IZRADA WEB APLIKACIJE ZA ORDINACIJU OPĆE PRAKSE

Pribudić, Ivo

Undergraduate thesis / Završni rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split / Sveučilište u Splitu**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:228:257127

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-08-04**

Repository / Repozitorij:

Repository of University Department of Professional Studies

zir.nsk.hr

UNIVERSITY OF SPLIT

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

IVO PRIBUDIĆ

Z A V R Š N I R A D

IZRADA WEB APLIKACIJE ZA ORDINACIJU OPĆE PRAKSE

Split, rujan 2022.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

Predmet: Programiranje na internetu

Z A V R Š N I R A D

Kandidat: Ivo Pribudić

Naslov rada: Izrada web aplikacije za ordinaciju opće prakse

Mentor: Marina Rodić, viši predavač

Split, rujan 2022.

Sadržaj

Sažetak 1
Summary 2
1. Uvod 3
2.1. Django 4
2.2. Sustav za upravljanje bazama podataka 5
2.3. HTML 5
2.4. CSS i Bootstrap 6
2.5. Chart.js 6
2.6. ReportLab 7
3. Model sustava 8
4. Opis aplikacije 10
4.1. Opis aplikacije – Doktor 10
4.2. Opis aplikacije – Administrator 15
4.3. Opis aplikacije – Pacijent 24
5. Zaključak 29
Literatura 30

Sažetak

Cilj ovog rada je napraviti sustav za ordinaciju opće prakse. Taj sustav bi razlikovao tri uloge: administrator, doktor i pacijent.

Administrator ima najviše obveza u sustavu, on dodjeljuje pacijentima doktore, dodaje nove doktore u sustav, dodaje nove kategorije bolesti te ima pregled u iste, može pregledati statistiku pojave tipova bolesti kroz godine i ima pregled u sve pacijente te ih može ažurirati. Administrator može obavljati iste dužnosti kao i doktor uz poneke dodatne funkcionalnosti koje su rezervirane samo za administratora.

Doktor može pregledavati listu svojih pacijenata, odobriti, odbiti i odgoditi njihove termine. Može pregledati profile svojih pacijenata gdje se nalaze podatci pacijenta, popis datuma kada je taj pacijent imao termin te može generirati PDF u kojem se vide podatci vezani za taj termin.

Pacijenti mogu pregledati svoj profil, poslati zahtjev za termin koji doktor treba odobriti te u slučaju da je doktor odgodio njihov termin prihvatiti novi ili otkazati.

Ključne riječi: django, web aplikacija, ordinacija opće prakse

Summary

General practice clinic application

The goal of this project is to create an application for a general practice clinic. The application would differentiate between three roles: administrators, doctors and patients.

The administrator has the most responsibilities, he assigns doctors to patients, adds new doctors to the application, adds new categories of diseases and has an overview of them, can review the statistics of the occurrence of types of diseases over the years and has an overview of all patients and can update them. An administrator can perform the same duties as the doctor while the administrator has a few functionalities meant only for his use.

The doctor can view the list of his patients, approve, reject and postpone their appointments. He can view the profiles of his patients, where the patient's data is located, and the list of dates when that patient had an appointment, and he can generate a PDF where the data related to that appointment can be seen.

Patients can review their profile, send a request for an appointment to be approved by the doctor, and if the doctor has postponed their appointment, accept a new one or cancel it.

Keywords: django, web application, general practice clinic

1. Uvod

S obzirom da je kontaktiranje doktora i ugovaranje pregleda u današnje vrijeme postalo jako teško, što zbog velikog broja pacijenata, a što zbog zaposlenosti doktora, bilo je potrebno osmisliti i kreirati aplikaciju koja bi pomogla i doktorima i pacijentima. Doktorima bi olakšala posao na način da imaju sve informacije na jednom mjestu, a pacijentima donijela mogućnost da odmah i u bilo koje doba dana vide slobodne termine svog doktora tako da unaprijed mogu planirati sve svoje obaveze. Na taj način bi obje strane uštedjele mnogo vremena i mnogo bi se lakše sporazumjele u bilo kojoj situaciji.

Pacijenti tako sami obavljaju registraciju u sustav nakon koje se elektroničkom poštom verificiraju pa zatraže termin pregleda kod svog doktora kojeg im administrator dodijeli nedugo nakon registracije i verifikacije, dok doktori upravljaju zatraženim terminima imajući mogućnost termine prihvatiti, premjestiti ili otkazati.

Svi u sustavu imaju mogućnost praćenja arhiviranih pregleda sukladno svojim ulogama, što znači da administratori mogu vidjeti sve arhivirane preglede, doktori za svoje pacijente, a pacijenti samo svoje preglede.

U prvom poglavlju opisane su korištene tehnologije i alati, njihove glavne karakteristike i način korištenja. U drugom poglavlju opisano je kako je strukturirana baza podataka. U trećem poglavlju detaljno je opisana sama aplikacija i njen rad kroz uloge koje postoje. U četvrtom i posljednjem poglavlju iznesen je zaključak o aplikaciji, o samom načinu izrade i popis korištene literature.

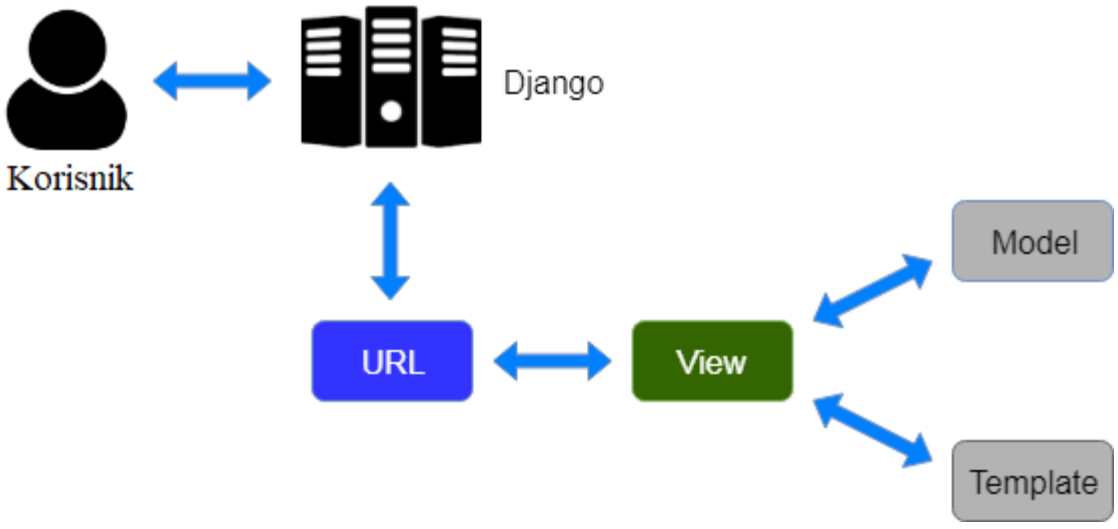
2. Korišteni alati

2.1. Django

Django je razvojni okvir (engl. *framework*) pisan u Python programskom jeziku. Pruža razne, već ugrađene pogodnosti, poput prijave u sustav, vezu s bazom podataka i CRUD (engl. *Create Read Update Delete*) operacije. Django prati MVT (engl. *Model View Template*) projektnu arhitekturu.

Model predstavlja sloj za rad s podatcima. Ti podatci najčešće dolaze iz baze podataka. View prima korisnikov zahtjev te vraća odgovor koji je generiran kroz obradu predloška (engl. *template*) s podatcima. Predložak je najčešće HTML (engl. *HyperText Markup Language*) datoteka na kojoj se nalazi izgled stranice. Upisivanjem URLa (engl. *Unform Resource Locator*) u adresnu traku zahtjev se preko usmjernika pošalje na odgovarajući view.

Kada korisnik zatraži URL, Django pošalje view koji je definiran za taj URL u `urls.py` datoteci. Django također ima mnogo ugrađenih paketa koji programeru olakšavaju pisanje programskog kôda. Paketi se lako dodaju i isključuju iz projekta stoga je lako napraviti modularnu aplikaciju [1].



