Web aplikacija za evidenciju i upravljanje članovima

Ledenčan, Jana

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:020874

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-21**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

WEB APLIKACIJA ZA EVIDENCIJU I UPRAVLJANJE ČLANOVIMA

Završni rad

Jana Ledenčan

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 07.07.2023.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Jana Ledenčan
Studij, smjer:	Programsko inženjerstvo
Mat. br. Pristupnika, godina upisa:	R4527, 27.07.2020.
OIB Pristupnika:	16897542193
Mentor:	prof. dr. sc. Krešimir Nenadić
Sumentor:	,
Sumentor iz tvrtke:	
Naslov završnog rada:	Web aplikacija za evidenciju i upravljanje članovima
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	Kratko opisati problematiku vođenja evidencije članova neke udruge ili kluba. Opisati dizajn baze podataka koju će koristiti web aplikacija. Dati kratak pregled mogućnosti koje će imati web aplikacija za evidenciju i upravljanje članocima. Opisati tehnologije i postupak izrade web aplikacije. Korištenje aplikacije omogućiti samo registriranim korisnicima. Tema rezervirana: Jana Ledenčan
Prijedlog ocjene završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	07.07.2023.
Datum potvrde ocjene od strane Odbora:	17.07.2023.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 17.07.2023.

Ime i prezime studenta:	Jana Ledenčan
Studij:	Programsko inženjerstvo
Mat. br. studenta, godina upisa:	R4527, 27.07.2020.
Turnitin podudaranje [%]:	6

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za evidenciju i upravljanje članovima**

izrađen pod vodstvom mentora prof. dr. sc. Krešimir Nenadić

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. PREGLED POSTOJEĆIH RJEŠENJA 2

2.1. Monday.com 2

2.2. LoveAdmin 3

2.3. Asana 4

2.4. GymMaster 4

2.5. EMS: Employee Management 5

3. VOĐENJE EVIDENCIJE ČLANOVA 7

4. PROGRAMSKO RJEŠENJE APLIKACIJE 8

4.1. Korišteni alati 8

4.2. Modeli 9

4.3. Autentifikacija 10

4.4. Profil 12

4.5. Rad s članovima 13

4.5.1. Dodavanje, uređivanje i uklanjanje članova 14

4.5.2. Prikaz informacija o članovima 15

4.5.3. Filtriranje i sortiranje 16

4.5.4. Izvoz podataka o članovima u Excel tablicu 16

4.6. Rad s grupama 17

4.6.1. Dodavanje, uređivanje i uklanjanje grupa 17

4.6.2. Pregled grupe 17

5. KORIŠTENJE APLIKACIJE 18

5.1. Početna stranica 18

5.2. Profil 19

5.3. Upravljački dio 20

5.4. Pregled grupe 21

5.5. Stvaranje novog člana i grupe 22

5.6. Uređivanje člana ili grupe **23**

6. ZAKLJUČAK **25**

LITERATURA **26**

SAŽETAK **27**

ABSTRACT **28**

1. UVOD

Pri vođenju evidencije članstva može se naići na razne probleme, neki od kojih su točnost, sigurnost i integritet podataka, njihova pohrana i organizacija, ažuriranja članstva te komunikacija. Točnost podataka može biti narušena pri zapisivanju podataka u papirnatom obliku i pri korištenju aplikacija koje to omogućuju. Takvo ručno unošenje podataka može u oba slučaja dovesti do pravopisnih grešaka, netočnih ili dupliciranih podataka. Prednost je što se u aplikaciji najčešće krivo uneseni podaci mogu lakše i brže ispraviti. Važno je očuvati sigurnost podataka jer su ti podaci često osobni podaci članova. Podacima u aplikaciji osoba može pristupiti samo ako je autentificirana. Time je osiguran i integritet podataka pa će sve izmjene zapisa podataka biti omogućene samo zaduženim osobama. Svakim dolaskom novog člana otežava se upravljanje količinom podataka. Zato je potrebna primjerena pohrana i organizacija istih. Ne samo što se svakim danom može učlaniti nova osoba već i postojeći članovi mogu se pridružiti ostalim grupama ili iz nekih otići.

Ovim radom želi se uputiti na moguću problematiku vođenja evidencije članova neke udruge ili kluba te olakšati isto izrađenom aplikacijom. Ona nudi svakom registriranom korisniku pregledno mjesto za sve svoje članove i jednostavnu evidenciju njihovog plaćanja članarine za pojedinu grupu. U nastavku rada dan je pregled mogućnosti web aplikacije za evidenciju i upravljanje članovima. Opisane su tehnologije i postupak izrade web aplikacije.

U drugom su poglavlju opisana postojeća rješenja, dok se u trećem poglavlju ukratko opisuje problem vođenja evidencije članova. Četvrto poglavlje obuhvaća sve tehnologije, alate i programske jezike koji su korišteni prilikom izrade aplikacije. U petom poglavlju objašnjava se izvedba aplikacije uz odgovarajući programski kod, dok se u šestom poglavlju opisuje korisničko sučelje. Nakon toga slijedi zaključak s preporukama za daljnja istraživanja i mogućnosti proširivanja aplikacije dane ovim radom.

1.1. Zadatak završnog rada

Kratko opisati problematiku vođenja evidencije članova neke udruge ili kluba. Opisati dizajn baze podataka koju će koristiti web aplikacija. Dati kratak pregled mogućnosti koje će koristiti web aplikacija. Dati kratak pregled mogućnosti koje će imati web aplikacija za evidenciju i upravljanje članovima. Opisati tehnologije i postupak izrade web aplikacije. Korištenje aplikacije omogućiti samo registriranim korisnicima.

2. PREGLED POSTOJEĆIH RJEŠENJA

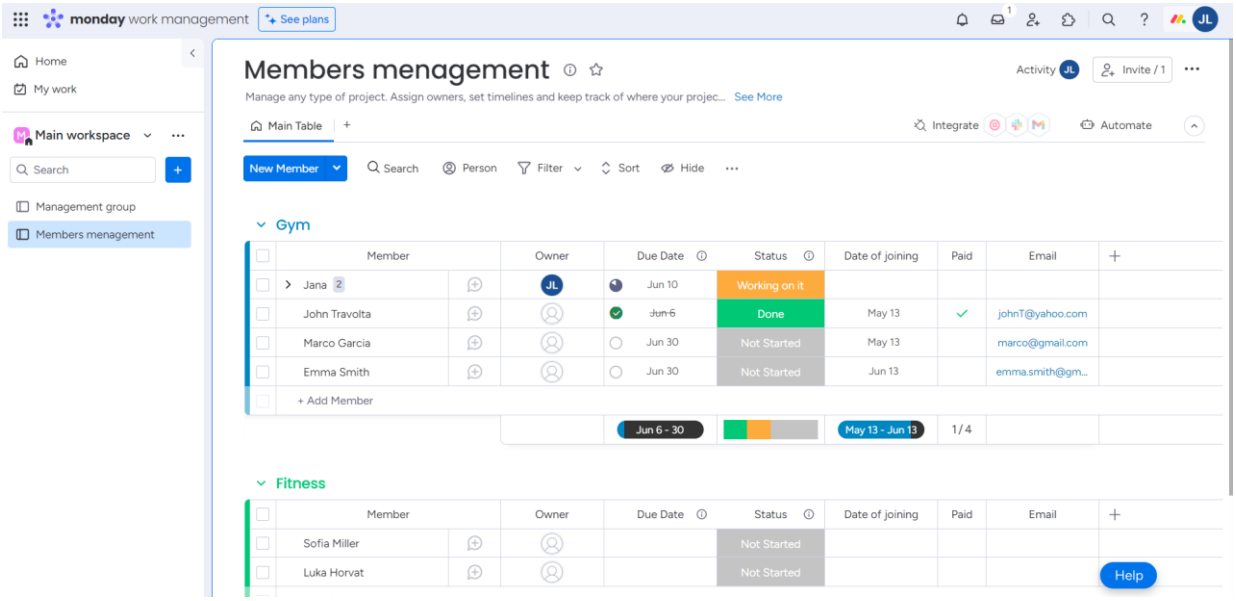
U ovom poglavlju opisane su prednosti i nedostaci postojećih web i mobilnih aplikacija koje rješavaju probleme vođenja tvrtki. Velik je broj aplikacija koje su stvorene kako bi se pratio rad zaposlenika ili članova grupa u slučaju teretana, većinom služe za praćenje dnevnih aktivnosti zaposlenika ili praćenje tijeka projekata.

2.1. Monday.com

Korisnik aplikacije prije samog rada s aplikacijom treba proći niz pred-koraka u kojima od ponuđenih pojmova označi one koje smatra da najbolje opisuju njegove potrebe.

Velike su mogućnosti rada s aplikacijom. Korisničko je iskustvo na visokoj razini jer su elementi aplikacije intuitivno posložena, ali bi se moglo dogoditi da osoba ni ne zna što sve s aplikacijom može napraviti ili da nekim težim putem dođe do jednakog rezultata.

