

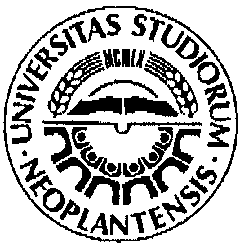
**UNIVERZITET U NOVOM SADU**

**PRIRODNO-MATEMATIČKI**

**FAKULTET**

**DEPARTMAN ZA MATEMATIKU**

**I INFORMATIKU**



Karolina Bicok, 63/19

Aplikacija za menadžment časova u školi stranih jezika

- seminarski rad iz predmeta Skript jezici-

Novi Sad, 2020.

# Sadržaj

Sadržaj

[Sadržaj 2](#_Toc49161585)

[1. Uvod 3](#_Toc49161586)

[O programskom jeziku ,,Python” 3](#_Toc49161587)

[Opis programa 3](#_Toc49161588)

[2. Klase 5](#_Toc49161589)

[Klasa Student 5](#_Toc49161590)

[Klasa Teacher 7](#_Toc49161591)

[Klasa Admin 10](#_Toc49161592)

[Klasa Course 11](#_Toc49161593)

[Klasa Appointment 13](#_Toc49161594)

[3. Moduli 19](#_Toc49161595)

[Modul CustomExceptions.py 19](#_Toc49161596)

[Modul input\_check.py 19](#_Toc49161597)

[Modul admin\_service.py 21](#_Toc49161598)

[Modul student\_service.py 23](#_Toc49161599)

[Modul teacher\_service.py 28](#_Toc49161600)

[Modul menu.py 33](#_Toc49161601)

[4. Rad aplikacije 39](#_Toc49161602)

[5. Zaključak 45](#_Toc49161603)

[6. Literatura 46](#_Toc49161604)

# 1. Uvod

## O programskom jeziku ,,Python”

Python je veoma popularan programski jezik opšte namene. Postao je poznat po svojoj jednostavnosti, lakoći učenja i brzini programiranja. Mnogi profesionalni programeri koriste Python bar kao pomoćni jezik, jer pomoću njega brzo i lako automatizuju razne poslove. Zbog pomenutih dobrih osobina koristi ga i sve više ljudi drugih profesija koji se na svom poslu služe programiranjem u različitim oblastima. Programski jezik Pajton je besplatan za upotrebu, pa se oko njega formirala brojna zajednica koja doprinosi njegovom daljem razvoju i podršci na internetu.

## Opis programa

U seminarskom radu biće prikazana aplikacija koja predstavlja simulaciju menadžmenta časova u nekoj školi stranih jezika. Aplikacija je istovremeno namenjena i za nastavnike i za učenike, dok sa druge strane administratori imaju posebna ovlašćenja.

Kada se pokrene aplikacija, postoje tri moguće opcije koje korisnik može odabrati. Prva služi za prijavljivanje korisnika, i to tako da korisnik treba da odabere da li se prijavljuje kao učenik, nastavnik, administrator, ili pak želi da odustane od prijave. U zavisnosti od izbora, kada se korisnik prijavi, imaće različite opcije.

Prijava kao učenik sadrži sledeće opcije: prikaz kurseva u ponudi, prikaz zakazanih časova, zakazivanje časa, otkazivanje časa, promena nastavnika, napuštanje škole i odjavu.

Prijava kao nastavnik sadrži opcije poput: prikaz zakazanih časova, opciju za završavanje časa, otkazivanje časa, prikaz današnje zarade, prikaz grafikona današnje zarade za svakog nastavnika i odjavu.

Prijava kao administrator sadrži opcije: dodavanje novog kursa, registracija novog nastavnika, registracija novog administratora, dodavanje novčanih sredstava studentu i odjavu.

Druga mogućnost jeste registracija novog učenika. Ta funkcionalnost se sastoji od unosa podataka poput imena, prezimena, korisničkog imena, šifre, jezika koji učenik želi da uči, nastavnika koji taj jezik predaje, kao i koliko ima novca na raspolaganju.

Treća, krajnja opcija jeste izlaz iz aplikacije.

Fajlovi koji se koriste za čitanje i pisanje podataka jesu students.txt, teachers.txt, admins.txt, courses.txt, appointments.txt i finished\_appointments.txt .

Students.txt fajl sadrži podatke o jednom učeniku u svakom redu fajla. Podaci koje jedan red sadrži su: ime, prezime, korisničko ime, lozinka, jezik, korisničko ime nastavnika i novac kojim učenik raspolaže. Svi podaci podeljeni su uspravnom crtom.

Teachers.txt fajl, slično kao i fajl sa učenicima, sadrži ime, prezime, jezik, korisničko ime i lozinku – za svakog nastavnika u redu.

Admins.txt fajl sadrži podatke o korisničkim imenima i lozinkama administratora.

Courses.txt sadrži naziv kursa i njegovu cenu.

Appointments.txt je fajl u kome se čuvaju svi zakazani časovi. Jedan red sadrži datum, vremenski broj, korisničko ime nastavnika, korisničko ime učenika, naziv kursa i njegovu cenu. Fajl finished\_appointments.txt sadrži iste podatke, ali ima drugačiju namenu u izvršavanju programa.

# 2. Klase

## Klasa Student

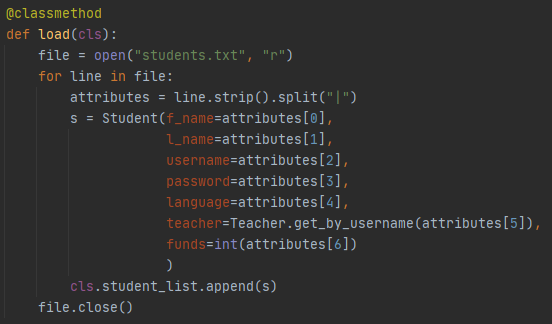
Zbog potrebe da se manipuliše raznim podacima i međusobnim vezama između nastavnika i učenika, termina i predmeta, aplikacija je zamišljena tako da koristi klasu za svaku od prethodno navedenih stavki. Tako imamo klasu Student koja je opisana sledećom funkcijom(slika 1.0):

Slika 1.0 - funkcija init i lista studenata


Slika 1.0 - funkcija init i lista

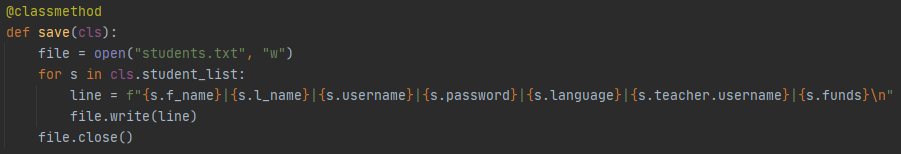
Na ovaj način opisuje se koje će podatke posedovati instanca naše klase Student. Ova funkcija je važna zbog dodavanja novih učenika, kao i zbog čitanja podataka iz fajla students.txt. Lista studenata tj. student\_list služiće da čuva sve učenike tj. objekte koji postoje.

Funkcija load() (slika 1.1) ima ulogu da iz fajla students.txt pročita redove, odvoji podatke u listu attributes, a potom napravi objekat Student sa određenim podacima iz tog reda fajla. Svakim čitanjem reda čuvaju se podaci i objekat se skladišti u listu student\_list.



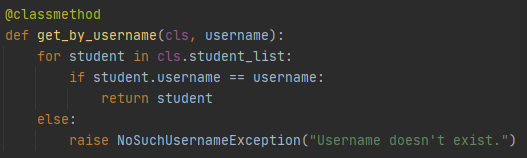
Slika 1.1 - funkcija load()

Funkcija save() (slika 1.2)služi da sačuva sve postojeće objekte koji se nalaze u student\_list tako što iterira kroz listu. Podaci svakog studenta se formatiraju pomoću f-strings a zatim se takav string upisuje u tekstualni fajl students.txt.



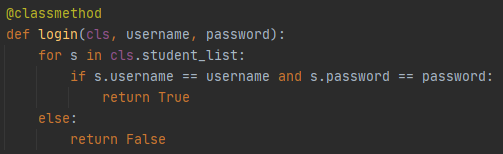
Slika 1.2 - funkcija save()

Sledeća funkcija koja je u sklopu klase Student jeste get\_by\_username(). Ona služi kako bismo pronašli određenog studenta tj. objekat na osnovu njegovog korisničkog imena. Kao argument, funkcija prima username, a zatim se iterira kroz listu objekata i proverava da li se poklapa sa nekim postojećim korisničkim imenom(slika 1.3)



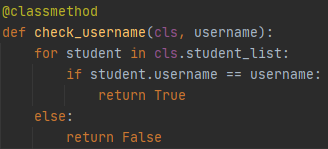
Slika 1.3 - get\_by\_username()

Funkcija login() ima ulogu da vrati True ili False vrednost u zavisnosti od poklapanja argumenata sa podacima u listi studenata. Ukoliko se istovremeno poklope username i password sa nekim postojećim u listi objekata, vraća se True. U suprotnom dobija se vrednost False(slika 1.4).



Slika 1.4 - login()

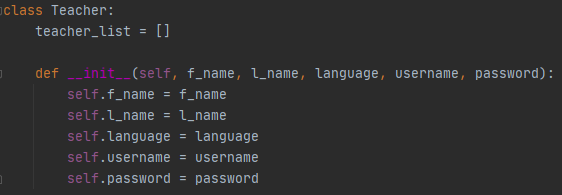
Funkcija check\_username(), slično kao prethodna, proverava postojanje određenog korisničkog imena u listi studenata. Ova funkcija(slika 1.5) služiće u delu aplikacije kada će biti potrebno da se osigura da ne postoje studenti sa istim korisničkim imenom.