Slika 1: Kontrolni tok MVT arhitekture

2.2. Sustav za upravljanje bazama podataka

Korišten je sustav za upravljanje bazama podatka MySQL. MySQL (engl. *Structured query language*) je RDBMS (engl. *relational database management system*) baziran na SQLu. Sustavi za upravljanje relacijskim bazama podataka koriste programeru za olakšani rad s bazama podatka koje su nužne za svaku veću aplikaciju. Baza podataka je strukturirana kolekcija podataka.

```
users =
User.objects.filter(doctor_id=user.id).order_by('last_name')
```

Ispis 1: Primjer dohvaćanja filtriranih podataka iz baze podataka

MySQL je otvorenog kôda. Otvoreni kôd znači da svatko može koristiti i mijenjati softver. Svatko može preuzeti softver MySQL s interneta i koristiti ga bez plaćanja. Također se njegov izvorni kôd može mijenjati kako bi odgovarao različitim potrebama. MySQL softver koristi GNU (engl. *GNU is not Unix*) Opću javnu licencu GPL (engl. *General Public License*) za definiranje što se smije, a što ne smije raditi sa softverom u različitim situacijama. Ukoliko GNU GPL nije zadovoljavajući, onda je potrebno ugraditi MySQL kôd u komercijalnu aplikaciju, mogućnost je kupiti komercijalno licenciranu verziju od Oraclea [2].

2.3. HTML

HTML (engl. *HyperText Markup Language*) je prezentacijski jezik za izradu web stranica. Dokument se stvara pomoću HTML jezika. HTML se koristi za oblikovanje sadržaja i stvaranje hiperveza hipertekst dokumenta. Jako je jednostavan za korištenje i učenje. HTML je besplatan i dostupan svima. Prikaz HTML dokumenta omogućuje web preglednik. Zamišljen je da radi jednako na svim preglednicima, no nekad neke funkcionalnosti su drugačije izvedene za različite preglednike.

2.4. CSS i Bootstrap

CSS (engl. *Cascading Style Sheets*) je stilski jezik koji se koristi za oblikovanje HTML datoteka, sa CSSom se oblikuje izgled pojedinačnih HTML elemenata, poput individualnih paragrafa ili tekstova. Uz pomoć CSSa, programer ima mogućnost uređivati izgled svojeg HTML dokumenta, a preko toga izgled same internetske stranice.

Bootstrap je razvojni okvir baziran u HTML, CSS i JavaScriptu za razvoj internetskih aplikacija. Besplatan je za korištenje, nudi mnogo različitih načina za organizirati podatke unutar stranice te mnogo načina kako iste prikazati. Tipični primjeri korištenja su uređivanje izgleda formi, stvaranja tablica i izrada navigacije. Neke napredne funkcionalnosti zahtijevaju korištenje JavaScripta, ali je isti lako uključiti preko raznih dostupnih JavaScript priključaka (engl. *plugin*) [3].

2.5. Chart.js

Chart.js je besplatna JavaScript biblioteka otvorenog kôda za vizualizaciju podataka. Predstavlja jednostavan način za uključivanje interaktivnih i animiranih grafikona. Ima više ugrađenih grafikona: trakasti, linijski, raspršeni itd. U ovoj aplikaciji korišten je linijski grafikon. Ova JavaScript biblioteka nema zavisnosti i zauzima vrlo malo prostora, jako je jednostavna za korištenje i koristi se u kombinaciji sa HTML5.

```
const myChart = new Chart(ctx, {

    type: 'line',

    data: {

        labels: [{%for item in data.keys%} "{{item}}", {%endfor%}],

        datasets: [

        {% for category in illness_categories%}

        {

            label: '# of Occurences of {{category.illness_category}}',

            data: [{%for item in data.values%}
{{item|get_item:category.illness_category}}, {%endfor%}],
```

Ispis 2: Primjer slanja podataka za iscrtavanje grafikona

2.6. ReportLab

ReportLab je programska biblioteka otvorenog kôda za generiranje PDF dokumenata i prilagođene vektorske grafike. ReportLab koristi Python programski jezik. Pomoću njega se može automatizirati produkcija, generirati više PDF dokumenata istovremeno i izrađivati grafove. Koristeći platno (engl. *canvas*) pretvara grafičke naredbe u PDF [4].

```
archive_obj = Archive.objects.get(id=id)
    buffer = io.BytesIO()
    canv = canvas.Canvas(buffer, pagesize=letter, bottomup=0)
    textobj = canv.beginText()
    textobj.setTextOrigin(inch, inch)
    textobj.setFont("Helvetica", 14)
    lines = []
    lines.append(f"Doctor: {str(archive_obj.doctor)}")
    lines.append(f"Appointment date: {str(archive_obj.date)}")
    lines.append(f"Patient: {str(archive_obj.patient)}")
    lines.append(f"Description: {str(archive_obj.description)}")
    lines.append(f"Doctor's note:
{str(archive_obj.doctors_note)}")
     for line in lines:
        textobj.textLine(line)
canv.drawText(textobj)
```

Ispis 3: Isječak kôda za generiranje PDF dokumenta

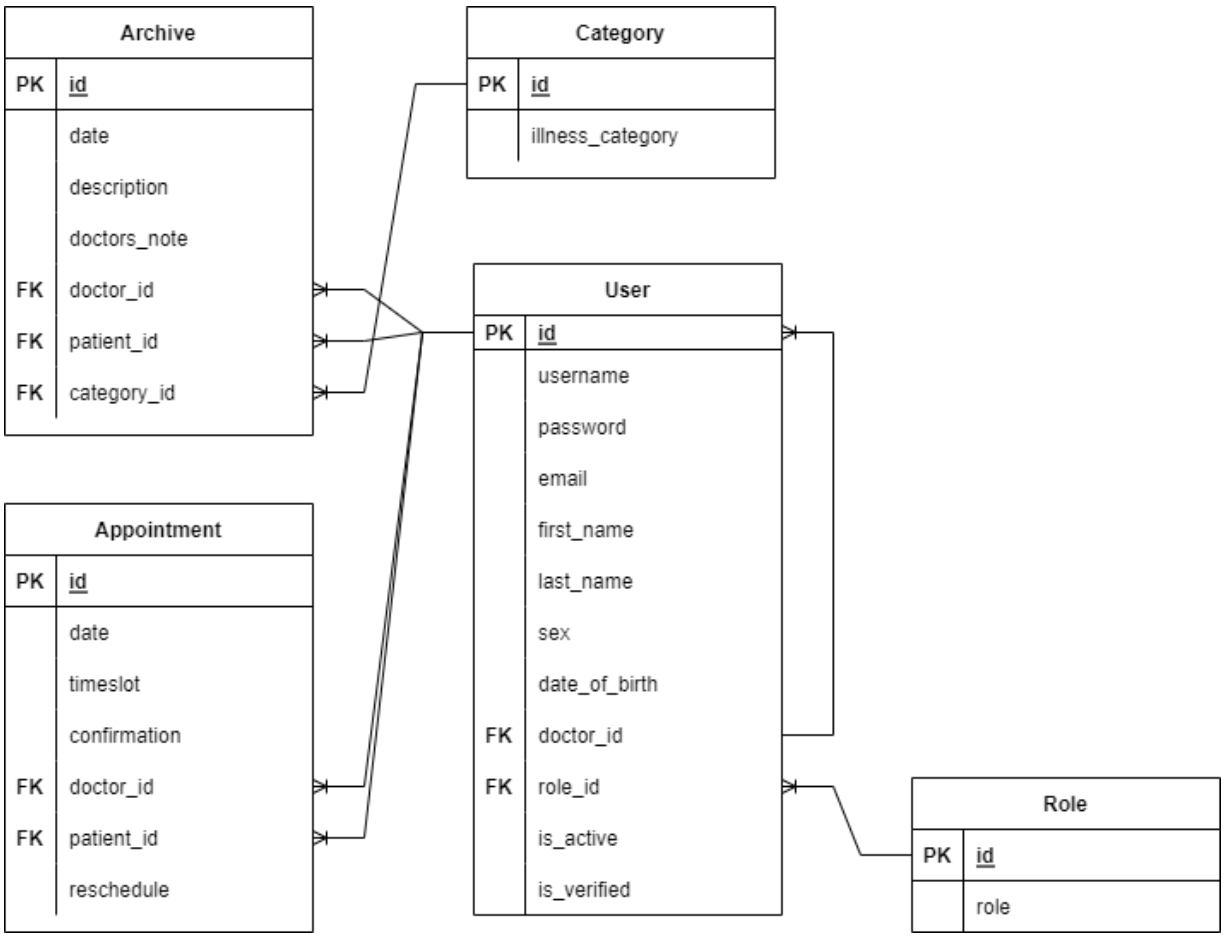
3. Model sustava

Sustav je modeliran kroz 5 tablica u bazi podataka.

