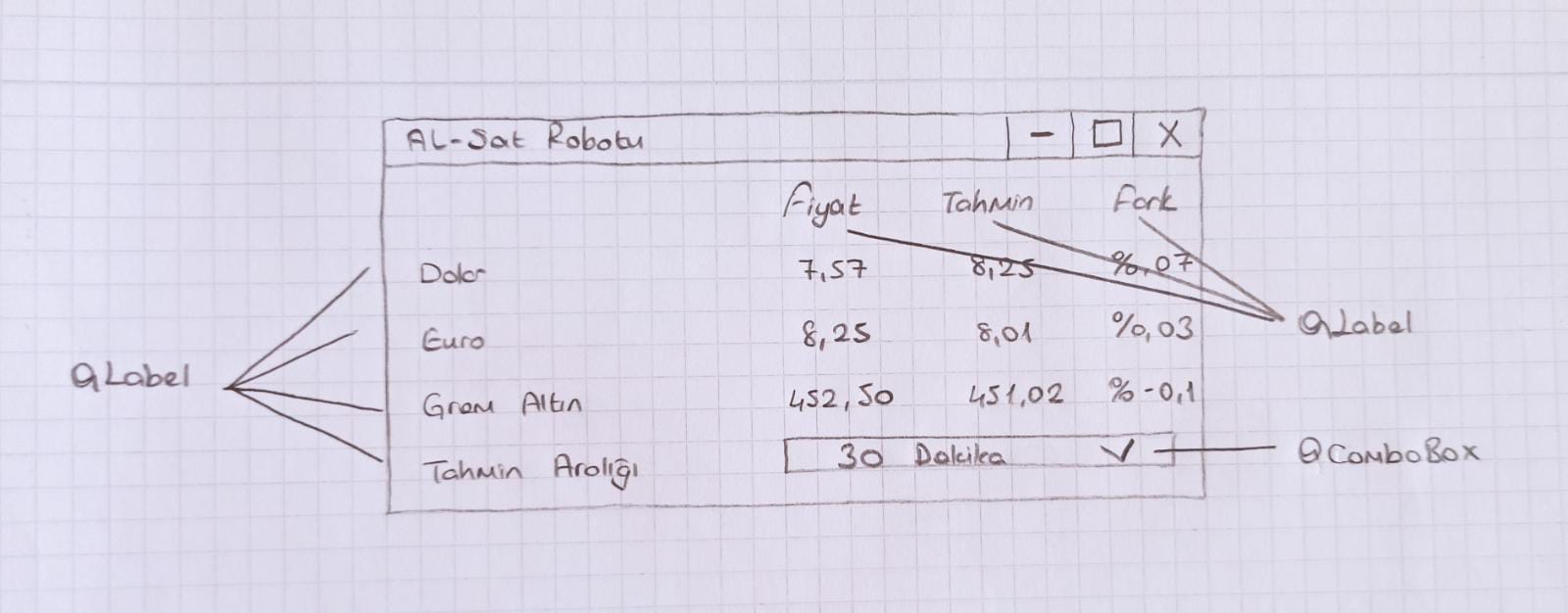
1. **Yazılım Tasarımı**
   1. **Problemin Tanımlanması**

Dünyada bir çok insan borsalardaki yükselme ve düşmelerden para kazanır, bunun asıl mantığı bir para birimi başka bir para birimi karşısında değer kazanırken paranızı ona geçirip paranızın değer kaybetmesini önlemek veya değer kazanmasını sağlamaktır. Fakat bu şekilde para kazanmak borsaların çok hassas olmasından ve hiçbir zaman artacak mı azalacak mı bilinememesinden dolayı oldukça zor ve risklidir. Çoğu kısa vadeli al-sat yapan insan gününü elindeki son verileri kontrol edip analiz ederek ve hangi para biriminin değer kazanacağını tahmin etmeye çalışarak geçirir. Buradaki çözülmeye çalışılan problem hangi paranın değer kazanacağını tahminleme işidir.

