UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI
FAKULTET
DEPARTMAN ZA MATEMATIKU
I INFORMATIKU

Mileta Avramović 435/21

Banka/Bankomat

- seminarski rad iz predmeta Skript jezici-

Novi Sad, 2022.

Sadržaj

1. Uvod

Sve se čuva u fajl tipa “csv“(Comma-separated values).Korisitiće se rečnik zarad provere prisutnosti vrednosti (korisnika) i ključevi su torke.Aplikacija poseduje korisnički i admin pristup.Pri pokretanju aplikacije, praviće se folder u kojem se nalazi fajl koji će se koristiti kao baza podataka.Korisnik ima opciju da izabere da li će da se prijavi,registruje,ugasi nalog ili izađe.Prijavljivanjem korisnika pod nazivom “admin pristup“ dolazi se do panela sa admin opcijama.Administrator može da vidi sve informacije o aktivnim i ugašenim korisnicima poput: stanje računa,pin,poslednje aktivnosti.Administrator može da ugasi nalog koji želi i da vidi grafikon stanja računa u odnosu na korisnike.Takođe može da vidi broj aktivinih i ugašenih korisnika i da pretraži nalog (tako što upiše tekući račun) i vidi sve informacije o tom korisniku. Pokretanjem fajla “login.py“ se pokreće aplikacija.

2. Opis programa

2.1 Klasa "Login"

Ova klasa ima nekoliko funkcija koje će se koristiti zarad prijavljivanja i registracije u sistem aplikacije.Funkcije poput prijavljivanja,dodavanja i gasenja korisnika kao i admin blok.Te funkcije ćemo videti u nastavku seminarskog rada.

Na slici 1 je prikazan početak klase "Login",odnosno importovanje biblioteka i ostalih modula zarad funkcionisanja programa.Takođe tu je i deklarisana promeniljiva "clear" koja je funkcija uz pomoć koje se čisti terminal.

```
import time,datetime
import os, sys, csv, re
import random as rd
from getpass import getpass as gp


from Bankomat import bankomat
from Data import data,join
from t_r_gen import tekuci_racun_gen
from slanje_mejla import sendmail
from graph import *



#Cisti terminal
clear = ('cls' if os.name == 'nt' else 'clear')
```

Slika 1. Početak kalse "Login"

Definisanje "login_user" funkcije (Slika 2.), d je rečnik koji će se kasnije dodeljivati kao parametar funkcije.Takođe prilikom pokretanja funkcije dolazi do kreiranja foldera i fajla koji će se koristiti kao baza podataka.Korisnik,odnosno "user", će unositi u terminal jednu od ponuđenih opcija.Ukoliko unese neku od opcija koje nisu ponuđene, odnosno podatak nekog drugog tipa,pokazaće mu se da je došlo do greške pri biranju i da ponovo izabere.

```
def login_user():

    d = data()

    user = input("Izaberite jedno : \n1. Prijavljivanje \n2. Registracija naloga \n3. Gasenje naloga  \n0. Izlaz \n")
    os.system(clear)

    if not str(user).isdigit():
        print ("Molimo Vas izaberite ponovo,greska pri biranju.")
        return login_user()
```

Slika 2. Definisanje funkcije login i ponuđeni izbori za korisnika

Unošenjem jednog od ponuđenih brojeva korisnik može da pozove funkciju koja će se izvršavati.Ako unese "1" pokrenuće se funkcija "login" koja služi za prijavljivanje, unošenjem "2" pokreće se funkcija "novi_nalog" koja služi za registraciju. Unošenjem "3" korisnik pokreće funkciju za gašenje naloga i na kraju unošenjem "0" korisnik gasi aplikaciju.Ukoliko

korisnik unese neki parametar koji nije jedan od ponuđenih (1,2,3,0) prikazaće se poruka da je došlo do greške pri biranju i da ponovo izabere. (Slika 3.)

```
#Prijavljivanje
if int(user) == 1:
    os.system(clear)
    login(d)

#Registracija naloga
elif int(user) == 2:
    os.system(clear)
    novi_nalog()
#Gasenje naloga
elif int(user) == 3:
    os.system(clear)
    gasenje_naloga()

#Izlaz
elif int(user) == 0:
    print ("Dovidjenja!")


#Greska pri unosu(ni jedan broj od navedenih) ponovno pokretanje programa
else:
    print ("Greska pri biranju,molimo Vas izaberite ponovo. '",user,"'")
    return login_user()

return
```

Slika 3. Pozivanje funkcija na uneti parametar

Definisanje funkcije „login(d)“,unose se ime i prezime.Promenljiva „entry“ se korisit kao vrednost koja će se povećavati svaki put kada korisnik pogreši pin. Ako u fajlu,u kojem se čuvaju korisnici, nema ni jednog korisnika izaći će poruka da je potrebno prvo napraviti nalog i pokrenuće funkciju za pravljenje naloga.Ako se ime i prezime sastoje od karaktera koji su mala ili velika slova ili razmak onda će program nastaviti da radi,u suprotnom izbaciće grešku.

Proverava se da li je tekući račun dodeljen unetom imenu,ako nije prikazaće se poruka da ne postoji korisnik ili ako tekući račun počinje “#” pokazaće se da je taj nalog ugašen. (Slika 4.)

```
def login(d):
    os.system(clear)
    ime_prezime = input("Prijavljivanje\n Unesite ime i prezime : ")
    entry = 0
    if (d == None):
        os.system(clear)
        print ("Molimo Vas prvo napravite nalog!")
        return novi_nalog()

    for a in ime_prezime:
        if ((ord(a) >= 65) and (ord(a) <= 90)) or ((ord(a) >= 97) and (ord(a) <= 122)) or (ord(a) == 32):
            continue
        else:
            os.system(clear)
            print ("Mozete koristiti mala ili velika slova i razmak!")
            return login_user()

    for parametar in d.keys():
        if ime_prezime.lower() in d[parametar]:
            t_r = parametar
            break
        else:
            t_r = None

    if t_r == None:
        os.system(clear)
        print("Korisnik ne postoji/Pogresno ime!")
        return login_user()

    elif t_r.startswith('#'):
        os.system(clear)
        print("Nalog je De-Aktiviran")
        return login_user()
```