Sljedeće tablice se nalaze u bazi podataka:

- User – sadrži osnovne podatke o korisniku
- Role – tablica Role je povezana sa tablicom User, ona sadrži uloge koje postoje u sustavu (administrator, doktor, pacijent)

- Appointment – ovo je tablica sa svim podatcima o terminima, povezana je sa tablicom User
- Category – tablica sa podatcima o kategorijama bolesti, povezana je sa tablicom Archive
- Archive – tablica sa podatcima o arhiviranim terminima, služi za statistiku



Slika 2: E-R dijagram baze podataka

4. Opis aplikacije

Aplikacija je napravljena kako bi se olakšalo vođenje liječničke ordinacije opće prakse.

4.1. Opis aplikacije – Doktor

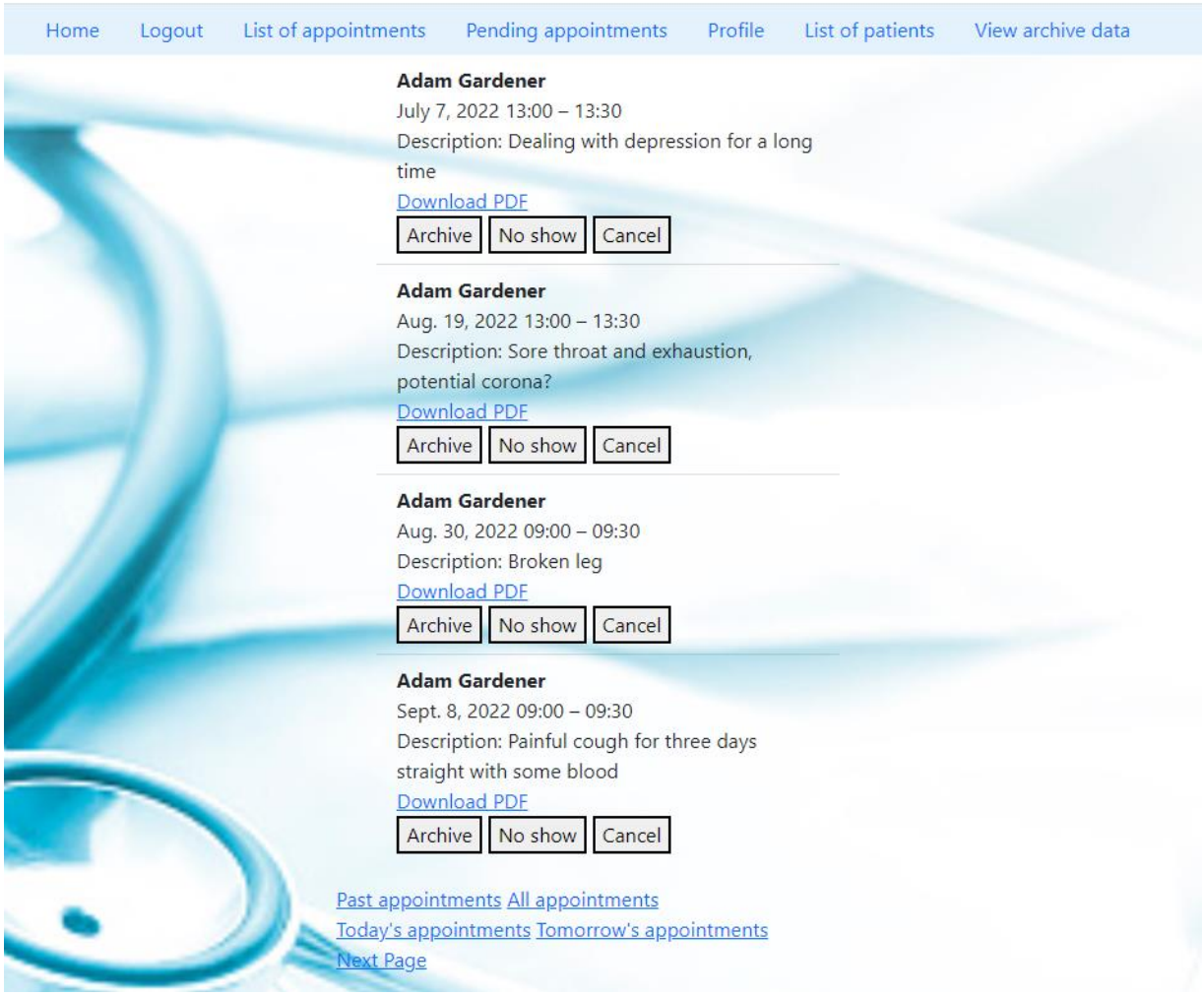
Uloga doktora jedna je od najvažnijih uloga u sustavima koji se baziraju na komunikaciji između primatelja i pružatelja zdravstvenih usluga.

Doktor ima sljedeće mogućnosti:

Pregledavanje liste ugovorenih sastanaka s pacijentima

U navedenoj funkcionalnosti doktor može vidjeti sve potrebne informacije o naručenim pacijentima, kao što su ime i prezime pacijenta, datum i vrijeme ugovorenog pregleda. Doktor također može i preuzeti PDF dokument (engl. *portable document file*) u kojem se nalazi prethodno navedeno, uz ime doktora koji izvršava pregled i sekcija opis u kojem je pacijent opisao razlog potrebe pregleda.

Svaki pojedini ugovoreni sastanak doktor može arhivirati, označiti kao da se pacijent nije pojavio, a isto tako ga može i otkazati. Pri arhiviranju, uz sve prethodno navedene elemente, doktor je obavezan unijeti kojoj kategoriji pripada pacijentova bolest uz opis njegovog specifičnog problema. Nakon što doktor popuni sve potrebne informacije za arhiviranje i klikne “Archive”, zapis o pregledu se briše iz sustava i više se ne prikazuje u listi ugovorenih sastanaka, već se prebacuje u arhivirane zapise.



Slika 3: Prikaz svih ugovorenih termina

Ukoliko se pacijent ne pojavi, odnosno ukoliko propusti svoj termin s doktorom, doktor tada treba označiti pregled kao “No show”. Ta radnja briše termin iz sustava i pacijentu šalje elektroničku poštu u kojoj ga informira o propuštenom terminu i navodi vrijeme kada se termin trebao odviti, kao i razlog zbog kojeg je pacijent htio doći.

Ukoliko doktor otkaže ugovoreni pregled klikom na tipku “Cancel”, pacijent prima elektroničku poštu u kojoj je objašnjeno kako je doktor otkazao dogovoreni pregled. Elektronička pošta isto tako sadržava informacije o vremenu i razlogu termina, nakon čega se termin briše iz sustava.

