

Stefan Obradović, 287/21

Aplikacija za ambulantu

-seminarski rad iz predmeta Skript jezici-

Novi Sad, 2022

Sadržaj

1	Uvod	4
2	Uputstvo za upotrebu	5
2.1	Pokretanje aplikacije	5
2.2	Prijavljivanje kao doktor	5
2.3	Doktorov prozor	6
2.3.1	Padajući meni „Izaberite pacijenta”	6
2.3.2	Dugme „Učitaj podatke”	6
2.3.3	Dugme „Sačuvaj podatke”	6
2.4	Prijavljivanje kao pacijent	7
2.5	Pacijentov prozor	7
2.5.1	Dugme „Zakaži termin”	8
2.5.2	Dugme „Prikaži kao grafik”	8
2.5.3	Dugme „Osveži listu”	8
2.5.4	Dugme „Prikaži moje termine”	8
2.5.5	Dugme „Prikaži moje podatke”	8
3	Opis programa	9
3.1	Potrebni moduli i inicijalizacija aplikacije	9
3.2	Klasa „UI_login”	9
3.2.1	Funkcija „open_doktor_login”	10
3.2.2	Funkcija „open_pacijent_login”	10
3.3	Klasa „UI_doktor_login”	10
3.3.1	Funkcija „doktor_ulogovan”	10
3.3.2	Funkcije „doktor_registrovanje” i „doktor_nazad”	12
3.4	Klasa „UI_doktor_registracija”	12
3.4.1	Funkcija „doktor_registracija”	13
3.4.2	Funkcija „nazad”	14
3.5	Klasa „UI_doktor”, dodaj povratak na pocetni	14
3.5.1	Funkcija „Ucitaj”	15
3.5.2	Funkcija „sacuvaj”	15
3.5.3	Funkcija „povratak_doktor”	15
3.6	Klasa „UI_pacijent_login”	16
3.6.1	Funkcija „pacijent_ulogovan”	16
3.6.2	Funkcije „pacijent_registrovanje” i „nazad”	18
3.7	Klasa „UI_pacijent_registracija”	18
3.7.1	Funkcija „pacijent_registrovan”	19
3.7.2	Funkcija „nazad”	21
3.8	Klasa „UI_pacijent”	21
3.9	Funkcija „zakazi”	21
3.9.1	Funkcija „osvezi”	23
3.9.2	Funkcija „prikaziGrafik”	23
3.9.3	Funkcija „termin”	24
3.9.4	Funkcija „podaci”	25

4 Zaključak	26
Literatura	27

1 Uvod

Program će biti namenjen da ga koriste i doktori i pacijenti. Funkcije koje će **doktor** imati na raspolaganju su: pregled pacijenata koji su zakazani kod njega i menjanje podataka izabranog pacijenta. Funkcije koje će **pacijent** imati na raspolaganju su: mogućnost zakazivanja termina kod doktora, pregled svojih zakazanih termina, pregled broja koliko svaki doktor ima zakazanih pacijenata u formi spiska i grafika.

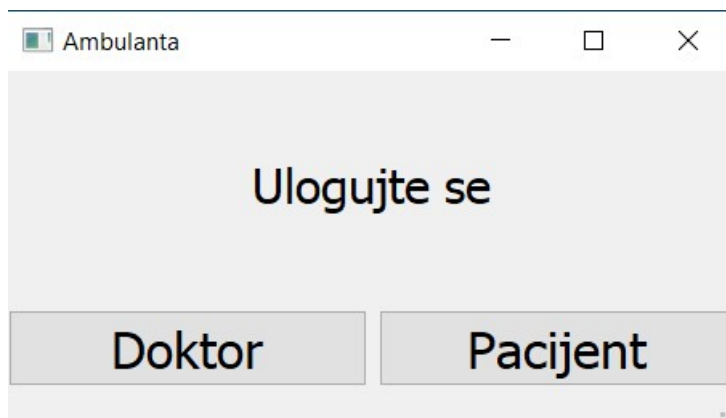
Formatiranje fajlova gde su upisani svi registrovani doktori je isto kao i ono gde su upisani svi registrovani pacijenti: **ID_doktora(pacijenta):Ime_doktora(pacijenta):šifra:**

Formatiranje fajlova za zakazane termine: **ID_pacijenta:ID_doktora:mesec:dan:sat:minut:**

2 Uputstvo za upotrebu

2.1 Pokretanje aplikacije

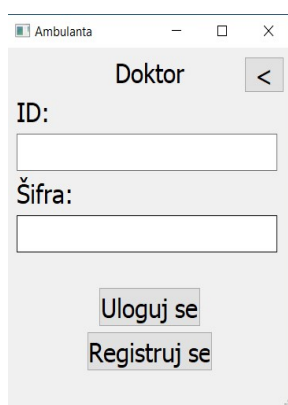
Korisnik ima dve mogućnosti. Da se prijavi ili pacijent.



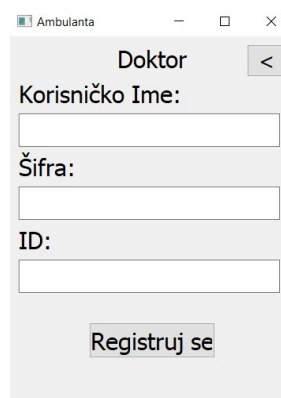
Slika 1: Početni prozor

2.2 Prijavljivanje kao doktor

U slučaju da korisnik izabere opciju doktor, biće mu prikazan prozor na kojem će moći da se prijavi ili da odabere opciju da se registruje, što će ga odvesti na drugi prozor.



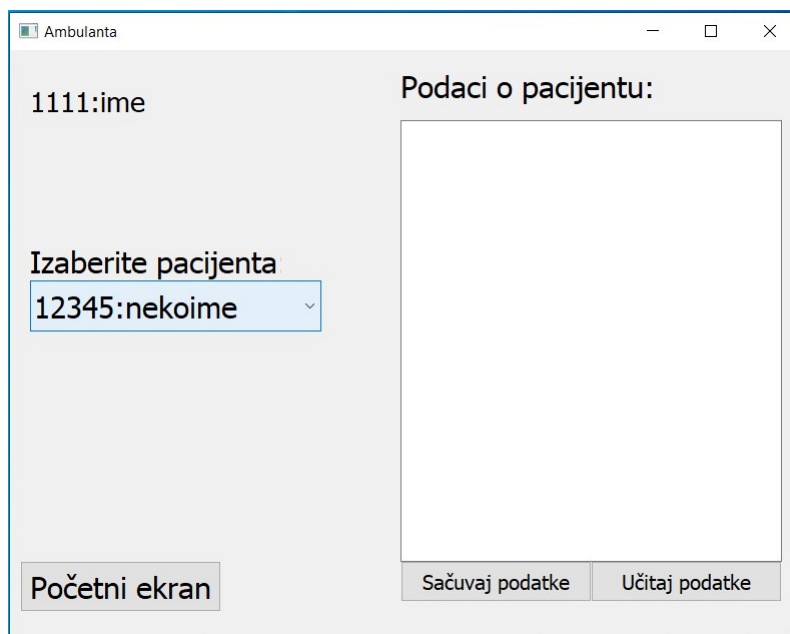
Slika 2: Doktor prijavljivanje



Slika 3: Doktor registrovanje

2.3 Doktorov prozor

Nakon prijavljivanja ili registrovanja, korisniku će biti prikazan doktorov prozor i biće mu dostupne sve njegove funkcije.



