

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

ZRINKA KOMIĆ

Z A V R Š N I R A D

IZRADA JAZZ & BLUES WEB APLIKACIJE

Split, rujan 2022.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

Predmet: Informacijski sustavi

Z A V R Š N I R A D

Kandidat: Zrinka Komić

Naslov rada: Izrada jazz & blues web aplikacije

Mentor: mr. sc. Karmen Klarin, viši predavač

Split, rujan 2022.

Sadržaj

Sažetak.....	1
Summary.....	1
1. Uvod.....	2
2. Tehnologije.....	3
2.1. Razvojni okvir Django.....	3
2.2. PostgreSQL.....	4
2.3. HTML, CSS i Javascript.....	4
3. Modeli i baza podataka.....	5
4. Opis praktičnog rada aplikacije.....	9
4.1. Registracija.....	9
4.2. Prijava.....	11
4.3. Navigacija i početna stranica.....	12
5. Funkcionalnosti aplikacije za goste.....	14
5.1. Pregled albuma i <i>eventova</i>	14
5.2. Košarica.....	19
5.3. Kupovina.....	21
5.4. Odabir sjedećih mjesta za koncerte.....	22
5.5. Pregled narudžbi albuma i karata.....	23
5.5.1. Pregled kupljenih albuma.....	24
5.5.2. Pregled i printanje kupljenih karata.....	25
6. Funkcionalnosti aplikacije za ulogu „ <i>staff</i> “.....	26
6.1. Upravljanje albumima i <i>eventovima</i>	26
6.1.1. Unos novog albuma/koncerta/festivala.....	27

6.1.2. Uređivanje postojećeg albuma/koncerta/festivala.....	30
6.1.3. Brisanje albuma/koncerta/festivala.....	32
6.2. Upravljanje narudžbama.....	33
7. Zaključak.....	36
Literatura.....	37

Sažetak

Jazz & blues web aplikacija izrađena je kako bi ljubiteljima tog tipa glazbe olakšala proces kupovine CD albuma, gramofonskih ploča te karata za festivale i koncerte. Cilj aplikacije je bio pojednostaviti pronalazak takvog sadržaja jer je najmanje zastupljen na sličnim stranicama.

Za izradu aplikacije korišteno je razvojno okruženje Django zasnovano na programskom jeziku Python. Aplikacija ima dijelove koji su dostupni svim korisnicima, postoje funkcionalnosti koje su omogućene isključivo registriranim korisnicima, a zaposlenici (engl. *staff*) imaju mogućnosti uređivanja sadržaja stranice, dodavanja artikala u ponudu i uklanjanja istih.

Ključne riječi: Django, Python, jazz, blues

Summary

Building a jazz & blues web application

The jazz & blues web application was created to help ease the process of buying CD albums, gramophone records and tickets for festivals and concerts for music lovers of that genre. The goal of the application was to simplify the search for such content because it is the least represented on similar web sites.

The Django development environment which is based on the Python programming language was used to create the application. The application has parts which are available to all users, there are functionalities that are only available to registered users, and staff have the ability to edit the page content, add new items to the webshop and to remove them.

Keywords: Django, Python, jazz, blues

1. Uvod

Tema ovog završnog rada je izrada web aplikacije za kupovinu jazz & blues albuma i karata za događaje – festivale i koncerte. Zbog lakše pretrage, korisnicima je omogućeno albume i događaje u aplikaciji filtrirati po njihovim atributima, pretraživati po imenu te sortirati na razne načine. Kupnju mogu realizirati samo registrirani korisnici, a neregistrirani korisnici, tj. gosti mogu pregledavati sadržaj i ponudu stranice bez mogućnosti dodavanja artikala u košaricu dok ne izvrše registraciju i prijavu u aplikaciju. Korisnici s ulogom *staff* imaju posebne mogućnosti uređivanja ponude unutar aplikacije – imaju pristup stranicama za uređivanje postojećih artikala, dodavanje novih artikala i brisanje artikala iz ponude.