* 1. **Programın Amacı**

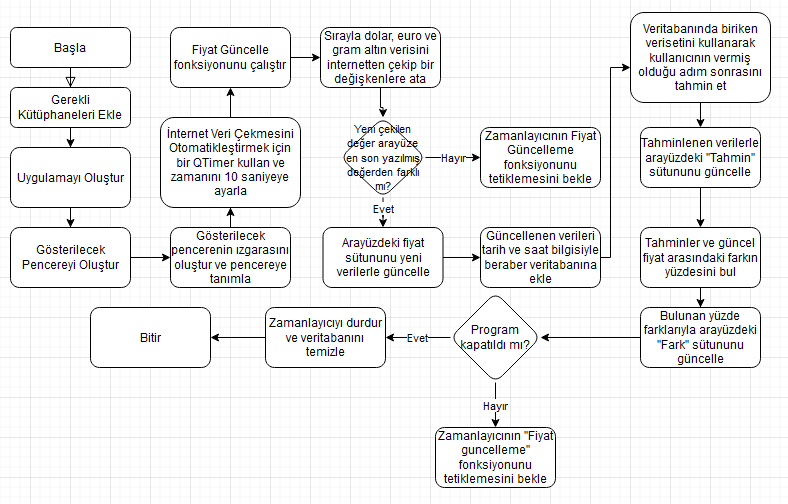
Bu program internet üzerinde gerçek zamanlı verileri çekip bir veritabanına kaydederek oluşan bu veriseti üzerinden yakın gelecekte “Dolar”, “Euro” ve “Gram Altın” paritelerinin “Türk Lirası” karşısında sahip oldukları değerlere göre çalıştıktan sonra veritabanında oluşturduğu verisetini kullanarak tahminlemesini yapacak ve anlık olarak kullanıcı arayüzünü güncelleyerek anlık fiyat bilgisini, seçilen zaman sonrası için yaptığı tahmini ve tahmin ile gerçek fiyat arasındaki fiyat farkını kullanıcının düşüş ve artışı takip edebilmesi amaçlı gösterecektir, bu sayede kullanıcının daha doğru yatırım yaparak daha doğru kararlar verebilmesini amaçlamıştır.

* 1. **Programın Tasarımı**
     1. **Arayüz Tasarımı**



* + 1. **Algoritma Tasarımı**

1. Başla  
2. Kütüphaneleri Ekle  
3. Uygulamayı Oluştur  
4. Gösterilecek pencereyi oluştur  
5. Gösterilecek pencerenin ızgarasını oluştur ve pencereye tanımla  
6. İnternet Veri Çekmesini Otomatikleştirmek için bir QTimer kullan ve zamanını 10 saniyeye ayarla  
7. Fiyat Güncelle fonksiyonunu çalıştır  
8. Sırayla dolar, euro ve gram altın verisini internetten çekip bir değişkenlere ata  
9. Yeni çekilen değer arayüze en son yazılmış değerden farklı mı?  
10. Hayır ise: Zamanlayıcının Fiyat Güncelleme fonksiyonunu tetiklemesini bekle  
11. Evet ise: Arayüzdeki fiyat sütununu yeni verilerle güncelle  
12. Güncellenen verileri tarih ve saat bilgisiyle beraber veritabanına ekle  
13. Veritabanında biriken verisetini kullanarak kullanıcının vermiş olduğu adım sonrasını tahmin et  
14. Tahminlenen verilerle arayüzdeki "Tahmin" sütununu güncelle  
15. Tahminler ve güncel fiyat arasındaki farkın yüzdesini bul  
16. Bulunan yüzde farklarıyla arayüzdeki "Fark" sütununu güncelle  
17. Program kapatıldı mı?  
18. Hayır ise: Zamanlayıcının "Fiyat guncelleme" fonksiyonunu tetiklemesini bekle  
19. Evet ise: Zamanlayıcıyı durdur ve veritabanını temizle  
20. Bitir