Slika 1.5 - check\_username()

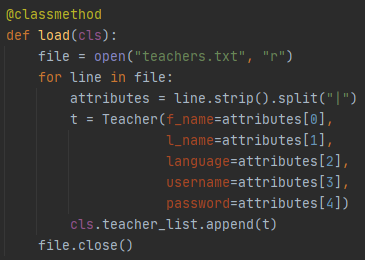
## Klasa Teacher

Klasa Teacher opisuje se sledećom funkcijom za inicijalizovanje(slika 2.0). Ona se karakteriše imenom, prezimenom, jezikom, korisničkim imenom i lozinkom. Pozivanjem funkcije \_\_init\_\_() kreira se instance tj. objekat klase Teacher. Lista teacher\_list čuva objekte tipa Teacher.



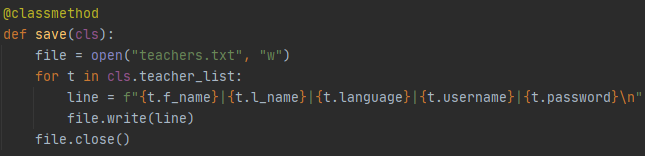
Slika 2.0 - \_\_init\_\_()

Funkcija load() (slika 2.1)ima ulogu da od podataka iz tekstualnog fajla teachers.txt napravi instance klase Teacher i doda ih u listu teacher\_list iterirajući kroz svaki red u fajlu.



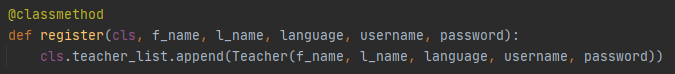
Slika 2.1

Funkcija save() čuva objekte iz liste teacher\_list u tekstualni fajl teachers.txt tako što formatira podatke objekata u string pomoću f-strings funkcije i upisuje ih u fajl pomoću write() funkcije(slika 2.2).



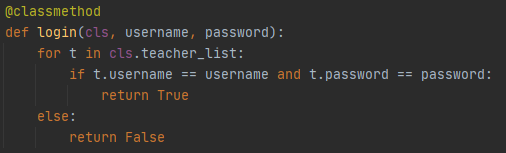
Slika 2.2

Funkcija register() (slika 2.3) sadrži određene parametre koji služe da se napravi instanca klase Teacher i sačuva u listi teacher\_list pomoću funkcije append().



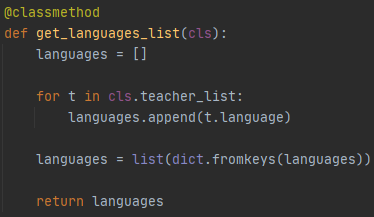
Slika 2.3

Funkcija login() (slika 2.4) ima istu ulogu kao i u klasi Student, gde se parametri username i password proveravaju tako što se iterira kroz listu nastavnika i traži se poklapanje sa podacima. Ukoliko je nastavnik sa takvim korisničkim imenom i lozinkom pronađen, vraća se True vrednost, a u suprotnom False.



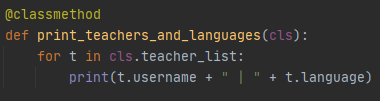
Slika 2.4

Funkcija get\_languages\_list() (slika 2.5) služi da se pribavi lista jezika koje predaju nastavnici. Iterira se kroz listu objekata nastavnika i svaki jezik dodaje se u listu languages. Na kraju se od liste pravi rečnik, pa ponovo lista jezika kako bi se izbegli duplikati. Na kraju funkcija vraća listu jezika.



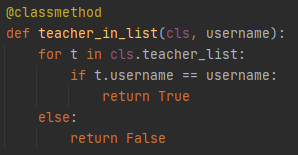
Slika 2.5

Funkcija print teachers\_and\_languages() (slika 2.6) ima ulogu da štampa korisničko ime nastavnika i jezik koji predaje, iterirajući kroz listu nastavnika.



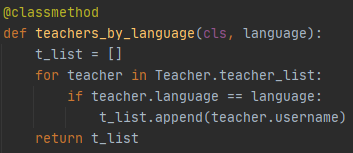
Slika 2.6

Funkcija teacher\_in\_list() (slika 2.7) služi da se proveri da li postoji nastavnik sa određenim korisničkim imenom u listi nastavnika. Ukoliko se pronađe poklapanje, vraća se vrednost True, a u suprotnom False. Ova funkcija imaće ulogu tokom registracije nastavnika.



Slika 2.7

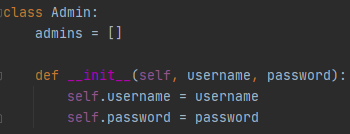
Funkcija teachers\_by\_language() (slika 2.8) ima ulogu da pronađe nastavnike koji predaju jezik koji je prosleđen kao parametar language. Kreira se lista t\_list i ako u toku iteriranja kroz listu nastavnika postoji nastavnik koji predaje određen jezik, njegovo korisničko ime biće dodato u listu pomoću append() funkcije. Kada se iteriranje završi, vraća se t\_list.



Slika 2.8

## Klasa Admin

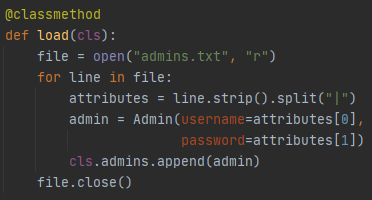
Klasa Admin opisana je funkcijom \_\_init\_\_() (slika 3.0) pomoću koje se kreira instanca klase koja sadrži korisničko ime i lozinku. Instance klase čuvaju se u listi admins.



Slika 3.0

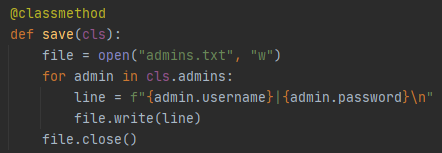
Naredne funkcije u klasi Admin rade na isti način kao i one u klasi Teacher i Student, s tim što se podaci dobijaju iz fajla admins.txt i čuvaju u isti:

**load()** (slika 3.1)



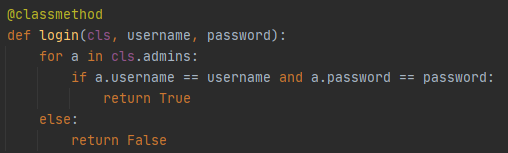
Slika 3.1

**save()** (slika 3.2).



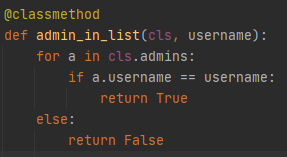
Slika 3.2

**login()** (slika 3.3)



Slika 3.3

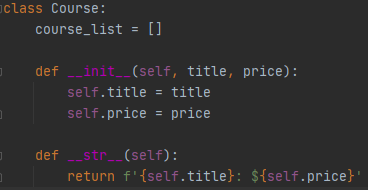
**admin\_in\_list()** (slika 3.4)



Slika 3.4

## Klasa Course

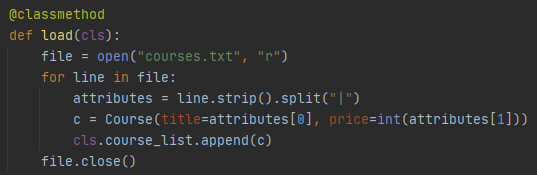
Objekti klase Course inicijalizuju se pomoću funkcije \_\_init\_\_() i sadrže naziv kursa i njegovu cenu. Objekti klase čuvaće se u listi course\_list. Pored toga, funkcija \_\_str\_\_() ima ulogu reprezentacije objekata, tako da kad se prikaže lista course\_list, naziv i cena kursa biće prikladno prikazani i čitljivi(slika 4.0).



Slika 4.0

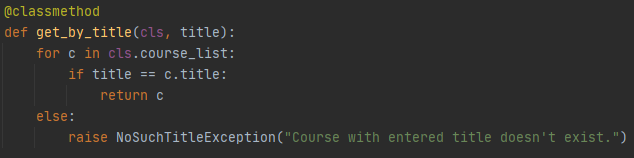
Funkcije koje se ponavljaju i imaju jednaku ulogu kao i prethodne su sledeće:

**load()** (slika 4.1)



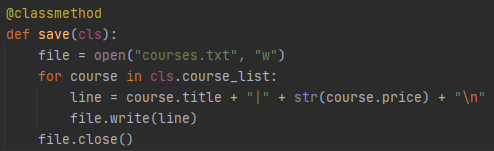
Slika 4.1

**get\_by\_title()** – kao get\_by\_username (slika 4.2)



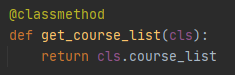
Slika 4.2

**save()** (slika 4.3)



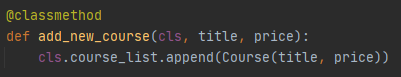
Slika 4.3

Funkcija get\_course\_list() vraća listu objekata tipa Course(slika 4.4).