Slika 4: Doktorov prozor

2.3.1 Padajući meni „Izaberite pacijenta”

Ovde doktor može da izabere sve pacijente koji imaju zakazan termin kod njega. U slučaju da nema pacijenata pišaće „Nemate pacijenata”.

2.3.2 Dugme „Učitaj podatke”

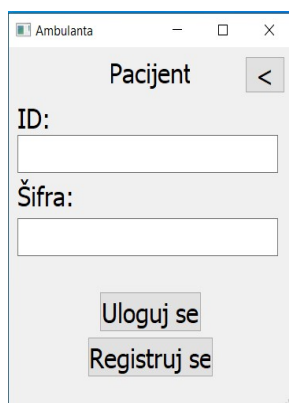
Pritiskom na dugme „**Učitaj podatke**”, u prostoru za tekst pojaviće se svi pacijentovi podaci, ako nema podataka taj prozor će ostati prazan i napraviće se novi fajl „ID_pacijenta.txt”.

2.3.3 Dugme „Sačuvaj podatke”

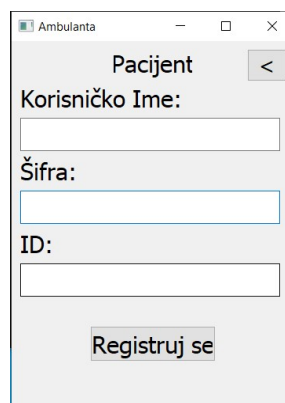
Pritiskom na dugme „**Sačuvaj podatke**”, svi podaci koji su napisani u prostoru za tekst će tako biti sačuvani u fajl „ID_pacijenta.txt”

2.4 Prijavljivanje kao pacijent

U slučaju da korisnik izabere opciju pacijent, biće mu prikazan prozor na kojem može da se prijavi ili da izabere opciju da se registruje.



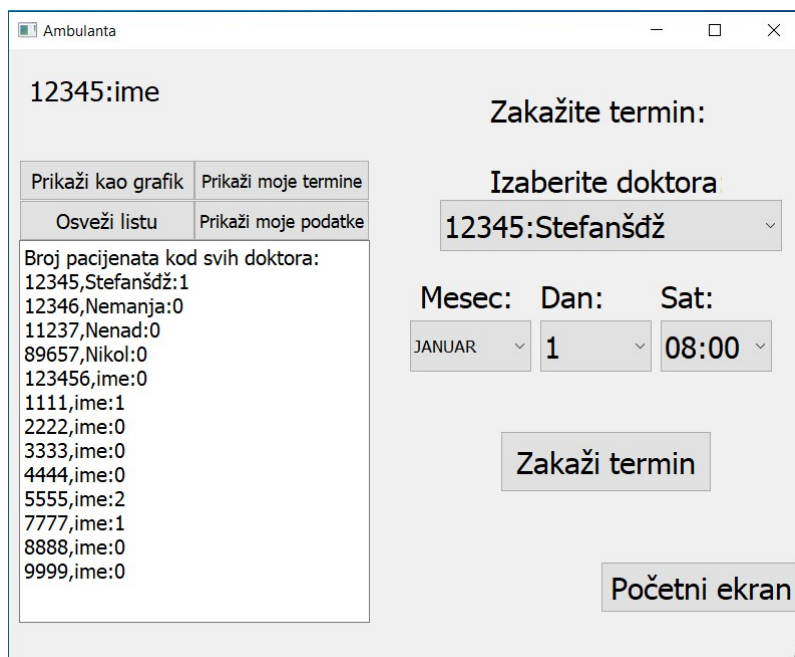
Slika 5: Pacijent prijavljivanje



Slika 6: Pacijent registrovanje

2.5 Pacijentov prozor

Nakon što se pacijent prijavi ili registruje, biće mu prikazan pacijentov prozor i biće mu dostupne sve pacijentove funkcije.



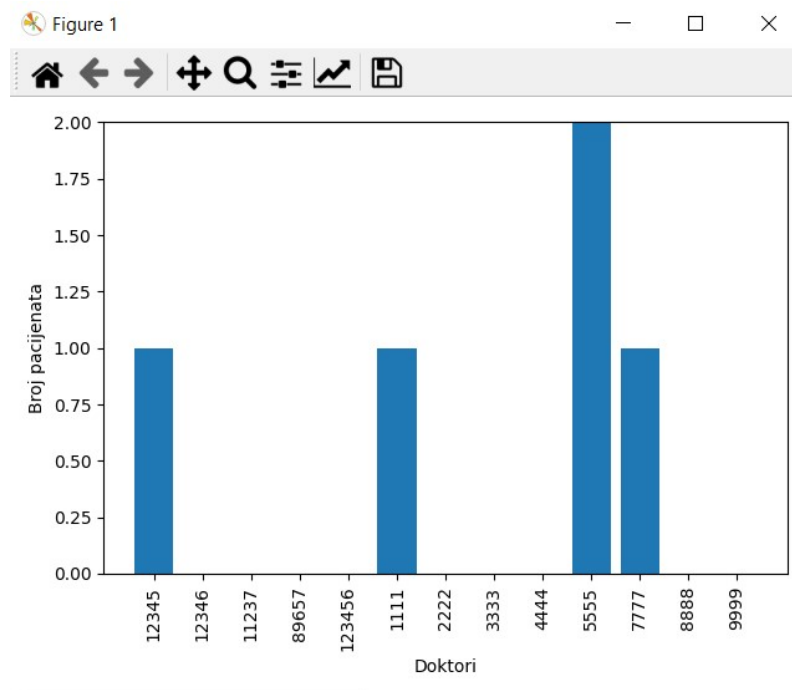
Slika 7: Pacijentov prozor

2.5.1 Dugme „Zakaži termin”

Pritiskom na dugme „**Zakaži termin**”, pacijent će zakazati termin kod izabranog doktora u izabranom terminu. U zavisnosti da li je termin zauzet ili nije, korisnik će dobiti odgovarajuću poruku.

2.5.2 Dugme „Prikaži kao grafik”

Pritiskom na dugme „**Prikaži kao grafik**”, spisak broja pacijenata kod svih doktora se ažurira i takav spisak će biti prikazan kao grafik u drugom prozoru.



Slika 8: Primer grafika

2.5.3 Dugme „Osveži listu”

Pritiskom na dugme „**Osveži listu**”, ažuriraće se spisak svih doktora sa brojem zakazanih pacijenata.