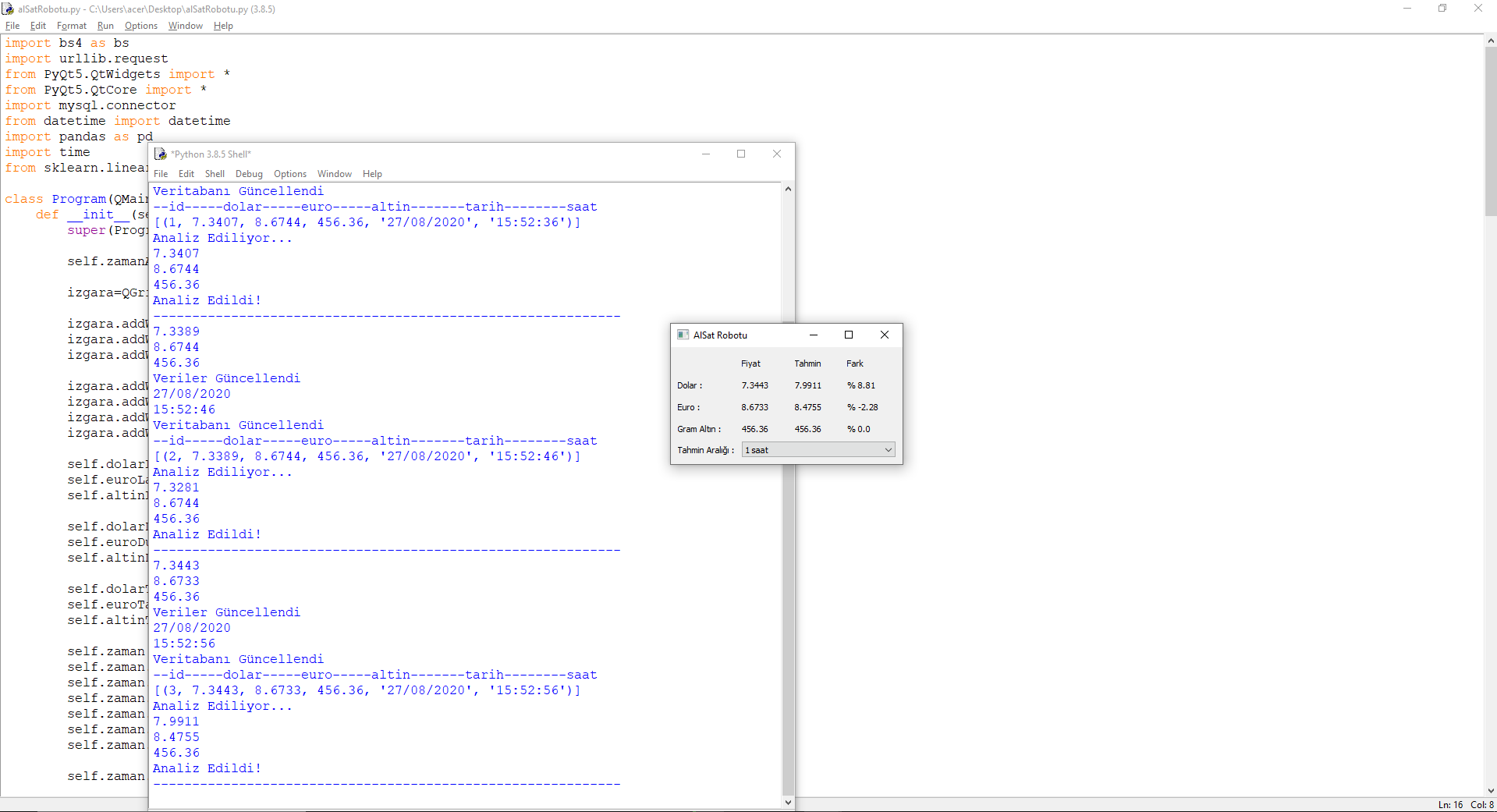


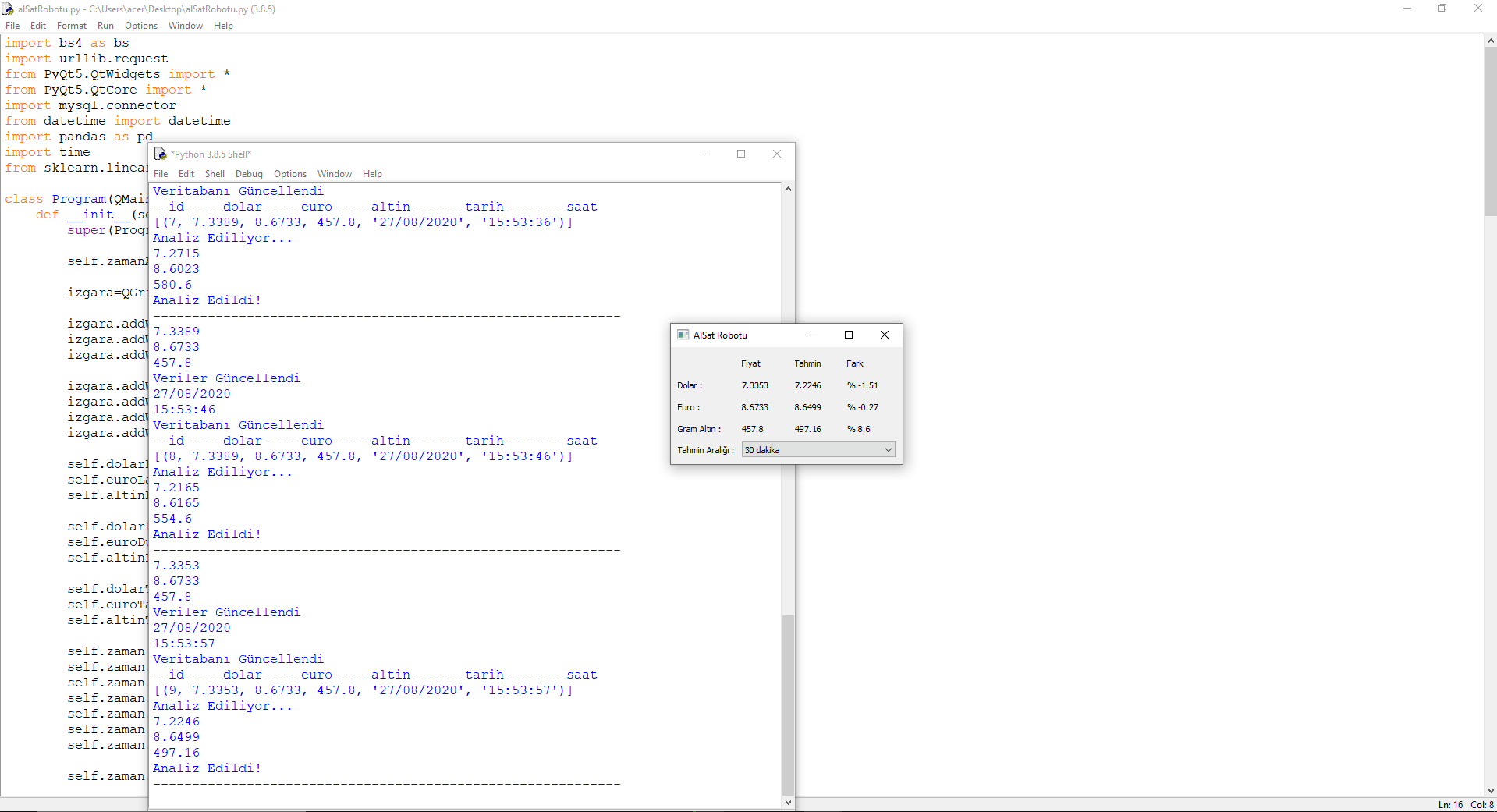
1. **Hatalar ve Eksiklikler**

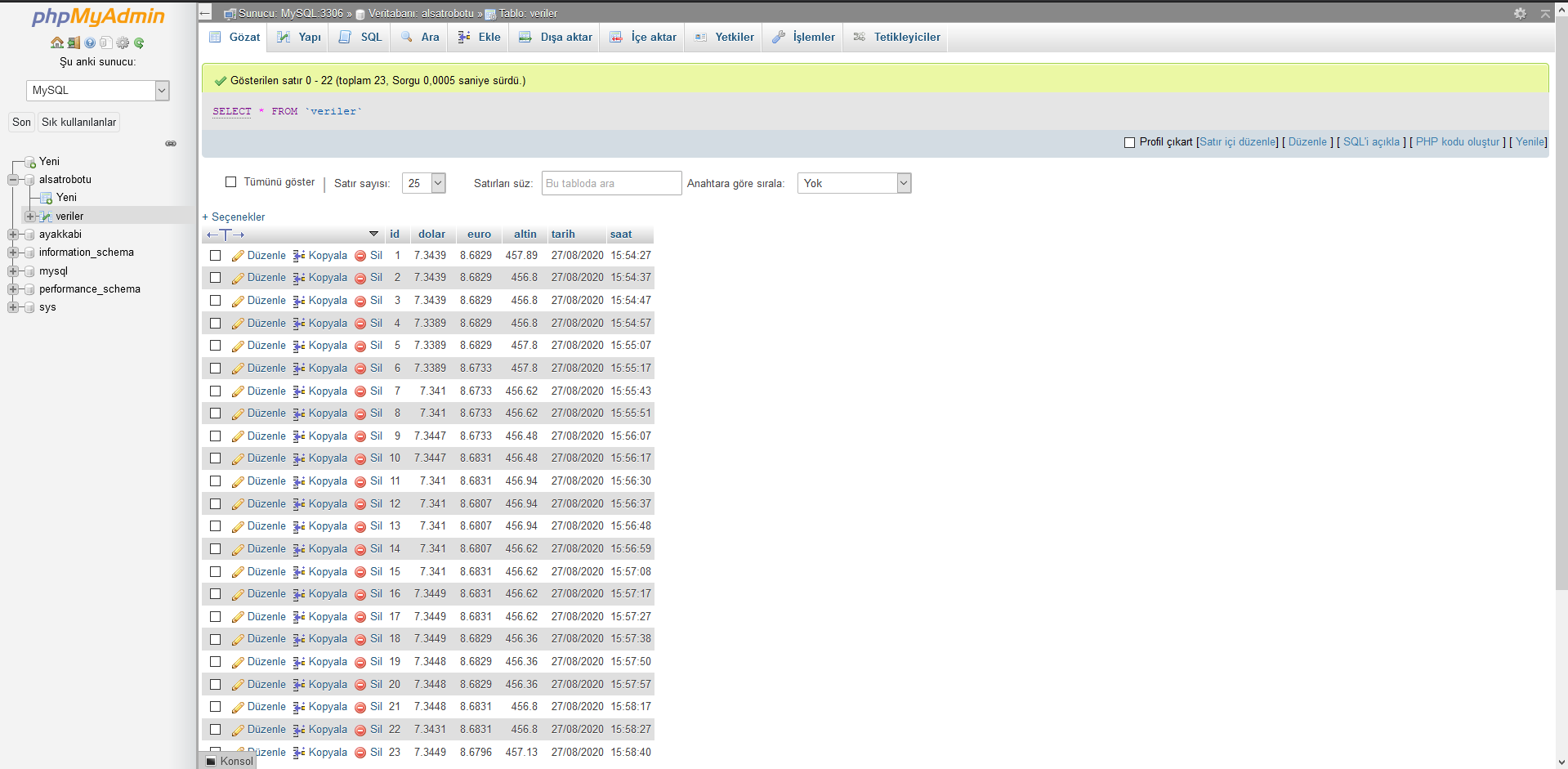
Program internetten güncel fiyat verilerini çekmeye çalışırken eğer verilerin çekildiği site cevap vermezse ve bağlantı kurulamazsa program çöküyor. Bunun olmaması için eğer veri çekilemezse fonksiyon hata vermek yerine tekrar bir 10 sn. daha beklemeye alınabilir.

Programdaki verisetinin ilk açılışta çok küçük olmasından dolayı program açık durduğu süre boyunca ne kadar fazla veri biriktirirse o kadar iyi tahminleme yapacaktır, ve tahmin aralığı ne kadar arttırılırsa o kadar tahmin doğruluğu düşer. İnternetten güncel verileri çekmeden önce daha önceki fiyat verileriyle de veritabanı beslenerek daha doğru ve tutarlı tahminle ve uzun zaman aralıkları eklenebilir. Aynı zamanda daha çok parite eklenerek program geliştirilebilir.

1. **Ekran Görüntüleri**







1. **Kodlar**

import bs4 as bs

import urllib.request

from PyQt5.QtWidgets import \*

from PyQt5.QtCore import \*

import mysql.connector

from datetime import datetime

import pandas as pd

import time

from sklearn.linear\_model import LinearRegression

class Program(QMainWindow):

def \_\_init\_\_(self,ebeveyn=None, \*args, \*\*kwargs):

super(Program,self).\_\_init\_\_(ebeveyn, \*args, \*\*kwargs)

self.zamanAraligi = 6

izgara=QGridLayout()

izgara.addWidget(QLabel("Fiyat"),0,1)