Korisnik prilikom pregleda ponude može dodati u košaricu artikle koji mu se sviđaju, a u pregledu košarice te iste proizvode može uklanjati ili uređivati količinu koju želi kupiti. Kada korisnik uredi artikle koje je odlučio naručiti, prosljeđuje ga se na zadnji korak gdje upisuje podatke za dostavu i plaćanje i tako završava narudžbu. Prije kupnje karte za koncert, korisnik odabire mjesto na kojem želi sjediti koje mu se ispisuje na karti koju može preuzeti u .pdf formatu kako bi je mogao isprintati.

Ovaj rad podijeljen je u nekoliko dijelova. U prvom dijelu rada ukratko su opisane tehnologije koje su bile neophodne za izradu aplikacije. U drugom dijelu rada opisana je struktura projekta i ukratko su opisani formirani modeli. U trećem dijelu nalazi se opis praktičnog rada aplikacije koji je jednak za sve korisnike, nakon kojeg slijede odvojena poglavlja za funkcionalnosti rada koje su različite za registriranog kupca i korisnika zaposlenika (s ulogom *staff*).

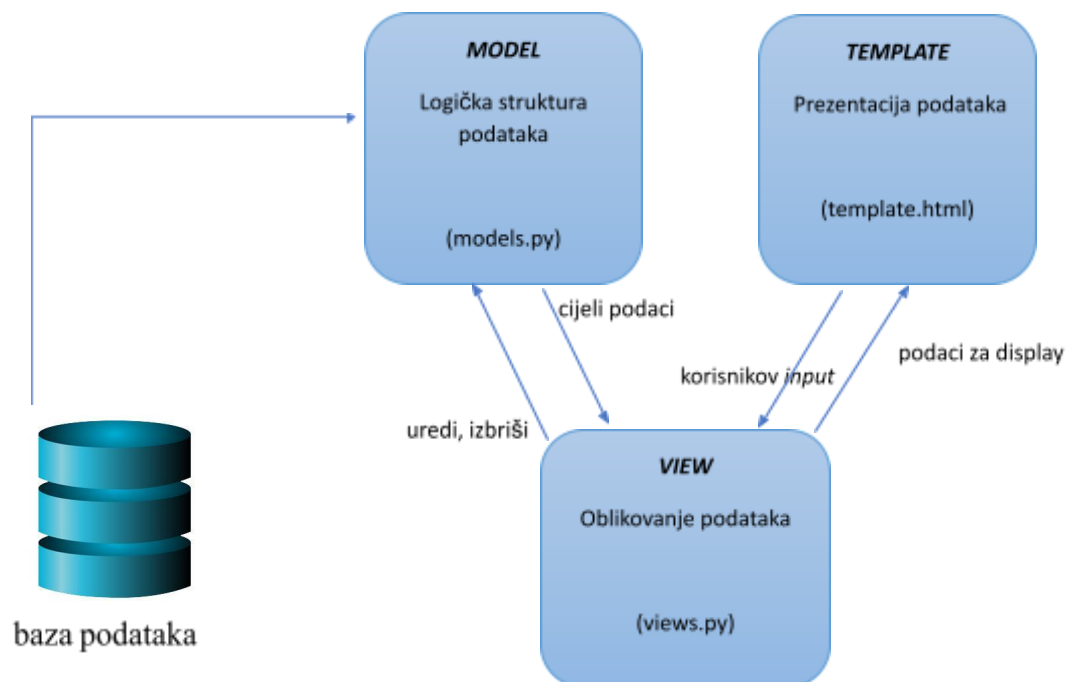
2. Tehnologije

2.1. Razvojni okvir Django

Razvojni okvir Django [1], u daljnjem tekstu samo Django, je Python razvojni okvir visoke razine koji omogućava brz razvoj sigurnih i održivih web stranica.

Njegov razvoj počeo je 2003. godine, a 2005. godine je okruženje objavljeno kao otvoreni server (engl. *open source*) te je nastavio rasti i razvijati se sve do danas. Za implementiranje korisničkog sučelja koristi se Model-Template-View (MTV) arhitektura slična Model-Viewer-Controller (MVC) arhitekturi kako bi se odvojio originalni prikaz informacija od načina kako su te informacije prikazane korisniku.