Slika 4.4

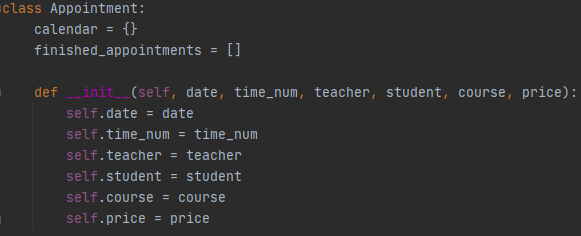
Funkcija add\_new\_course() dodaje novu istancu klase u listu kurseva tako što prenosi parametre u parametre za inicijalizaciju objekta(slika 4.5).



Slika 4.5

## Klasa Appointment

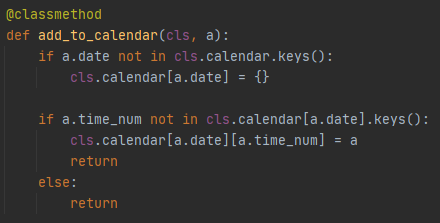
Klasa Appointment složenija je od svih prethodnih klasa. Ona je karakteristična po tome što se instance čuvaju u rečniku koji je podrečnik rečnika calendar. Rečnik calendar sadrži ključeve koji su datumi, a vrednosti su rečnici. Vrednosti su sada ključevi podrečnika i to tipa int, a vrednosti podrečnika objekti tipa Appointment. Pored kalendara koji čuva podatke o zakazanim terminima tj. časovima, postoji i lista završenih časova.



Slika 5.0

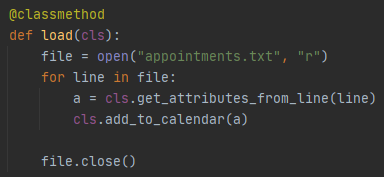
Funkcija \_\_init\_\_() sadrži podatke date, time\_num, objekat tipa Teacher, objekat tipa Student, objekat tipa Course i cenu kursa tipa int(slika 5.0).

Funkcija add\_to\_calendar() dodaje objekat tipa Appointment u calendar. Prvo je potrebno proveriti da li je određeni datum upisan u calendar. Ukoliko nije, potrebno je dodati ključ s određenim datumom i njegovu vrednost – rečnik. Sledeća stvar koja se proverava je postojanje vremenskog broja(time num) u podrečniku. Ukoliko ne postoji, to znači da u tom terminu nije zakazan nijedan drugi, pa se tom vremenskom broju(ključu) dodeljuje objekat a(slika 5.1).



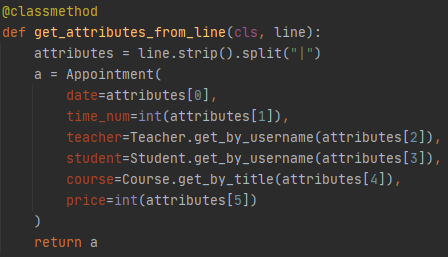
Slika 5.1

Funkcija load() (slika 5.2) ista je kao i u prethodnim klasama, s tim što je pribavljivanje podataka iz jednog reda fajla skraćeno pomoću funkcije get\_attributes\_from\_line() (slika 5.3) koja će se koristiti na više mesta.



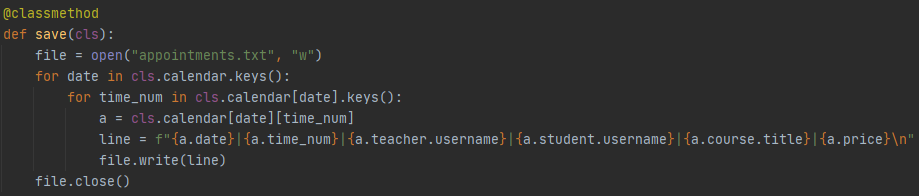
Slika 5.2

Kao što se može primetiti na slici 5.3, argumenti za teacher, student i course dobijaju se pozivom funkcija koje vraćaju objekte na osnovu imena.



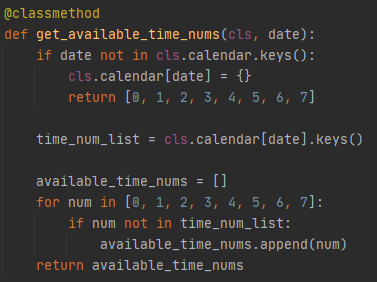
Slika 5.3

Funkcija save() ima ulogu da sačuva sve zakazane termine, prolazeći kroz svaki datum i vremenski broj(slika 5.4).



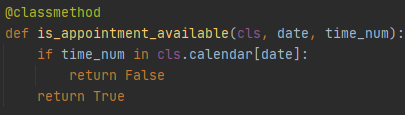
Slika 5.4

Funkcija get\_available\_time\_nums() (slika 5.5) služi kako bi se vratila lista slobodnih vremenskih brojeva. Ukoliko određeni datum nije u kalendaru, biće dodat i vratiće se lista svih brojeva od 0 do 7, što će značiti da su svi termini za taj dan slobodni. Ako se datum već nalazi u kalendaru, potrebno je pribaviti listu brojeva koji se nalaze u podrečniku uz pomoć cls.calendar[date].keys(). Zatim se iterira kroz listu brojeva od 0 do 7 i proverava da li se neki broj ne nalazi u listi koju smo prethodno pribavili. Ako je slučaj potvrdan, taj broj se dodaje u listu slobodnih vremenskih brojeva, a zatim se vraća lista.



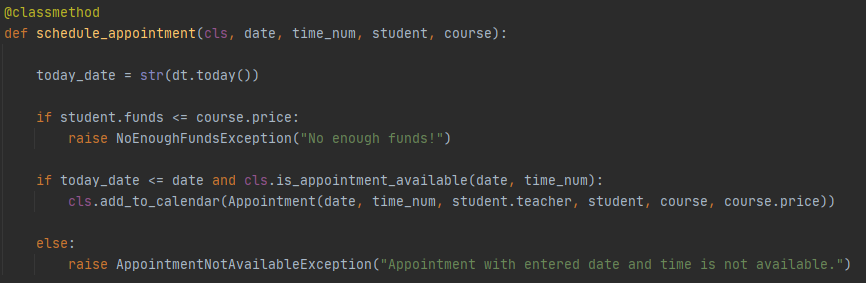
Slika 5.5

Funkcija is\_appointment\_available() proverava da li je vremenski broj slobodan. Ukoliko datum nije ni upisan u kalendar ili se time num ne nalazi u podrečniku, vraća se vrednost True, a u suprotnom False (slika 5.6). Ova funkcija biće važna pri zakazivanju časa.



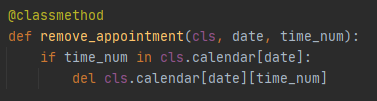
Slika 5.6

Funkcija schedule\_appointment() (slika 5.7) ima ulogu da doda određeni objekat tipa Appointment u kalendar pomoću prethodno navedene funkcije. Međutim, potrebno je proveriti da li se uneti datum nalazi u stvarnom kalendaru posle današnjeg dana, jer nema smisla zakazivati termin koji je prošao. dt je skraćenica za datetime biblioteku uz pomoć koje može da se dobije stvarno trenutno vreme kada se pozove funkcija today(). Pored provere ispravnosti datuma, potrebno je proveriti da li student koji zakazuje termin ima dovoljno sredstava za određeni kurs. Kada se utvrdi da postoji slobodan vremenski broj tj. da je termin slobodan, moguće je zakazati čas i Appointment se dodaje u calendar.



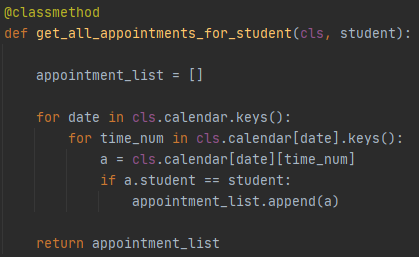
Slika 5.7

Funkcija remove\_appointment() (slika 5.8) proverava da li se određeni vremenski broj nalazi u podrečniku kalendara i briše ga ako ga pronađe.



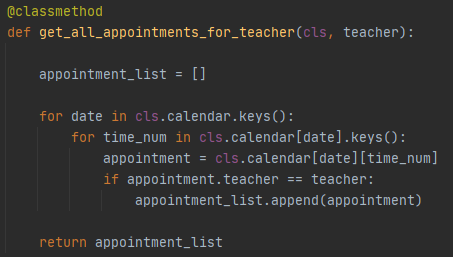
Slika 5.8

Funkcija get\_all\_appointments\_for\_student() (slika 5.9) ima ulogu da vrati listu objekata tipa Appointment za određenog studenta. Zakazani termini nalaze se tako što se iterira kroz rečnik calendar i podrečnik, proveravajući da li objekat koji se nalazi na tom mestu sadrži korisničko ime prosleđenog studenta. Ukoliko je to potvrdno, objekat tipa Appointment dodaje se u listu appointment\_list, a na kraju se vraća ta lista.



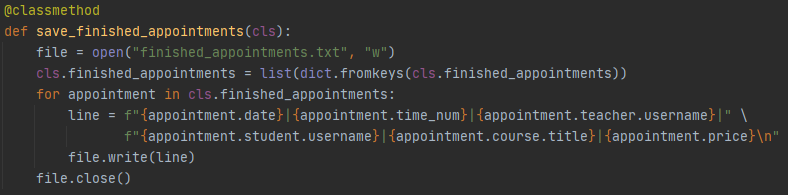
Slika 5.9

Funkcija get\_all\_appointments\_for\_teacher() radi isto kao i prethodna, samo što se umesto korisničkog imena studenta sada proverava korisničko ime nastavnika(slika 5.10).