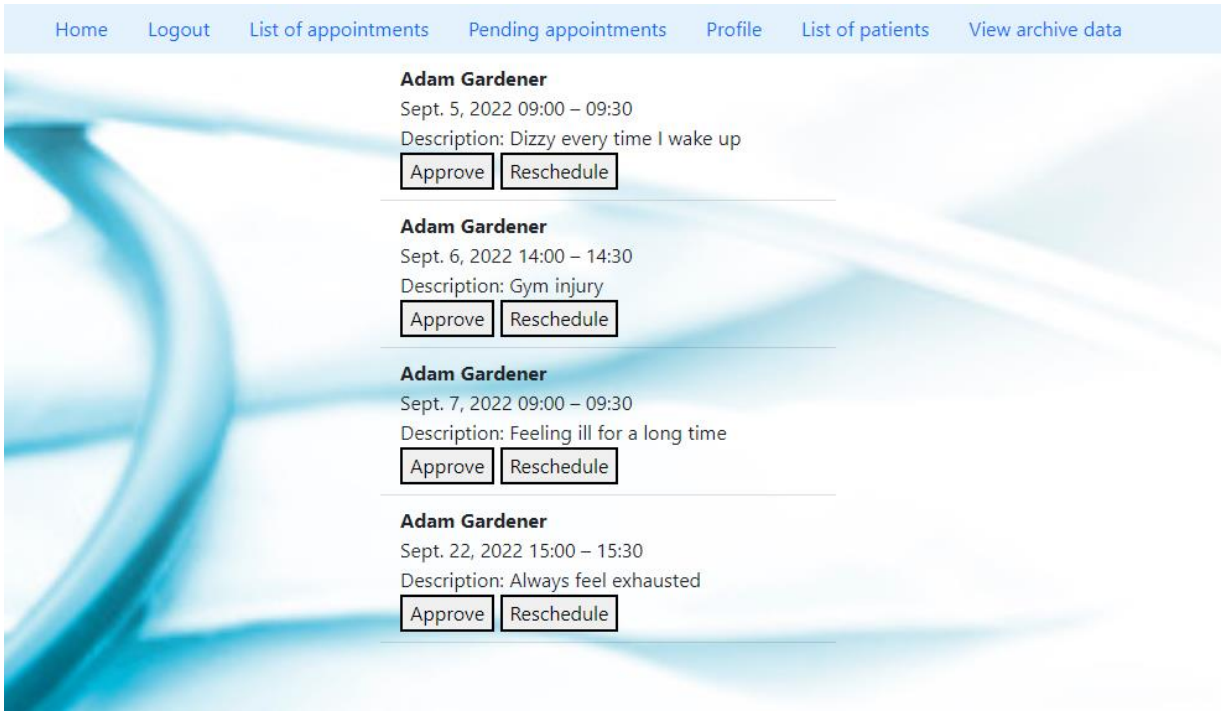
Osim pregleda za danas, sutra i svih pregleda općenito, na listi ugovorenih pregleda doktor može vidjeti i preglede koji su vremenom u prošlosti. Razlog postojanja te mogućnosti je za slučajeve u kojima je doktor odlučio iz nekog razloga zadržati pregled u toj listi bez da ih arhivira ili ga je jednostavno ljudskom greškom zaboravio arhivirati.

Pregledavanje liste neriješenih pregleda

U ovoj funkcionalnosti nalaze se svi pregledi koje su pacijenti zatražili, a koji još nisu odobreni od strane doktora. Prikazuju se osnovne informacije o pacijentu i vrijeme zatraženog pregleda. Ukoliko doktor odobri pregled, on se prebacuje na listu ugovorenih pregleda. Za slučaj da doktor iz nekog razloga želi odgoditi pregled, tada treba odabrati datum na kojeg ga želi prebaciti, pod uvjetom da taj datum nije vikend i da nije u prošlosti.

```
    def clean_date(self):
        date = self.cleaned_data['date']
        if date < date.today():
            raise forms.ValidationError("The date cannot be in the
past!")
        if date.weekday() > 4:
            raise forms.ValidationError("Date cannot be a
weekend!")
        return date
```

Ispis 4: Primjer validacije datuma unutar forme



Slika 4: Prikaz svih neriješenih termina

Nakon toga, za odabrani datum se otvaraju dostupni termini između kojih će odabrati onog koji mu najviše odgovara. Nakon što pacijent odabere termin, pošalje mu se elektronička pošta u kojoj je obaviješten o odgođenom terminu te mu je ponuđen novi termin od strane doktora kojeg pacijent tada može prihvatiti ili odbiti. Ako pacijent prihvati ponuđeni termin, navedeni termin se tada prebacuje u listu svih ugovorenih pregleda, a ako odbije, termin se briše iz sustava te pacijent treba zatražiti novi termin.

Pregledavanje vlastitog profila

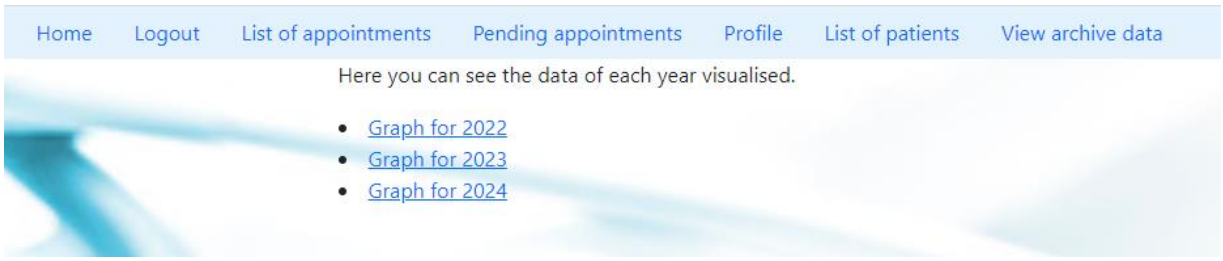
Na svom profilu doktor može vidjeti neke svoje osnovne podatke kao što su ime i prezime, spol, elektroničku poštu, datum rođenja i ulogu u sustavu.

Pregledavanje liste pacijenata

Na listi pacijenata doktor ima mogućnost vidjeti sve svoje pacijente. Tu se prikazuju osnovne informacije o pacijentu, kao što su njegov mail, ime i prezime. Klikom na navedene informacije otvara se pacijentov profil sa svim ostalim informacijama, uključujući pacijentovog doktora i njegove arhivirane preglede.

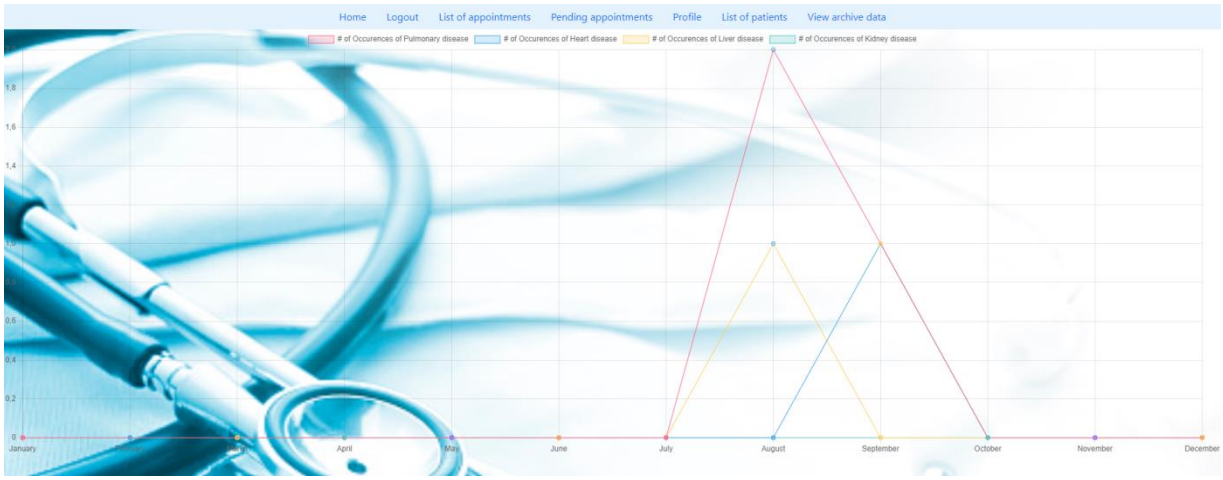
Pregledavanje arhiviranih podataka

U ovom odjeljku doktor može pristupiti arhiviranim podatcima od prethodno obavljenih pregleda. Podatci su podijeljeni po godinama i grafički su prikazani. Ukoliko postoji barem jedan obavljeni i arhivirani pregled za barem jednu godinu, u odjeljku arhiviranih podataka može se naći graf u kojem se ti podatci mogu vidjeti.



Slika 5: Podatci podijeljeni po godinama

Klikom na jednu od godina otvori se graf u kojem se po mjesecima prikazuje statistika pojavljivanja određenih bolesti. Statistika se vrši na razini cijele ordinacije.



Slika 6: Grafički prikazana arhiva podataka

4.2. Opis aplikacije – Administrator

Uloga administratora

Osim što ima sve mogućnosti koje ima i doktor, uloga administratora ima još neke dodatne opcije na raspolaganju.

U nastavku slijede administratorske mogućnosti:

Pregledavanje liste ugovorenih sastanaka s pacijentima

U listi ugovorenih sastanaka administrator vidi sve sastanke od svih doktora, sve potrebne informacije o pacijentu i terminu kojeg je pacijent rezervirao. On također može preuzeti PDF u kojem se nalaze dodatne informacije o navedenom. Administrator je također u mogućnosti arhivirati obavljene termine, pri čemu je isto tako obvezan dodati kategoriju bolesti. Termin se tada prebacuje u listu arhiviranih sastanaka.