U MVC arhitekturi modeli se koriste za definiranje tablica baze podataka i odnose među tablicama. Model je povezan s bazom podataka i pogledom (engl. *view*) kao što je vidljivo na slici 1. Od baze podataka dohvaća tražene podatke i prosljeđuje ih pogledu, ali nema uvid u funkcije koje se izvode u njemu. Pogled određuje koji će podaci dohvaćeni iz modela biti prikazani te koji će prikazani podaci biti uređeni i spremljeni natrag u bazu u novom obliku. Predložak (engl. *template*) je najčešće HTML stranica koja omogućava prikaz podataka poslanih iz pogleda te dopušta korisniku unos koji se prosljeđuje u pogled.



Slika 1. Interakcija baze podataka, modela, pogleda i predloška

2.2. PostgreSQL

PostgreSQL [2] je besplatna objektno-relacijska baza podataka s temeljem u klijent-poslužitelj modelu. Njeno porijeklo seže u 1986. kao dio projekta POSTGRES na Kalifornijskom sveučilištu i već se preko 30 godina aktivno razvija, ali je otvorenog kôda što znači da je dobrovoljno razvijana i održavana od strane programera. S obzirom na snažnu reputaciju zbog odlične arhitekture, pouzdanosti i mogućnosti proširivanja te predanosti zajednice koja i dalje radi na novim i boljim rješenjima, nije iznenađenje da je PostgreSQL među prvim izborima za korištenje.

2.3. HTML, CSS i Javascript

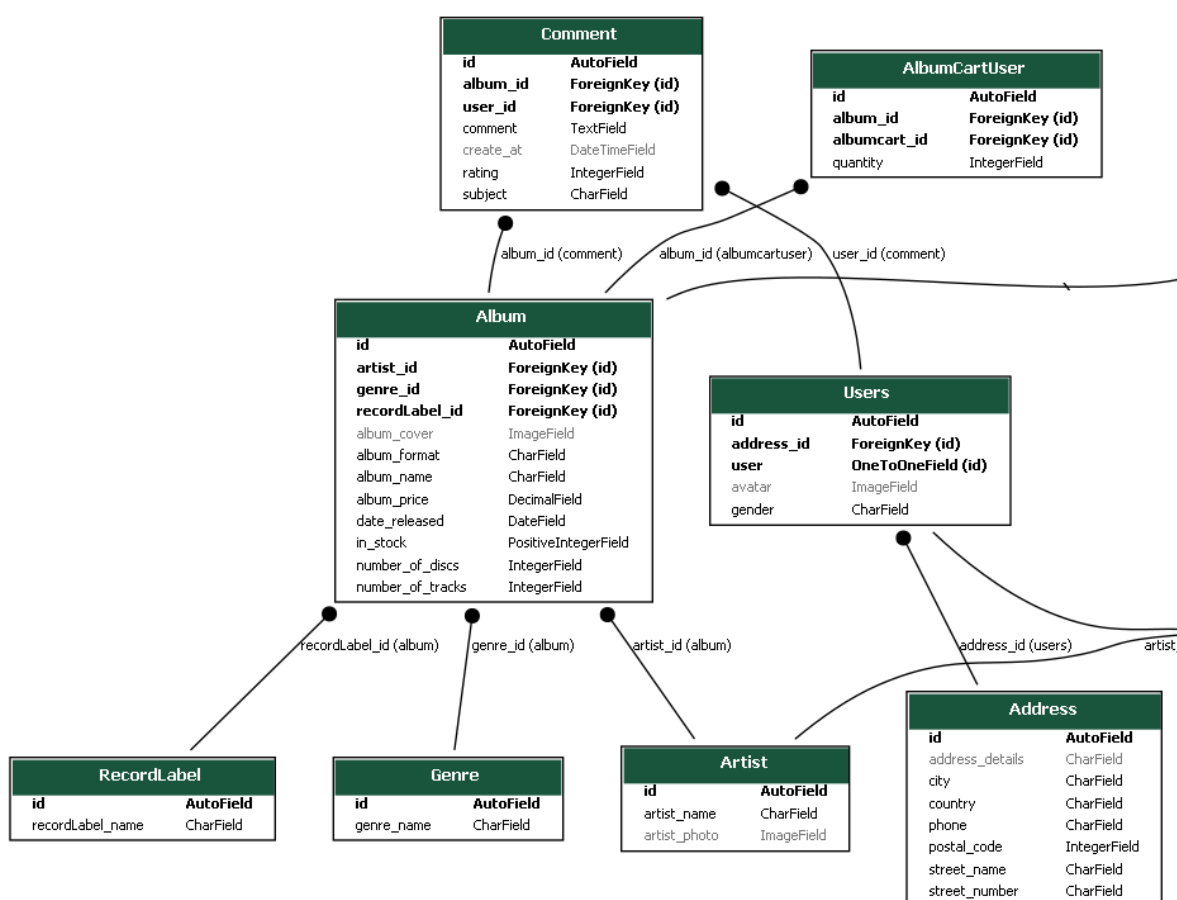
HTML ili HyperText Markup Language je prezentacijski jezik za izradu web stranica [3]. Glavni zadatak HTML-a je uputiti preglednik kako prikazati hipertekstualni dokument, a to se postiže *tagovima* – osnovnim elementima HTML jezika. Preglednik (engl. *browser*) čita sadržaj i strukturu datoteke i na taj način oblikuje stranicu. HTML je jednostavan za uporabu i lako se uči, što ga čini popularnim i široko prihvaćenim.

