GENERAL EXPLANATIONS

In this project, we selected a topic which is really getting in our lives recently. We are simulating one of the A.I supported customer services. Our application searches for pre-defined keywords in sentences. So basically if our customer writes a sentence which has 'çek' in it, application forwards to "withdrawing" function. Here is a table below which shows keywords for functions. We have also defined english keyboard versions for keywords like "yatır" and "yatir".

	KEYWORDS					
Para Çekme	çek	cek				
Para Yatırma	yatır	yatir				
Para Gönderme	gönder	gonder	havale	eft	transfer	
Fatura Ödeme	fatura	öde	ode			
Hesap Değiştirme	hesap	değiş	degis	başka	baska	
Çıkış	çık	cik	çıkış	cikis	kapat	iptal

We have searched for some kind of database systems to keep datas. We didn't want all variables in our code. So results were "Sqlite" and "MySql" but we don't need this kind of advanced systems. Also we found out it was a library called " tinyDB". This library works like simple databases but runs on local machine which is better for simple usage.

TinyDB creates a local json file. This file has no security protocols, editing json file with any editor is possible but as long as it is school project we can use it. Here is our json code.

```
1. {"_default": {"1": {"tcno": "12345678910", "isim": "Barış Demirezen", "password": "0
   000", "balance": "20000"},
2.  "2": {"tcno": "12345678911", "isim": "Ege Özfırıncı", "password": "0000", "balance"
   : "2000"},
3.   "3": {"tcno": "12345678912", "isim": "Hakan Canbiçen", "password": "0000", "balanc
   e": "100"}}}
```

Our object have four attributes. Those are ; tcno, isim, password, balance. We prefered storing them as string. There is three object is avaliable at the moment but adding more user is possible with editing json file.

More documents about tinyDB can be find in here:
https://tinydb.readthedocs.io/en/latest/index.html

Installing " pip " and " tinydb " is mandatory to run code.

To install " tindb " run the following command:

```
pip install tinydb
```

It's possible to move around documents by using hyperlinks that defined for functions in explainings. We didn't wanted to cut codes so we have skipped to new pages if needed.

CODE EXPLANATION

Connecting To Database

```
1.  db = TinyDB('db.json',ensure_ascii=False, encoding='utf-8')
2.  User = Query()
3.  login()
```

We are introducing our database file to code with parameters. Encoding utf-8 supports turkish characters in file so we must use it to keep turkish names. Also we are defining an object to use Query() function named “User”. After that we are calling our main function named “ login() ”.

Login Function

```
1.  def login():
2.      taken_pass="null"
3.      taken_tcno = input("Lütfen 11 haneli tc numaranızı giriniz\n")
4.      testint(taken_tcno)
5.      if len(taken_tcno)!=11 or int(taken_tcno) <0:
6.          print("Tc numaranızı hatalı girdiniz. Tekrar deneyiniz.\n")
7.          login()
8.      else:
9.          taken_pass= input("Lütfen şifrenizi giriniz\n")
10.         testint(taken_pass)
11.         control = db.search((User.tcno == taken_tcno) & (User.password == taken_pass
    ))
12.         if control ==[]:
13.             print("Hatalı Giriş\n")
14.             login()
15.         else:
16.             control = db.get(User.tcno == taken_tcno)
17.             isim = control.get('isim')
18.             tcno = control.get('tcno')
19.             print("Giriş Başarılı!\n")
20.             print("Merhaba, "+ isim + " hoşgeldiniz.\n")
21.             islemsecici(tcno)
```

In this part user is trying to login. We are controlling if “tcno” is 11 digit. Else, application asks for new tcno. After that application calls testint() function. It will be explained next. If “ tcno “ is valid we are moving to next step. Password is also goes through testint(). Application checks password for tcno. If it’s matched we are calling islemsecici(tcno). We are using “ tcno “ as parameter for all functions because of it’s unique for every user.

Test Integer Function

```
1.  def testint(tested_input):
2.      try:
3.          return int(tested_input)
4.      except ValueError:
5.          print("Lütfen sadece sayı giriniz.\n")
6.          login()
```

It’s the testint() function that we mentioned. This codeblock tests user input type. It returns input if input is number. Else gives error and starts login function again.