Označavanje termina kao “No show” se odvija isto kao kada doktor odluči napraviti tu radnju. Termin se briše iz sustava i pacijent je elektroničkom poštom obaviješten o prethodnom. Otkazivanje termina od strane administratora isto tako funkcionira na način da pacijent o tome bude obaviješten, nakon čega se termin briše iz sustava.

Administrator nema mogućnost pregleda termina koji su vremenom u prošlosti te je to rezervirano isključivo za doktore, ali zato ima mogućnost pretraživati postojeće termine filtriranjem prema prezimenu doktora.

```
elif request.POST.get("No show"):

    appointment = Appointment.objects.get(id=request.POST["No
    show"])

    send_mail( "Appointment missed", f"You have missed an
    appointment scheduled for {appointment.date} at
    {time_dict[appointment.timeslot]} for
    {appointment.description}", settings.EMAIL_HOST_USER,
    [appointment.patient.email], )

    Appointment.objects.get(id=request.POST["No show"]).delete()
```

Ispis 5: Primjer kôda za slanja elektroničke pošte

Pregledavanje liste neriješenih pregleda

Dinamika upravljanja listom neriješenih pregleda funkcionira slično kao što funkcionira upravljanje od strane doktora uz razliku da administrator vidi sve neriješene termine za sve doktore: pacijent zatraži termin kojeg administrator može prihvatiti ili premjestiti u drugi termin. U slučaju premještanja termina, pacijent dobije elektroničku poštu u kojoj to potvrdi pa se ugovoreni sastanak premjesti na listu svih termina odabranom doktoru. Ako pacijent odbije zamjenski termin, tada se taj termin briše iz sustava.

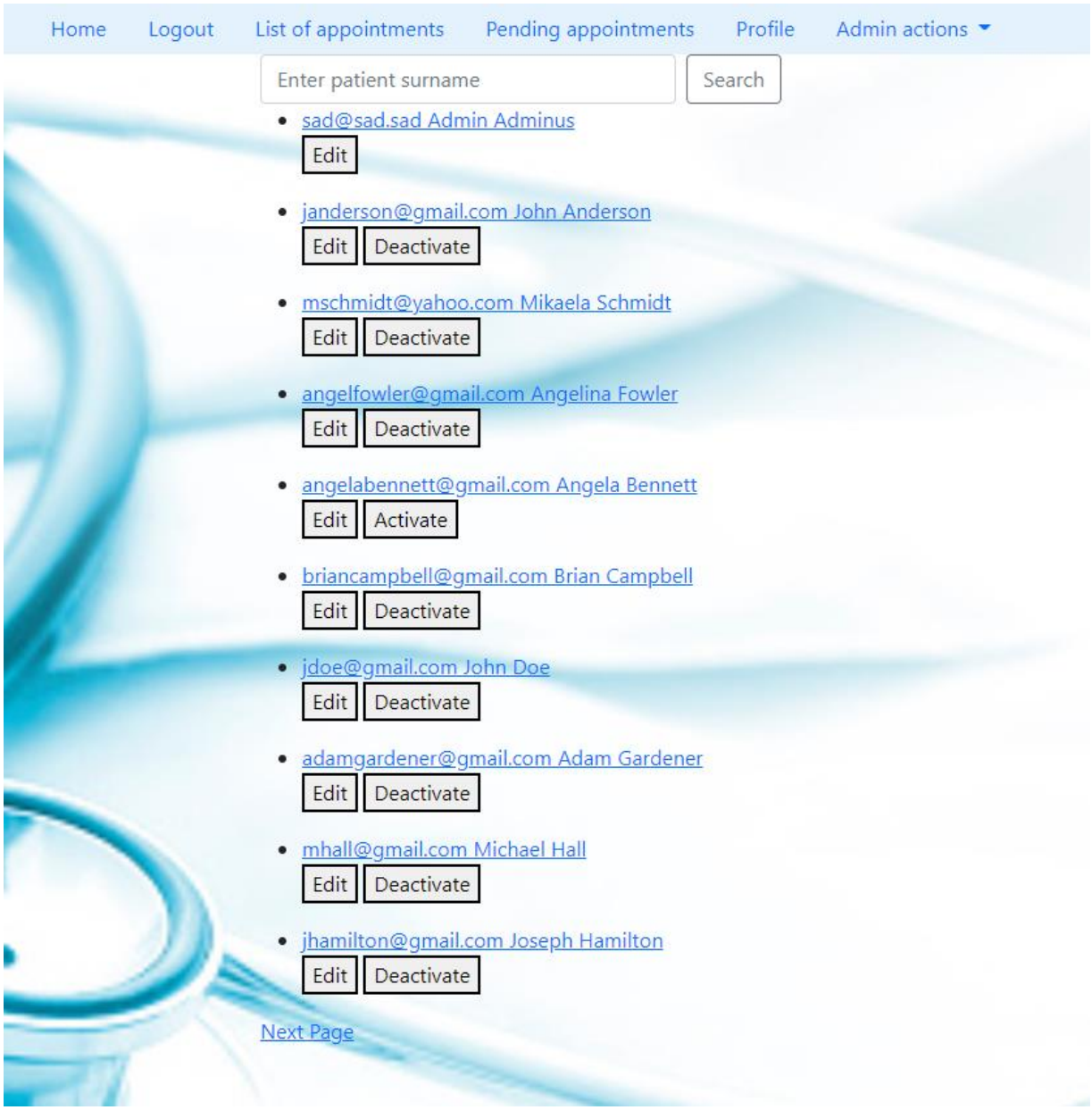
Pregledavanje vlastitog profila

Administrator također može vidjeti neke svoje osnovne podatke kao što su ime i prezime, spol, elektronička pošta, datum rođenja i uloga u sustavu.

Pregledavanje liste korisnika

U listi korisnika administrator ima mogućnost vidjeti popis svih korisnika koji se nalaze u sustavu. Klik na određenog korisnika odvede administratora na njegov profil. Dostupna mu je i opcija uređivanja postojećih informacija o korisniku kao i mogućnost aktivacije odnosno deaktivacije korisnika. Ukoliko administrator želi urediti informacije o određenom korisniku, tada mu se otvori forma koja sadrži već postojeće korisnikove informacije te ih može urediti ili izmijeniti. Izmijenjene informacije se pohrane i vidljive su na korisnikovom profilu.

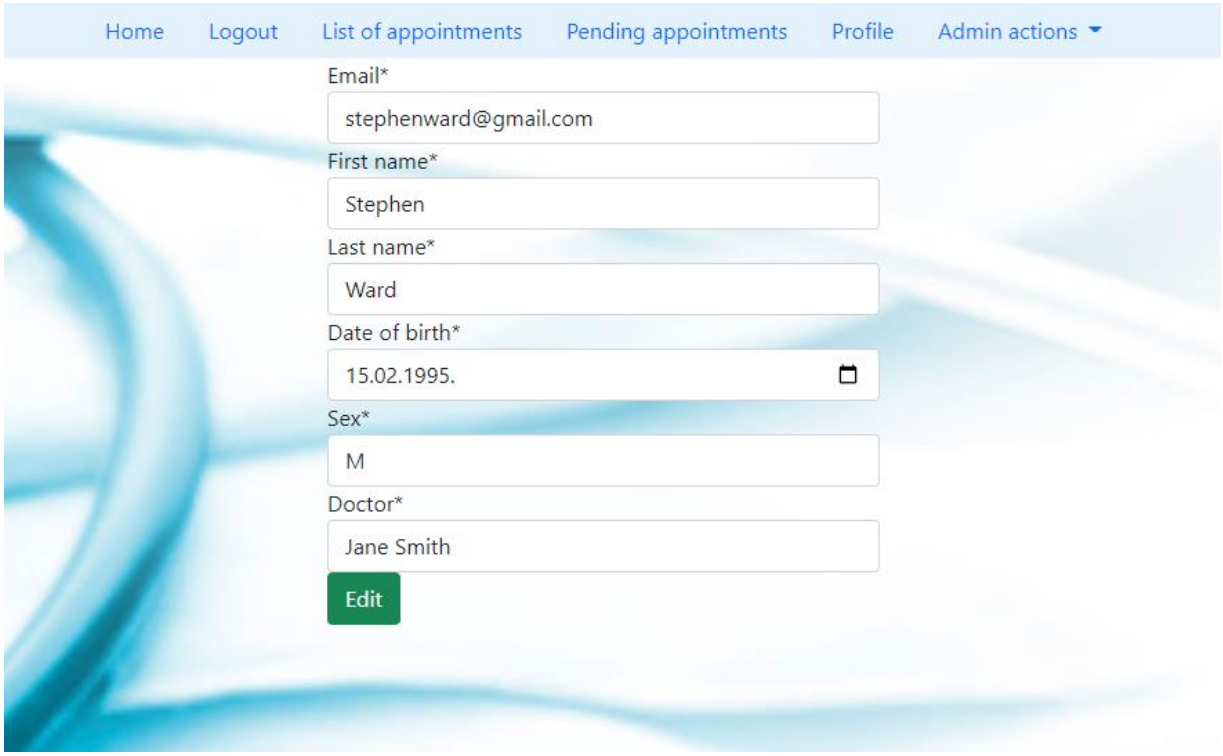
Uz svakog korisnika na listi je prikazan njegov status koji može biti aktivan ili neaktivan. Ukoliko je korisnikov status aktivan, tada ima mogućnost pristupati sustavu i koristiti akcije sustava sukladno svojoj ulozi (bilo da se radi o administratoru, doktoru ili pacijentu). Za slučaj da je status neaktivan, korisnik ne može pristupiti sustavu. Administrator u svakom trenutku može mijenjati navedeno.