2.5.4 Dugme „Prikaži moje termine”

Pritiskom na dugme „**Prikaži moje termine**”, u prostoru za tekst ispisaće se zakazani termini prijavljenog pacijenta, u formatu **Mesec:dan:sat kod doktora: ID.doktora**. U slučaju da pacijent nema zakazanih termina dobiće odgovarajuću poruku.

2.5.5 Dugme „Prikaži moje podatke”

Pritiskom na dugme „**Prikaži moje podatke**”, učitace se pacijentovi podaci. U slučaju da nema podataka, napraviće fajl „ID.pacijenta.txt” i ispisaće poruku „Nema unetih podataka”.

3 Opis programa

Program je podeljen na nekoliko klasa: `UI_login`, `UI_doktor_login`, `UI_doktor_registracija`, `UI_doktor`, `UI_pacijent_login`, `UI_pacijent_registracija`, `UI_pacijent`. Svaka klasa predstavlja jedan prozor programa.

3.1 Potrebni moduli i inicijalizacija aplikacije

Iz modula `PyQt5` potrebni su moduli koji su upotrebljeni u programu, a upotrebljeni su u pravljenju korisničkog interfejsa, na primer svi prozori su „`QMainWindow`”.

Aplikacija se može pokrenuti dvoklikom (nije potrebno pokretanje kroz IDE), za pokretanje bez konzole koristi se fajl sa ekstenzijom `.pyw`. Pri pokretanju aplikacije `UIwindow` promenljiva postaje instanca klase „`UI_login`”.

Listing 1: Potrebni moduli

```
1 from array import array
2 from ast import Try
3 from encodings import utf_8
4 import matplotlib.pyplot as plt
5 from PyQt5 import QtWidgets
6 from PyQt5.QtWidgets import QMainWindow, QApplication, QLabel, QPlainTextEdit
7 from PyQt5.uic import loadUi
8 import sys
```

Listing 2: Pokretanje aplikacije

```
642 app = QApplication(sys.argv)
643 UIWindow = UI_login()
644 app.exec_()
```

3.2 Klasa „`UI_login`”

Klasa „`UI_login`” predstavlja prozor sa *slike 1*. Ima dve funkcije: „`open_doktor_login`” i „`open_pacijent_login`”.

Listing 3: Klasa: `UI_login`

```
12 class UI_login(QMainWindow):
13     def __init__(self):
14         super(UI_login, self).__init__()
15
16         #ucitavanje UI
17         loadUi("Interfejs/login.ui", self)
18
19         self.dugmePocetniDoktor.clicked.connect(self.open_doktor_login)
20         self.dugmePocetniPacijent.clicked.connect(self.open_pacijent_login)
21
22         #prikazivanje intefejsa
23         self.show()
```

3.2.1 Funkcija „open_doktor_login”

Funkcija „open_doktor_login” ima ulogu da korisniku prikaže prozor za prijavljivanje doktora.

Listing 4: Funkcija: open_doktor_login

```
25 def open_doktor_login(self):
26     self.window = UI_doktor_login()
27     self.close()
```

3.2.2 Funkcija „open_pacijent_login”

Funkcija „open_pacijent_login” ima ulogu da korisniku prikaže prozor za prijavljivanje pacijenata.

Listing 5: Funkcija: open_pacijent_login

```
29 def open_pacijent_login(self):
30     self.window = UI_pacijent_login()
31     self.close()
```

3.3 Klasa „UI_doktor_login”

Klasa „UI_doktor_login” predstavlja prozor sa *slike 2*. Klasa ima funkcije: „doktor_ulogovan”, „doktor_registrovanje” i „nazad”. Polje „setEchoMode” sakriva tekst pri unosu.

Listing 6: Klasa: UI_doktor_login

```
34 class UI_doktor_login(QMainWindow):
35
36     def __init__(self):
37         super(UI_doktor_login, self).__init__()
38
39
40         loadUi("Interfejs/dokLogin.ui", self)
41         self.doktor_login.clicked.connect(self.doktor_ulogovan)
42         self.doktor_registruj.clicked.connect(self.doktor_registrovanje)
43         self.doktorLoginNazad.clicked.connect(self.nazad)
44         #sakrivena sifra
45         self.sifra_doktor_login.setEchoMode(QtWidgets.QLineEdit.Password)
46
47         self.show()
```

3.3.1 Funkcija „doktor_ulogovan”

Funkcija „doktor_ulogovan”, prvo proverava da li su u prostore za tekst uneti odgovarajući karakteri. U ID ne mogu biti uneta slova i „:”, a u šifru ne može biti uneta „:”.

Nakon toga se proverava da li je fajl sa doktorima prazan, a ako jeste korisnik će dobiti poruku da nije registrovan, ako nije nastavlja se sa proverom. Prvo se linija fajla podeli na četiri dela i upoređuje se uneti ID i šifra sa podacima is fajla. Ukoliko je doktor registrovan otvoriće se prozor za doktora i biće ispisano njegov ID i ime u gornjem levom ćošku. Pored ovoga imaće spisak njegovih pacijenata u padajućem meniju. Ovaj spisak se pravi tako što se kroz fajl „termini.txt” prolazi i upoređuju se ID doktora, sa ID-om doktora iz fajla i tamo gde se taj ID poklapa, taj pacijent će biti dodat u spisak. U slučaju da doktor nema pacijenata dobiće poruku da nema pacijenata.

Listing 7: Klasa: UI_doktor_login

```

53 def doktor_ulogovan(self):
54     #podeljeno je u više redova da bi stalo na stranicu
55     slova = ["a", "b", "v", "g", "d", "đ", "e", "ž", "z", "i", "j", "k", "l", "m", "n", "o",
56     "p", "r", "s", "t", "ć", "u", "f", "h", "c", "š", "A", "B", "V", "G", "D", "Đ", "E",
57     "Ž", "Z", "I", "J", "K", "L", "M", "N", "O", "P", "R", "S", "T", "Ć", "U", "F", "H", "C", "Š", "."]
58     dvotacka = [":"]
59
60     doktor_id = self.id_doktor_login.text()
61     doktor_sifra = self.sifra_doktor_login.text()
62     termini = open("termini.txt", "r", encoding="utf-8")
63     doktor = open("doktori.txt", "r", encoding="utf-8")
64
65     if any(element in doktor_id for element in slova):
66         self.greska_doktor_log.setText("Uneli ste pogresan karakter u ID.")
67     #podeljeno je na više redova da bi stalo na stranicu
68     elif any(element in doktor_sifra for element in dvotacka) or any(element in doktor_id for element in slova):
69         self.greska_doktor_log.setText("Uneli ste neodgovarajuće karaktere")
70
71     else:
72         test = doktor.read()
73         if test != "":
74             doktor.seek(0,0)
75             for line in doktor:
76                 #deljenje linije učitane iz fajla
77                 ID, ime, sifra, prostor = line.split(":", 3)
78                 #provera da li su parametri tacni
79                 if doktor_id == ID and doktor_sifra == sifra:
80
81                     self.window = UI_doktor()
82                     self.window.imeDoktora.setText(doktor_id + ":" + ime)
83                     pacijenti = open("pacijenti.txt", "r", encoding="utf-8")
84
85                     lista_za_doktora = []
86                     for line in termini:
87                         pacijent_broj, doktor_broj, ostalo = line.split(":", 2)
88                         if doktor_id == doktor_broj:
89                             lista_za_doktora.append(pacijent_broj)
90
91                     lista_pacijenata = []
92                     for line in pacijenti:
93                         broj, pacijent, nebitno = line.split(":", 2)
94                         if broj in lista_za_doktora:
95                             self.window.doktorCombo.addItem(broj + ":" + pacijent)