izgara.addWidget(QLabel("Tahmin"),0,2)

izgara.addWidget(QLabel("Fark"),0,3)

izgara.addWidget(QLabel("Dolar : "),1,0)

izgara.addWidget(QLabel("Euro : "),2,0)

izgara.addWidget(QLabel("Gram Altın : "),3,0)

izgara.addWidget(QLabel("Tahmin Aralığı : "),4,0)

self.dolarLabel = QLabel("Lütfen Güncelleyiniz")

self.euroLabel = QLabel("Lütfen Güncelleyiniz")

self.altinLabel = QLabel("Lütfen Güncelleyiniz")

self.dolarDurum = QLabel("---")

self.euroDurum = QLabel("---")

self.altinDurum = QLabel("---")

self.dolarTahminLabel = QLabel("Analiz Edilmedi")

self.euroTahminLabel = QLabel("Analiz Edilmedi")

self.altinTahminLabel = QLabel("Analiz Edilmedi")

self.zaman = QComboBox()

self.zaman.addItem("1 dakika")

self.zaman.addItem("5 dakika")

self.zaman.addItem("15 dakika")

self.zaman.addItem("30 dakika")

self.zaman.addItem("45 dakika")

self.zaman.addItem("1 saat")

self.zaman.currentTextChanged.connect(self.onComboBoxChanged)

izgara.addWidget(self.dolarLabel,1,1)

izgara.addWidget(self.euroLabel,2,1)

izgara.addWidget(self.altinLabel,3,1)

izgara.addWidget(self.dolarTahminLabel,1,2)

izgara.addWidget(self.euroTahminLabel,2,2)

izgara.addWidget(self.altinTahminLabel,3,2)

izgara.addWidget(self.dolarDurum,1,3)

izgara.addWidget(self.euroDurum,2,3)

izgara.addWidget(self.altinDurum,3,3)

izgara.addWidget(self.zaman,4,1,1,3)

w = QWidget()

w.setLayout(izgara)

self.setWindowTitle("AlSat Robotu")

self.resize(200,150)

self.setCentralWidget(w)

self.show()

self.timer = QTimer(self)

self.timer.setInterval(10000) # 60000 - 1 dakika

self.timer.timeout.connect(self.fiyatGuncelle)

self.timer.start()

self.fiyatGuncelle()

def fiyatGuncelle(self):

self.dolarFiyatiAl()

self.euroFiyatiAl()

self.altinFiyatiAl()

if self.dolarLabel.text() != self.dolar or self.euroLabel.text() != self.euro or self.altinLabel.text() != self.altin:

self.dolarLabel.setText(str(self.dolar))

self.euroLabel.setText(str(self.euro))

self.altinLabel.setText(str(self.altin))

print("Veriler Güncellendi")

self.veritabaninaEkle()

self.analizEt()

def analizEt(self):

print("Analiz Ediliyor...")

self.dolarTahminEt()

self.euroTahminEt()

self.altinTahminEt()

print("Analiz Edildi!")

print(30\*"--")

self.dolarTahminLabel.setText(str(self.dolarTahmin))

self.euroTahminLabel.setText(str(self.euroTahmin))

self.altinTahminLabel.setText(str(self.altinTahmin))

dolarDegisim =(self.dolarTahmin-self.dolar)/self.dolar\*100

euroDegisim =(self.euroTahmin-self.euro)/self.euro\*100

altinDegisim =(self.altinTahmin-self.altin)/self.altin\*100

self.dolarDurum.setText("% "+str(round(dolarDegisim,2)))

self.euroDurum.setText("% "+str(round(euroDegisim,2)))

self.altinDurum.setText("% "+str(round(altinDegisim,2)))

def dolarTahminEt(self):

baglanti=mysql.connector.connect(user='root',password='',host='127.0.0.1',database='alsatrobotu')

data = pd.read\_sql("Select \* from veriler", con=baglanti)

lr = LinearRegression()

lr.fit(data[["id"]],data.dolar)

sonID = int(data[-1:]['id'])

self.dolarTahmin= lr.predict([[sonID+self.zamanAraligi]])

self.dolarTahmin = float(self.dolarTahmin)

self.dolarTahmin= round(self.dolarTahmin,4)

print(self.dolarTahmin)

def euroTahminEt(self):