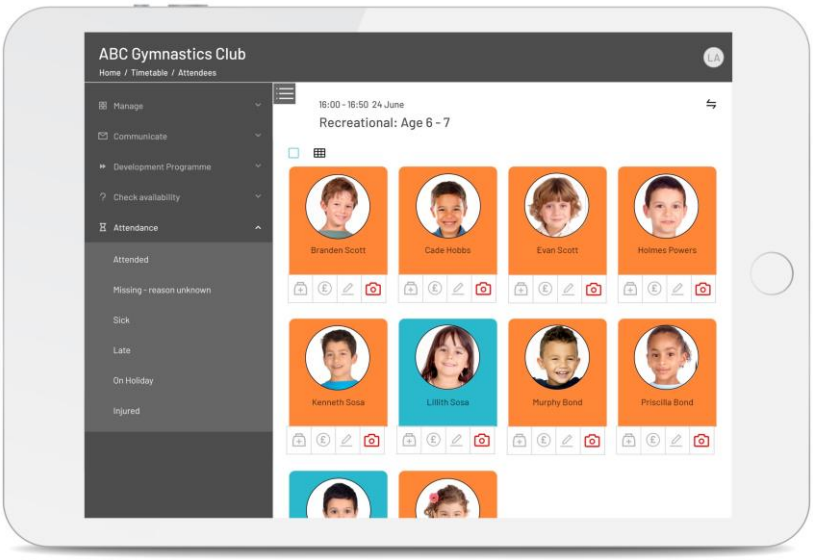
Na slici 2.1. prikazan je pokušaj rekreiranja mogućnosti aplikacije ovog završnog rada na *Monday.com* aplikaciji [1]. Napravljena je ploča (engl. *board*) simboličnog imena *Members management* s grupama *gym* i *fitness*. U svaku grupu dodani su redci s imenima članova i proizvoljno stvarani stupci za *Gym* grupu. Vrlo korisno je što su se automatski izgenerirali jednaki stupci za drugu grupu. Svaki novi stupac može biti različitog sadržaja od uobičajenog teksta, datuma, adrese elektroničke pošte do postavljanja statusa projekata i dokumenata. *Due date* i *status* su stupci koji su bili automatski izgenerirani i međusobno su povezani. Posebno je dodan stupac *Paid* kako bi se vidjelo može li se aplikacijom *Monday.com* odraditi sličan posao koji omogućuje aplikacija završnoga rada. Aplikacija omogućuje i povezivanje s raznim uslugama i aplikacijama. Usmjerena je na olakšavanje organiziranja posla i praćenje tijeka rada. Ima korisnu opciju razgovora sa svakim pojedincem. Sadrži traku za pretraživanje, filtriranje po raznim kriterijima te sortiranje po bilo kojem stupcu. Velik broj predložaka za brže kreiranje ploča kakve osoba treba. Postoji i mobilna verzija.



Sl. 2.1. Sučelje aplikacije Monday.com

2.2. LoveAdmin

LoveAdmin je aplikacija prilagođena za sportske klubove, organizacije s članovima i svima onima koji omogućuju tečajeve [2]. Implementirane su dvije vrste računa. Jedan je za administratora, drugi za članove. Administrator ima kalendar za upis događaja, može ih filtrirati, vidjeti polaznike, provjeriti plaćanja, zabilježiti prisutnost. Slika 2.2. prikazuje sučelje aplikacije gdje su vidljivi polaznici grupe. O svakom korisniku može vidjeti više detalja odlaskom na njihov profil. Postoji poseban dio za pregled svih kontakata gdje administrator može dodavati korisnike u grupe, dodijeliti im uloge i ostvariti komunikaciju s njima. Administrator može izraditi izvješća,

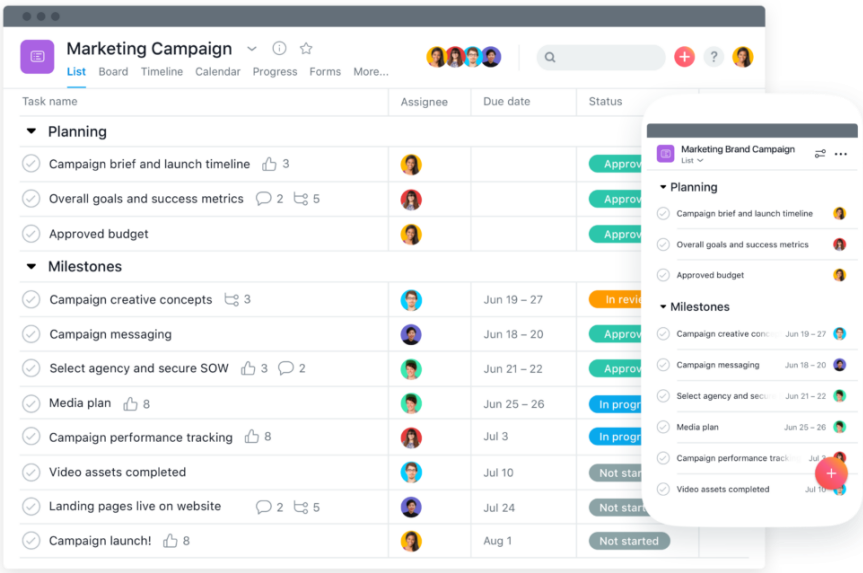


Sl. 2.2. LoveAdmin [2]

izmijeniti fakture te vidjeti statističke podatke na raznim dijagramima. Automatski se uključuju popusti, plaćanja i prikupljanje podataka o istim kako bi se smanjila količina posla administratora. Članovi mogu upravljati vlastitim informacijama, pratiti svoje grupe i jednostavno i brzo platiti članarinu. Iako je implementirano manje funkcionalnosti, u usporedbi s ostalim aplikacijama, ostvarena je jednostavnost korištenja.

2.3. Asana

Softver namijenjen upravljanju proizvodima i projektima za timove tvrtke, prikazan na slici 2.3. Kvalitetu proizvoda potvrđuje podatak da 80% kompanija s liste *Fortune 100* upotrebljava Asanu [3]. Za timove je omogućena raspodjela poslova, praćenje napretka, izrada izvješća, održavanje komunikacije unutar timova. Dodano je mnoštvo automatiziranih značajki koje olakšavaju upravljanje poslovnih procesa kao što je upravljanje zadacima i njihova podjela. Administratoru su osigurane razne funkcionalnosti od upravljanja dopuštenjima, postavki privatnosti, sigurnosnih zahtjeva do centraliziranog praćenja projekata. Omogućeni su razni predlošci za projekte čime je osigurano lakše praćenje zadataka.



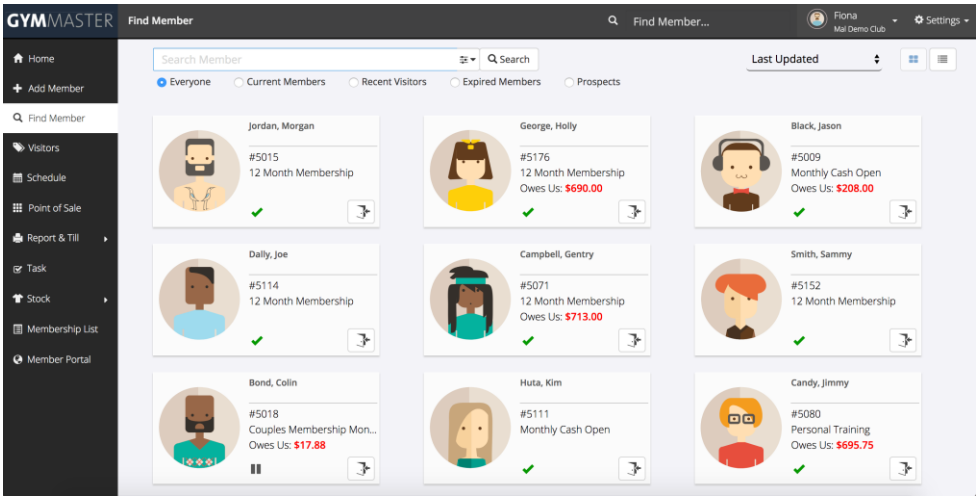
Sl. 2.3. Sučelje aplikacije Asana [3]

2.4. GymMaster

GymMaster navodi kako je njihov sustav najfleksibilniji i najskalabilniji sustav upravljanja članovima teretane [4]. Omogućuje izgradnju zajednice i trajne veze s klijentima koji imaju pristup aplikaciji sa svih uređaja koji imaju pristup internetu. Ulaskom u aplikaciju dolazi se na stranicu

za osoblje koja je vizualno podijeljena na lijevu i desnu stranu. Na lijevoj strani nalazi se raspored događaja prvotno za trenutni dan, ali može se potražiti raspored za bilo koji prošli ili budući datum.

Na desnoj strani nalaze se zbirni podaci o članovima kao što je statistika broja članova, novih učlanjenja, broja posjetitelja te raznih grafičkih prikaza. Omogućeno je dodavanje raznih načina plaćanja od tjednih do godišnjih kao i prikaz svih dosadašnjih transakcija. Svaki trener može poslati poruku članu putem elektroničke pošte ili SMS-a. Na slici 2.4. je prikazan dio aplikacije koji se bavi pretragom članova. Za razliku od dosad navedenih aplikacija implementirano je stvaranje automatiziranih poruka. Mogu se vidjeti razni podaci o svakom članu, tako i programi i razna mjerenja kao što je BMI. U kalendar je moguće dodati nove aktivnosti s točnim vremenom održavanja i brojem osoba koje se mogu prijaviti za taj termin vježbanja. Ostvareno je izdvajanje članova u razne formate po željenim stupcima jedino što je vizualno vrlo neprivlačno stvoren dio u kojemu se odabiru stupci ili brišu jer na jednom mjestu ima previše gumba. Uočava se praćenje trendova pa su tako jedini sustav za upravljanje teretanom s komunikacijom generiranom umjetnom inteligencijom.

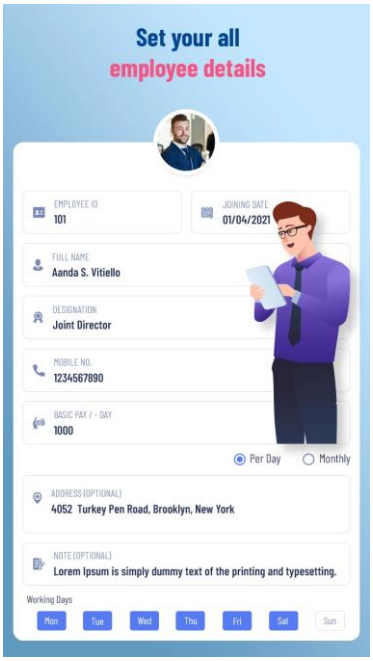


Sl. 2.4. Sučelje aplikacije GymMaster [4]

2.5. EMS: Employee Management

EMS sustav za upravljanje zaposlenicima koristi se za praćenje dnevnih aktivnosti zaposlenika. Zbog osjetljivosti osobnih podataka *Techno Information* upoznaje korisnika s uvjetima pružanja aplikacije pri prvom korištenju aplikacije. Omogućen je unos osnovnih podataka o zaposleniku, praćenje prisutnosti, odsutnosti, kada je zaposlenik na godišnjem, poludnevnih i prekovremenih sati. Može se pratiti podatke u aplikaciji i generirati PDF izvješće o plaćama zaposlenika. Kako bi se dodatno osigurali podaci omogućena je sigurnosna kopija svih podataka o zaposlenicima u

oblaku. Izgled aplikacije [5], prikazan slikom 2.5. , ne odražava profesionalnu stranu aplikacije kao što se to moglo vidjeti u prijašnjim aplikacijama.