```

```

96             lista_pacijenata.append(broj)
97
98         if lista_pacijenata == []:
99             self.window.doktorCombo.addItem("Nemate_pacijenata.")
100
101
102         pacijenti.close()
103         doktor.close()
104         self.close()
105         break
106
107         self.greska_doktor_log.setText("Niste_registrovani")

```

3.3.2 Funkcije „doktor_registrovanje” i „doktor_nazad”

Funkcija „**doktor_registrovanje**”, služi da odvede korisnika na prozor za registrovanje (*slika 3*). Funkcija „**doktor_nazad**”, služi da korisnika na prethodni prozor.

Listing 8: Funkcija: doktor_registrovanje

```

49 def nazad(self):
50     self.window = UI_login()
51     self.close()

```

Listing 9: Funkcija: doktor_nazad

```

107 def nazad(self):
108     self.window = UI_login()
109     self.close()

```

3.4 Klasa „UI_doktor_registracija”

Klasa „**UI_doktor_registracija**” predstavlja prozor sa *slike 3*. Služi da omogući registraciju doktora. Ima dve funkcije: „**doktor_registracija**” i „**nazad**”.

Listing 10: Klasa UI_doktor_registracija

```

112 class UI_doktor_registracija(QMainWindow):
113
114     def __init__(self):
115         super(UI_doktor_registracija, self).__init__()
116
117         loadUi("Interfejs/dokRegistracija.ui", self)
118         self.doktor_registruj_reg.clicked.connect(self.doktor_registracija)
119         self.doktorRegistracijaNazad.clicked.connect(self.nazad)
120         #sakrivena sifra
121         self.sifra_doktor_reg.setEchoMode(QtWidgets.QLineEdit.Password)
122
123         self.show()

```

3.4.1 Funkcija „doktor_registracija”

Prvo se izvršava provera da li su uneti odgovarajući karakteri u prostore za unos teksta. U ime ne može biti unet broj, u ID ne mogu biti uneti slova, a ni u jedan prostor za tekst ne može biti unesena dvotačka. Nakon toga se proverava da li je unet ID već postojeći, a ako jeste korisnik će dobiti odgovarajuću poruku. Nakon uspešne registracije izvršava se deo koda koji uneti podatke u fajl gde se čuvaju podaci o svim doktorima, i otvoriće se prozor za doktore gde oni imaju pregled svojih pacijenata.

Listing 11: funkcija „doktor_registracija”

```
129 def doktor_registracija(self):
130     #podeljeno na delove da stane na stranicu
131     slova = ["a", "b", "v", "g", "d", "d", "e", "z", "z", "i", "j", "k", "l", "m", "n", "o",
132     "p", "r", "s", "t", "c", "u", "f", "h", "c", "s", "A", "B", "V", "G", "D", "D", "E",
133     "Z", "Z", "I", "J", "K", "L", "M", "N", "O", "P", "R", "S", "T", "C", "U", "F", "H", "C", "S", ":", "]
134     dvotacka = [":"]
135     brojevi = ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9"]
136     doktor_id = self.id_doktor_reg.text()
137     doktor_ime = self.ime_doktor_reg.text()
138     doktor_sifra = self.sifra_doktor_reg.text()
139
140     if len(doktor_id) == 0 or len(doktor_ime) == 0 or len(doktor_sifra) == 0:
141         self.greska_doktor_reg.setText("Popunite sva polja")
142
143     elif len(doktor_id) > 6:
144         self.greska_doktor_reg.setText("Uneli ste previše karaktera za ID (>6)")
145
146     #podeljeno u dva reda da stane na stranicu
147     elif any(element in doktor_id for element in dvotacka) or any(element in doktor_sifra for element in dvotacka)
148     or any(element in doktor_ime for element in dvotacka) :
149         self.greska_doktor_reg.setText("Ne mozete upotrebiti: " + ":", ")
150
151     elif any(element in doktor_id for element in slova):
152         self.greska_doktor_reg.setText("Ne mozete upotrebiti: " + ":", ")
153
154     elif any(element in doktor_ime for element in brojevi):
155         self.greska_doktor_reg.setText("Ne mozete koristiti brojeve u imenu.")
156
157     else:
158         termini = open("termini.txt", "r", encoding="utf-8")
159         doktor = open("doktori.txt", "r", encoding="utf-8")
160         doktor_provera_lista = []
161         for line in doktor:
162             doktor_id_provera, ostalo = line.split(":", 1)
163             doktor_provera_lista.append(doktor_id_provera)
164
165         if doktor_id in doktor_provera_lista:
166             self.greska_doktor_reg.setText("Ovaj ID je zauzet")
167
168         else:
169             #podesiti promenu imena doktora
170             doktor.close()
171             doktor = open("doktori.txt", "a", encoding="utf-8")
```

```

172         doktor.write(doktor_id + ":" + doktor_ime + ":" + doktor_sifra + ":" + "\n")
173         doktor.close()
174
175         self.window = UI_doktor()
176         self.window.imeDoktora.setText(doktor_id + ":" + doktor_ime)
177         pacijenti = open("pacijenti.txt", "r", encoding="utf-8")
178
179         lista_za_doktora = []
180         for line in termini:
181             pacijent_broj, doktor_broj, ostalo = line.split(":", 2)
182             if doktor_id == doktor_broj:
183                 lista_za_doktora.append(pacijent_broj)
184
185         lista_pacijenata = []
186         for line in pacijenti:
187             pacijent_id, pacijent_ime, pacijent_sifra = line.split(":", 2)
188             if pacijent_id in lista_za_doktora:
189                 self.window.doktorCombo.addItem(pacijent_id + ":" + pacijent_ime)
190                 lista_pacijenata.append(pacijent_id)
191
192         if lista_pacijenata == []:
193             self.window.doktorCombo.addItem("Nemate pacijenata.")
194
195         termini.close()
196         pacijenti.close()
197         self.close()

```

3.4.2 Funkcija „nazad”

Kod funkcije za povratak je isti kao i u prethodnoj klasi (*listing 9*), samo umesto na početni prozor, korisnik će biti vraćen na prozor za prijavljivanje doktora.