Slika 4. Definisanje i prvi deo prijavljivanja korisnika

Ukoliko korisnik unese “admin pristup” program će vratiti funkciju “admin_block(t_r)” u kojoj imaju opcije koje poseduje administrator.Ukoliko unos nije “admin pristup” program će tražiti od korisnika da unese pin.Korisnik ima tri pokušaja, ukoliko sve lepo unese vratiće mu se funkcija “bankomat”.Ako prekorači dozvoljeni broj pokušaja unosa pina ili pogreši prilikom unosa imena, neće se uspešno prijaviti na nalog. (Slika 5.)

```
#Admin pristup
elif (ime_prezime.lower() == 'admin pristup'):
    return admin_block(t_r)
#korisnici block
elif not (ime_prezime.lower() == 'admin pristup'):
    while int(entry) != 3:
        print("Broj preostalih pokusaja :",(3-entry))
        pin = str(gp("Unesite pin : "))

        if pin == d[t_r][1]:
            Pin = pin
            Stanje = d[t_r][2]
            E_mail_adresa = d[t_r][4]
            os.system(clear)
            return bankomat(ime_prezime,Stanje,Pin,t_r,E_mail_adresa)

        else:
            entry += 1
            os.system(clear)
            print ("Pogresan Pin!")
    os.system(clear)
    print ("Neuspelo prijavljivanje\n")
    return login_user()

else:
    os.system(clear)
    print ("Korisnik ne postoji!")
    return login_user()
```

Slika 5. Provera da li se korisnik ili administrator

Funkcija "novi_nalog()" je funkcija uz pomoć koje se korisnik registruje.Poziva se funkcija "join()" koja vodi do fajla u kojem se skladište podaci.Promenljiva "ime_prezime1" je ime korisnika, "ime_prezime2" je prezime korisnika dok je "ime_prezime" promenljiva koju čine ime i prezime.Ako korisnik unese isto ime i prezime ili unese nešto što nije slovo prikazaće se greška i moraće da ponovi unos."auto_gen_pin" je pin koji se automatski generiše od strane aplikacije.Uz pomoć "tekuci_racun_gen" generiše se tekući račun za uneto ime i prezime. (Slika 6.)

```
def novi_nalog():
    import time,datetime

    filename = join()
    ime_prezime1 = input("Registracija korisnika\n Ime : ")
    os.system(clear)
    ime_prezime2 = input("Prezime: ")

    if (ime_prezime1.isalpha() == False) or (ime_prezime2.isalpha() == False) or (ime_prezime1 == ime_prezime2):
        os.system(clear)
        print ("Pogresan unos!")
        return novi_nalog()

    #automatski generisan pin
    auto_gen_pin = rd.randint(1000,9999)
    os.system(clear)
    ime_prezime = (ime_prezime1.lower())+' '+(ime_prezime2.lower())
    t_r = tekuci_racun_gen(ime_prezime)
```

Slika 6. Početak funkcije "novi_nalog"

Ako želi korisnik može da unese email adresu.Provera da li je dobro uneta adresa.Poruka ako nije uneta a ako je uneta dodelju se promeljivoj.(Slika 7)

```
E_mail_adresa = input("Unesite e-mail adresu : ")
if not re.match(r"^[A-Za-z0-9\.\+_-]+@[A-Za-z0-9\._-]+\.[a-zA-Z]*$", E_mail_adresa):
    os.system(clear)
    E_mail_adresa = "Nije unet email"
    print("Mail adresa ne postoji")

else:
    E_mail_adresa = E_mail_adresa
```

Slika 7. Unos email adrese

Korisniku se prikazuje pin koji program dodeljuje i odlučuje da li želi taj pin ili neki svoj.Ako želi automatski generisan pin ostaje mu još da potvrdi unete podatke. (Slika 8)

```
print("Pin : ",auto_gen_pin)
confirm = input("Da li Zelite da koristite ovaj pin ? \n1. Da \n2. Ne \n")

if (confirm == '1'):
    os.system(clear)

    print ("Ime i Prezime :",ime_prezime1+' '+ime_prezime2,"\nTekuci Racun :",t_r,"\nPin :",auto_gen_pin)
    confirm = input("Potvrdi. \n1. Da \n2. Ne \n")
```

Slika 8. Prikaz pina dodeljenog od strane programa i potvrda korisnika

Ukoliko korisnik potvrdi,počinje upis u fajl.Koristeći funkcije iz metode csv.Nakon uspešnog pravljenja naloga,ukoliko je korisnik uneo email , stiže mail sa informacijama o nalogu (tekući račun i pin).(Slika 9)

```
if (confirm == '1'):
    os.system(clear)
    with open(filename, "a",newline="") as wr:
        ime_prezime = (ime_prezime)
        new = [t_r,ime_prezime,auto_gen_pin,'0.0',time.strftime('%d-%b-%Y at %I:%M %p'),E_mail_adresa]

        w = csv.writer(wr)
        w.writerow(new)
        wr.close()
        MSG = "Informacije o nalogu"+"\n\n"+"Tekuci Racun : "+t_r +"\n" "Pin:" + str(auto_gen_pin)
        vr = sendmail(E_mail_adresa, MSG)
        vr = E_mail_adresa
        os.system(clear)
        if not (vr == True): print(vr)
        print ("Uspesno ste napravili nalog! \n")
        return login_user()
```

Slika 9. Upis u bazu podataka i slanje mail-a sa informacijama o nalogu

Ukoliko korisnik ne želi da napravi nalog,nalog se neće napraviti.Ako unese nešto što nije (1 ili 2) ispisaće se greška i moraće ponovo da se registruje. (Slika 10)

```
elif (confirm == '2'):
    os.system(clear)
    print ("Nalog nije napravljen!")
    return login_user()

else:
    os.system(clear)
    print ("Nije dobar unos,pokusajte ponovo!")
    return novi_nalog()
```