Sl. 2.5. EMS [5]

3. VOĐENJE EVIDENCIJE ČLANOVA

Na problematiku vođenja evidencije članova nailaze gotovo sve udruge i klubovi. Povećanjem broja članova postaje sve teže pratiti tko je platio članarinu ili se pridružio u još neku od mogućih grupa ili tko je postao novi član. Uz sve je to najčešće potrebno poznavati ne samo ime i prezime svojih članova već i podatke kao što su njihove elektroničke adrese i datumi rođenja. Kako sve te podatke ne bi voditelj morao zapisivati na uobičajen način, na papir, korisno je poslužiti se raznim aplikacijama u kojima sve ono što je važno osoba može trajno zapisati, ali i lako izmijeniti. Time se dolazi do pojma menadžmenta jer ne samo što želimo te podatke upisati već je poželjno logički organizirati sve te podatke.

Prema [6] menadžment je djelatnost planiranja, organiziranja, kombiniranja, koordiniranja i kontroliranja aktivnosti i organizacijskih resursa kojemu je svrha postizanje organizacijskih ciljeva. Predstavlja vrlo široku djelatnost i uključenost raznih zadataka. Cilj mu je ostvariti željene rezultate na raznim razinama društva i na svim razinama organizacije. Konstituiranje menadžmenta kao znanosti veže se za sam kraj XIX. i početak XX. st. Jedan od prvih teoretičara menadžmenta bio je rudarski inženjer, Francuz Henry Fayol. Prvi je izdvojio menadžment kao posebnu djelatnost poduzeća i definirao ga kao niz međusobno povezanih zadataka i funkcija zato se smatra „ocem modernoga menadžmenta“ [6].

Kvalitetan menadžment prikazuje uspješnost organizacije stoga je jedno od najvažnijih resursa organizacije i društva. Svaki voditelj želi biti što produktivniji i uspješniji u obavljanju svojih zadataka. Na primjeru manjeg sportskog kluba razmotrit će se na koje sve prepreke može naići nadležna osoba za vođenje evidencije plaćanja tog kluba. Za neke specifične situacije aplikacija može odgovarati i voditeljima teretana koji traže jednostavnu aplikaciju samo za evidenciju članova bez mogućnosti izrade programa svakom klijentu. U sportskom klubu od velike je važnosti da treneri budu u prijateljskom odnosu s članovima. Žele znati koliko je dugo neki član u klubu ili kada mu je rođendan, ali u manjim klubovima često treneri vode brigu i o plaćanju članarina. Kako sve te podatke ne bi morali pamtiti, aplikacija za vođenje evidencije članova omogućuje im sve navedeno. Za svakog pojedinca imaju njihove osobne podatke, adresu elektroničke pošte kojom mogu stupiti u kontakt s njima ili za plaćanje članarine ili kako bi prenijeli obavijesti. Na slici 5.3. prikazan je izgled aplikacije gdje su za primjer podataka stvoreni korisnici teretane. Lako se može uočiti kako je moguće kreirati razne grupe koje pojedina teretana može nuditi, od grupnih treninga do evidencije članova koji plaćaju članarinu za rad na spravama. Za svaku grupu samostalno se definira cijena i uključuju članovi.

4. PROGRAMSKO RJEŠENJE APLIKACIJE

U ovom poglavlju objašnjeni su pojedini dijelovi aplikacije uz blokove programskog koda i načini korištenja tehnologija. Opisani su modeli, autentifikacija korisnika i korisnikov profil, te sve ono što je omogućeno za članove i grupe koje osoba sama stvara.

4.1. Korišteni alati

Pri izradi ove aplikacije korišteno je razvojno programsko okruženje Visual Studio Code te ekstenzija *Black Formatter* za automatsko uređivanje koda prilikom spremanja datoteke kako bi sav Python kod bio jednako formatiran.

Svi slikovni primjeri aplikacije prikazani su u pregledniku Chrome. Glavni je *font* putem poveznice uključen u aplikaciju s Google Fonts [7], a logo je izrađen u Canva alatu [8].

Za praćenje tijeka izrade aplikacije korišten je Git koji je distribuirani sustav za verzioniranje koda.

Django je besplatan okvir otvorenog koda koji može ubrzati razvoj web aplikacije koja se gradi u programskom jeziku Python i korišten je za izradu ove web aplikacije. Django koristi paradigmu MVC (engl. *model-view-controller*) koju naziva MTV (engl. *model-template-view*) o kojoj će više riječi biti u nastavku rada.

HTML datotekama prikazuju se rezultati upita nad bazom zato uz uobičajene HTML oznake u Django predloščima nalaze se i varijable, oznake, filteri, petlje i drugo [9]. S varijablama se ispisuje sadržaj koji je proslijeđen iz konteksta. Varijable se pišu unutar vitičastih zagrada. Objekt nalikuje Python rječniku (engl. *dictionary*) gdje je sadržaj mapiran u odnosu ključ-vrijednost (engl. *key-value)*. Oznake se pišu unutar *{% oznaka %}*. Označavaju proizvoljnu logiku unutar prikaza. Oznake mogu biti ispis sadržaja, logička cjelina ili pristup drugim oznakama iz predloška. Petlje se zapisuju unutar *{% %}* gdje se definiraju njihove oznake.

CSS se koristi za definiranje stilova web stranica, uključujući dizajn, izgled i varijacije prikaza za različite uređaje i veličine zaslona. Kako bi se primijenila oblikovanja definirana u CSS dokumentu postavlja se poveznica unutar oznaka *<head></head>*. Uključivanje unutar ovog rada prikazano je na slici 4.1.. Unutar aplikacije potrebno je stvoriti direktorij *static* unutar kojeg se postavlja *.css* dokument. Uz oznaku *{% load static %}* referencira se na taj dokument. Za izradu ove aplikacije u CSS dokumentu definirane su prilagođene boje pozadine, navigacijske trake, linkova i teksta.

```
<link rel="stylesheet" href="{% static 'app/css/base.css' %}">
```

Sl. 4.1. Uključivanje CSS dokumenta u aplikaciju

Ostala oblikovanja odrađena su uz pomoć Bootstrapa koji omogućuje korištenje unaprijed izgrađenih komponenti kao što su tablice, kartice i sustav rešetke, ali i određenih boja čime je ostvarena brža i jednostavnija izrada Web stranica.

Slika 4.2. prikazuje div element s atributom *class* unutar kojega su Bootstrap oblikovanja. Kontejner (engl. *Container*) je osnovni gradivni element Bootstrapa koji osigurava responzivan dizajn. Ima predefinirane margine, ispune, poravnanja za HTML elemente. Boja je pozadine *bg-light*, predstavlja predefiniranu boju imena *light* koja je svijetlo sive boje. *My-4* su vertikalne margine dimenzije 4 od predefiniranih veličina u kojima je najveća dimenzija označena znamenkom pet. Klasa *p* je oznaka za ispune, uz *x* označava samo horizontalne ispune, te je *mb-5* oznaka za donju marginu najveće dimenzije. U aplikaciji je korištena peta verzija Bootstrapa koja je uvela male promijene u nazivima margina i ispuna iz oznake za lijevo kao *l* (engl. *left*) u oznaku *s* (engl. *start*) te je oznaka za desno od prijašnje oznake *r* (engl. *right*) zamijenjena sa *e* (engl. *end*).

```
<div class="container bg-light my-4 px-5 py-4 mb-5">
```

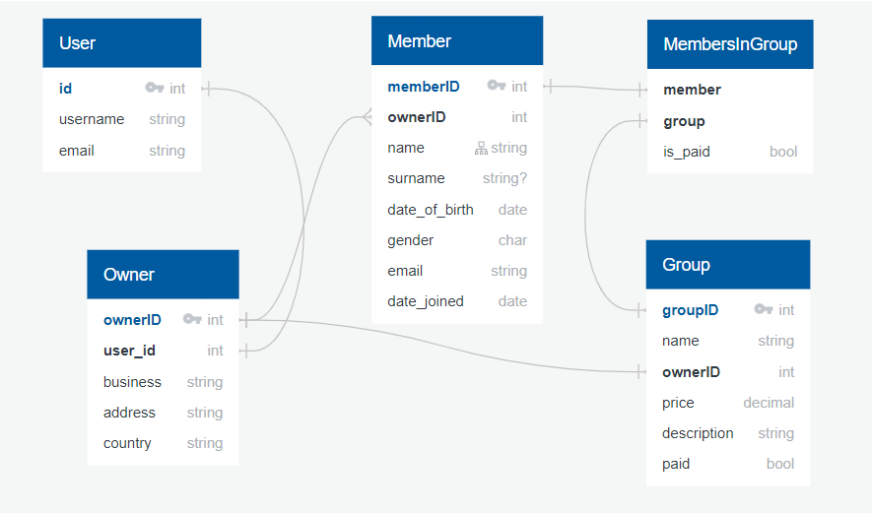
Sl. 4.2. Isječak Bootstrap oblikovanja u aplikaciji

4.2. Modeli

Django koristi objektno-relacijsko preslikavanje kojim se baza podataka i odnosi među entitetima opisuju Python kodom. Pruža automatiziranu podršku za mapiranje podatkovnih primjera (engl. *tuples*) u objekte i natrag, uzimajući u obzir sve razlike između SQL baze podataka i Python koda. U izradi aplikacije korišten je Djangov predodređen tip baze podataka - SQLite 3. Tip baze

podataka može se mijenjati u *settings.py*. Na slici 4.3. prikazani su svi modeli aplikacije, s njihovim vezama i tipovima podataka kojima su predstavljeni atributi klase. Slikom 4.4. prikazano je kako Python klasa *Group* izgleda u samoj aplikaciji unutar *models.py* datoteke. Modeli se u Djangu vežu za tablice baze podataka, a predstavljeni su Python klasama u *models.py* datoteci. Kada se stvore nisu trajno zadani već se mogu izmjenjivati i dopunjavati nakon primjene migracija.