Listing 12: Funkcija: nazad

```

125 def nazad(self):
126     self.window = UI_doktor_login()
127     self.close()

```

3.5 Klasa „UI_doktor”, dodaj povratak na pocetni

Klasa „UI_doktor” predstavlja prozor sa *slike 4*. i omogućava sve doktorove funkcije. Klasa ima tri funkcije: „povratak_doktor”, „sacuvaj” i „ucitaj”.

Listing 13: Klasa: UI_doktor

```

197 class UI_doktor(QMainWindow):
198
199     def __init__(self):
200         super(UI_doktor, self).__init__()
201
202         loadUi("Interfejs/doktor.ui", self)
203

```

```

204         self.podaci_pacijenta = self.findChild(QPlainTextEdit, "podaciPacijenata2")
205         self.doktor_pocetni.clicked.connect(self.povratak_doktor)
206         self.doktorSacuvaj.clicked.connect(self.sacuvaj)
207         self.doktorUcitaj.clicked.connect(self.ucitaj)
208
209         self.show()

```

3.5.1 Funkcija „Ucitaj”

Funkcija „**Ucitaj**” služi da učitava podatke nekog pacijenta. Prvo se proveriti da li doktor ima pacijenata. U slučaju da ima, biće prikazani podaci odabranog pacijenta (ukoliko pacijent nema fajl, biće napravljen), u suprotnom neće se ništa desiti.

Listing 14: Funkcija: Ucitaj

```

211 def ucitaj(self):
212
213     pacijent = self.doktorCombo.currentText()
214     if pacijent != "Nemate_pacijenata.":
215         id_pacijenta, ime_pacijenta = pacijent.split(":", 1)
216         ime_fajla = "pacijent/" + id_pacijenta + ".txt"
217
218         try:
219             fajl_pacijent = open(ime_fajla, "r", encoding="utf-8")
220             self.podaciPacijenata.setPlainText(fajl_pacijent.read())
221
222         except:
223             fajl_pacijent = open(ime_fajla, "x", encoding="utf-8")
224             fajl_pacijent.close()

```

3.5.2 Funkcija „sacuvaj”

Funkcija „**sacuvaj**” ima ulogu da ispisan tekst u namenjenom prostoru sačuva u fajl pacijenta.

Listing 15: Funkcija: sacuvaj

```

226 def sacuvaj(self):
227     pacijent = self.doktorCombo.currentText()
228     if pacijent != "Nemate_pacijenata.":
229         id_pacijenta, ime_pacijenta = pacijent.split(":", 1)
230         ime_fajla = "pacijent/" + id_pacijenta + ".txt"
231
232         fajl_pacijent = open(ime_fajla, "w", encoding="utf-8")
233         fajl_pacijent.write(self.podaciPacijenata.toPlainText())
234         fajl_pacijent.close()

```

3.5.3 Funkcija „povratak_doktor”

Funkcija „**povratak_doktor**” ima ulogu da korisnika vrati na početni prozor (*slika 1*).

Listing 16: Funkcija: povratak_doktor

```
236 def povratak_doktor(self):
237     self.window = UI_login()
238     self.close()
```

3.6 Klasa „UI_pacijent_login”

Klasa „**UI_pacijent_login**” predstavlja prozor, koji je namenjen za prijavljivanje pacijenata (*slika 5*). Ima tri funkcije: „**pacijent_ulogovan**”, „**pacijent_registrovanje**” i „**nazad**”.

Listing 17: Klasa: UI_pacijent_login

```
241 def __init__(self):
242     super(UI_pacijent_login, self).__init__()
243
244     loadUi("Interfejs/pacLogin.ui", self)
245
246     self.pacijent_login.clicked.connect(self.pacijent_ulogovan)
247     self.pacijent_registrui.clicked.connect(self.pacijent_registrovanje)
248     self.pacijentLoginNazad.clicked.connect(self.nazad)
249     #skrivena sifra
250     self.sifra_pacijent_login.setEchoMode(QtWidgets.QLineEdit.Password)
251
252     self.show()
```

3.6.1 Funkcija „pacijent_ulogovan”

Funkcija „**pacijent_ulogovan**” služi da već registrovanog pacijenta prijavi u njegov profil i prikaže mu prozor za pacijente (*slika 7*). Prvo se izvršava provera unetih karaktera i nakon toga počinje provera da li je pacijent registrovan. U slučaju da nije dobiće odgovarajuću poruku, a ako jeste otvoriće se prozor za pacijente i imaće pristup svim funkcijama. Kada se otvori prozor za pacijenta u prostoru za tekst ispisaće se broj pacijenata kod svih doktora (postupak objašnjen u klasi „**UI_pacijent**” *listing 26*) i u padajućem meniju se dodaju svi doktori.

Listing 18: Funkcija: pacijent_ulogovan

```
260 def pacijent_ulogovan(self):
261
262     #podeljeno na više delova da stane na stranu
263     slova = ["a", "b", "v", "g", "d", "d", "e", "ž", "z", "i", "j", "k", "l", "m", "n", "o",
264     "p", "r", "s", "t", "ć", "u", "f", "h", "c", "š", "A", "B", "V", "G", "D", "Đ", "E",
265     "Ž", "Z", "I", "J", "K", "L", "M", "N", "O", "P", "R", "S", "T", "Ć", "U", "F", "H", "C", "Š", ":"]
266     dvotacka = [":"]
267
268     pacijent_sifra = self.sifra_pacijent_login.text()
269     pacijent_id = self.id_pacijent_login.text()
270     pacijent = open("pacijenti.txt", "r", encoding="utf-8")
```



```

271 fajl_doktori = open("doktori.txt", "r", encoding="utf-8")
272 termini = open("termini.txt", "r", encoding="utf-8")
273
274 if len(pacijent_sifra) == 0 or len(pacijent_id) == 0:
275     self.pacijent_login_greska.setText("Popunite_sva_polja")
276
277 elif any(element in pacijent_sifra for element in dvotacka) or any(element in pacijent_id for element in slova):
278     self.pacijent_login_greska.setText("Uneli_ste_neodgovarajuće_karaktere")
279
280 else:
281     test = termini.read()
282     if test != "":
283         termini.seek(0,0)
284         for line in pacijent:
285             ID, ime, sifra, prostor = line.split(":", 3)
286
287             if pacijent_id == ID and pacijent_sifra == sifra:
288
289                 lista_doktora = []
290                 lista_doktora_sa_imenima = []
291                 niz_termina = array("i", [])
292
293                 for line in fajl_doktori:
294                     id_doktora, ime_doktora, sifra_doktora, prazan = line.split(":", 3)
295                     lista_doktora.append(id_doktora)
296                     lista_doktora_sa_imenima.append(id_doktora + ":" + ime_doktora)
297
298                 fajl_doktori.seek(0,0)
299                 for line in fajl_doktori:
300                     niz_termina.append(int(0))
301
302                 test = termini.read()
303
304                 if test != "":
305                     termini.seek(0,0)
306                     for line in termini:
307                         id_pacijenta, id_dok, mesec, termin = line.split(":", 3)
308                         niz_termina[lista_doktora.index(id_dok)] += 1
309
310                 lista_termina = []
311                 for i in range(len(niz_termina)):
312                     lista_termina.append(str(niz_termina[i]))
313
314                 #self.window.
315                 fajl_doktori.seek(0,0)
316                 self.window = UI_pacijent()
317
318                 self.window.brojPacijenata.setText("Broj_pacijenata_kod_svih_doktora:")
319                 for i in range(len(lista_doktora)):
320                     #podeljeno na dva reda da stane na stranu
321                     self.window.brojPacijenata.append(lista_doktora_sa_imenima[i] + ":" +
322                     lista_termina[i])
323
324                 for line in fajl_doktori:
325                     id_doktora, ime, sifra, prazan = line.split(":", 3)
326                     self.window.pacijentCombo.addItem(id_doktora + ":" + ime)