Process Selector Function

```
1.  def islemsecici(tcno):
2.      balance = balancecheck(tcno)
3.      print("Mevcut Bakiyeniz: " + balance + "\n")
4.      islem = input("Lütfen yapmak istediğiniz işlemi yazınız\n")
5.      islem = islem.lower()
6.      if 'cek' in islem or 'çek' in islem:
7.          paracekme(tcno)
8.      elif 'yatir' in islem or 'yatır' in islem:
9.          parayatirma(tcno)
10.     elif 'gonder' in islem or 'gönder' in islem or 'havale' in islem or 'eft' in isl
    em or 'transfer' in islem:
11.         paragonderme(tcno)
12.     elif 'fatura' in islem or 'öde' in islem or 'ode' in islem:
13.         if int(balance)>0:
14.             fatura(tcno)
15.         else:
16.             print("Bu işlem için yeterli bakiyeniz bulunmamakta.\n")
17.             islemsecici(tcno)
18.     elif 'hesap' in islem or 'değiş' in islem or 'degis' in islem or 'başka' in isle
    m or 'baska' in islem:
19.         print("Hesabınızdan çıkış yapılıyor.\nYeni hesap için ")
20.         login()
21.     elif 'çık' in islem or 'cik' in islem or 'çıkış' in islem or 'cikis' in islem or
     'kapat' in islem or 'iptal' in islem:
22.         quit()
23.     else:
24.         print("İstediğiniz islem anlasilamadi. Şunları deneyebilirsiniz:\npara çekme
    \npara yatırma \npara gönderme\nfatura öde\nhesap değiştir\nçıkış \n")
25.         islemsecici(tcno)
```

Actually this part is brain of our code. Code checks users balance with balancecheck(tcno) function. Since " tcno " is unique, it's possible to check values with this parameter. We will explain balancheck() function later.

User inputs a sentence or a word. In python there is no use case like other high-level languages so using if and if elses are obligatory. In line 5, islem.lower() changes all strings to lower case. You can clearly read commands to call functions about sentences. Also it was a table that shows commands in " General Explanations ". Every word calls functions but fatura statement checks if user balance is greater than zero if it is, moves to " fatura(tcno) " function.

Balance Check Function

```
1.  def balancecheck(tcno):
2.      balancechecked = db.get(User.tcno == tcno)
3.      balancechecked = balancechecked.get('balance')
4.      return balancechecked
```

Since we need to show balance many times in application, we used a function to use it in case needed. This function is also called with " tcno " parameter. After controlling database, returns balance of user back.

Withdraw Function

```
1.  def paracekme(tcno):
2.      bakiye = balancecheck(tcno)
3.      print("Mevcut bakiyeniz = " + bakiye)
4.      cekilecek = input("Lütfen çekmek istediğiniz miktarı giriniz.\nÇıkış için 'iptal
    ' yazınız.\n")
5.      if cekilecek.lower() == "iptal":
6.          islemsecici(tcno)
7.      else:
8.          try:
9.              int(cekilecek)
10.             if int(cekilecek) <= int(bakiye) and int(cekilecek) > 0:
11.                 yeni_bakiye = int(bakiye) - int(cekilecek)
12.                 db.update({'balance': str(yeni_bakiye)}, User.tcno == tcno)
13.                 print("İşlem başarılı.\nHesabınızdan "+ cekilecek + "tl çektiniz. Ye
    ni bakiyeniz: "+ str(yeni_bakiye))
14.                 islemsecici(tcno)
15.             else:
16.                 print("İşleminizi gerçekleştiremiyoruz. Lütfen tekrar deneyiniz!\n")

17.                 paracekme(tcno)
18.         except ValueError:
19.             print("Lütfen sadece  tam sayı giriniz.\n")
20.             paracekme(tcno)
```

Function calls for user balance with " balancecheck(tcno) ". User can abort process with writing " iptal ". This code checks if input is integer or " iptal " too. In case of entering integer, we are controlling if there is enough balance to withdraw from account. After valid inputs balance gets updated with new one and goes back to islemsecici(tcno).

Deposit Function

```
1.  def parayatirma(tcno):
2.      bakiye = balancecheck(tcno)
3.      print("Mevcut bakiyeniz = " + bakiye)
4.      yatirilacak = input("Lütfen yatırmak istediğiniz miktarı giriniz.\nÇıkış için 'i
    ptal' yazınız.\n")
5.      if yatirilacak.lower() == "iptal":
6.          islemsecici(tcno)
7.      else:
8.          try:
9.              int(yatirilacak)
10.             if int(yatirilacak) > 0:
11.                 yeni_bakiye = int(bakiye) + int(yatirilacak)
12.                 db.update({'balance': str(yeni_bakiye)}, User.tcno == tcno)
13.                 print("İşlem başarılı.\nHesabınıza "+ yatirilacak + "tl yatırdınız.
    Yeni bakiyeniz: "+ str(yeni_bakiye))
14.                 islemsecici(tcno)
15.             else:
16.                 print("İşleminizi gerçekleştiremiyoruz. Lütfen tekrar deneyiniz!\n")

17.                 parayatirma(tcno)
18.         except ValueError:
19.             print("Lütfen sadece tam sayı giriniz.\n")
20.             parayatirma(tcno)
```

This is our deposit function. It works almost like withdraw. Deposits must be bigger than zero and must be integer. It's only condition on this part. We are updating balance on deposit after valid inputs. We will continue next part in other page.