Slika 10. Odbijanje pravljenja naloga

Korisnik može da napiše pin koji želi.Potrebno je da unese i potvrdi željeni pin.Ima tri pokušaja i pin može da budu bilo koje četiri cifre.(Slika 11)

Ako korisnik unese sve kako treba ispis,potvrda i slanje email-a se isto realizuju kao u slikama 8 i 9 samo sa željenim pinom.

```
else:
    os.system(clear)
    pin_count = 0
    print("Zeljeni Pin....")
    while pin_count != 3:
        print("Preostalo pokusaja :",(3-pin_count))
        pin = str(gp ("Unesite Pin: "))
        os.system(clear)

        if (len(pin) == 4) and (pin.isdigit() == True):
            os.system(clear)
            confirm_pin = str(gp ("Potvrdite Pin : "))

            if pin == confirm_pin:
```

Slika 11. Željeni pin

Ukoliko se potvrdni i unešen pin ne poklapaju,brojač se poveća za jedan i tako do tri,ukoliko se ne upišu isti pinovi nalog se neće napraviti.

```
            else:
                print ("Pin-ovi nisu isti!")
                pin_count +=1

else:
    pin_count = pin_count
    os.system(clear)
    print ("Pogresan pin")
```

Slika 12. Brojač i greška

Definisanje funkcije “gašenje_naloga()”,pozivaju se “data()” i “join()” zbog promene u bazi podataka.Unosi se tekući račun koji želimo da ugasimo,ako se nalazi u bazi,upisuje se pin i gasi se nalog. Ako se unešeni pin poklapa sa pinom iz baze,izaći će poruka sa imenom naloga koji se gasi.Skoro kao i uvek biće potrebna potvrda. (Slika 13)

```
def gasenje_naloga():
    d = data()
    filename = join()

    os.system(clear)
    t_r = input("Gasenje naloga\nUnesite Tekuci Racun : ")

    if t_r in d.keys():
        acc_pin = str(gp("Unesite Pin: "))

        if acc_pin == d[t_r][1]:
            os.system(clear)
            print ("Gasenje naloga :",d[t_r][0])
            confirm = input("Potvrdi. \n1. Da \n2. Ne \n")
```

Slika 13. Definisanje funkcije i unos tekućeg računa za gašenje

Prilikom gašenja naloga,sve promene koje su upisane se brišu i ostaje poslednja promena.Razlika između aktivnog i ugašenog naloga je “#“ koja se nalazi pre tekućeg računa ugašenog naloga. (Slika 14) Predstavlja proces prolaska kroz bazu i dodavanje “#“.

```
if (confirm == '1') :
    os.system(clear)
    d[('#'+t_r)] = d.pop(t_r)
    with open(filename,"w",newline="") as rd:
        r = csv.writer(rd)
        r.writerow([])
        rd.close()
    with open(filename,"a",newline="") as ow:
        for parametar in d.keys():
            parametri = (d[parametar][0])
            over_write = [parametar,parametri,str(d[parametar][1]),str(d[parametar][2]),str(d[parametar][3]),str(d[parametar][4])]
            o = csv.writer(ow)
            o.writerow(over_write)
        ow.close()
        print ("Gasenje naloga je uspesno \n")
        return login_user()
```

Slika 14. Proces gašenja naloga

(Slika 15) U slučaju da korisnik ne želi da ugasi nalog,nalog je već ugašen ili dođe do neke greške.

```
        elif (confirm == '2'):
            os.system(clear)
            print ("Gasenje naloga je neuspesno")
            return login_user()

        else:
            os.system(clear)
            print ("Gasenje naloga je neuspesno")
            return login_user()

    else:
        os.system(clear)
        print ("Pin-ovi nisu isti!")
        return login_user()

elif ("#"+t_r) in d.keys():
    os.system(clear)
    print("Nalog je već ugasen!")
    return login_user()

else:
    os.system(clear)
    print ("Doslo je do greske,pokusajte ponovo.")
    return login_user()
```

Slika 15. Opcije koje mogu da se dese prilikom unosa

```
def admin_block(t_r):
    d = data()
    pin = str(gp("Unesite Pin: "))
    if pin == d[t_r][1]:
        del d[t_r] #da se admin nalog ne prikazuje
        os.system(clear)
        print (time.strftime('Datum:%d-%b-%Y \nVreme:%I:%M %p'))
        print ("::: Dobrodosli(admin panel) :::\n\n:: Izaberite nesto od ponudjenih opcija ::")
        ad = input("1. Broj korisnika \n2. Broj aktivnih naloga \n3. informacije o aktivni korisnicima
```

Slika 16. početak funkcije “admin_block(t_r)”

Definiše se funkcija “admin_block(t_r)”,opet se poziva “data()” i administrator ima nekoliko opcija,tačnije osam.Prve tri se vide na (Slika 16) osatle su: “4” aktivnost korisnika, “5” pronađi nalog, “6” ugasite nalog, “7” grafikon i “0” izlaz.