```

```

327
328             #self.window.
329             self.window.imePacijenta.setText(pacijent_id + ":" + ime)
330
331             pacijent.close()
332             termini.close()
333             fajl_doktori.close()
334             self.close()
335             break
336
337     else:
338         self.pacijent_login_greska.setText("Niste registrovani")

```

3.6.2 Funkcije „pacijent_registrovanje” i „nazad”

Funkcija „**pacijent_registrovanje**” će korisnika odvesti na prozor za registraciju pacijenata (*slika 6*), a funkcija „**nazad**” će korisnika odvesti na prethodni prozor (*slika 1*).

Listing 19: Funkcija: pacijent_registrovanje

```

335 def pacijent_registrovanje(self):
336     self.window = UI_pacijent_registracija()
337     self.close()

```

Listing 20: Funkcija: nazad

```

335 def nazad(self):
336     self.window = UI_login()
337     self.close()

```

3.7 Klasa „UI_pacijent_registracija”

Klasa „**UI_pacijent_registracija**” predstavlja prozor sa slike 6, i daje korisniku mogućnost da se registruje. Takođe ima dugme za povratak na prethodni prozor.

Listing 21: Klasa: UI_pacijent_registracija

```

340 class UI_pacijent_registracija(QMainWindow):
341
342     def __init__(self):
343         super(UI_pacijent_registracija, self).__init__()
344
345         loadUi("Interfejs/pacRegistracija.ui", self)
346
347         self.pacijent_registruj.clicked.connect(self.pacijent_registrovan)
348         self.pacijentRegistracijaNazad.clicked.connect(self.nazad)
349         #skrivenasifra
350         self.sifra_pacijent_reg.setEchoMode(QtWidgets.QLineEdit.Password)
351
352         self.show()

```

3.7.1 Funkcija „pacijent_registrovan”

Funkcija „pacijent_registrovan” služi da registruje pacijenta i prikaže korisniku prozor za pacijente (slika 7), koji će mu dati pristup svim pacijentovim funkcijama. Prvo se vrši provera unetih podataka, tj. da li je negde unet neodgovarajući karakter. Nakon toga se proverava da li je korisnik sa tim ID-om registrovan, a ako jeste korisnik će dobiti poruku, ako nije uspešno će se registrovati. Nakon registracije korisniku se prikazuje prozor za pacijente na kojem ima pristup svim pacijentovim funkcijama. Na isti način kao i u prethodnoj klasi kada se doktor prijavi, u prostoru za ispis teksta biće prikazan spisak svih doktora sa brojem njihovih pacijenata.

Listing 22: Funkcija: `pacijent_registrovan`

```
35 def pacijent_registrovan(self):
36     termini = open("termini.txt", "r", encoding="utf-8")
37
38     #podeljeno na vise delova da stane na stranicu
39     slova = ["a", "b", "v", "g", "d", "d", "e", "z", "z", "i", "j", "k", "l", "m", "n", "o",
40     "p", "r", "s", "t", "c", "u", "f", "h", "c", "s", "A", "B", "V", "G", "D", "D", "E", "Z",
41     "Z", "I", "J", "K", "L", "M", "N", "O", "P", "R", "S", "T", "C", "U", "F", "H", "C", "S", ":", "]
42     dvotacka = [":"]
43     brojevi = ["0", "1", "2", "3", "4", "5", "6", "7", "8", "9"]
44
45     pacijent_ime = self.ime_pacijent_reg.text()
46     pacijent_sifra = self.sifra_pacijent_reg.text()
47     pacijent_id = self.id_pacijent_reg.text()
48
49     if len(pacijent_ime) == 0 or len(pacijent_sifra) == 0 or len(pacijent_id) == 0:
50         self.pacijent_greska_reg.setText("Popunite sva polja")
51
52     #podeljeno na više linija da stane na stranicu
53     elif any(element in pacijent_id for element in dvotacka)
54     or any(element in pacijent_sifra for element in dvotacka)
55     or any(element in pacijent_ime for element in dvotacka):
56         self.pacijent_greska_reg.setText("Ne mozete upotrebiti: " + "_.")
57
58     elif len(pacijent_id) > 6:
59         self.pacijent_greska_reg.setText("Uneli ste previše karaktera za ID (>6)")
60
61     elif any(element in pacijent_id for element in slova):
62         self.pacijent_greska_reg.setText("Ne mozete upotrebiti: " + "_.")
63
64     elif any(element in pacijent_ime for element in brojevi):
65         self.pacijent_greska_reg.setText("Ne mozete koristiti brojeve u imenu.")
66
67     else:
68         pacijent = open("pacijenti.txt", "r", encoding="utf-8")
69
70         pacijent_provera_lista = []
71         for line in pacijent:
72             pacijentov_id, ostalo = line.split(":", 1)
73             pacijent_provera_lista.append(pacijentov_id)
74
75         if pacijent_id in pacijent_provera_lista:
```

```

76         self.pacijent_greska_reg.setText("Ovaj ID_je_već_registrovan")
77
78     else:
79         pacijent.close()
80         pacijent = open("pacijenti.txt", "a", encoding="utf-8")
81         pacijent.write(pacijent_id + ":" + pacijent_ime + ":" + pacijent_sifra + ":" + "\n")
82         pacijent.close()
83
84
85         fajl_doktori = open("doktori.txt", "r", encoding="utf-8")
86         lista_doktora = []
87         lista_doktora_sa_imenima = []
88         niz_termina = array("i", [])
89
90         for line in fajl_doktori:
91             id_doktora, ime_doktora, sifra_doktora, prazan = line.split(":", 3)
92             lista_doktora.append(id_doktora)
93             lista_doktora_sa_imenima.append(id_doktora + ":" + ime_doktora)
94
95         fajl_doktori.seek(0,0)
96
97         for line in fajl_doktori:
98             niz_termina.append(int(0))
99
100         test = termini.read()
101         if test != "":
102             termini.seek(0,0)
103             for line in termini:
104                 id_pacijenta, id_dok, mesec, termin = line.split(":", 3)
105                 niz_termina[lista_doktora.index(id_dok)] += 1
106
107         lista_termina = []
108         for i in range(len(niz_termina)):
109             lista_termina.append(str(niz_termina[i]))
110
111         # self.window.
112         fajl_doktori.seek(0,0)
113         self.window = UI_pacijent()
114
115         for i in range(len(lista_doktora)):
116             self.window.brojPacijenata.append(lista_doktora_sa_imenima[i] + ":" + lista_termina[i])
117
118
119         for line in fajl_doktori:
120             id_doktora, ime, sifra, prazan = line.split(":", 3)
121             self.window.pacijentCombo.addItem(id_doktora + ":" + ime)
122
123         self.window.imePacijenta.setText(pacijent_id + ":" + pacijent_ime)
124
125         termini.close()
126         fajl_doktori.close()
127         self.close()