baglanti=mysql.connector.connect(user='root',password='',host='127.0.0.1',database='alsatrobotu')

data = pd.read\_sql("Select \* from veriler", con=baglanti)

lr = LinearRegression()

lr.fit(data[["id"]],data.euro)

sonID = int(data[-1:]['id'])

self.euroTahmin= lr.predict([[sonID+self.zamanAraligi]])

self.euroTahmin = float(self.euroTahmin)

self.euroTahmin= round(self.euroTahmin,4)

print(self.euroTahmin)

def altinTahminEt(self):

baglanti=mysql.connector.connect(user='root',password='',host='127.0.0.1',database='alsatrobotu')

data = pd.read\_sql("Select \* from veriler", con=baglanti)

lr = LinearRegression()

lr.fit(data[["id"]],data.altin)

sonID = int(data[-1:]['id'])

self.altinTahmin= lr.predict([[sonID+self.zamanAraligi]])

self.altinTahmin = float(self.altinTahmin)

self.altinTahmin= round(self.altinTahmin,2)

print(self.altinTahmin)

def dolarFiyatiAl(self):

kaynak = urllib.request.urlopen("http://bigpara.hurriyet.com.tr/doviz/dolar/").read()

sayfa = bs.BeautifulSoup(kaynak,"lxml")

self.dolar = sayfa.find\_all("span",{"class":"value up"})

self.dolar = self.dolar[0].string.strip()

self.dolar = self.dolar.replace(",",".")

self.dolar = float(self.dolar)

print(self.dolar)

def euroFiyatiAl(self):

kaynak = urllib.request.urlopen("http://bigpara.hurriyet.com.tr/doviz/euro/").read()

sayfa = bs.BeautifulSoup(kaynak,"lxml")

self.euro = sayfa.find\_all("span",{"class":"value up"})

self.euro = self.euro[0].string.strip()

self.euro = self.euro.replace(",",".")

self.euro = float(self.euro)

print(self.euro)

def altinFiyatiAl(self):

kaynak = urllib.request.urlopen("http://bigpara.hurriyet.com.tr/altin/gram-altin-fiyati/").read()

sayfa = bs.BeautifulSoup(kaynak,"lxml")

self.altin = sayfa.find\_all("span",{"class":"value up"})

self.altin = self.altin[0].string.strip()

self.altin = self.altin.replace(",",".")

self.altin = float(self.altin)

print(self.altin)

def veritabaninaEkle(self):

baglanti=mysql.connector.connect(user='root',password='',host='127.0.0.1',database='alsatrobotu')

isaretci=baglanti.cursor()

dolar = self.dolarLabel.text()

euro = self.euroLabel.text()

altin = self.altinLabel.text()

now = datetime.now()

tarih = now.strftime("%d/%m/%Y")

saat = now.strftime("%H:%M:%S")

print(tarih)

print(saat)

isaretci.execute('''INSERT INTO veriler(dolar, euro, altin, tarih, saat) VALUES ("%s","%s","%s","%s","%s")'''%(dolar, euro, altin, tarih, saat))

print("Veritabanı Güncellendi")

print("--id-----dolar-----euro-----altin-------tarih--------saat")

isaretci.execute('''SELECT \* FROM veriler ORDER BY id DESC LIMIT 0,1''')

sonuc = isaretci.fetchall()

print(sonuc)

isaretci.close()

baglanti.close()

def onComboBoxChanged(self):

if self.zaman.currentText() == "1 dakika":

self.zamanAraligi = 6

elif self.zaman.currentText() == "5 dakika":

self.zamanAraligi = 30

elif self.zaman.currentText() == "15 dakika":

self.zamanAraligi = 90

elif self.zaman.currentText() == "30 dakika":

self.zamanAraligi = 180

elif self.zaman.currentText() == "45 dakika":

self.zamanAraligi = 270

elif self.zaman.currentText() == "1 saat":

self.zamanAraligi = 360

def closeEvent(self, event):

print("kapandım")

self.timer.stop()

baglanti=mysql.connector.connect(user='root',password='',host='127.0.0.1',database='alSatRobotu')

isaretci=baglanti.cursor()

isaretci.execute('''TRUNCATE TABLE veriler''')

isaretci.close()

baglanti.close()

uyg=QApplication([])

pencere=Program()

uyg.exec\_()