Dokle god se ne unese “0” program radi.Unošenjem “1” prikazuju se aktivni i ugašeni korisnici. Postoje dva brojača jedan za aktivne drugi za ugašene naloge. (Slika 17)

```
while ad != '0':

    if ad == '1':
        os.system(clear)
        aktivni_korisnici, ugaseni_korisnici = 0, 0
        for korisnici in d.keys():
            if not korisnici.startswith('#'):
                aktivni_korisnici += 1
            else:
                ugaseni_korisnici += 1

        print(":: Korisnici ::")
        print("Aktivni korisnici :",aktivni_korisnici)
        print("Ugaseni korisnici :",ugaseni_korisnici,'\n')
```

Slika 17. Broj aktivnih i ugašenih korisnika

Unošenjem “2” prikazuje se broj aktivnih naloga.(Slika 18)

```
elif ad == '2':
    os.system(clear)
    aktivni_korisnici = 0
    print (":: Broj aktivnih naloga ::")
    for korisnici in d.keys():
        if not korisnici.startswith('#'):
            aktivni_korisnici += 1
            print ("Aktivni korisnici",aktivni_korisnici,':',d[korisnici][0])
    print('\n')
```

Slika 18. Broj aktivnih naloga

Unošenjem “3” prikazuju se tekući račun,ime i prezime, pin i stanje računa aktivnih korisnika.(Slika 19)

```
elif ad == '3':
    os.system(clear)
    print (":: O korisnicima ::")
    for korisnik_info in d.keys():
        if not korisnik_info.startswith('#'):
            print ("Tekuci Racun :",korisnik_info,"Ime i Prezime =",d[korisnik_info][0],", Pin :",d[korisnik_info][1],
    print('\n')
```

Slika 19. O korisnicima

Unošenjem "4" prikazuju se aktivnosti korsinika.Tekući račun,koji korisnik i kada se poslednji put prijavio na nalog. (Slika 20)

```
elif ad == '4':
    os.system(clear)
    print (":: Aktivnosti korisnika ::")
    for korisnik_info in d.keys():
        if not korisnik_info.startswith('#'):
            print ("Tekuci Racun :",korisnik_info,"Korisnik:",d[korisnik_info][0],"je prethodno bio prijavljen",d[korisnik_info][3])
    print('\n')
```

Slika 20. Aktivnosti korisnika

Unošenjem "5" vrši se pretraga naloga uz pomoć tekućeg računa.Unese se tekući račun i ako nalog postoji prikazaće se sve informacije o nalogu.Status naloga,ime korisnika,pin i stanje računa.(Slika 21)

```
elif ad == '5':
    os.system(clear)
    t_r_find = str(input('Unesite Tekuci Racun(12 cifara) : '))

    if t_r_find in d.keys():
        print("Status naloga : Aktiviran")
        print("Ime i Prezime :",d[t_r_find][0])
        print("Pin :",d[t_r_find][1])
        print("Stanje Racuna :","{:,}".format(d[t_r_find][2]))

    elif ("#"+t_r_find) in d.keys():
        print("Status naloga : Ugasen")
        print("Ime i Prezime :",d[("#"+t_r_find)][0])
        print("Pin :",d[("#"+t_r_find)][1])
        print("Stanje Racuna :","{:,}".format(d[("#"+t_r_find)][2]))

    else:
        os.system(clear)
        print("Nalog nije pronadjen.")
```

Slika 21. Pretraga naloga

Unošenjem "6" ili "7" vrši se poziv funkcije za gašenje naloga ili prikaz grafikona.(Slika 22)

```
elif ad == '6':
    os.system(clear)
    return gasenje_naloga()
elif ad == '7':
    os.system(clear)
    grafikon()
    return admin_block(t_r)
```

Slika 22. Gašenje naloga i grafikon

Na kraju se zatvara petlja i zatvaraju se uslovi grananja.Takođe ukoliko dođe do greške biće ispis. (Slika 23)

```
            ad = input("1. Broj korisnika \n2. Broj aktivnih naloga \n3. informacije o aktivnim korisnicima
        os.system(clear)
        return login_user()

    else:
        os.system(clear)
        return login_user()

try:
    os.system(clear)
    login_user()

except Exception as exc:
    os.system(clear)
    print ("Doslo je do greske: %s" %exc)
    print ("Izvinjavamo se.\nDovidjenja!")
```

Slika 23. Kraj "login" klase

2.2 Klasa "Bankomat"

Nakon uspešnog prijavljivanja ili registrovanja pokreće se funkcija "bankomat()". Funkcija je zadužena za korišćenje naloga.Neke funkcije su identične kao funkcije iz klase "login".

Počinje uvozom modula i funkcija iz klasa.Takođe se deklariše globalna promenljiva "stanje" zato što je to početna vrednost količine novca koliko ima svaki nalog,kasnije se na nju dodaje "Stanje" koje je promenljiva koja se odnosi na promene količine novca na nalogu. (Slika 24)

```
from getpass import getpass as gp
import os, csv

from Data import join,data
from slanje_mejla import sendmail
import time,datetime

stanje = 0.0
```

Slika 24. početak klase "Bankomat"

Definiše se funkcija “bankomat“ sa argumentima: ime i prezime,stanje,pin,tekući račun i email adresu ako je uneta.Povezuje se sa bazom podataka.Poziva se globalna promenljiva i njoj se dodeljuje pormenjeno stanje računa.

Korisnik ima opciju da izabere: “1“ prikaz stanja, “2“ prikaz tekućeg računa,“3“ uplaćivanje novca(depozit), “4“ podizanje novca, “5“ prenos novca,“6“ promena pina i “0“ izlaz.(Slika 25)

```
def bankomat(ime_prezime,Stanje,Pin,t_r,adresa):
    filename = join()
    clear = ('cls' if os.name == 'nt' else 'clear')
    print(("Postovani"),(ime_prezime.upper())+("!"))
    print("Dobrodosli \n")


    #izberi
    global stanje
    stanje += Stanje
    Opr = input("Izaberite jedno : \n1. Provera stanja \n2. Provera tekuceg racuna \n3. Depozit \n4. Podizanje novca \n5. Prenos novca
    os.system(clear)
```