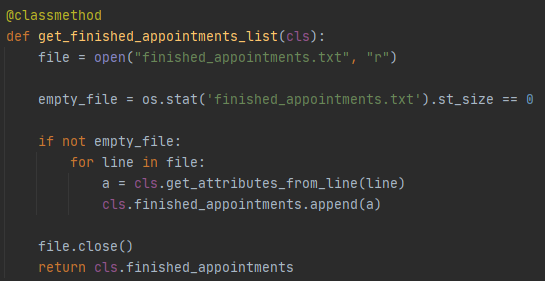
Slika 5.10

Funkcija save\_finished\_appointments() (slika 5.11) otvara fajl za upis podataka o završenim časovima. Kao u prethodnim funkcijama za čuvanje, podaci o terminu se ređaju u string, a potom se on upisuje u fajl.



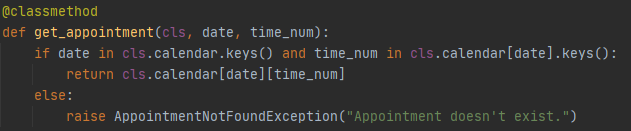
Slika 5.11

S druge strane, funkcija get\_finished\_appointments\_list() (slika 5.12) vraća listu objekata termina koji su završeni. Pre dodavanja objekata u listu potrebno je proveriti da li je fajl prazan.



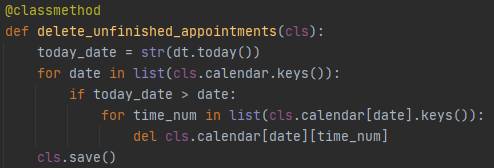
Slika 5.12

Funkcija get\_appointment() (slika 5.13)proverava da li određeni datum postoji u kalendaru, a potom i vremenski period. Ukoliko postoji, vraća objekat termina koji je sačuvan kao vrednost za određeni vremenski broj.



Slika 5.13

Funkcija delete\_unfinished\_appointments() (slika 5.14) briše sve termine koji se nalaze u kalendaru, a čiji datum je prošao. Da bi bilo moguće brisati određene vrednosti u rečniku kroz koji se iterira, mora se pribaviti kopija ključeva pomoću list().

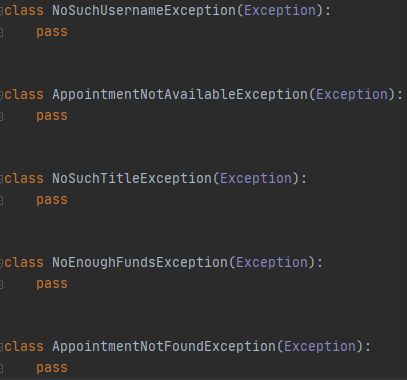


Slika 5.14

# 3. Moduli

Moduli u ovoj aplikaciji podeljeni su tako da opisuju i sadrže funkcije koje su povezane s određenim klasama. Pored toga, postoje moduli za proveru unosa i posebno napravljene izuzetke.

## Modul CustomExceptions.py

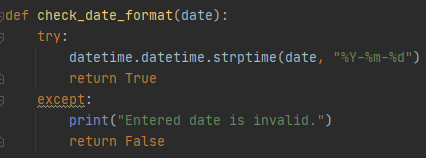


Slika 6.0

Modul CustomExceptions.py sadrži nazive izuzetaka koji se pojavljuju tokom provere konačnih rezultata rada određene funkcije ili uslova(slika 6.0)

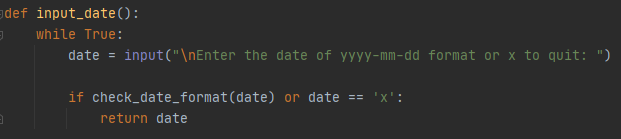
## Modul input\_check.py

Funkcija check\_date\_format() (slika 7.0) proverava da li je uneti datum validan, u smislu da se ne može prekoračiti broj dana u određenom mesecu, broj meseci i slično. Potrebno je uvesti biblioteku datetime koja sadrži određenu funkciju za proveru. Pored toga, istovremeno se proverava i način unosa datuma, tako da je važno da prve 4 cifre predstavljaju godinu, druge dve mesec i poslednje dve dan. Uz to, godina, mesec i dan moraju biti odvojeni crtom.



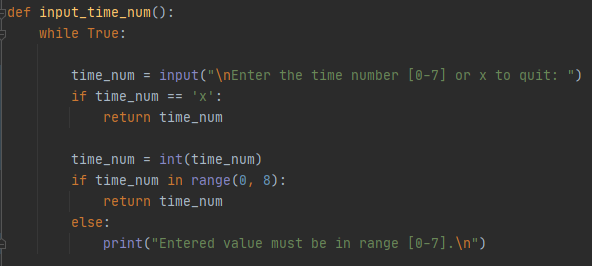
Slika 7.0

Funkcija input\_date() (slika 7.1) poziva se pri svakom unosu datuma. Ukoliko datum ispunjava uslove iz if naredbe, funkcija vraća datum, a u suprotnom se vri while petlja.



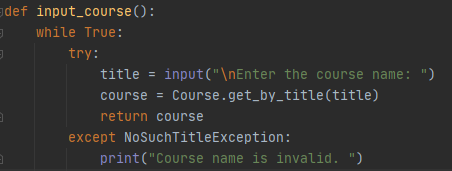
Slika 7.1

Funkcija input\_time\_num() (slika 7.2) poziva se pri unosu vremenskog broja. Posebna pažnja obraća se na tip unetog podatka, tako da se dozvoljava jedino unos prirodnih brojeva od 0 do 7.



Slika 7.2

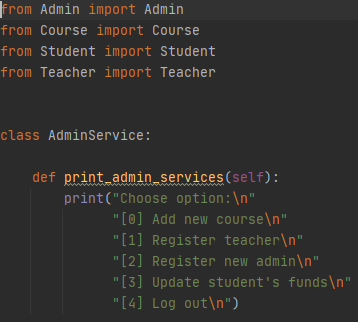
Funkcija input\_course() (slika 7.3) služi za proveru unetog naziva kursa. Pomoću funkcije iz klase Course nalazimo kurs sa datim nazivom, a ukoliko funkcija baca izuzetak, štampamo određenu poruku za nepravilan unos i petlja nastavlja da se vrti.



Slika 7.3

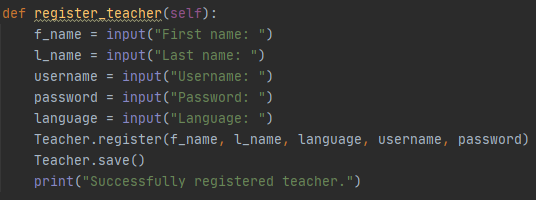
## Modul admin\_service.py

Modul admin\_service.py sadrži istoimenu klasu s određenim funkcijama.



Slika 8.0

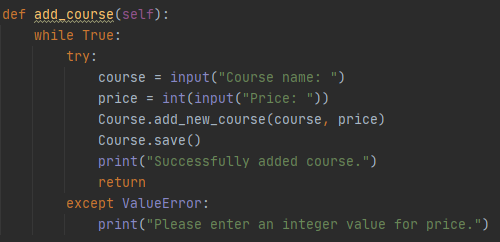
Prva funkcija, print\_admin\_services() (slika 8.0) služi za prikaz svih opcija koje može odabrati korisnik koji se prijavi kao administrator. U to spadaju dodavanje novog kursa, nastavnika i administratora, kao i dodavanje sredstava na raučun studenta.



Slika 8.1

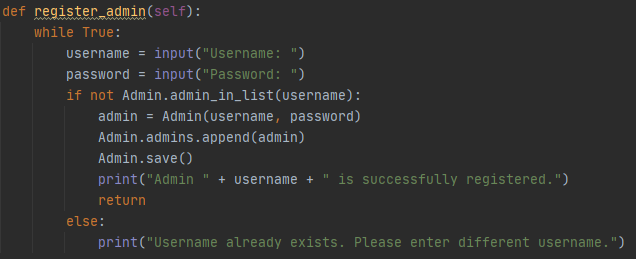
Funkcija register\_teacher() (slika 8.1) služi za unos podataka koji će se proslediti funkciji register() iz klase Teacher. Nakon inicijalizovanja novog objekta, potrebno je sačuvati listu u fajl pomoću Teacher.save().

Funkcija add\_course() (slika 8.2) ima ulogu za unos podataka o novom kursu, pri čemu se poziva funkcija za dodavanje novog kursa iz klase Course. Kao i u prethodnoj funkciji, potrebno je sačuvati listu objekata pomoću Course.save(). U slučaju da nije unet ceo broj, od korisnika će se zahtevati da ponovi unos.



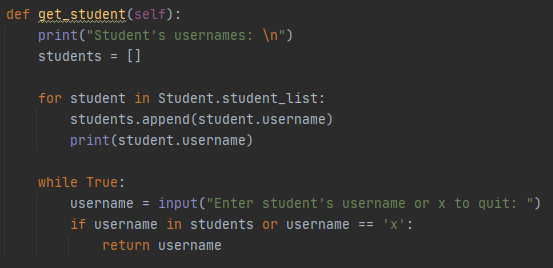
Slika 8.2

Funkcija register\_admin() (slika 8.3) zahteva unos korisničkog imena i lozinke za novog administratora. Ukoliko ne postoji administrator sa datim imenom, kreira se novi objekat u listi admins, a potom se lista čuva u fajl pomoću Admin.save(). U suprotnom se zahteva ponovni unos podataka.