Money Transfer Function

```
1.  def paragonderme(tcno):
2.      bakiye = balancecheck(tcno)
3.      print("Mevcut bakiyeniz = " + bakiye)
4.      yatirilacak = input("Lütfen göndermek istediğiniz miktarı giriniz.\nÇıkış için '
    iptal' yazınız.\n")
5.      if yatirilacak.lower() == "iptal":
6.          islemsecici(tcno)
7.      else:
8.          try:
9.              int(yatirilacak)
10.             if int(bakiye)> int(yatirilacak) and int(yatirilacak)>0:
11.                 alici_tc = input ("Lütfen göndermek istediğiniz kişinin tc numarasın
    ı giriniz\n")
12.                 if len(alici_tc)==11:
13.                     try:
14.                         int(alici_tc)
15.                         alici_tc = str(alici_tc)
16.                         control = db.search(User.tcno == alici_tc)
17.                         if control ==[]:
18.                             print("Sistemde bu tc numarası bulunamadı.\n")
19.                             paragonderme(tcno)
20.                         else:
21.                             control = db.get(User.tcno == alici_tc)
22.                             alici_isim = control.get('isim')
23.                             alici_bakiye = control.get('balance')
24.                             onay = input(alici_isim + " adlı kullanıcıya "+ yatirila
    cak + " tl gönderiyi onaylıyor musunuz? evet/hayır\n")
25.                             if onay.lower() == "evet":
26.                                 yeni_bakiye = int(bakiye) - int(yatirilacak)
27.                                 db.update({'balance': str(yeni_bakiye)}, User.tcno
    == tcno)
28.                                 control = db.search(User.tcno == alici_tc)
29.                                 control = db.get(User.tcno == alici_tc)
30.                                 alici_bakiye = control.get('balance')
31.                                 alici_bakiye = int(alici_bakiye) + int(yatirilacak)
32.                                 db.update({'balance': str(alici_bakiye)}, User.tcno
     == alici_tc)
33.                                 print("İşlem başarılı.\n")
34.                                 islemsecici(tcno)
35.                             else:
36.                                 print("İşleminiz iptal edilmiştir.\n")
37.                                 islemsecici(tcno)
38.                     except ValueError:
39.                         print("Lütfen 11 karakterlik tc numarasını giriniz.\n")
40.                         paragonderme(tcno)
41.                 else:
42.                     print("Lütfen 11 karakterlik tc numarasını giriniz.\n")
43.                     paragonderme(tcno)
44.             else:
45.                 print("İşleminizi gerçekleştiremiyoruz. Lütfen tekrar deneyiniz!\n"
46.                 islemsecici(tcno)
47.         except ValueError:
48.             print("Lütfen sadece tam sayı giriniz.\n")
49.             paragonderme(tcno)
```

There are many of exception handlings in this part. Here is what code checks for errors.

- Money to be transfered must be integer and bigger than zero but less than balance
- Receiver " tc " must be integer and eleven digit.
- Receiver "tc " must be exist on database.

In line 28 we are refreshing our queried values to handle error if users send money to itself.

Bill Selector Function

```
1.  def fatura(tcno):
2.      bakiye = balancecheck(tcno)
3.      print("Mevcut bakiyeniz = " + bakiye)
4.      secim = input("Lütfen yatırmak istediğiniz faturanın numarasını seçiniz.\n1 - El
    ektrik\n2 - Su\n3 - Doğalgaz\nÇıkış için 'iptal' yazınız.\n")
5.      if secim == "iptal":
6.          islemsecici(tcno)
7.      else:
8.          if secim == "1":
9.               fatura_tur = "Elektrik"
10.         elif secim == "2":
11.             fatura_tur = "Su"
12.         elif secim == "3":
13.             fatura_tur = "Doğalgaz"
14.         else:
15.             print("Hatalı giriş yaptınız. Lütfen tekrar deneyiniz.\n")
16.             fatura(tcno)
17.         faturaode(tcno,fatura_tur)
```

This is our " pay the bills " function. This part only asks users to select a bill to pay. There are three options. If input is valid calls faturaode(tcno,fatura_tur) function.

Pay Selected Bill Function

```
1.  def faturaode(tcno,fatura_tur):
2.      bakiye=balancecheck(tcno)
3.      if int(bakiye) > 202:
4.          rastgele_tutar = random.randrange(10,201)
5.      else:
6.          rastgele_tutar = random.randrange(1,int(bakiye)+1)
7.      onay = input(str(rastgele_tutar) + " tl tutarındaki " + fatura_tur + " faturanız
    ı ödemeyi onaylıyor musunuz? evet/hayır\n")
8.      if onay.lower() == "evet":
9.           yeni_bakiye = int(bakiye) - int(rastgele_tutar)
10.          db.update({'balance': str(yeni_bakiye)}, User.tcno == tcno)
11.          print("Faturanız başarıyla yatırıldı.\n")
12.          islemsecici(tcno)
13.     else:
14.         print("İşleminiz iptal edilmiştir.\n")
15.         islemsecici(tcno)
```

We have parameters from fatura(tcno) functions. This is where this parameters gonna be processed. We added approving question, to pay bills. There is a random function to create random bills. If user has more than 202 liras, our bill prices will be maximum 200 liras. In case of user have less than 200 liras, our maximum bill price will be same as users balance. For example, if user have 150 liras, expected bill must be maximum 150 liras. User can abort process with writing " iptal ".