Slika 25. Definisanje funkcije “bankomat“

Ovaj kod je identičan kodu iz klase “login“,ukoliko korisnik unese nešto sto nisu cifre od 0 do 6 izbaciće mu grešku,ukoliko unese neki od ponuđenih brojeva pozivaće se funkcije.(Slika 26)

```
if not Opr.isdigit():
    Opr = 7

while int(Opr) != 0:

    if int(Opr) == 1:
        os.system(clear)
        print (":: Moje Stanje : ","{:,} ::".format(stanje),"\n")

    elif int(Opr) == 2:
        os.system(clear)
        print(":: Tekuci Racun =",t_r,":: \n")

    #Depozit
    elif int(Opr) == 3:
        os.system(clear)
        Depozit(stanje, adresa)
    #Podizanje novca
    elif int(Opr) == 4:
        os.system(clear)
        podizanjenovca(stanje, adresa)
```

Slika 26. Pozivanje funkcija

Prenos novca se neće izvršiti ukoliko korisnik nema dovoljno novca ili unese isti tekući račun.Promena pina identična je kao kod klase "login".(Slika 27)

```
#Prenos novca
elif int(Opr) == 5:
    os.system(clear)
    if stanje < 0.0:
        print ("Nemate dovoljno novca za tranzakciju")

    else:
        tekuci_racun = input('Unesite tekuci racun: ')
        if (tekuci_racun == t_r):
            os.system(clear)
            print("Prenos nije moguc,to je Vas tekuci racun")
        else:
            kolicina_novca = prenos_novca(tekuci_racun, stanje, t_r, adresa)
            stanje -= float(kolicina_novca)


#Promena Pina
elif int(Opr) == 6:
    os.system(clear)
    Pin = promena_pina(Pin, adresa)


else:
    os.system(clear)
    print (":: Doslo je do greske,molimo pokusajte ponovo ::")
```

Slika 27. Prenos novca,promena pina i preška pri unosu

Zatvaranje beskonačne petlje i upis u bazu podataka.Upisuje se na poslednje mesto u bazi podataka i upisuje se svako koršišćenje bankomata iz razloga da bi admin mogao da zna kada je se poslednji put nalog koristio. (Slika 28)

```
    Opr = input(":: Izaberite jedno : \n1. Provera stanja \n2. Provera tekuceg racuna \n3. Depoz
    if not Opr.isdigit():
        Opr = 7
        os.system(clear)

os.system(clear)
print ("::: Hvala Vam na koriscenju! :::\n::: Srdacan Pozdrav! :::")


with open(filename,'a+',newline="") as ap: #a+ zapisuje na kraju
    re_new = [t_r,ime_prezime,str(Pin),str(stanje),time.strftime('%d-%b-%Y at %I:%M %p'),adresa]
    w = csv.writer(ap)
    w.writerow(re_new)
    ap.close()
return
```

Slika 28. Kraj funkcije "bankomat"

U funkciji “Depozit“ korisnik unosi količinu novca koju želi da uplati.Takođe se poziva globalna promenljiva. (Slika 29)

```
def Depozit(Stanje, adresa):
    clear = ('cls' if os.name == 'nt' else 'clear')
    global stanje
    print(":: Depozit ::")
    try:
        kolicina_uplate = input("Unesite zeljenu kolicinu u dinarima: ")
```

Slika 29. Definisanje funkcije “Depozit“

Ukoliko je željena uplata veća ili jednaka nuli vrši se provera da li je suma veća od broja sa 14 cifara ukoliko jeste prikazaće se greška.Ukoliko je sve kako treba stanje će se povećati za količinu novca koju ste vi odredili,poslaće se email o uspešno obavljenoj uplati. (Slika 30)

```
if float(kolicina_uplate) >= 0.0:
    #ne sme preko 14 cifri
    if (len(kolicina_uplate) > 14) or ((len(str(float(kolicina_uplate)+stanje))) > 14):
        os.system(clear)
        print (':: Ogranicenje iznosa je prekoraceno! ::')
        return

    else:
        stanje += float(kolicina_uplate)
        os.system(clear)
        MSG = "Uspesno ste uplatili novac "+str(kolicina_uplate)+"\n\nVase stanje: "+str(stanje)
        msg = sendmail(adresa, MSG)
        os.system(clear)
        if not (msg == True): print(msg)
        print(":: Uspesno ste uplatili novac",kolicina_uplate,"::",'\n')
        return
```

Slika 30. Depozit

Kao i uvek,proveravaju se različiti slučajevi poput pogrešnog unosa i završava se funkcija “Depozit“.(Slika 31)

```
    elif float(kolicina_uplate) < 0.0:
        os.system(clear)
        #Ako unese negativno
        print (":: Greska pri unosu. ::\n")
        return Depozit(stanje, adresa)

    else:
        os.system(clear)
        print (":: Greska pri unosu. ::\n")
        return Depozit(stanje, adresa)

except ValueError:
    os.system(clear)
    print (":: Greska pri unosu. ::\n")
    return Depozit(stanje, adresa)
```

Slika 31. Nastavak provera i kraj funkcije

Funkcija uz pomoć koje se podiže novac je slična kao funkcija "Depozit".Ralika je u početnoj proveri.Kod depozita je bila provera da stanje ne pređe broj od 14 cifara a ovde je provera da li ima novca na računu.Ako nema prikazaće se greška.(Slika 32)

```
def podizanjenovca(Stanje, adresa):
    clear = ('cls' if os.name == 'nt' else 'clear')
    global stanje
    print(":: Podizanje novca ::")
    #Ako nema novca
    if float(stanje) <= 0.0:
        print (":: Podizanje novca nije uspelo! ::\n:: Stanje racuna: ",stanje,"::","\n:: Molimo Vas prvo uplatite novac! ::\n")
        return
```

Slika 32. Definisanje funkcije "podizanjenovca" i provera stanja računa

Identično kao i kod depozita,korisinik unosi željenu količinu novca i onda idu provere.Dakle ista je provera samo za podizanje novca.(Slika 33)(Slika 31)

```
else:
    try:
        podizanje_novca = input("Unesite zeljenu kolicinu(din): ")
        os.system(clear)

        #Ako unese negativno
        if float(podizanje_novca) < 0.0:
            os.system(clear)
            print (":: Greska pri unosu. ::\n")
            return podizanje_novca(stanje, adresa)
```