Slika 7: Prikaz svih korisnika

Pregledavanje liste pacijenata bez doktora

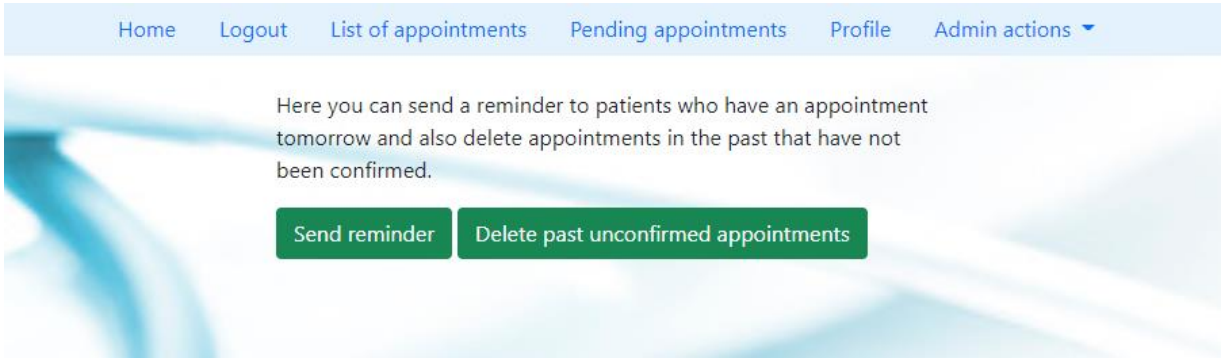
Na navedenoj listi administrator može vidjeti sve pacijente koji trenutno postoje u sustavu, a još uvijek nemaju dodijeljenog doktora. Administrator može dodijeliti doktora pacijentima sa liste tako da klikne na njega, što otvori formu u kojoj može izmjenjivati informacije, pa tako i izmijeniti informaciju o doktoru. Kada se pacijentu dodijeli doktor, vise nije na listi nedodijeljenih pacijenata. Također, ukoliko administrator pacijentu promijeni doktora, pacijent o tome dobije obavijest putem elektroničke pošte.



Slika 8: Uređivanje korisnikovih podataka

Slanje podsjetnika na pregled

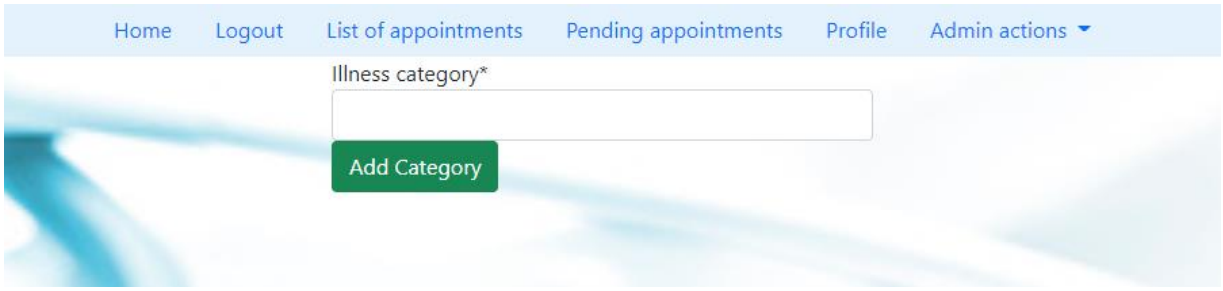
Administrator ima mogućnost slanja podsjetnika pacijentima kojima je pregled zakazan na taj dan ili na sutrašnji dan. Ukoliko pošalje podsjetnike, tada se pošalje elektronička pošta svim pacijentima koji imaju pregled koji odgovara gore navedenim uvjetima. Elektronička pošta sadrži informacije o ugovorenom terminu i razlogu termina. Osim toga, administrator može i izbrisati sve prethodno zatražene termine koji nisu potvrđeni. To uključuje termine koje su pacijenti zatražili i zamjenske termine koje je doktor zatražio od pacijenata. Ta radnja sve takve termine briše iz sustava.



Slika 9: Slanje podsjetnika i brisanje nepotvrđenih termina

Dodavanje kategorija bolesti

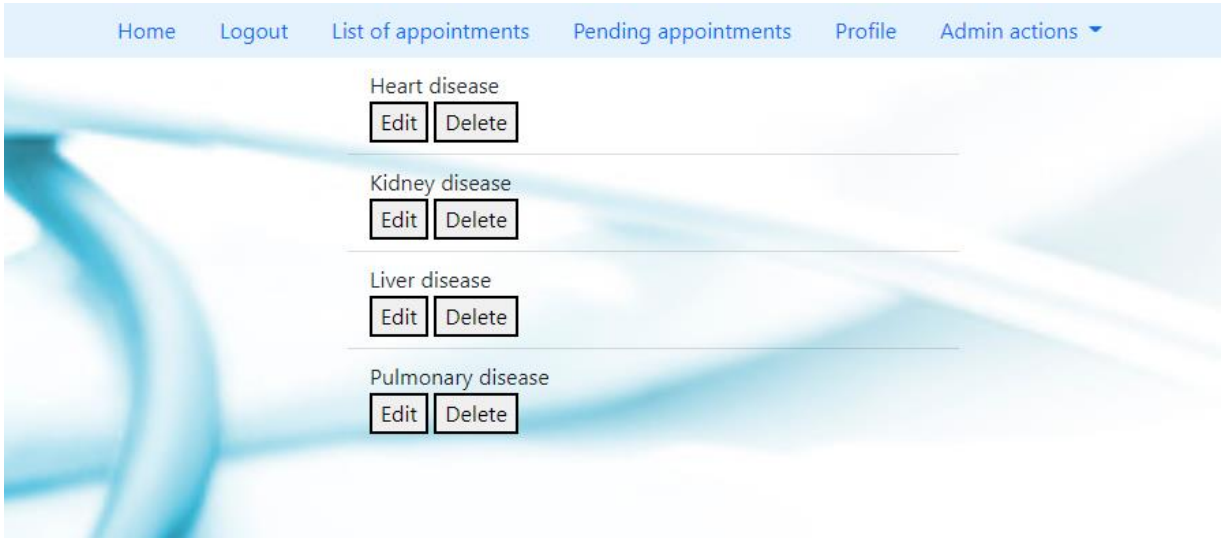
Ova sekcija služi za dodavanje kategorije bolesti u sustav. Nakon dodavanja nove kategorije, ta kategorija se dodaje i na listu svih kategorija bolesti.