U aplikaciji „*Member Management*“ kao što je slikovito prikazano na slici 4.3. postoji pet modela *User*, *Owner*, *Member*, *Group* i *MembersInGroup*. *User* je Djangov generički model povezan s *Owner* modelom vezom jedan-na-jedan. On ubrzava izgradnju osnovnih korisničkih modela i omogućuje jednostavnu implementaciju autentifikacije korisnika. Ostali modeli bit će više objašnjeni u nastavku poglavlja zajedno s funkcionalnostima uz koje su vezani.



Sl. 4.3. Modeli aplikacije

```
class Group(models.Model):
    owner = models.ForeignKey(User, on_delete=models.CASCADE)
    name = models.CharField(max_length=50)
    price = models.DecimalField(max_digits=8, decimal_places=2)
    description = models.TextField(max_length=300, default=None)
    paid = models.BooleanField(default=False)
```

Sl. 4.4. Klasa modela *Group*

4.3. Autentifikacija

U ovom poglavlju objašnjen je način rada registracije, prijave i odjave korisnika.

Za registraciju korisnika stvoren je poseban HTML dokument *bootstrap_form.html* u kojemu su navedena polja koja zahtijevaju popunjavanje pri registraciji uz html elemente forme za unos podataka, ali zbog manjkavosti izgleda i pravilnog rasporeda korišten je *widget_tweeks* i bootstrap oblikovanja. Nakon što korisnik unese odgovarajuće podatke za registraciju i pritisne gumb *Sign up* izvršit će se metoda POST koja će prikupljene podatke spremiti kao objekt tablice Owner.

Slika 4.5. prikazuje način na koji je implementirano ograničenje kako niti jedna osoba ne bi imala jednaku adresu elektroničke pošte pri registraciji. Podiže se iznimka ako osoba unese već postojeću adresu elektroničke pošte. Za registraciju i prijavu korištena je jednaka Bootstrap forma što pridonosi smanjenju dupliciranja koda, a poštuje se i DRY načelo (engl. *Don't repeat yourself*) korištenjem *base.html* dokumenta i skupnih CSS oblikovanja.

```
def clean(self):
    email = self.cleaned_data.get("email")
    if User.objects.filter(email=email).exists():
        raise ValidationError("Email exists")
    return self.cleaned_data
```

Sl. 4.5. Isječak koda klase *UserRegistrationForm*

Slika 4.6. prikazuje *login.html* koji služi za prijavu korisnika. Korisnik prilikom prijave unosi korisničko ime i zaporku. Nakon što korisnik klikne na gumb *Sign in*, obavlja se operacija POST. Ako su podaci točni korisnika se preusmjerava na stranicu menadžmenta aplikacije.

```
{% extends 'app/base.html' %}
{% block content %}

<h1 class="text-center">Sign in</h1>

<form method="post">
    {% csrf_token %}

    {% include 'includes/bootstrap_form.html' with form=form %}

    <div class="row mt-4 text-center">
        <div clss="col-md-8 ms-2">
            <input type="submit" value="Sign in" class="custom-btn btn px-4 py-2">
        </div>
    </div>
</form>
<p class="text-center mt-5">Don't have an account? <a href="/accounts/sign-up">Create an account</a>.</p>
{% endblock %}
```

Sl. 4.6. Isječak aplikacije - *login.html*

U svim Django stranicama (HTML dokumentima) potrebno je navesti *{% csrf_token %}*, vidljiv na slici 4.6. , koji osigurava da se svi zahtjevi koji se vraćaju unakrsno provjere kako ne bi došlo do *Cross Site Request* prijevare. Kako se može uočiti na slici 4.6. , korisniku je omogućen odlazak na stranicu za registraciju ako još nema račun te je na stranici za registraciju ostvareno suprotno.

U radu je jedan od najčešće korištenih atributa *class* koji služi za dodavanje stilskih oblikovanja CSS i/ili Bootstrap oblikovanja vidljiv i u isječku koda na slici 4.6. Oznake se pišu unutar *{% oznaka %}*. Označavaju proizvoljnu logiku unutar prikaza. Oznake mogu biti ispis sadržaja, logička cjelina ili pristup drugim oznakama iz predloška. Petlje se zapisuju unutar *{% %}* gdje se definiraju njihove oznake. Kako bi se osoba odjavila unutar *base.html* dokumenta u navigaciju je dodana poveznica, a uz Djangov ugrađeni sustav za autentifikaciju funkcionalnost je ostvarena.

4.4. Profil

Django koristi paradigmu MVC (engl. *model-view-controller*) koju naziva MTV (engl. *model-template-view*) koji će biti opisan na primjeru korisnikova profila iz aplikacije uz pomoć slika 4.7. 4.8. i 4.9. Prvi je sloj MTV-a model koji je objašnjen u potpoglavlju 4.2.. Idući je sloj *template*, zapravo prikaz, koji u Djangu predstavlja korisničko sučelje. Uzima web zahtjev i vraća web odgovor koji može biti preusmjeravanje, statusni kod, HTML sadržaj neke web stranice. Slika 4.9. prikazuje navedeno. Treći je dio *View*, u drugim MVC sustavima predstavlja *Controller*, koji sadrži logiku koja omogućuje odgovor korisniku na traženi zahtjev, zapravo povezuje cjeline, uspostavlja vezu s bazom podataka. Na slici 4.8. prikazan je *View* profila na kojemu je moguće

```
path("profile/<int:user_id>", views.profile, name="profile"),
```

Sl. 4.7. Putanja za stranicu profila unutar *urls.py*

```
@login_required()
def profile(request, user_id):
    user = request.user
    email_form = EmailChangeForm(user)
    password_form = SetPasswordForm(user)
    context = {
        "user": user,
        "email_form": email_form,
        "password_form": password_form,
    }
    return render(request, "accounts/profile.html", context)
```

Sl. 4.8. *View* profila

```
{% extends 'app/base.html' %}
{% block content %}

<h1 class="mt-5 mb-5">Profile</h1>
<div class="ms-3">
  <h5 class="mb-3"><b>Username:</b> {{ user.username }}</h5>
  <h5 class="mb-3"><b>Email address:</b> {{ user.email }}</h5>
  <h5 class="mb-3"><b>The type of work for which this application will be used:</b> {{user.owner.business}}</h5>
  <h5 class="mb-3"><b>Company address:</b> {{user.owner.address}}</h5>
  <h5 class="mb-3"><b>Country:</b> {{user.owner.country}}</h5>
</div>
```

Sl. 4.9. Dio *profil.html*

uočiti dekorater *@login_required()* koji osigurava samo prijavljenim korisnicima uvid u vlastiti profil. Na profilu je osim prikaza podataka unesenih pri registraciji omogućena izmjena adrese elektroničke pošte korisnika te njegove zaporke. Oba koriste već spomenuti dekorater *@login_required* kako ne bi bilo potrebno tražiti trenutnu zaporku ili adresu elektroničke pošte jer je sigurno da će samo autentificirani korisnik moći tome pristupiti.

4.5. Rad s članovima

Prije nego li se opišu sve mogućnosti s članovima (engl. *members*) opisat će se korišteni CRUD pristup. CRUD označava izvođenje operacija *Create*, *Retrieve*, *Update* i *Delete* nad tablicom u bazi podataka.

```
@login_required()
def members(request):
    if request.method == "POST":
        form = MemberForm(request.POST)
        if form.is_valid():
            member = Member.objects.create(
                owner=request.user,
                name=form.cleaned_data.get("name"),
                surname=form.cleaned_data.get("surname"),
                date_of_birth=form.cleaned_data.get("date_of_birth"),
                gender=form.cleaned_data.get("gender"),
                email=form.cleaned_data.get("email"),
            ).save()

            return HttpResponseRedirect(reverse("app:management", args=[]))
        else:
            context = {"form": form}
            print(form.errors)
            return render(request, "management/add_member.html", context)
    else:
        form = MemberForm()
    return render(request, "management/add_member.html", {"form": form})
```

Sl. 4.10. *Members view*

Create View odnosi se na pogled (logiku) za stvaranje instance tablice u bazi podataka. To je nalik primanju unosa korisnika i pohranjivanje u određenu tablicu. Kako bi se kreirao novi član moraju se ispuniti sva polja stvorena za člana, od imena do adrese elektroničke pošte. Na slici 4.10. vidljivo je dohvaćanje korisničkog unosa i spremanje novog objekta u tablicu *Member*.

4.5.1. Dodavanje, uređivanje i uklanjanje članova

Kao što je navedeno u prošlom potpoglavlju za dodavanje novog člana korišten je CRUD pristup, zaslužan je i za uređivanje te uklanjanje članova. Za dodavanje novog člana stvoren je *add_member.html* koji predstavlja formu u kojoj se nalaze polja za upis imena, prezimena, datuma rođenja, spola te adrese elektroničke pošte. Kako bi se podaci kreiranoga člana mogli izmijeniti unutar *edit.html* stvorena je forma s odgovarajućim porukama za greške. Neke greške na koje korisnik može naići su prazno polje bilo kojeg od gore navedenih elemenata ili datum rođenja koji je postavljen na budući datum. Uklanjanje člana iz tablice menadžmenta ostvareno je definiranjem *destroy viewa* kojega je moguće vidjeti na slici 4.12. Pristupa članu preko indeksa metodom

```
@classmethod
def get_owner_members(self, user):
    return Member.objects.filter(owner=user)

@classmethod
def get_member_by_id(self, id):
    return Member.objects.get(pk=id)
```

Sl. 4.11. Isječak koda iz klase *Member* unutar *models.py*

get_member_by_id klase *Member* prikazane na slici 4.11. te uz Djangovu preddefiniranu metodu *delete* izbriše tog člana za koji je osoba kliknula poveznicu s natpisom *Delete*. Na slici 4.12.

```
@login_required()
def edit(request, id):
    member = Member.get_member_by_id(id)
    return render(request, "management/edit.html", {"member": member})


@login_required()
def update(request, id):
    member = Member.get_member_by_id(id)
    form = MemberForm(request.POST, instance=member)
    if form.is_valid():
        form.save()
        return HttpResponseRedirect(reverse("app:management", args=[]))

    return render(request, "management/edit.html", {"member": member, "form": form})


@login_required()
def destroy(request, id):
    member = Member.get_member_by_id(id)
    member.delete()
    return HttpResponseRedirect(reverse("app:management", args=[]))
```

Sl. 4.12. Isječak koda iz *views.py*

vidljivi su *edit* i *update viewovi* zaslužni za odlazak s menadžment stranice na *edit.html* te nakon izvršene izmjene ponovni povratak na menadžment stranicu uz određene putanje definirane u *urls.py*.