Slika 33. Unos željene količine novca za podizanje

Ukoliko je željena suma manja od stanja na računu,stanje će se umanjiti za podignutu sumu.Prikazaće se poruka da je novac uspešno podignut i poslaće se email u kojem pišu informacije o promeni. (Slika 34)

```
elif float(podizanje_novca) <= stanje:
    stanje -= float(podizanje_novca)
    MSG = "Uspesno ste podigli novac "+str(podizanje_novca)+"\n\nVase stanje: "+str(stanje)
    msg = sendmail(adresa, MSG)
    os.system(clear)
    if not (msg == True): print(msg)
    print(":: Uspesno ste podigli novac",podizanje_novca,"::",'\n')
    return

else:
    os.system(clear)
    print (":: Podizanje novca nije uspelo! ::\n:: Moje stanje :",stanje,"::","\n")
return podizanjenovca(stanje, adresa)
```

Slika 34. Podizanje novca

Na redu je funkcija za promenu pina.Ona je identična kao i u klasi “login“ iz tog razloga neću stavljati fotografije.Isti su uslovi,iste su provere.(Slika 11) (Slika 12)

Funkcija za prenos novca ima argumente “tekuci_racun“ to je tekući račun primaoca,“balans“ ukoliko se pokuša prenos više novca od raspoloživog,“t_r“ je tekući račun pošiljaoca i email adresa.Deklariše se količina novca na početnu vrednost i proverava se da li je nalog ugašen. (Slika 35)

```
def prenos_novca(tekuci_racun, balans, t_r, adresa):
    import time,datetime
    clear = ('cls' if os.name == 'nt' else 'clear')
    os.system(clear)

    d = data()
    filename = join()
    kolicina_novca = 0.0

    print (":: Unesite kolicinu za prenos ::")
    ugasen_nalog = str('#'+tekuci_racun)
    #ako nije aktivan(ima # pre tekuceg racuna)
    if ugasen_nalog in d.keys():
        os.system(clear)
        print (":: Tekuci racun nije aktivan ::")
        return kolicina_novca
```

Slika 35. Prenos novca

Ukoliko tekući račun postoji u bazi podatak,korisnik unosi željenu količinu za prenos ukoliko je sve dobro,prenos će se izvršiti. (Slika 36)

```
elif tekuci_racun in d.keys():
    try:
        kolicina_novca = input("Unesite zeljenu kolicinu(din): ")
        #ne sme minus
        if float(kolicina_novca) < 0.0 or ('-' in kolicina_novca):
            os.system(clear)
            print (":: Greska pri unosu. ::\n")
            return prenos_novca(tekuci_racun, balans, t_r, adresa)
        #nema dovoljno novca
        elif float(kolicina_novca) > float(balans):
            os.system(clear)
            print (":: Ne mozete izvrsiti tranzakciju! ::\n:: Moje stanje : ",balans,"::","\n")
            return prenos_novca(tekuci_racun, balans, t_r, adresa)

        else:
            os.system(clear)
            print(":: Tekuci Racun :",tekuci_racun,"::")
            print(":: Ime i Prezime :",d[tekuci_racun][0],"::")
            print(":: Uneta kolicina za prenos = ","{:,} ::".format(float(kolicina_novca)),"\n")
```

Slika 36. Provere unosa količine novca za prenos

(Slika 37)Sledi potvrda i kreće upis.Menja se stara vrednost i upisuju se nove.Putem mejla se obaveštavaju nalozi o uspešnom prenosu novca.Tu su i provere ukoliko dođe do neke greške (Slika 38)

```
confirm = input("Molimo Vas potvrdite \n1. Da \n2. Ne \n")

if (confirm == '1'):
        with open(filename,'a+',newline="") as ap:
            trenutni_balans = balans
            balans = str(float(d[tekuci_racun][2]) + float(kolicina_novca))
            re_new = [tekuci_racun,d[tekuci_racun][0],d[tekuci_racun][1],balans,d[tekuci_racun][3],d[tekuci_racun][4]]
            w = csv.writer(ap)
            w.writerow(re_new)
            ap.close()

        os.system(clear)
        MSG_from = "Uspesno ste uplatili(din) :  "+str(kolicina_novca)+" Na tekuci racun :"+str(tekuci_racun)+"\n\nVas
        MSG_to = "Uspesno ste primili novac(din) "+str(kolicina_novca)+" Od tekuci racun "+str(t_r)+"\n\nVase stanje:
        sendmail(adresa, MSG_from)
        msg2 = sendmail(d[tekuci_racun][4], MSG_to)
        os.system(clear)
        if not (msg2 == True): print(msg2)
        print(":: Zeljena kolicina uspesno uplacena!::")
        return kolicina_novca
else:
    kolicina_novca = 0
    os.system(clear)
    print(":: Zeljena kolicina nije uspesno uplacena! ::")
    return kolicina_novca
```

Slika 37. Upis promene u bazu.

```
    except ValueError as err:
        os.system(clear)
        print("Greska :",err)
        print(":: Greska pri unosu. ::\n")
        return prenos_novca(tekuci_racun, balans, t_r, adresa)
else:
    os.system(clear)
    print (":: Doslo je do neslaganja,pokusajte ponovo ::")
    return kolicina_novca
```

Slika 38. Ispis poruka ukoliko dođe do greške.