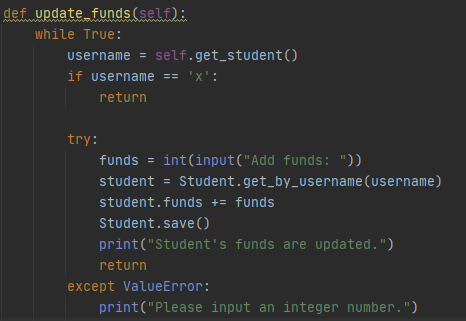
Slika 8.3

Funkcija get\_student() (slika 8.4) prikazuje listu korisničkih imena studenata, kako bi administrator odabrao onog čije podatke želi izmeniti. Kada unese korisničko ime koje postoji u listi studenata, funkcija vraća korisničko ime studenta.



Slika 8.4

Funkcija update\_funds() (slika 8.5) daje mogućnost administratoru da unese vrednost koja će se dodati na račun određenog studenta. To će biti moguće tek kada se pronađe student s datim korisničkim imenom. Kada se dodavanje izvrši, podaci o studentu moraju se sačuvati pomoću Student.save(). Ukoliko se unese vrednost za sredstva koja nije tipa int, zatražiće se ponovni unos.



Slika 8.5

## Modul student\_service.py

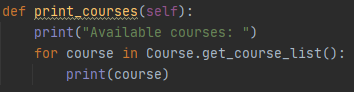
Modul student\_service.py čini istoimena klasai funkcije unutar nje.

Funkcija schedule\_appointment() (slika 9.0) koristi se za zakazivanje časa. U početku se poziva funkcija input\_date() za unos datuma. Zatim se dobijaju slobodni vremenski brojevi za taj datum. Da bi korisnik znao na šta se odnosi svaki vremenski broj tj. koje je njegovo pravo vreme u stvarnosti, pravi se rečnik pomoću get\_time\_num\_dict(). Pomoću print\_available\_time\_nums() korisnik vidi vremenske brojeve koji su slobodni. Promenljiva time\_num prima vrednost pomoću funkcije input\_time\_num(). Nakon toga, štampaju se nazivi kurseva pomoću funkcije self.print\_courses() i traži se od korisnika da unese naziv kursa sa liste. Ukoliko je sve prvilno uneto, poziva se istoimena funkcija Appointment.schedule\_appointment(), a zatim se čuva novi termin pomoću Appointment.save(). Ukoliko je nemoguće odraditi zakazivanje zbog nekog od izuzetaka, štampa se prigodna poruka i unos se ponavlja.



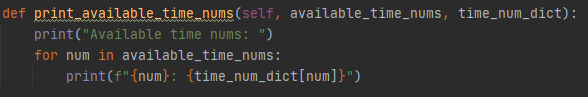
Slika 9.0

Funkcija print\_course() (slika 9.1) iterira kroz listu kurseva i štampa svaki od njih.



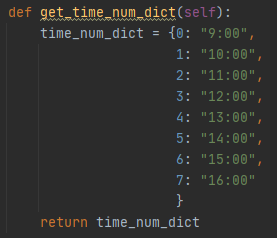
Slika 9.1 1

Funkcija print\_available\_time\_nums() (slika 9.2) ima dva parametra. Prvi parameter je lista slobodnih brojeva, dok je drugi rečnik. Štampaju se samo oni brojevi i njegovi stvarni sati koji se nalaze u listi slobodnih parametara.



Slika 9.2

Funkcija get\_time\_num\_dict() (slika 9.3) vraća rečnik koji se vidi na slici.



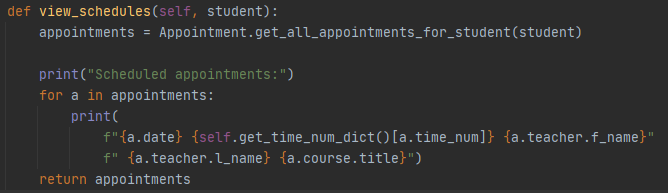
Slika 9.3

Funkcija cancel\_appointment() (slika 9.4) prikazuje zakazane termine za određenog studenta, a promenljiva appointments prima listu objekata termina. Ako je lista prazna, ispisuje se poruka o tome i izlazi se iz funkcije. Ukoliko nije, korisnik ima mogućnost da izabere datum i vremenski broj za čas koji želi da otkaže. Iterira se kroz listu appointments, kako bi se pronašao željeni termin. Ukoliko je datum ispravan, a vremenski broj nije, od korisnika se zahteva da unese validan broj. Nakon toga, pozivaju se funkcije Appointment.cancel\_appointment() i Appointment.save(). Ako termin nije pronađen, izlazi se iz funkcije.



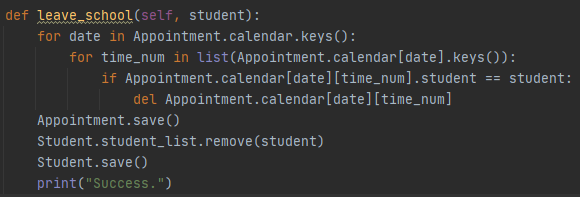
Slika 9.4

Funkcija view\_schedules() (9.5) iterira kroz listu zakazanih termina za određenog studenta i pomoću f-strings prikazuje formatirane podatke o zakazanom terminu. Nakon toga, vraća se lista zakazanih termina.



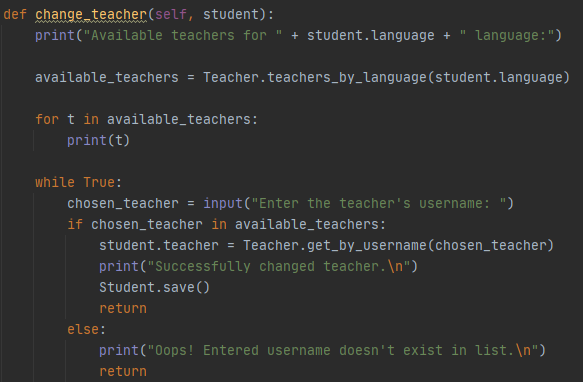
Slika 9.5

Funkcija leave\_school() (slika 9.6) pronalazi sve zakazane termine u kalendaru iterirajući kroz rečnik i podrečnik. Ukoliko pronađe zakazan termin, briše ga i čuva se lista zakazanih termina bez njega. Na kraju se briše student iz liste studenata i čuva se novi fajl.



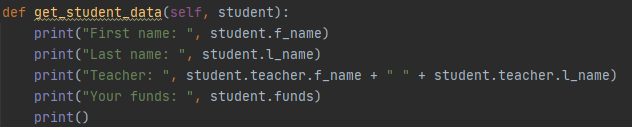
Slika 9.6

Funkcija change\_teacher() (slika 9.7) prikazuje listu korisničkih imena nastavnika koji predaju jezik koji je student odabrao pri registraciji. Potrebno je da korisnik ispravno unese neko od ponuđenih imena. Ukoliko ga pravilno unese, promenljiva teacher koja je deo objekta tipa Student menja se i na njeno mesto dolazi novi objekat pomoću get\_by\_username(). Potrebno je sačuvati izmenjenu listu studenata. Ukoliko je ime neispravno, izlazi se iz funkcije.



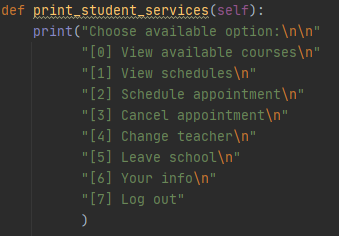
Slika 9.7

Funkcija get\_student\_data() (slika 9.8) štampa osnovne podatke o korisniku.



Slika 9.8 1

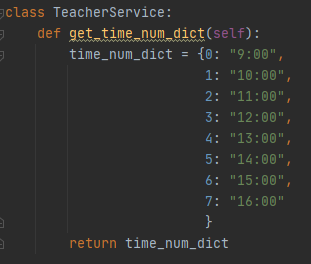
Funkcija print\_student\_service() (slika 9.9) štampa dostupne opcije koje korisnik može odabrati unoseći određeni broj iz uglaste zagrade.



Slika 9.9

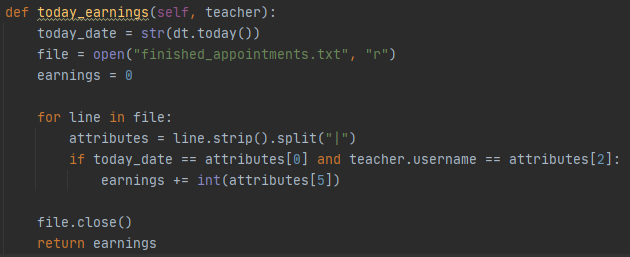
## Modul teacher\_service.py

Modul teacher\_service.py sastavljen je od istoimene klase i određenih funkcija. Prva funkcija (slika 10.0) je već opisana u modulu student\_service.py.