4.5.2. Prikaz informacija o članovima

Django nudi admin sučelje koje je vrlo korisno za popunjavanje baze podataka. Kada su definirani modeli, Django kreira sučelje na kojem *super user* može dodavati, izmjenjivati i uklanjati objekte. Korišteno je dok nije stvoren vlastito sučelje za dodavanje, uređivanje te brisanje članova i grupa. Kako bi se omogućio prikaz modela aplikacije potrebno je unutar *admin.py* napisati prikazano na slici 4.13.

```
admin.py
from django.contrib import admin
from .models import Owner, Group, Member, MembersInGroup


admin.site.register(Owner)
admin.site.register(Group)
admin.site.register(Member)
admin.site.register(MembersInGroup)
```

Sl. 4.13. *admin.py*

Za prikaz svih informacija o članu, kako nisu sve informacije prikazane izravno u tablici menadžmenta, korišten je skočni prozor stvoren s Bootstrapovom *modal* komponentom. Bootstrapov JavaScript modal koristi se za dodavanje dijaloških okvira na web stranicu za skočne prozore, korisničke obavijesti ili potpuno prilagođeni sadržaj [10].

Na slici 4.14. vidljiv je *button* element za informacije koji na klik otvara *modal* definiran unutar *management.html*. Unutar *views.py* definiran je *view_member* te unutar *urls.py* stvorena putanja za prikaz svih informacija o pojedinom članu. Kako bi u tom prikazu bile navedene i sve grupe u

```
<button type="button" class="btn" href="{% url 'app:view_member' member.id %}" data-bs-toggle="modal" data-bs-target="#myModal{{member.id}}">
    <div class="mt-2 center"><img class="info-icon" src="{% static 'app/images/info.png' %}" alt="Info"></img></div>
</button>
```

Sl. 4.14. Isječak koda iz *management.html*

koje je uključen pojedini član stvorena je metoda *get_all_members_in_group* unutar modela *MembersInGroup*.

4.5.3. Filtriranje i sortiranje

Kako sučelje ne bi bilo nepregledno i pretrpano gumbima, pretraživanje uz filtriranje i sortiranje omogućeno je klikom na ikonicu *search*. Klikom na nju odlazi se na Bootstrapov *modal*. Sastoji se od polja za upit prema kojemu se želi pretražiti članove, izbornika s više opcija za filtriranje i sortiranje silazno ili uzlazno. Na slici 4.15. prikazana je definicija takve forme unutar *forms.py*.

```
class SearchForm(forms.Form):
    search_query = forms.CharField(required=False)
    filter_choice = forms.ChoiceField(
        choices=[
            ("name", "Name"),
            ("surname", "Surname"),
            ("email", "Email"),
            ("gender", "Gender (M/F)"),
            ("date_joined", "Date of joining"),
        ],
        widget=forms.RadioSelect,
        required=False,
    )

    sort_choice = forms.ChoiceField(
        choices=[
            ("asc", "Ascending"),
            ("desc", "Descending"),
        ],
        widget=forms.RadioSelect,
        required=False,
    )
```

Sl. 4.15. Forma za pretragu članova

4.5.4. Izvoz podataka o članovima u Excel tablicu

Na slici 4.16. prikazan je dio koda iz *management.html* gdje ikonica *Excel* predstavlja poveznicu na *export_members_xls* definiranu u views.py. U njoj je definirano ime Excel dokumenta, stupci koji će se spremati za sve korisnikove članove.

```
<span class="col-1">
  <a class="excel-icon" href="{% url 'app:export_excel' %}">
    <img class="excel mt-2" src="{% static 'app/images/excel.png' %}" alt="Excel">
  </a>
</span>
```

Sl. 4.16. Isječak iz *management.html*

4.6. Rad s grupama

4.6.1. Dodavanje, uređivanje i uklanjanje grupa

Rad s grupama slične je implementacije kao prethodno definiran rad s članovima jer je riječ o CRUD pristupu. Za dodavanje nove grupe stvoren je *add_new_group.html* koji predstavlja formu u kojoj se nalaze polja za upis imena grupe, opisa te cijene. Izmjena podataka grupe ostvarena je unutar *edit_group.html* u kojemu svako polje prvotno prikazuje stare podatke, kako korisnik ne bi izgubio stare podatke ako nije želio mijenjati sve podatke grupe, te je ostvaren prikaz poruka za greške. Uklanjanje je grupe ostvareno dohvaćanjem te određene grupe preko ID-a grupe i Djangovom *delete* metodom.

4.6.2. Pregled grupe

Unutar *management.html* postavljena je poveznica na svako ime grupe. Klikom na poveznicu usmjeri se korisnika na *group_details.html*. Prvo su prikazani podaci o grupi uneseni stvaranjem te grupe. Slijedi prikaz svih članova te grupe unutar kartica čije je oblikovanje napravljeno unutar *member_card.html*. Svakom članu postavljen je ikonica minusa za uklanjanje iz grupe. Klikom na nju pojavi se oblačić stvoren Bootstrapovim modal oblikovanjem kako bi korisnik potvrdio uklanjanje tog člana iz te grupe u kojoj se trenutno nalazi. Navedeno je ostvareno uz metodu *remove_member_from_group* unutar *views.py* filtriranjem po članu i grupi objekata *MembersInGroup*, prikazano na slici 4.17. Za svakog člana postavljena je i kućica za označavanje je li platio članarinu za tu grupu. Klikom na *Approve payment* gumb pamti se stanje i nakon izlaska iz te grupe. Na kraju stranice navedeni su ostali članovi, oni koji nisu uključeni u tu grupu. Izlistani su unutar tablice uz gumb za dodavanje u grupu. Klikom na njega pojavljuje se prozor za potvrdu dodavanja osobe za člana te grupe.

```
@login_required()
def remove_member_from_group(request, group_id, member_id):
    member = Member.objects.get(pk=member_id)
    group = Group.objects.get(pk=group_id)

    MembersInGroup.objects.filter(Q(member=member), Q(group=group)).delete()

    return HttpResponseRedirect(reverse("app:group_details", args=[group.id]))
```

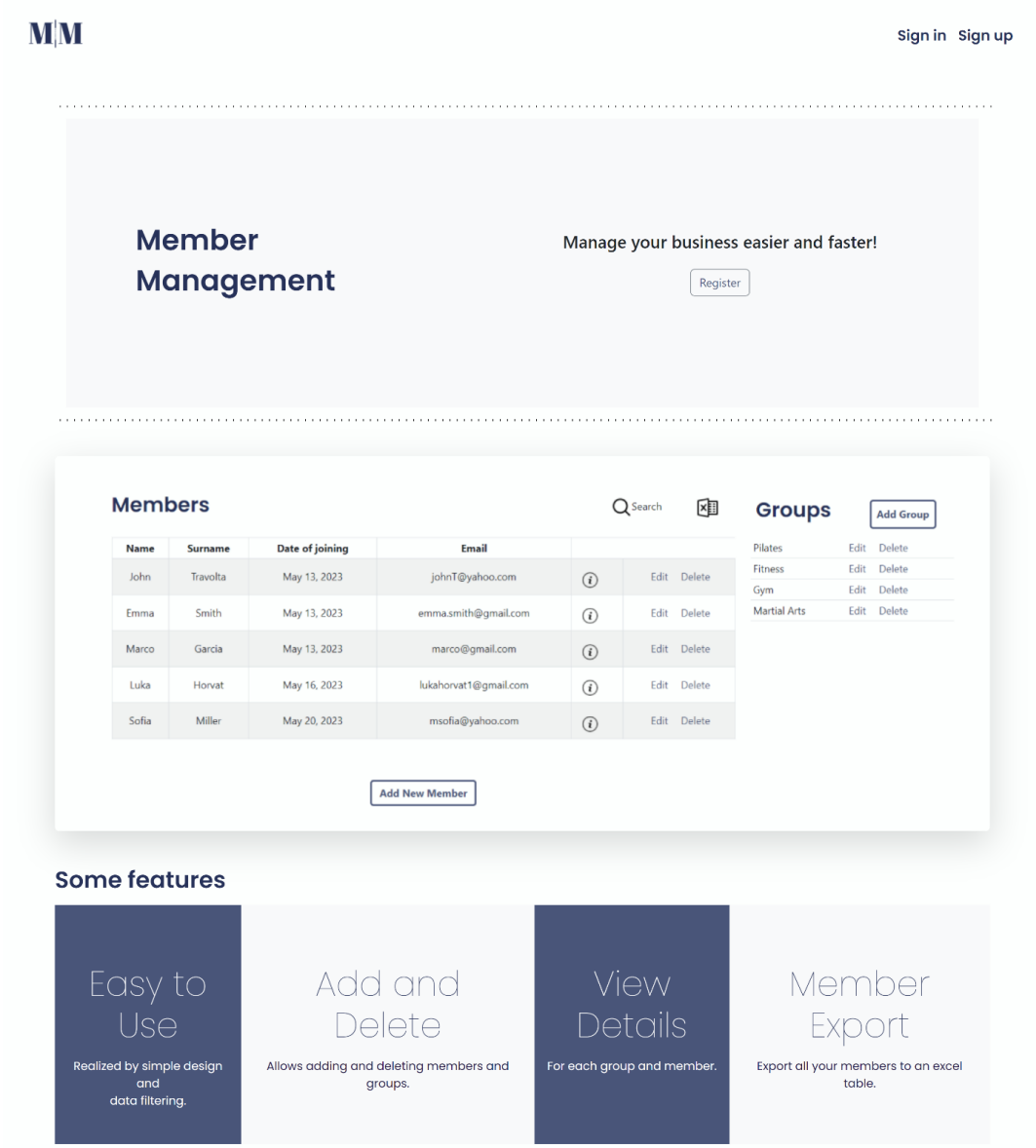
Sl. 4.17. Isječak iz *views.py*

5. KORIŠTENJE APLIKACIJE

U ovome su poglavlju opisane sve stranice i njihovo korištenje. Svim je korisnicima dostupna uvodna stranica. Na njoj je vidljiv prikaz aplikacije registriranoj osobi uz popis dijela značajki aplikacije, dok samo prijavljeni korisnici imaju uvid u ostatak aplikacije. Omogućen im je pregled profila i menadžment dio na kojem mogu stvarati i uređivati članove i grupe.