2.3 Klasa "slanje_mejla"

Slanje mejla se izvršava uz pomoć modula "smtplib",potrebno je na gmailu dozvoliti pristup manje bezbednim aplikacijama. "sever.login(mejl,šifra mejla)" je mejl koji se koristi za slanje korisnicima.Ukoliko se unese pogrešan mail prikazaće se greška.(Slika 39)

```
import os, sys
import smtplib

def sendmail(adresa, msg, sbj = "BANKA"):
    clear = ('cls' if os.name == 'nt' else 'clear')
    try:
        try:
            server = smtplib.SMTP('smtp.gmail.com', 587)
            print ("Molimo sačekajte...")
            server.starttls()
            server.login("sendermail112@gmail.com", "SenderMail123")
            os.system(clear)
            print ("Molimo sačekajte....")
            message = 'Subject: {}\n\n{}'.format(sbj, msg)
            server.sendmail("sendermail112@gmail.com", adresa, message)
            os.system(clear)
            print ("Molimo sačekajte.....")
            server.sendmail("sendermail112@gmail.com",str(adresa)+"\n"+str(msg))
            os.system(clear)
            print ("Molimo sačekajte......")
            server.quit()
            return True

        except Exception:
            server = smtplib.SMTP('smtp.gmail.com', 587)
            os.system(clear)
            print ("Molimo sačekajte.....")
            server.starttls()
            server.login("sendermail112@gmail.com", "SenderMail123")
            os.system(clear)
            print ("Molimo sačekajte......")
            server.sendmail("sendermail112@gmail.com", str(adresa)+"\n"+str(msg))
            os.system(clear)
            print ("Molimo sačekajte.......")
            server.quit()
            return "Pogrešan mail"
    except Exception:
        return ""
```

Slika 39. Funkcija koja šalje mejlove

2.4 Klasa "graph"

"Matplotlib" je jedna od najpopularnijih biblioteka za plotovanje podataka.Uz pomoć funkcije "grafikon()" se prikazuje stanje računa u odnosu na korisnike.U taj grafikon ima uvid samo administrator.Otvara se csv fajl,čita se,za graničnik se koristi "," i preskače se prva linija u fajlu zato što je prazna i pravi problem pri pokretanju grafikona.Ugašeni tekući računi i admin nalog se prikazuju u grafikonu.(Slika 40)

```
import matplotlib.pyplot as plt
import csv
from Data import *
def grafikon():
    x = []
    y = []

    with open(r'C:\Users\admin\Desktop\Bankomat\Data\usersdata.csv','r') as csvfile:
        r = csv.reader(csvfile, delimiter = ',')
        next(r) #preskace prvu liniju

        for indiv_korisnik_info in r:
            if indiv_korisnik_info[0].startswith('#'):
                pass
            elif indiv_korisnik_info[1].startswith('admin pristup'):
                pass

            else:
                x.append(indiv_korisnik_info[1])
                y.append((indiv_korisnik_info[3]))

    plt.bar(x, y, color = 'b', width = 0.3, label = "kolicina novca")
    plt.xlabel('Imena')
    plt.ylabel('Stanje racuna')
    plt.title('Kolicina novca kod korisinika')
    plt.legend()
    plt.show()
```

Slika 40. Grafikon

2.5 Klasa "Data"

Definiše se funkcija "join()" koja kreira folder i fajl u kojem se skladište svi podaci.(Slika 41)

```
import os
import csv

def join():
    directory = "Data"
    name = "usersdata.csv"
    filename = os.path.join(directory, name)
    return filename
```

Slika 41. Kreiranje baze podataka

Funkcija "data()" sortira potrebne podatke u "filename" odnosno u fajl koji je funkcija "join()" napravila.Imamo listu "new" i rečnik "d".Ukoliko ne postoje korisnici izlazi poruka da je prvo potrebno da se napravi nalog. (Slika 42)

```
def data():
    filename = join()
    d = {}
    new = []
    try:
        with open(filename, "a",newline="") as ap:
            if (os.path.getsize(filename)) <= 0:
                wr = csv.writer(ap)
                wr.writerow(new)
                ap.close()
                print ("Prvo napravite nalog!")
                return
```

Slika 42. Definisanje funkcija "data()"

Čita podatke,zanemaruju se prazna mesta i prazni redovi.Ključ je "indiv_korisnik_info[0]" što je zapravo tekući račun."indiv_korisnik_info[0]" se dodeljuju ime i prezime,pin,stanje,vreme i adresa.Drugo je isto to samo ukoliko se ne unese mail.Ukoliko dođe do greške napravi folder.(Slika 43)

```
        else:
            with open(filename, "r",newline="") as rd:
                r = csv.reader(rd)
                for indiv_korisnik_info in r:
                    if (indiv_korisnik_info == ['']) or (indiv_korisnik_info == []):
                        continue

                    else:
                        try:
                            indiv_korisnik_info[1] = (indiv_korisnik_info[1])
                            indiv_korisnik_info[3] = float(indiv_korisnik_info[3])
                            d[indiv_korisnik_info[0]] = indiv_korisnik_info[1],indiv_korisnik_info[2],indiv_korisnik_info[3],indiv_korisnik_info[4],

                        except IndexError:
                            d[indiv_korisnik_info[0]] = indiv_korisnik_info[1],indiv_korisnik_info[2],indiv_korisnik_info[3],indiv_korisnik_info[4],
                return d

except (IOError or FileNotFoundError):
    os.mkdir("Data")
    data()
```

2.6 Klasa "t_r_gen"

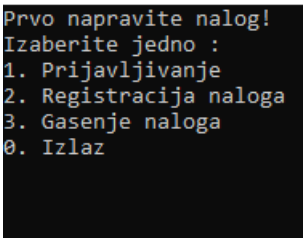
Funkcija koja uzima unos "ime_prezime" i vraća string "t_r" koji čini 12 nasumičnih brojeva odnosno tekući račun. (Slika 44)

```
from random import randint, randrange
from Data import data

def tekuci_racun_gen(ime_prezime):
    d = data()
    alphabets = 'abcdefghijklmnopqrstuvwxyz'
    t_r = ''
    t_r += str(len(d.keys()) + 10)
    for alpha in ime_prezime:
        if (len(t_r) < 12):
            if alpha in alphabets:
                index = alphabets.rfind(alpha)
                t_r += str(index + 1)

            else:
                t_r += '0'

        else:
            break

    if len(t_r) > 12:
        final_t_r = ''
        for index in t_r:
            if len(final_t_r) < 12:
                final_t_r += index

        t_r = final_t_r
        return t_r

    if len(t_r) < 12:
        remain_index = 12 - len(t_r)
        for index in range(remain_index):
            t_r += str(randint(0,9))

        return t_r

    else:
        return t_r
```