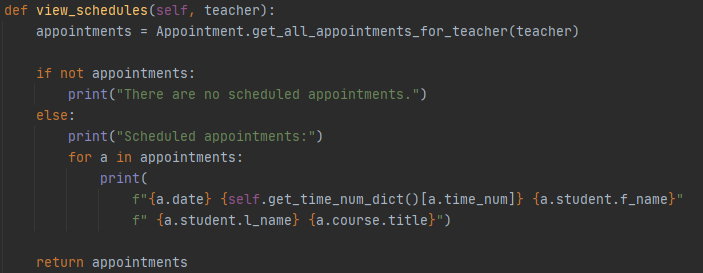
Slika 10.0

Funkcija today\_earnings() (slika 10.1) ima ulogu da vrati količinu novca koju je nastavnik zaradio tog dana. Čita se fajl finished\_appointments.txt i proverava da li postoji neki termin koji se poklapa sa današnjim danom, kao i korisničko ime nastavnika za taj termin. Ukoliko postoji, promenljiva earnings sabira se sa atributom u redu koji se nalazi na 6.mestu, a koji predstavlja cenu određenog kursa.



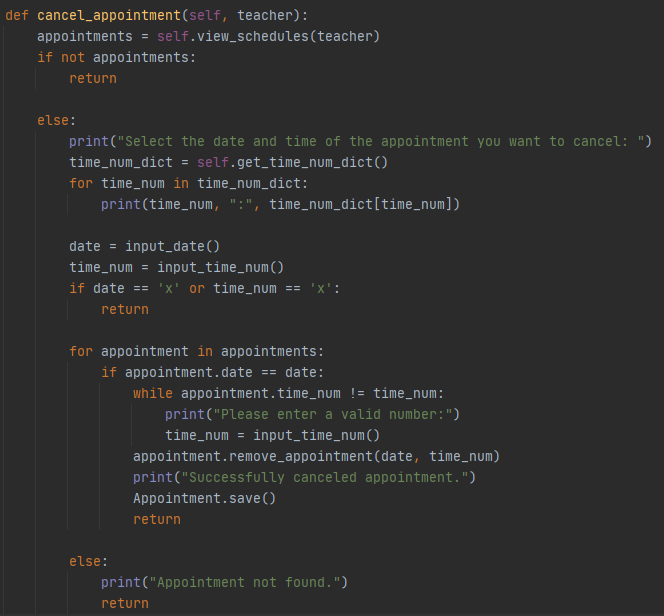
Slika 10.1

Funkcija view\_schedules() (slika 10.2) ima istu ulogu kao u prethodnom modulu, samo što se nalaze termini u zavisnosti od korisničkog imena nastavnika.

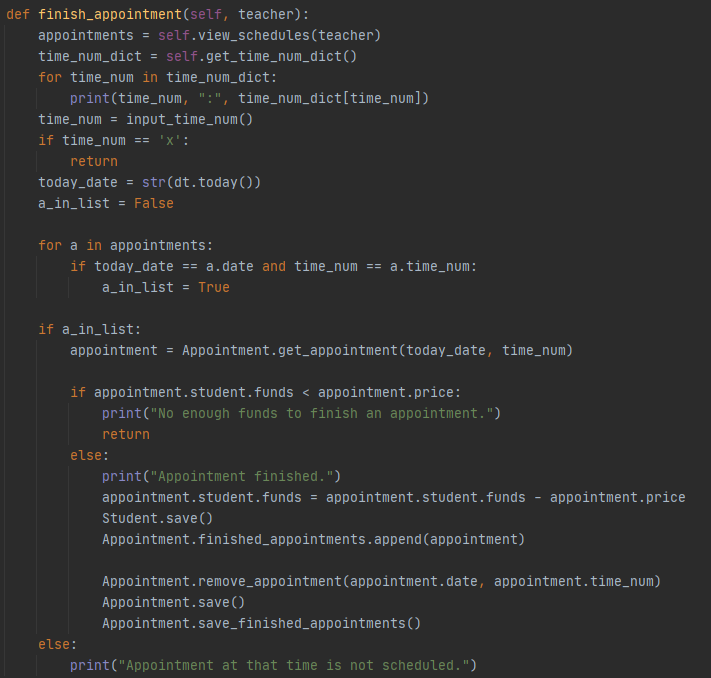


Slika 10.2

Funkcija cancel\_appointment() (slika 10.3) ne razlikuje se mnogo od istoimene funkcije iz studentskog modula, osim što se zakazani termin pretražuje na osnovu korisničkog imena nastavnika.

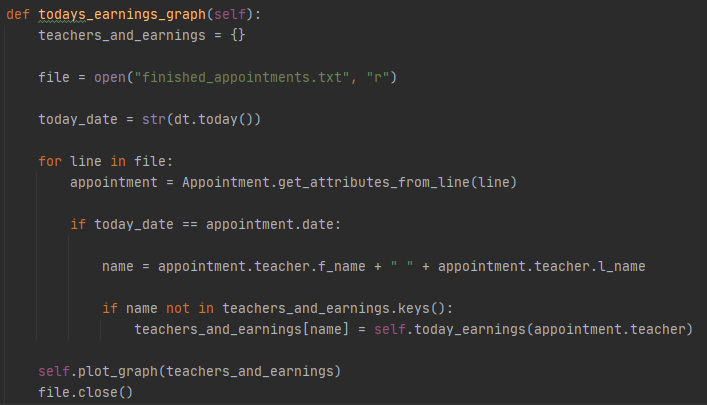


Slika 10.3

Funkcija finish\_appointment() (slika 10.4) traži od korisnika da unese određeni vremenski broj da bi odabrao termin koji će se završiti. Termin koji se može završiti je istog datuma kao i današnji dan, pa nema potrebe za unosom datuma. Ukoliko postoji termin sa unetim vremenskim brojem, potrebno je proveriti da li student ima dovoljno sredstava na računu. Ukoliko ima, cena određenog kursa oduzima se od tih sredstava, a podaci o studentu, završenim terminima i uklonjenom terminu, kao i o trenutnim zakazanim terminima čuvaju se u fajlovima za to. Ukoliko ne postoji određeni termin, ispisuje se poruka i funkcija se završava.

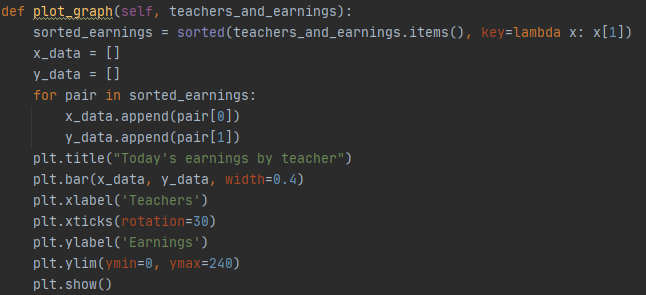
Slika 10.4

Funkcija today\_earnings\_graph() (slika 10.5) pravi rečnik za svakog nastavnika koji je održao časove tog dana, tako da su ključevi imena nastavnika, a vrednosti njihova današnja primanja tj. zarade. Prolazi kroz fajl finished\_appointments.txt, pazeći da čita samo one termine koji se poklapaju sa današnjim danom. Kada se rečnik popunio, poziva se funkcija za prikaz statistike.



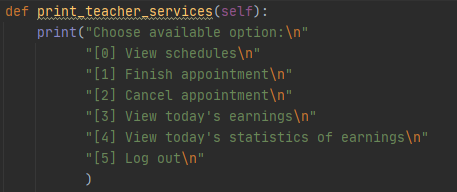
Slika 10.5

Funkcija plot\_graph() (10.6) sortira rečnik teachers\_and\_earnings i stvara listu koja sadrži liste od po dva elemenata, gde je prvi element ime nastavnika a drugi njegova zarada. Imena nastavnika smeštaju se u listu x\_data, a zarade u y\_data. Na osnovu toga prikazuje se grafikon koji prikazuje koliko je koji nastavnik zaradio tog dana.



Slika 10.6

Funkcija print\_teacher\_services() (slika 10.7) prikazuje koje opcije su dostupne za nastavnika koji se prijavi.

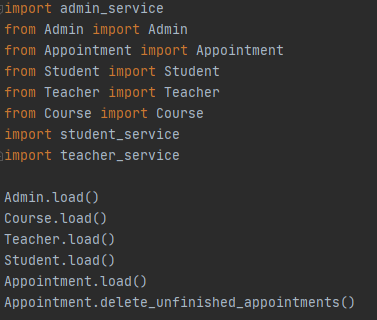


Slika 10.7

## Modul menu.py

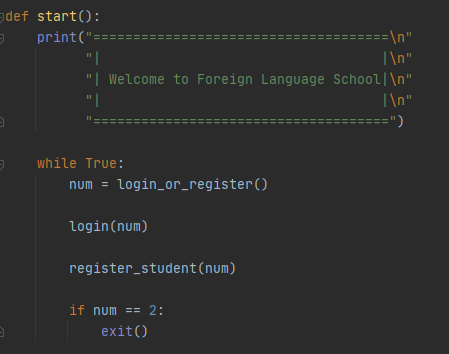
U ovom modulu izvršavaju se sve dosadašnje glavne funkcije koje su prethodno navedene u opcijama studenata, administratora i nastavnika.

Na slici 11.0 vidi se uvezivanje modula i klasa koje su potrebne da bi se pozivale određene funkcije. Pre svega, potrebno je dobiti sve objekte s kojima će se manipulisati, pa se pozivaju funkcije load() iz svih klasa. Posle toga se brišu termini koji nisu završeni.