Slika 10: Dodavanje kategorije bolesti

Pregledavanje liste kategorija bolesti

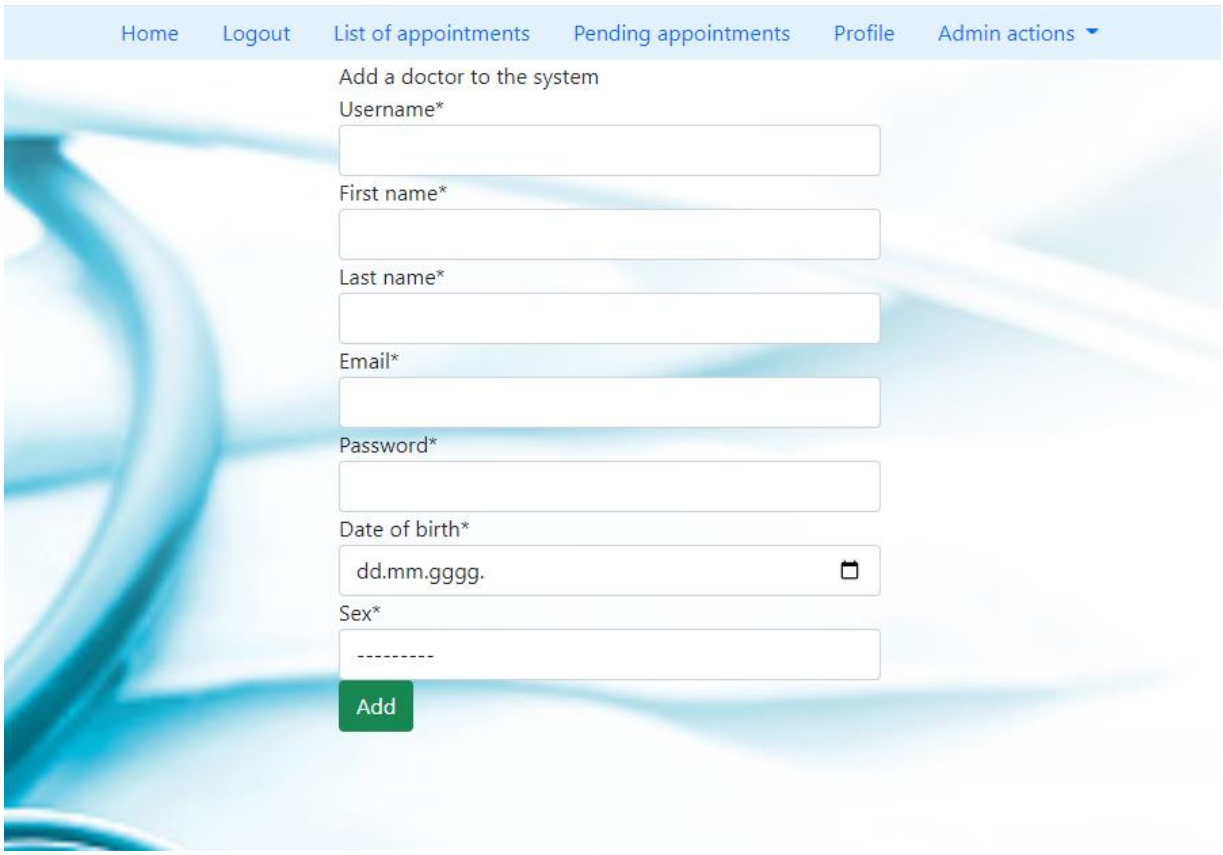
Ovdje administrator može vidjeti sve unesene kategorije bolesti sortirane abecednim redom. Naziv kategorije moguće je urediti ili izbrisati. Ukoliko se kategorija izbriše, ona tada nestaje iz sustava.



Slika 11: Lista kategorija bolesti

Dodavanje novog doktora

Dok se pacijenti trebaju sami registrirati u sustav, nove doktore u sustav je obvezan unijeti administrator. Pazeći na to da je korisničko ime za svakog doktora jedinstveno, administrator unosi informacije o novom doktoru. Nakon što se novi doktor pohrani u sustav, na svoju elektroničku poštu primi podatke za daljnje pristupanje sustavu te je vidljiv je na listi svih korisnika.



Slika 12: Dodavanje doktora u sustav

Pregledavanje arhiviranih podataka

Slično kao što doktor može pristupati arhiviranim podatcima, tako i administrator može vidjeti informacije o prethodno obavljenim pregledima na razini cijele ordinacije. Svi podatci su grafički prikazani po godinama ako za navedenu godinu postoji barem jedan obavljeni pregled.

```
all_archives = Archive.objects.filter(date__startswith=year)
    data = {}
    illness_categories = Category.objects.all()
    months = ["January", "February", "March", "April", "May", "June",
"July", "August", "September", "October", "November", "December"]
    for i in months:
        data[i] = {}
        for category in illness_categories:
            data[i][category.illness_category] = 0
    for archive in all_archives:
      data[months[archive.date.month]][archive.category.illness_category]
+= 1
```

Ispis 6: Dodavanje podataka u rječnik koji se koristi za generiranje grafikona

Pregledavanje statistike doktora

Ukoliko administrator želi provjeriti koliko obavljenih pregleda je pojedini doktor imao, tada treba otići na profil doktora čije podatke želi provjeriti. Ako je doktor u barem jednoj godini obavio barem jedan pregled, na doktorovom profilu naći će se graf koji će sadržavati te podatke. Klikom na željenu godinu, otvorit će se graf u kojem se prikazuje statistika doktorovih pregleda podijeljenih po mjesecima u odabranoj godini.

4.3. Opis aplikacije – Pacijent

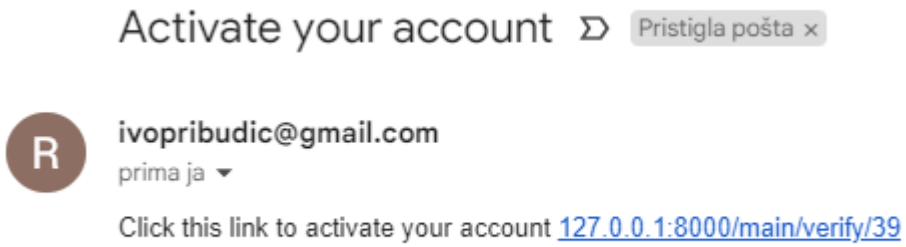
Za razliku od administratora i doktora koji imaju ovlasti upravljati sustavom u skladu sa svojim ulogama, pacijentovo kretanje sustavom je ograničeno.

Njegove mogućnosti su sljedeće:

Registracija u sustav

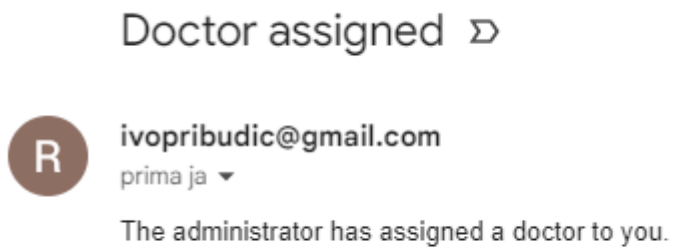
Prilikom pristupanja sustavu, pacijent ima mogućnost pristupiti sustavu putem "Login" ako se njegovi podatci već nalaze u sustavu. Za slučaj da mu je to prvi put da ide pristupiti aplikaciji, potrebno je registrirati se. Pacijent se u sustav registrira samostalno unoseći neke svoje osnovne informacije kao što su korisničko ime, ime, prezime, elektronička pošta itd. pritom odabirući korisničko ime koje nije već zauzeto.

Kada upotpuni registracijsku formu potrebnim informacijama i registrira se, pacijent na svoju adresu elektroničke pošte primi poruku koju sustav automatski pošalje nakon što pacijent klikne "Register". Navedena pošta sadrži poveznicu koja vodi na početnu stranicu na kojoj je pacijent obaviješten o uspješnoj verifikaciji u sustav.



Slika 13: Elektronička pošta za verificiranje računa

Pacijentu je još potrebno da mu administrator dodijeli doktora. Nakon toga, on se slobodno može prijaviti u sustav i koristiti sve mogućnosti uloge pacijent.