Slika 44. Generisanje broja tekućeg računa

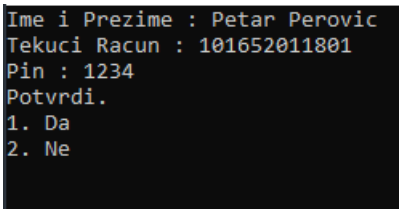
2.7 Za kraj delovi programa kako to izgleda i radi

Login i opcije (Slika 45)



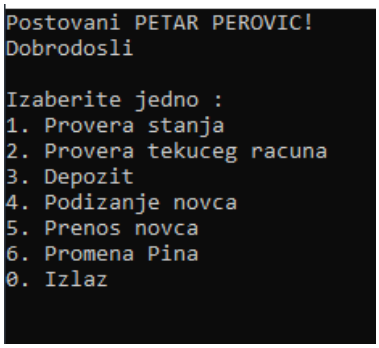
Slika 45. Login

Potvrda podataka pre kreiranja naloga (Slika 46)



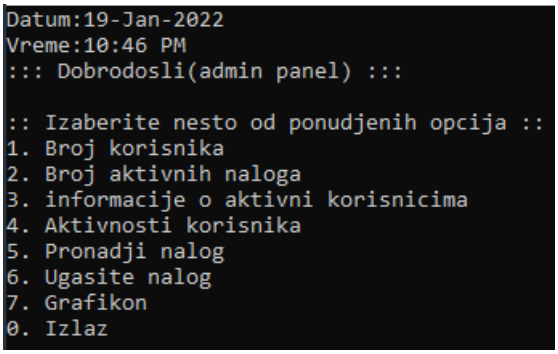
Slika 46. Potvrda

Nakon prijavljivanja punuđene opcije (korisnik panel)(Slika 47)



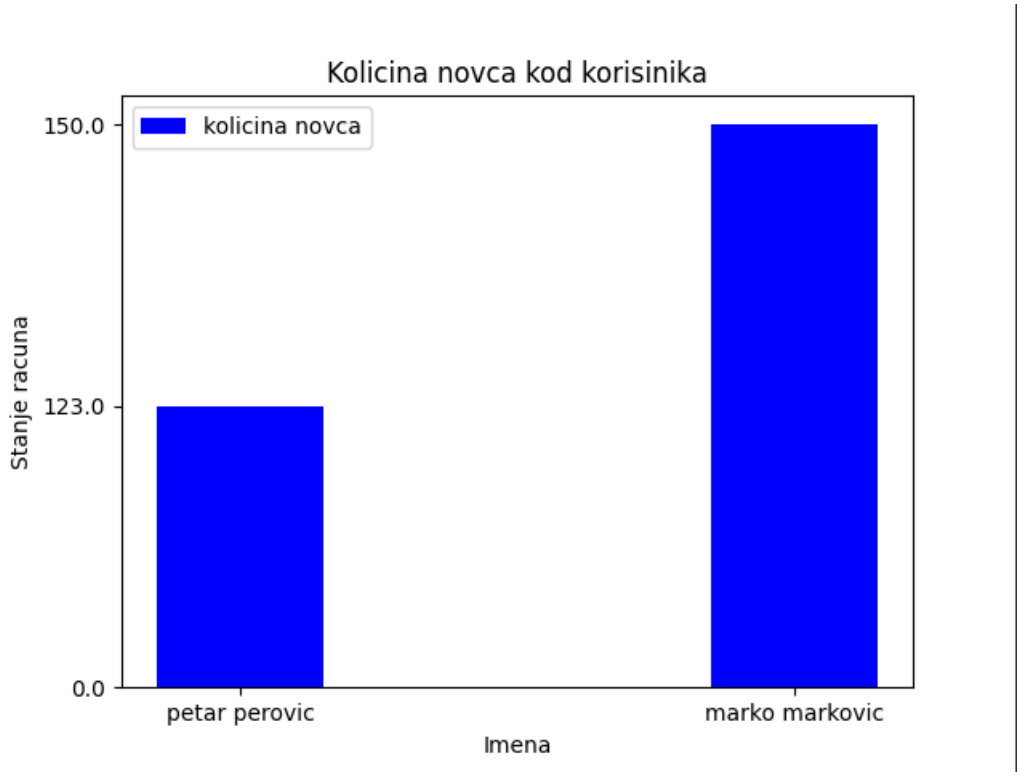
Slika 47. Korisnik panel

Nakon prijavljivanja ponuđene opcije (admin panel) (Slika 48)



Slika 48. Admin panel

Grafikon (Slika 49)



Slika 49. Grafikon

Fajl u kojem se čuvaju podaci.Tekući račun,ime i prezime,pin,stanje računa,datum poslednjeg prijavljivanja i email adresa. (Slika 50)

```
101652011801,petar perovic,1234,0.0,19-Jan-2022 at 10:45 PM,Nije unet email
101652011801,petar perovic,1234,0.0,19-Jan-2022 at 10:45 PM,Nije unet email
111413914016,admin pristup,1234,0.0,19-Jan-2022 at 10:46 PM,Nije unet email
101652011801,petar perovic,1234,123.0,19-Jan-2022 at 10:47 PM,Nije unet email
```

Slika 50. “usersdata.csv“

3. Zaključak

Projekat je moguće unaprediti dodavanjem grafički korisničkog interfejsa i nekom boljom bazom podataka.Uz nadogradnju ovaj program bi se mogao koristi kao pravi bankomat.

4. Literatura

1. CSV File Reading and Writing, https://docs.python.org/3/library/csv.html
2. Matplotlib, https://www.python-graph-gallery.com/matplotlib/
3. w3schools, https://www.w3schools.com/python/