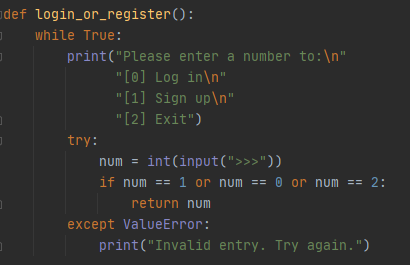
Slika 11.0

Funkcija start() (slika 11.1) je glavna funkcija gde se izvršavaju funkcije koje će biti definisane u nastavku. U zavisnosti od broja koji se unese za promenljivu num, izvršavaće se određene funkcije.



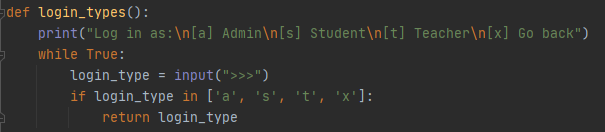
Slika 11.1

Funkcija login\_or\_register() (slika 11.2) nudi korisniku da unese određeni broj kako bi se prijavio, registrovao ili napustio program. Kao vrednost, vraća se uneti broj ukoliko je ispravan. U suprotnom se vrti while petlja.



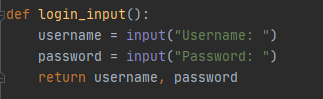
Slika 11.2

Funkcija login\_types() (slika 11.3) nudi korisniku da se prijavi na određen način. Ovakav unos osmišljen je zbog jednostavnijeg pretraživanja korisničkih imena i lozinki. Potrebno je uneti neko od ponuđenih slova kako bi se izašlo iz petlje i funkcije.



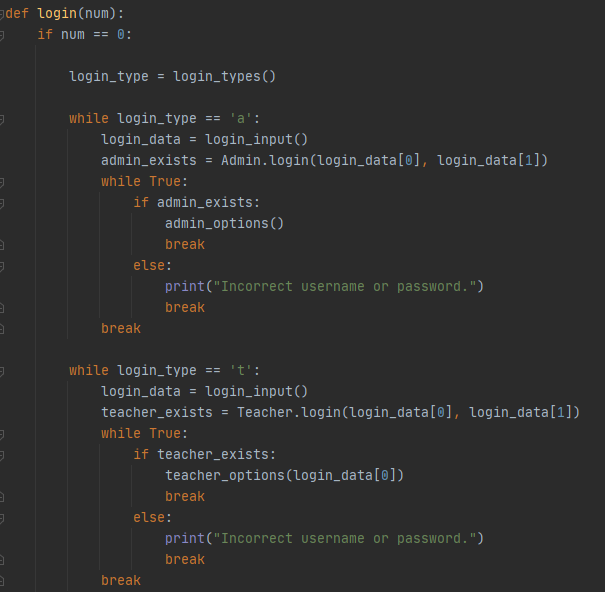
Slika 11.3

Zbog potrebe za unosom korisničkog imena i lozinke, pravi se funkcija login\_input() (slika 11.4) koja će vratiti unete vrednosti.

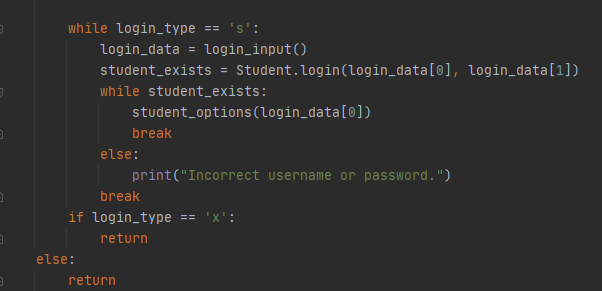


Slika 11.4

Funkcija login() (slika 11.5 i 11.6) u zavisnosti od promenljive koju prima preko funkcije login\_types() ulazi u određene petlje. Te petlje služe kako bi opcije za određeni tip korisnika bile dostupne sve dok je prijavljen. Kada se unesu korisničko ime i lozinka, i kada se potvrdi da takav korisnik postoji u listi objekata, prikazuju se dostupne opcije. U suprotnom, ispisuje se poruka o neispravno unetom imenu ili lozinki i potrebno je ponovo uneti tip korisnika koji se prijavljuje.

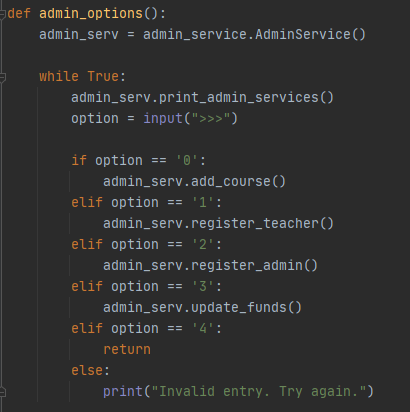


Slika 11.5



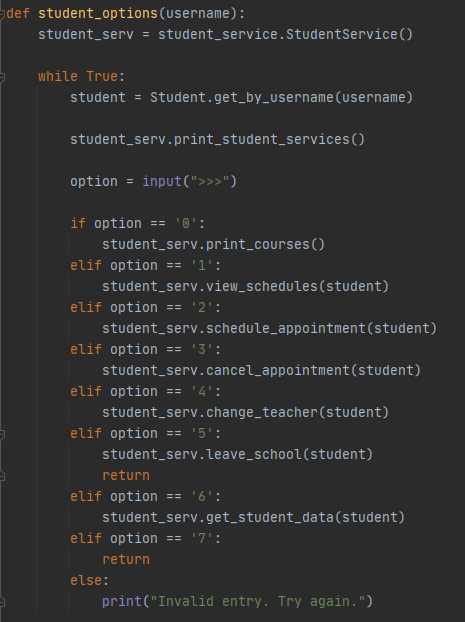
Slika 11.6

Funkcija admin\_options() (slika 11.7) prikazuje dostupne opcije i zahteva od korisnika unos željene opcije tj. broja kome pripada ta opcija. Ukoliko se uneti broj poklapa sa onim navedenim u listi opcija, poziva se ta opcija iz modula admin\_service. Ukoliko se unese vrednost koja nije predviđena, ispisuje se poruka i očekuje se ponovni unos.



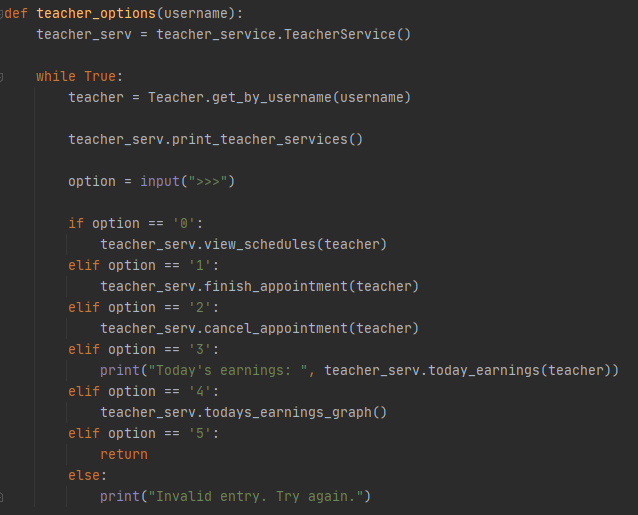
Slika 11.7

Funkcija student\_options() (slika 11.8) radi na isti način kao i prethodna, s tim što se u zavisnosti od korisničkog imena dobija objekat koji je povezan s njim.



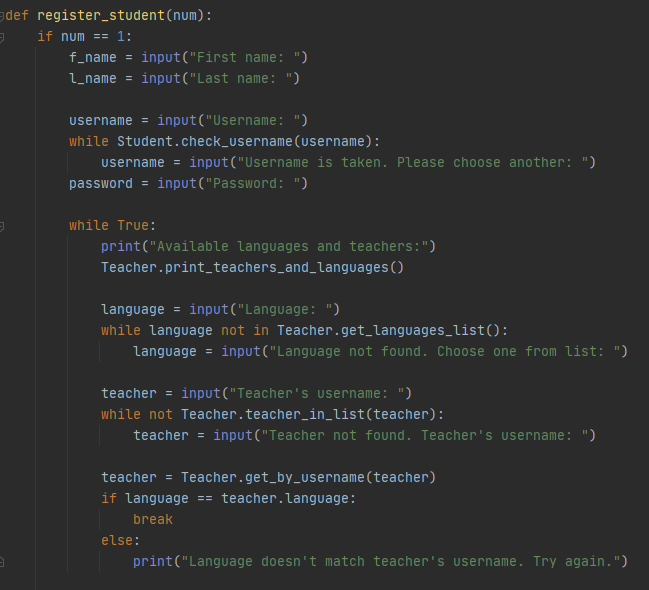
Slika 11.8

Funkcija teacher\_options() (slika 11.9) radi na isti način kao i prethodna. Dobija se objekat tipa Teacher.

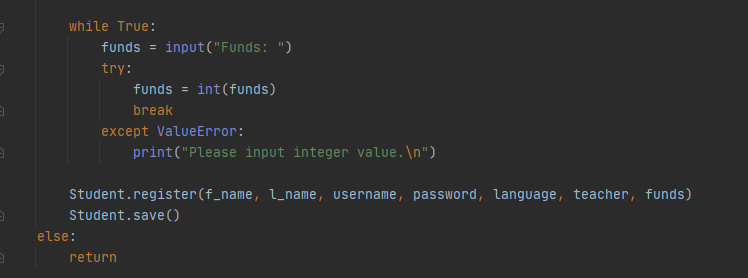


Slika 11.9

Funkcija register() (slika 11.10 i 11.11) nudi korisniku da unese svoje podatke. Pri unosu korisničkog imena proverava se da li takvo ime već postoji u listi. Ukoliko korisničko ime nastavnika , naziv jezika ili broj sredstava nije ispravno unet, mora se ponoviti unos. Na kraju se poziva funkcija register() i čuva se novi korisnik u tekstualnom fajlu.