```

3.7.2 Funkcija „nazad”

Ima istu ulogu kao druge funkcije „nazad”, ali uz drugačiji prozor na koji vraća korisnika. U ovom slučaju korisnik će biti vraćen na prozor za prijavu pacijenta (*slika 5*).

Listing 23: Funkcija: nazad

```
354 def nazad(self):
355     self.window = UI_pacijent_login()
356     self.close()
```

3.8 Klasa „UI_pacijent”

Klasa „UI_pacijent” predstavlja prozor sa slike 7. i daje korisniku pristup svim pacijentovim funkcijama. Ima šest funkcija: „zakazi”, „osvezi”, „prikaziGrafik”, „termin”, „podaci” i „povratak_pacijent”.

Listing 24: Klasa: UI_pacijent

```
447 class UI_pacijent(QMainWindow):
448     def __init__(self):
449         super(UI_pacijent, self).__init__()
450
451         loadUi("Interfejs/pacijent.ui", self)
452
453         self.pacijent_pogresan = self.findChild(QLabel, "pacijent_greska")
454
455         self.pacijentZakazi.clicked.connect(self.zakazi)
456
457         self.pacijent_pocetni.clicked.connect(self.povratak_pacijent)
458
459         self.pacijentPrikazi.clicked.connect(self.prikaziGrafik)
460
461         self.prikazOsvezi.clicked.connect(self.osvezi)
462
463         self.pacijentTermini.clicked.connect(self.termin)
464
465         self.pacijentPodaci.clicked.connect(self.podaci)
466
467         self.show()
```

3.9 Funkcija „zakazi”

Funkcija „zakazi” služi da omogući korisniku zakazivanje termina kod izabranog doktora u padajućem meniju. Korisnik može da izabere doktora, mesec, dan i sat pregleda. Prvo se proverava da li termin postoji zato što neki meseci nemaju 31 dan, a februar za potrebe projekta ima 28 dana bez obzira na godinu. Nakon toga se u fajlu termina uzima linija i deli se na doktorov ID, i zakazan termin, i to se stavlja u listu zakazanih termina. Termin koji je pacijent izabrao upoređuje se sa svim terminima u listi. U slučaju da je

termin zakazan, pacijent će dobiti poruku da je termin zauzet, a u suprotnom će dobiti poruku da je termin uspešno zakazan.

Listing 25: Funkcija: zakazi

```
474 def zakazi(self):
475     termini = open("termini.txt", "r", encoding="utf-8")
476
477     mesec = self.zakazivanje_mesec.currentText()
478     dan = self.zakazivanje_dan.currentText()
479     sat = self.zakazivanje_sat.currentText()
480
481     labela_ime_pacijenta = self.imePacijenta.text()
482     id_pacijenta, ime_pacijenta = labela_ime_pacijenta.split(":", 1)
483
484     zakazan = mesec + ":" + dan + ":" + sat + ":"
485
486     doktor = self.pacijentCombo.currentText()
487     id_doktora, ime_doktora = doktor.split(":", 1)
488
489     trideset = ["APRIL", "JUN", "SEPTEMBAR", "NOVEMBAR"]
490     trideset_jedan = ["JANUAR", "MART", "MAJ", "JUL", "AVGUST", "OKTOBAR", "DECEMBAR"]
491
492     if mesec == "FEBRUAR" and dan > "28":
493         self.pacijent_pogresan.setText("Termin ne postoji")
494
495     elif mesec in trideset and dan == "31":
496         self.pacijent_pogresan.setText("Termin ne postoji")
497     else:
498         lista_termina = []
499
500         test = termini.read()
501         if test != "":
502             termini.seek(0,0)
503             for line in termini:
504                 pacijent_id, doktor_id, mesec, dan, sat, minut, ostatak = line.split(":", 6)
505                 lista_termina.append(doktor_id + ":" + mesec + ":" + dan + ":" + sat + ":" + minut + ":")
506
507                 provera = id_doktora + ":" + zakazan
508
509                 if provera in lista_termina:
510                     termini.close()
511                     self.pacijent_pogresan.setText("Termin je zauzet.")
512
513                 else:
514                     termini.close()
515                     termini = open("termini.txt", "a", encoding="utf-8")
516                     termini.write(id_pacijenta + ":" + id_doktora + ":" + zakazan + "\n")
517                     termini.close()
518                     self.pacijent_pogresan.setText("Zakazali ste termin")
```

3.9.1 Funkcija „osvezi”

Funkcija „osvezi” ima ulogu da osveži spisak svih doktora i njihov broj pacijenata. Prvo se naprave liste, koje će čuvati ID-ove svih doktora i ID-ove sa imenima doktora kao i niz celobrojnih vrednosti, koji će čuvati broj zakazanih pacijenata kod svih doktora. Prvo se popune liste, nakon čega se nizu dodaje element (jednak 0) za svakog doktora iz liste. Nakon toga se prolazi kroz fajl sa zakazanim terminima i beleži se ID-doktora. U nizu se inkrementira vrednost elementa, kojem je index jednak indexu ID-u doktora u listi doktora. Nakon toga se u prostor za tekst ispisuju elementi iz liste sa imenima doktora (zbog lakšeg pregleda korisniku) i broj pacijenata tog doktora.

Listing 26: Funkcija: osvezi

```
522 def osvezi(self):
523     fajl_doktori = open("doktori.txt", "r", encoding="utf-8")
524     lista_doktora = []
525     lista_doktora_sa_imenima = []
526     niz_termina = array("i", [])
527
528     for line in fajl_doktori:
529         id_doktora, ime_doktora, sifra_doktora, prazan = line.split(":", 3)
530         lista_doktora.append(id_doktora)
531         lista_doktora_sa_imenima.append(id_doktora + "," + ime_doktora)
532
533     fajl_doktori.seek(0,0)
534     for line in fajl_doktori:
535         niz_termina.append(int(0))
536
537     termini = open("termini.txt", "r", encoding="utf-8")
538
539     for line in termini:
540         id_pacijenta, id_dok, mesec, termin = line.split(":", 3)
541         niz_termina[lista_doktora.index(id_dok)] += 1
542
543     lista_termina = []
544     for i in range(len(niz_termina)):
545         lista_termina.append(str(niz_termina[i]))
546
547     self.brojPacijenata.setText("Broj_pacijenata_kod_svih_doktora.")
548     for i in range(len(lista_doktora)):
549         self.brojPacijenata.append(lista_doktora_sa_imenima[i] + ":" + lista_termina[i])
550
551     termini.close()
552     fajl_doktori.close()
```

3.9.2 Funkcija „prikaziGrafik”

Funkcija „prikaziGrafik” će prvo osvežiti listu na način opisan u funkciji „osvezi”, pa će uzeti te podatke (listu doktora, i niz termina) i prikazati ih kao grafik u zasebnom prozoru (*slika 8*).