5.1. Početna stranica

Nakon pokretanja aplikacije, prikazat će se početna stranica. Na slici 5.1. prikazan je izgled početne stranice. Može se uočiti da sadrži navigacijsku traku, * * element u kojemu su naslov



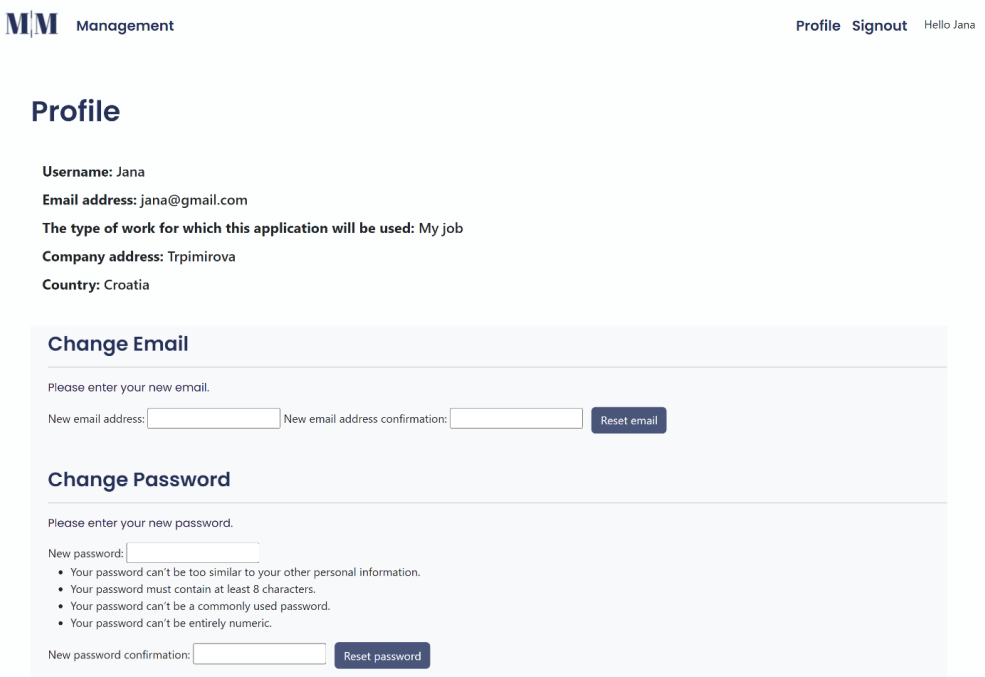
Sl. 5.1. Izgled početne stranice

i gumb za registraciju uz prikladan tekst, nakon toga postavljena je slika aplikacije i nabrojana su poneka svojstva aplikacije.

Navigacijska traka sastoji se od dva dijela. Na lijevom dijelu navigacijske trake nalazi se logotip stranice i poveznica na menadžment, dok su na desnom dijelu poveznice za prijavu i registraciju korisnika. Ako je osoba prijavljena, u desnom dijelu navigacijske trake nalaze se poveznice za korisnikov profil, odjavu i pozdravna poruka s korisničkim imenom.

5.2. Profil

Ako korisnik u bilo kojem trenutku želi promijeniti svoju adresu elektroničke pošte koju je koristio pri registraciji za ulazak u aplikaciju ili zaporku, to može učiniti na svome profilu, slika 5.2. . Do stranice profila može otići klikom na *Profile* na desnoj strani navigacijske trake. Tu poveznicu korisnik vidi kada je prijavljen u aplikaciji. Na profilu korisnik može pogledati sve svoje podatke koje je unio pri registraciji te već spomenuto promijeniti adresu elektroničke pošte i zaporku za prijavu u aplikaciju. Kako bi zamjena adrese elektroničke pošte i zaporke bila što točnija za korisnika, postavljena su dva polja u koja je potrebno upisati jednake vrijednosti da bi se izvršila promjena.



Sl. 5.2. Izgled korisničkog profila

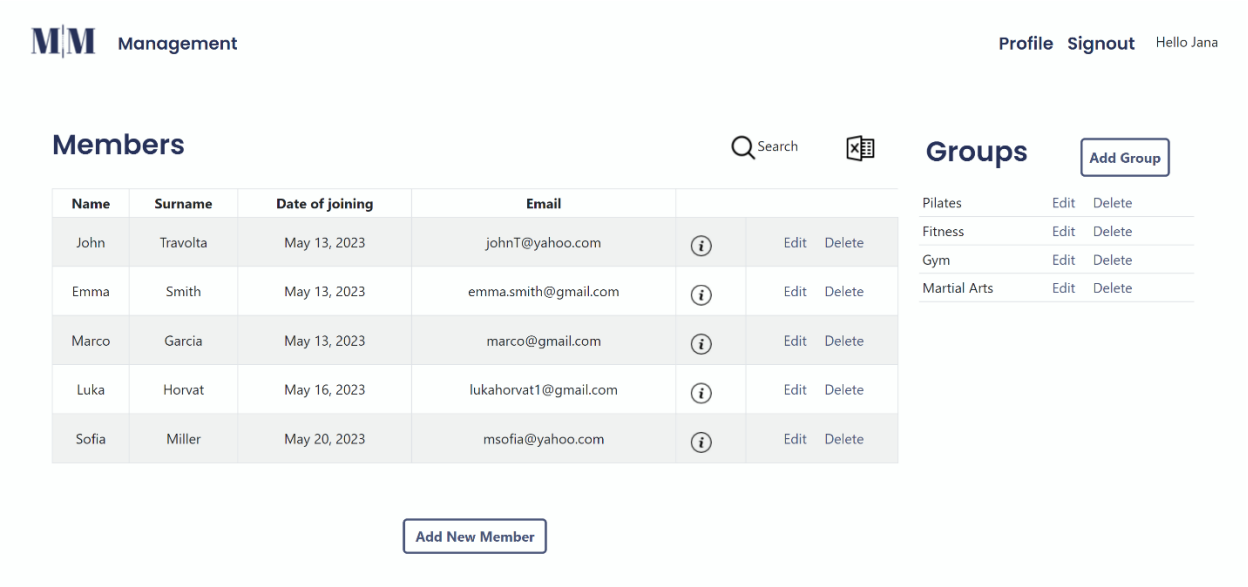
5.3. Upravljački dio

Na slici 5.8. vidljivo je iz kojih je sve slučajeva moguće otići na stranicu menadžmenta. Osnovni je preduvjet odlaska na stranicu menadžmenta uspješna autentifikacija korisnika. Neprijavljeni korisnik nema pristup tom dijelu aplikacije. Pri prvom odlasku na menadžment stranicu korisnik vidi naslove *Members* i *Groups* te pripadne gumbe za dodavanje istih. Stalno prisutne su i ikone *Search* i *Excel* ikona.

Korisnik s menadžment stranice može pristupiti svim dijelovima aplikacije koji su omogućeni za prijavljenog korisnika. S poveznicama na navigacijskoj traci korisnik može pristupiti prije objašnjenom profilu te se odjaviti iz aplikacije. Kao što je prikazano na slici 5.8. korisnik u je omogućen odlazak na stranicu za stvaranje novog člana. Kada ima barem jednog upisanog člana, poveznicom *Edit* omogućen mu je odlazak na stranicu za izmjenu informacija o članu.

Jednako tako ako korisnik želi stvoriti grupu, klikom na gumb *Add group* odlazi na stranicu za stvaranje grupe. Stvorenoj grupi na menadžment stranici, klikom na poveznicu Edit, odlazi na stranicu za izmjenu informacija grupe. Klikom na ime grupe korisnik odlazi na stranicu za prikaz informacija grupe o kojemu će biti riječ u nastavku rada.

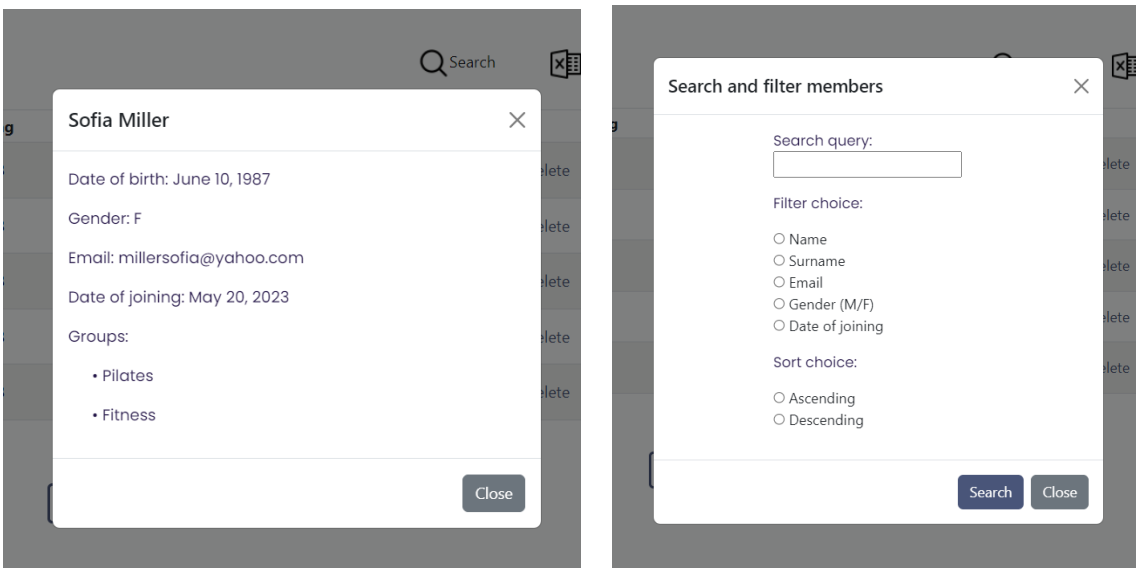
Slika 5.3. prikazuje izgled menadžment stranice s upisanih pet članova i četiri grupe.