Slika 14: Elektronička pošta obavijesti o dodijeljenom doktoru

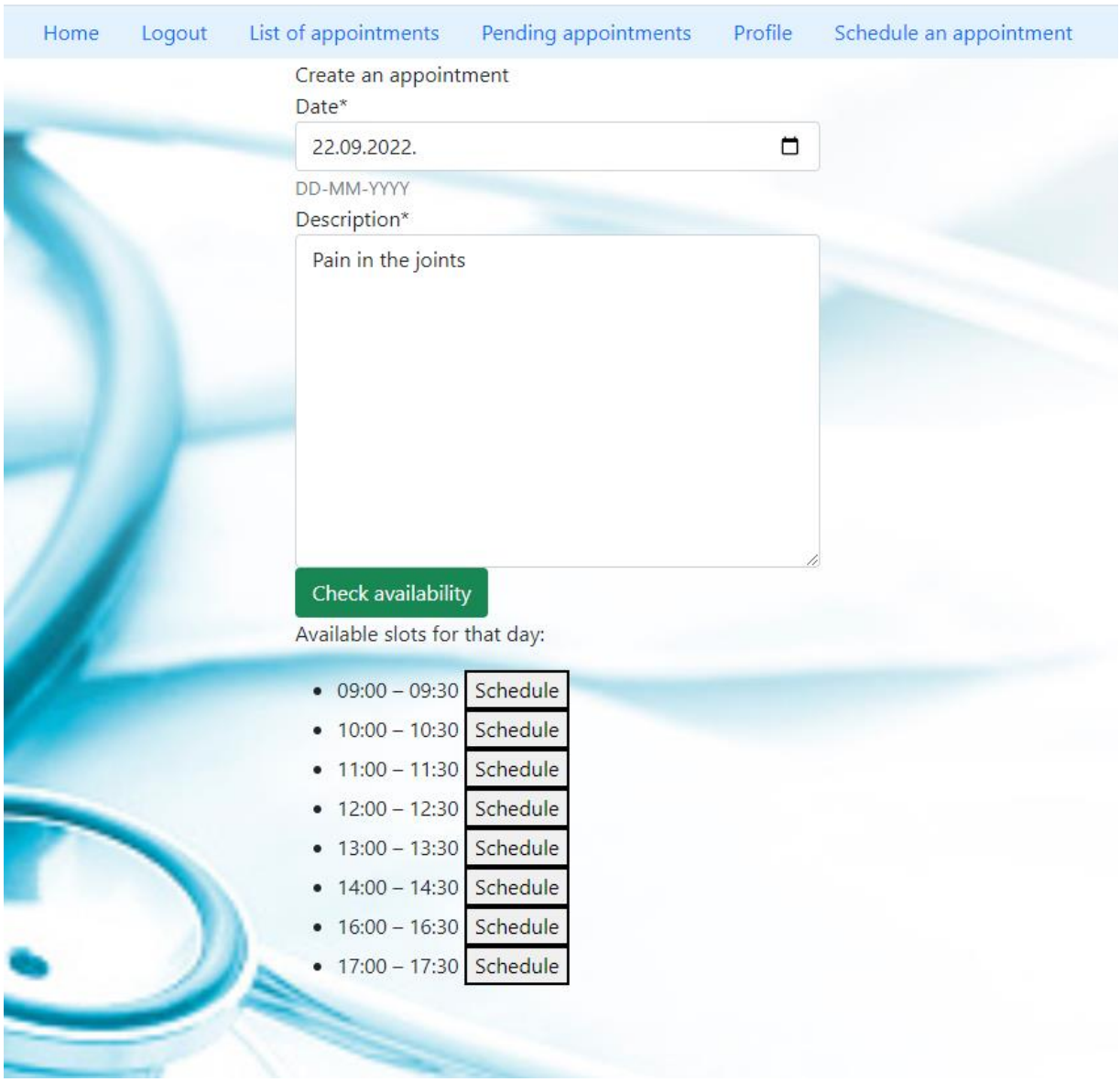
Zakazivanje pregleda

Nakon registracije u sustav i primitka dodijeljenog doktora, pacijent može zatražiti pregled. Pri kreiranju pregleda, obvezan je unijeti datum na kojeg želi posjetiti svog doktora. Nakon što odabere željeni datum, potrebno je unijeti razlog dolaska i zatim kliknuti "Check availability" kako bi se odmah ispod prikazali slobodni termini za taj dan. Ukoliko mu barem jedan od ponuđenih termina odgovara, tada zakaže pregled u to određeno vrijeme, nakon čega se zatraženi pregled prebaci u listu neriješenih pregleda.

Ako mu ne odgovara nijedan ponuđeni termin za dan kojeg je odabrao, tada je potrebno promijeniti datum na idućeg kojeg želi pa ponovno pregledati dostupne termine dok ne nađe termin koji mu odgovara.

Ako doktoru ne odgovara termin kojeg je pacijent zatražio, doktor tada premjesti termin na vrijeme kada bi mu to odgovaralo i to se elektroničkom poštom pošalje pacijentu koji isto tako može prihvatiti ili odbiti termin. Ako ga prihvati, on se premješta na listu

ugovorenih pregleda. Ako ga odbije, briše se iz sustava te pacijent treba ponoviti postupak zakazivanja pregleda.



Slika 15: Ugovaranje termina

Pregledavanje liste neriješenih pregleda

Na ovoj listi pacijent može vidjeti sve preglede koje je zatražio i koje doktor još nije odobrio te one preglede koje je zatražio, a koji su premješteni na neki drugi termin od strane doktora. Navedeni pregledi prikazuju informacije kao što su datum i vrijeme pregleda i razlog traženja pregleda, uz dodatak da premješteni termini imaju odgovarajuću naznaku kako bi se pregledi mogli lakše razlikovati.

Tu pacijent ima mogućnost otkazati zatraženi pregled kojeg doktor još nije odobrio. Ukoliko ga otkaže, taj pregled se briše iz sustava. Također, pacijent ima mogućnost potvrditi ili otkazati pregled kojeg je doktor odgodio. Prihvaćeni pregled se premješta na listu ugovorenih pregleda, a otkazani se briše iz sustava.

Pregledavanje liste ugovorenih pregleda

Na navedenoj listi pacijent može vidjeti sve preglede koji su potvrđeni, bilo to od strane pacijenta ili doktora. Lista prikazuje osnovne informacije o pregledu. Moguće je preuzeti PDF dokument u kojem se nalaze i neke dodatne informacije o pregledu. Također je moguće otkazati pregled klikom na “Cancel”, što taj pregled onda briše iz sustava.

Pregledavanje vlastitog profila

Isto kao administrator i doktor, pacijent može pregledavati svoje podatke na profilu.

Slika 16: Pacijentov profil

Osim pregledavanja podataka, također može vidjeti i arhivirane podatke od prethodno obavljenih pregleda. Klikom na željeni pregled, preuzima se PDF dokument koji sadrži informacije o navedenom, kao što su ime doktora koji je obavio pregled, datum pregleda, razlog dolaska i doktorove bilješke.

Doctor: Jane Smith
Appointment date: 2022-07-06
Patient: Adam Gardener
Description: I have pains in my lower back
Doctor's note: I have prescribed a painkiller and recommended lots of bed rest

Slika 17: PDF arhiviranog pregleda

5. Zaključak

U svrhu završnog rada razvijena je funkcionalna aplikacija kojoj je cilj olakšati komunikaciju između pacijenata i doktora.

Administrator ima najveći broj mogućnosti, od unosa novih djelatnika u sustav pa do administriranja ugovorenih i neugovorenih termina za svakog pojedinog doktora. Doktor vodi računa o svojim obavezama tako da redovito pregledava i potvrđuje ili odgađa zatražene termine sukladno svojim željama i mogućnostima. Na pacijentu je da traži termine od svog doktora kada god mu je to potrebno.

Kroz ovaj rad prikazano je kako je korišteni okvir Django brz i siguran za razvijanje aplikacija raznih vrsta: od manjih, jednostavnijih aplikacija pa do onih kompliciranih koje sadržavaju veliki broj različitih komponenata ili imaju naglasak na bazi podataka.

Potencijalne nadogradnje same aplikacije bi mogle biti u vidu proširivanja funkcionalnosti, kao na primjer mogućnost potpisivanja recepata za lijekove koje bi ljekarna potom izdala i slanje uputnica odabranim bolnicama. Također, proširivanje razvojnim okruženjem poput Angulara umjesto korištenja Djangovog templating enginea. Konačno, dobar odabir bi bila i automatizacija pojedinih administratorskih obveza poput slanja podsjetnika.

Literatura

[1] W3Schools, „Django Introduction“, https://www.w3schools.com/django/django_intro.php (posjećeno 02.09.2022.)

[2] Oracle, „What is MySQL“, https://www.oracle.com/mysql/what-is-mysql/

(posjećeno 24.08.2022.)

[3] Bootstrap, „About“, https://getbootstrap.com/docs/5.2/about/overview/

(posjećeno 27.08.2022.)

[4] ReportLab PDF Generation User Guide, „What is the ReportLab PDF Library?“, https://www.reportlab.com/docs/reportlab-userguide.pdf (posjećeno 4.09.2022.)