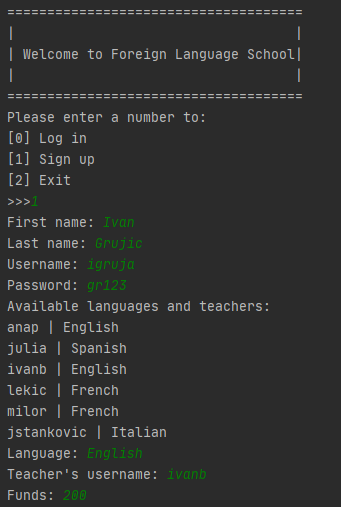


Slika 11.10

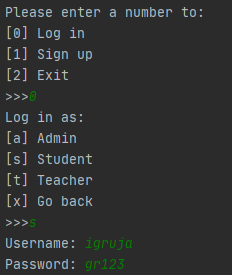


Slika 11.11

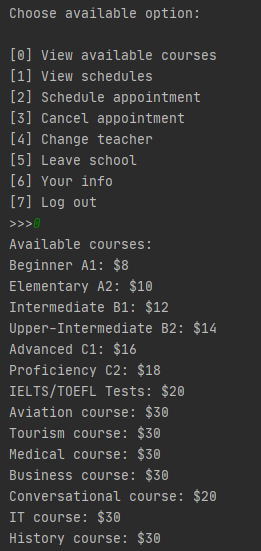
# 4. Rad aplikacije



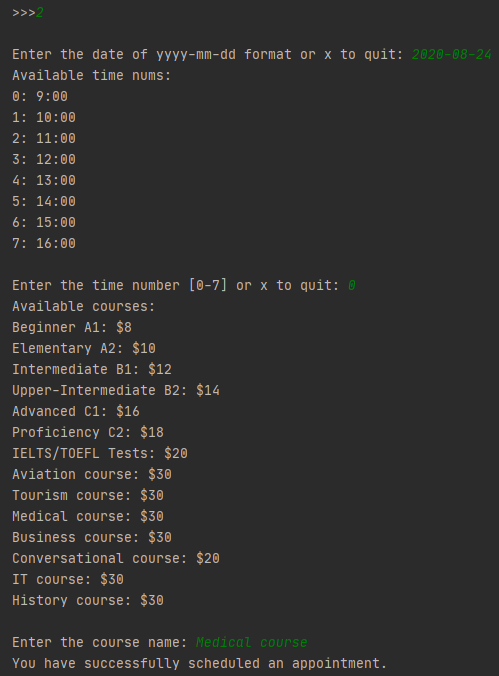
12.0 Registracija korisnika



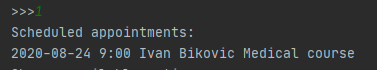
12.1 Prijava studenta



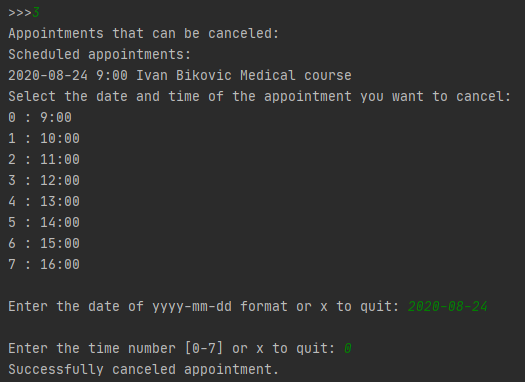
12.2 Student - opcija 0



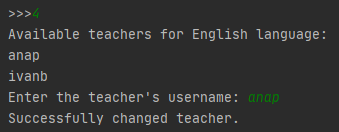
12.3 Student - opcija 2



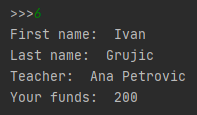
12.4 Student - opcija 1



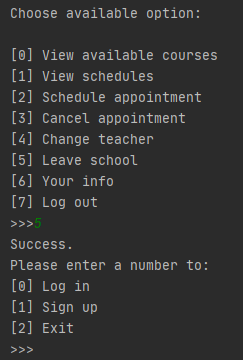
12.5 Student - opcija 3



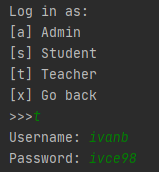
12.5 Student - opcija 4



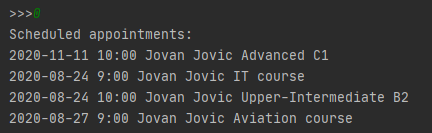
12.6 Student - opcija 6



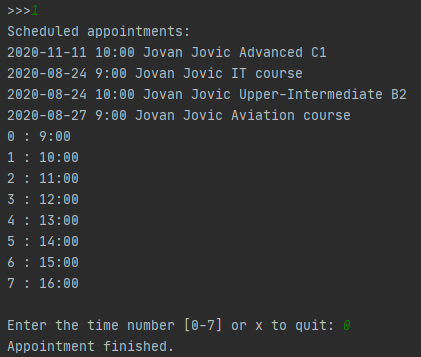
12.7 Student - opcija 5



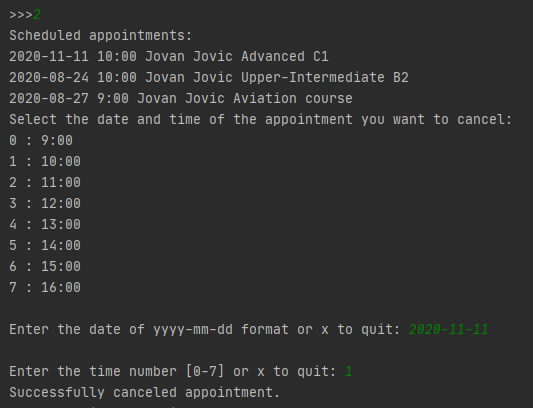
12.8 Teacher - login



12.9 Teacher - opcija 0



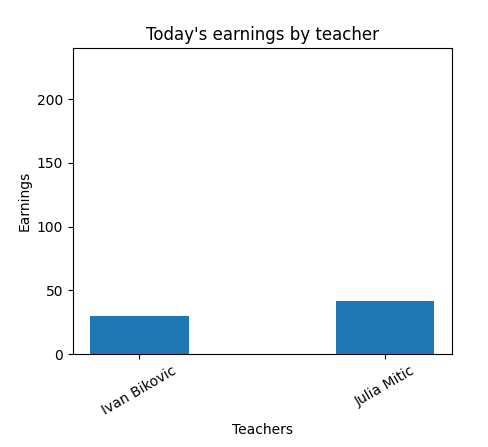
12.10 Teacher - opcija 1



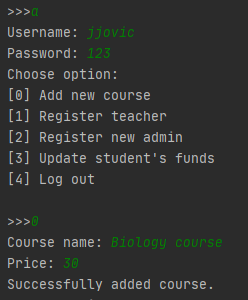
12.11 Teacher - opcija 2



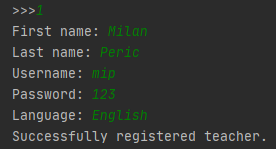
12.12 Teacher - opcija 3



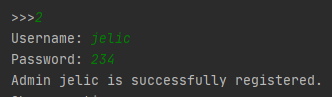
12.13 Teacher - opcija 4



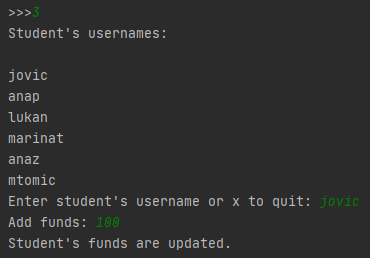
12.14 Admin - login i opcija 0



12.15 Admin - opcija 1



12.16 Admin - opcija 2



12.17 Admin - opcija 3

# 5. Zaključak

U ovoj aplikaciji urađene su osnovne stvari koje se tiču menadžmenta časova u nekoj školi stranih jezika. Obuhvaćeno je zakazivanje i otkazivanje od strane učenika, kao i završavanje(evidentiranje) časova od strane nastavnika. Aplikacija trenutno ne daje mogućnost održavanja časova istovremeno, međutim, ukoliko bi tako nešto bilo moguće(recimo da se radi o online školi), tada bi glavni fokus bio na svakom nastavniku i njegovom kalendaru. Ova aplikacija mogla bi se koristiti u realnim uslovima. Mana bi jedino bila ta što sredstva učenika ne bi bila povezana sa nekom od kartica, već bi unos sredstava morao fizički da se reguliše i proveri od strane administratora.

Unapređenje aplikacije sastojalo bi se u tome da postoji uređen korisnički interfejs koji bi olakšao korišćenje aplikacije svim korisnicima.

# 6. Literatura

1. Python Software Foundation, https://www.python.org/
2. Python Tutorial, <https://www.w3schools.com/python/default.asp>
3. <https://petlja.org/>