Sl. 5.3. Menadžment stranica

Svaki član ima prikazano ime, prezime, datum dodavanja u aplikaciju, adresu elektroničke pošte, gumb za sve informacije o članu, *Edit* i *Delete* poveznice. Na slici 5.4. je prikaz prozora kada

korisnik pritisne ikonu informacija. Ispisani su podaci o članu: datum rođenja, rod, adresa elektroničke pošte, datum učlanjenja te sve grupe u koje je član uključen. Na desnoj slici prikazan je prozor za pretraživanje i filtriranje.

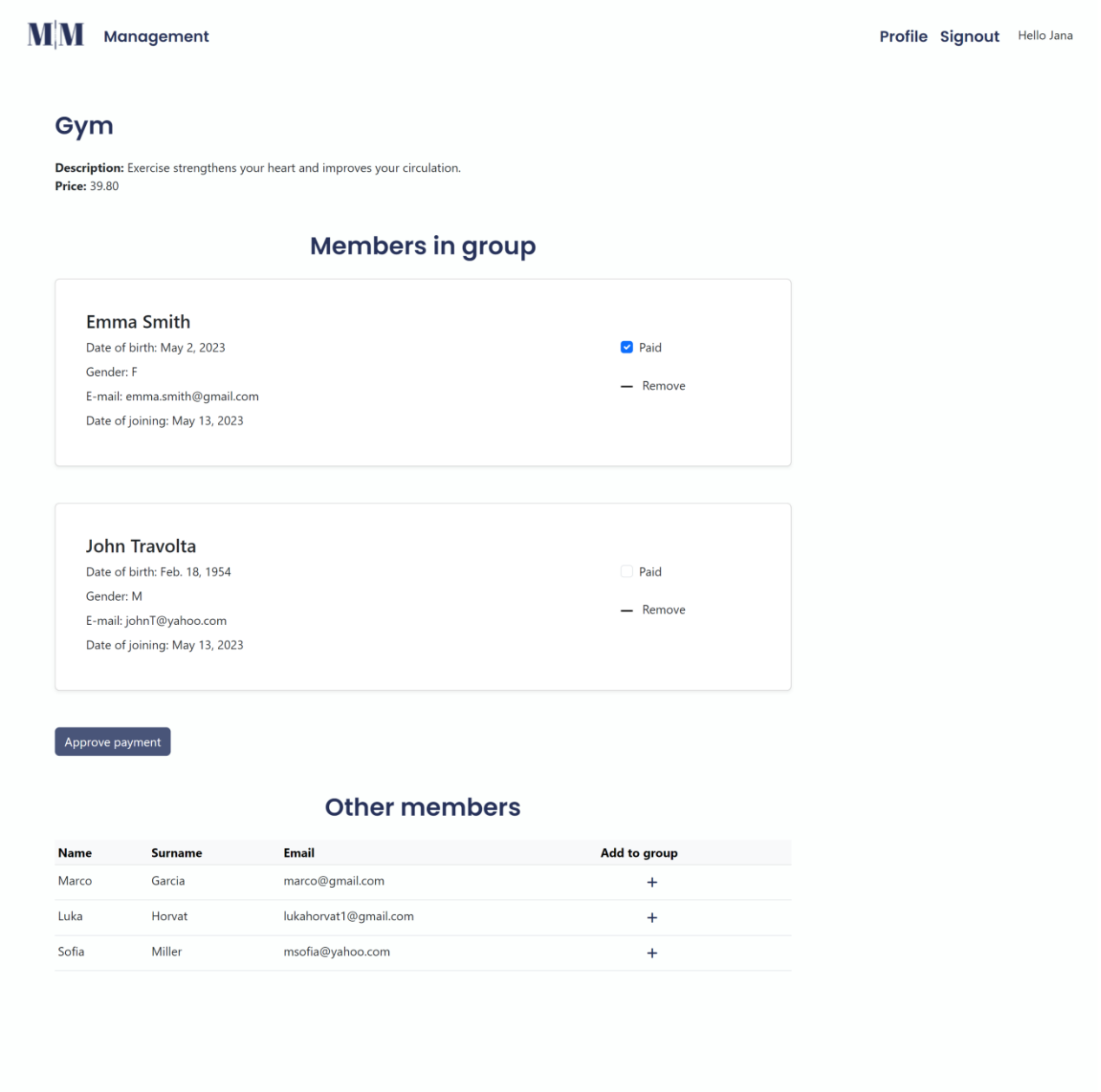


Sl.5.4. Isječak prikaza informacija člana (lijevo) i prozor za filtriranje i pretraživanje članova (desno)

5.4. Pregled grupe

Kada korisnik klikne na naziv grupe dolazi na stranicu za prikaz informacija grupe prikazano na slici 5.5. Na početku stranice su informacije unesene stvaranjem grupe na menadžment stranici s gumbom *Add Group*. Zatim su izlistani članovi koji su dodani u tu grupu uz njihove podatke i kućicom za označavanje (engl. *checkbox*). Ako je osoba platila članarinu za tu grupu, u koju je član uvršten, gumbom *Approve payment* postavit će se kvačica u kućici za označavanje. *Remove* s ikonicom minusa prikazuje oblak s porukom potvrde za uklanjanje člana iz trenutne grupe. U zadnjem dijelu stranice nabrojane su sve osobe unesene u aplikaciju koje nisu članovi trenutne

grupe. Ako bi korisnik poželio dodati osobu klikom na plus ikonicu dobije se prozor za potvrdu istog.

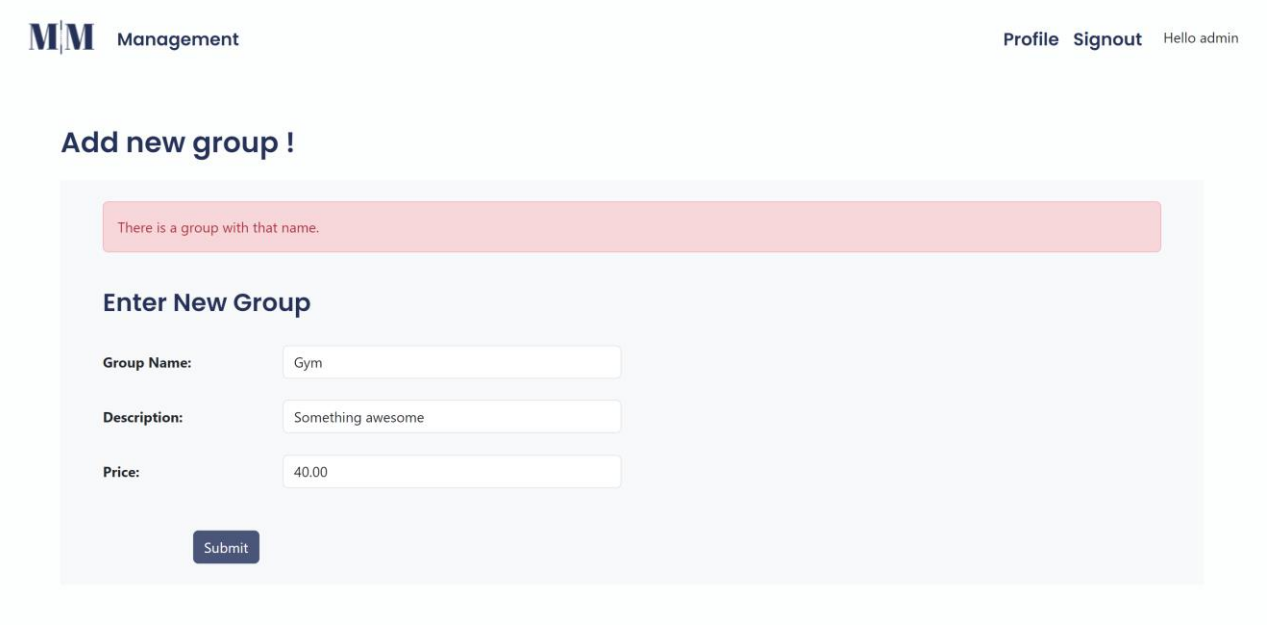


Sl.5.5. Izgled stranice grupe

5.5. Stvaranje novog člana i grupe

Za stvaranje novog člana potrebno je na menadžment stranici kliknuti *Add New Member*. Time korisnik dolazi na stranicu za stvaranje novog člana u kojoj popunjava formu podacima imena, prezimena, datuma rođenja, roda, adrese elektroničke pošte. Ako neko od navedenih polja ne ispuni, dobije poruku kako treba ispuniti određeno prazno polje.

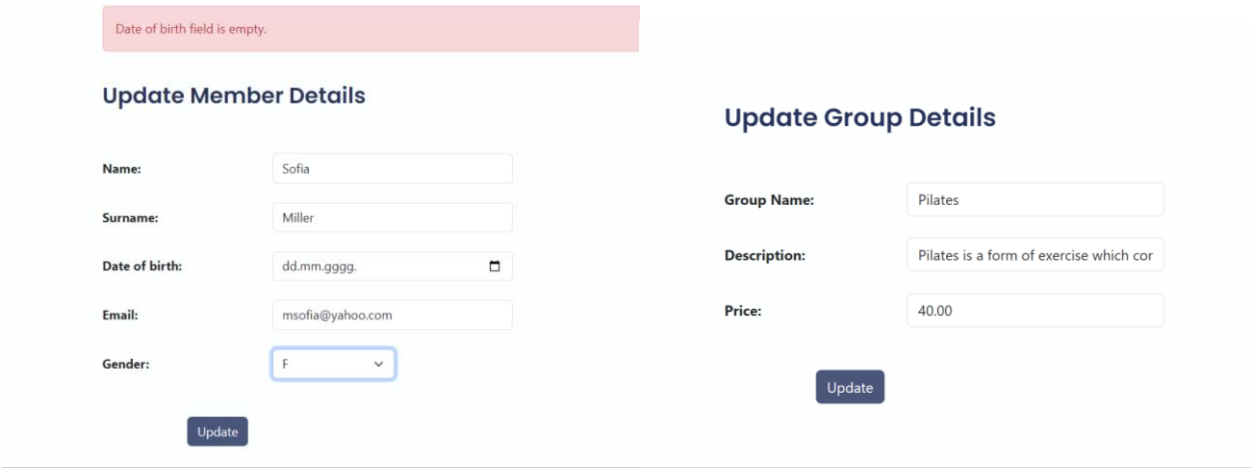
Kada korisnik želi stvoriti novu grupu na menadžment stranici klikne *Add Group*. Time dolazi na stranicu za stvaranje nove grupe kao što je prikazano na slici 5.6. Ako korisnik unese već postojeći naziv grupe dobije prikladnu poruku koja je vidljiva na slici.