Listing 27: Funkcija: prikaziGrafik

```
554 def prikaziGrafik(self):
555     termini = open("termini.txt", "r", encoding="utf-8")
556     fajl_doktori = open("doktori.txt", "r", encoding="utf-8")
557     lista_doktora = []
558     lista_doktora_sa_imenima = []
559     niz_termina = array("i", [])
560
561     for line in fajl_doktori:
562         id_doktora, ime_doktora, sifra_doktora, prazan = line.split(":", 3)
563         lista_doktora.append(id_doktora)
564         lista_doktora_sa_imenima.append(id_doktora + "," + ime_doktora)
565
566     fajl_doktori.seek(0,0)
567     for line in fajl_doktori:
568         niz_termina.append(int(0))
569
570     test = termini.read()
571     if test != "":
572         termini.seek(0,0)
573         for line in termini:
574             id_pacijenta, id_dok, mesec, termin = line.split(":", 3)
575             niz_termina[lista_doktora.index(id_dok)] += 1
576
577     lista_termina = []
578     for i in range(len(niz_termina)):
579         lista_termina.append(str(niz_termina[i]))
580
581     self.brojPacijenata.setText("Broj_pacijenata_kod_svih_doktora:")
582     for i in range(len(lista_doktora)):
583         self.brojPacijenata.append(lista_doktora_sa_imenima[i] + ":" + lista_termina[i])
584
585     plt.bar(lista_doktora, niz_termina)
586     plt.xlabel("Doktori")
587     plt.xticks(rotation = 90)
588     plt.ylabel("Broj_pacijenata")
589     plt.ylim(ymin = 0, ymax = max(niz_termina))
590     plt.tight_layout()
591     plt.show()
592     termini.close()
593     fajl_doktori.close()
```

3.9.3 Funkcija „termin”

Funkcija „**termin**” koristi se da prikaže zakazane termine prijavljenog pacijenta. Prvo se napravi lista termina, u koju će se pisati svi termini prijavljenog pacijenta. Prolaskom kroz fajl u kom su zakazani termini, iz svake linije se izvlači ID pacijenta, i ako se ID pacijenta podudara sa ID-om prijavljenog pacijenta, taj termin će se dodati u listu. Na kraju se lista proverava da li je prazna, u tom slučaju pacijent će dobiti poruku da nema zakazan termin.

Listing 28: Funkcija: termin

```
595 def termin(self):
596
597     termini = open("termini.txt", "r", encoding="utf-8")
598     lista_termina = []
599     pacijent = self.imePacijenta.text()
600     pacijent_id, pacijent_ime = pacijent.split(":", 1)
601
602     self.brojPacijenata.setText("Vaši zakazani termini:")
603
604     test = termini.read()
605     if test != "":
606         termini.seek(0,0)
607         for line in termini:
608             id_pacijent, id_doktor, zakazan_termin = line.split(":", 2)
609
610             if pacijent_id == id_pacijent:
611                 lista_termina.append(pacijent_id)
612                 #podeljeno na 2 reda da stane na stranicu
613                 self.brojPacijenata.append(zakazan_termin + "_kod_doktora:"
614                                             + id_doktor + "\n")
615
616         if lista_termina == []:
617             self.brojPacijenata.append("Nemate zakazan termin.")
618
619     termini.close()
```

3.9.4 Funkcija „podaci”

Funkcija „**podaci**” služi da pacijentu prikaže njegove podatke. U slučaju da fajl postoji, podaci iz fajla biće ispisani u prostoru za tekst, a ako fajl ne postoji, taj fajl će biti napravljen i pacijent će dobiti poruku da nema unetih podataka (ova poruka neće biti upisana u fajl).

Listing 29: Funkcija: podaci

```
623 def podaci(self):
624     labela = self.imePacijenta.text()
625     pacijent_id, pacijent_ime = labela.split(":", 1)
626
627     fajl = "pacijent/" + pacijent_id + ".txt"
628     try:
629         fajl_pacijenta = open(fajl, "x", encoding="utf-8")
630         self.brojPacijenata.setText("")
631         self.brojPacijenata.setText("Nema unetih podataka")
632         fajl_pacijenta.close()
633
634     except:
635         fajl_pacijenta = open(fajl, "r", encoding="utf-8")
636         ispis = fajl_pacijenta.read()
637         self.brojPacijenata.setText("")
638         self.brojPacijenata.append(ispis)
639         fajl_pacijenta.close()
```

4 Zaključak

Napravljena je aplikacija za ambulantu. Funkcije su zakazivanje termina i upisivanje podataka u fajl pacijenta.

Moguće nadogradnje su dodavanje još funkcija, izbor doktora specijaliste za neku oblast (zubar, dermatolog, opšta praksa...). Pored nadogradnji dodavanjem funkcija, može se dodatno optimizovati trenutni kod, pravljenje više .py fajlova za svaku klasu da kod bude pregledniji.

Moguće primene ovog programa su u ambulantama da se modernizuje zdravstveni sistem. Prvenstveno zbog podataka o pacijentima jer mnogo ljudi ima nekoliko kartona u različitim mestima i u svakom piše mali deo cele zdravstvene istorije pacijenta.

Dodatne prednosti digitalnih kartona bile bi te što zbog činjenice da su svi podaci na jednom mestu i kad se desi neka nesreća doktori u urgentnom centru mogu da traže podatke o nekom pacijentu, gde imaju napisane sve njihove prethodne terapije da pri lečenju tom pacijentu ne daju lek na koji su alergični ili bi imao kontra efekat zbog drugih lekova.

Literatura

- [1] dr Jovana Vidaković, kurs Skript jezici
<https://moodle.pmf.uns.ac.rs/course/view.php?id=517>
- [2] Python Software Foundation
<https://www.python.org/doc/>
- [3] Qt for Python, dokumentacija QT dizajnera
<https://doc.qt.io/qtforpython-5/>
- [4] Geeks-for-Geeks, internet stranica
<https://www.geeksforgeeks.org/python-programming-language/>
- [5] W3Schools, internet stranica
<https://www.w3schools.com/python/>
- [6] Code First with Hala, youtube profil
<https://www.youtube.com/c/CodeFirstio>