Sl.5.6. Stranica za stvaranje grupe

5.6. Uređivanje člana ili grupe

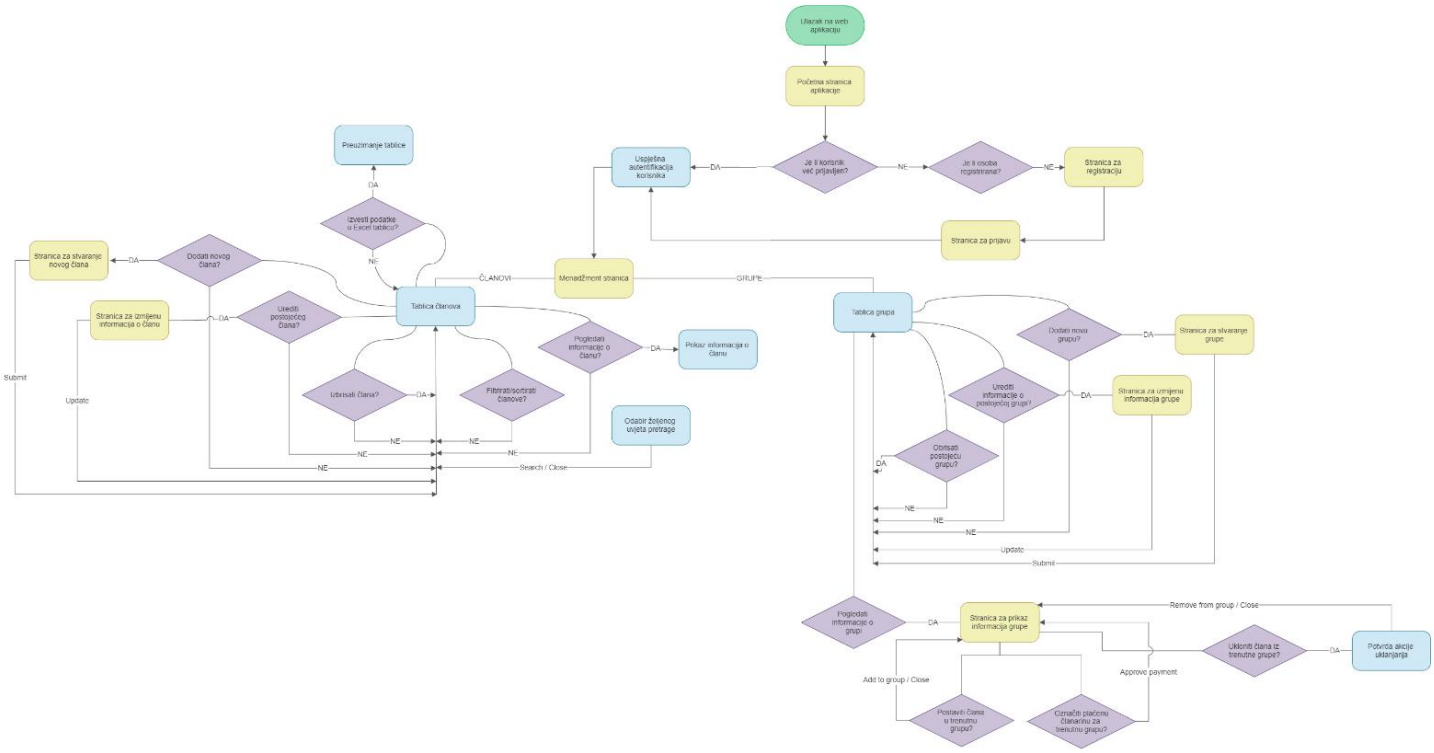
Kada korisnik želi urediti podatke, koje je upisao pri stvaranju člana na menadžment stranici, to može učiniti klikom na poveznicu *Edit*. Podaci će biti prikazani kao na slici 5.7. Ako se ne popuni jedno polje prikaže se poruka i ništa se ne izmijeni dok svi uvjeti ne budu zadovoljeni. Kod izmjene



Sl.5.7. Isječak stranica za izmjenu informacija o članu (lijevo) i grupi (desno)

podataka grupe jedini je uvjet da naziv grupe ne bude jednak već postojećoj grupi kao što je bio slučaj pri stvaranju grupe.

Na slici 5.8. prikazan je dijagram mogućeg kretanja korisnika kroz web aplikaciju. Na strelicama su prikazane definirane radnje kao i radnje pritiska gumba nazvanih kao u aplikaciji na engleskom jeziku. Iz dijagrama je isključen odlazak na profil jer korisnik u bilo kojem trenutku može odlučiti otići na stranicu profila ili iz nje preko navigacije, a otežalo bi čitanje dijagrama.



Sl.5.8. Dijagram kretanja kroz aplikaciju

6. ZAKLJUČAK

Ovim radom opisani su mogući problemi pri vođenju evidencije članstva. Navedena je važnost točnosti, sigurnosti i integriteta podataka, njihova pohrana i organizacija, ažuriranja članstva i komunikacija te do kakvih se problema može doći.

Prednost je aplikacije što nema preširok raspon funkcionalnosti pa nije kao u promatranim aplikacijama dosta često teško pronalaženje onoga što korisniku treba. Većina je razmotrenih aplikacija vizualno lijepo izrađena i bogata funkcionalnostima. Imaju mogućnost povezivanja s drugim aplikacijama.

Za razvoj Web aplikacije za evidenciju i upravljanje članovima korišten je Django, Bootstrap te za uvođenje drugačijih boja i tekstova CSS. Korištena je Djangova SQLite 3 baza podataka za koju su definirani odgovarajući modeli i veze među njima. Nakon što se korisnih registrira odlazi na menadžment stranicu iz koje može pristupiti svim ostalim dijelovima aplikacije. Omogućeno mu je stvaranje, uređivanje te brisanje članova i grupa. Filtriranje i sortiranje članova. U svakoj grupi, za članove te grupe, može označiti jesu li platili članarinu, a u tablici s članovima koji nisu u toj grupi može ih jednostavnim klikom dodati u grupu u kojoj se nalazi.

Kako bi se aplikacija unaprijedila korisno bi bilo omogućiti filtriranje po članovima koji nisu platili članarinu. Druga značajka koja bi pridonijela boljoj upotrebljivosti aplikacije je slanje maila članovima izravno iz aplikacije. Omogućiti dodatnu stranicu za bilježenje napretka svakog člana i potrepština za klub kao što je oprema što nudi veliki broj aplikacija za vođenje teretana.

LITERATURA

[1] Monday.com, dostupno na: http://www.monday.com. [pristupano svibanj 2023.]

[2] LoveAdmin, dostupno na:: https://loveadmin.com/. [pristupano lipanj 2023.]

[3] Asana, dostupno na: https://asana.com/. [pristupano lipanj 2023.]

[4] GymMaster, dostupno na: https://www.gymmaster.com/. [pristupano svibanj 2023.]

[5] EMS: Employee Management, dostupno na: https://play.google.com/store/apps/details?id=com.technoapps.employeeattendance. [pristupano lipanj 2023.]

[6] H. enciklopedija, Leksikografski zavod Miroslav Krleža, 2021., dostupno na: https://www.enciklopedija.hr/Natuknica.aspx?ID=40070. [pristupano 26. svibanj 2023.]

[7] Google fonts, dostupno na: https://fonts.google.com/ . [pristupano svibanj 2023.]

[8] Canva, dostupno na: https://www.canva.com/ . [pristupano svibanj 2023.]

[9] Django documentation, dostupno na: https://docs.djangoproject.com/en/4.2/. [pristupano srpanj 2023.]

[10] Bootstrap 5.3, dostupno na: https://getbootstrap.com/docs/5.3/components/modal/. [pristupano lipanj 2023.]

SAŽETAK

U ovome je radu izrađena aplikacija za vođenje evidencije članova neke udruge ili kluba. Aplikacija omogućuje, svakom registriranom korisniku, pregledno mjesto za sve svoje članove i jednostavnu evidenciju njihovog plaćanja članarine u pojedinoj grupi. Na početku rada pružen je uvid u aplikacije sa sličnim rješenjima koje su dostupne na internetu. Dan je uvid u problematiku vođenja evidencije članstva. Potom su objašnjene korištene tehnologije uz opis programskog dijela aplikacije. Objašnjeni su modeli baze podataka. Definiran je i objašnjen CRUD pristup te povezan s implementiranim funkcionalnostima aplikacije. Objašnjena je MVC paradigma koja se u Djangu naziva MTV. Na kraju je opisano korištenje aplikacije na temelju sučelja koje korisnik vidi. Izrađen je dijagram koji prikazuje sve mogućnosti prijelaza iz svih dijelova aplikacije. Pri razvoju aplikacije korišteni su Django, Bootstrap te CSS.

Ključne riječi: Bootstrap, CRUD, Django, evidencija članova, web aplikacija

ABSTRACT

Web Application for Membership Management

In this final paper, an application was created for managing members of an association or club. The application provides every registered user with an overview of all their members and a simple record of their payment of membership fees in a particular group. At the beginning of this paper, it is provided an insight into applications with similar solutions available on the Internet. Second, it was given an insight into the issue of keeping membership records. Further, the used technologies were explained along with a description of the program part of the application. The database models are described. The CRUD approach is defined and connected to the implemented functionalities of the application. The MVC paradigm, called MTV in Django, is analyzed. Finally, the use of the application is described based on the interface that the user sees. A created diagram shows all transition possibilities from all parts of the application. Django, Bootstrap and CSS were used in the development of the application.

Keywords: Bootstrap, CRUD, Django, member record, web application