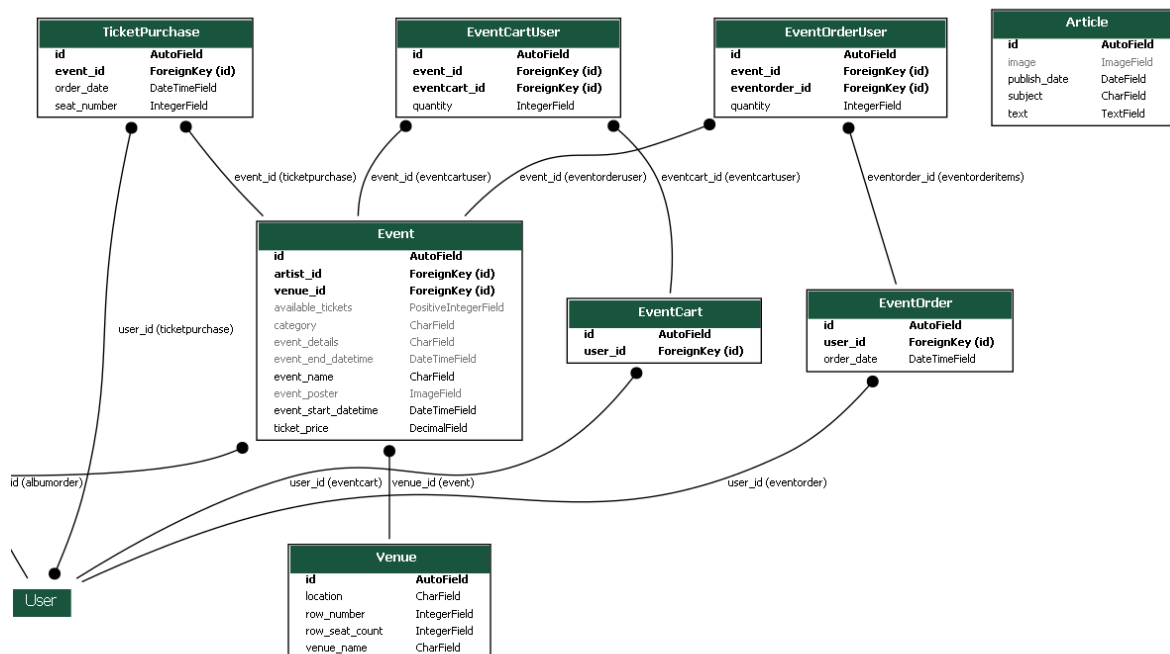
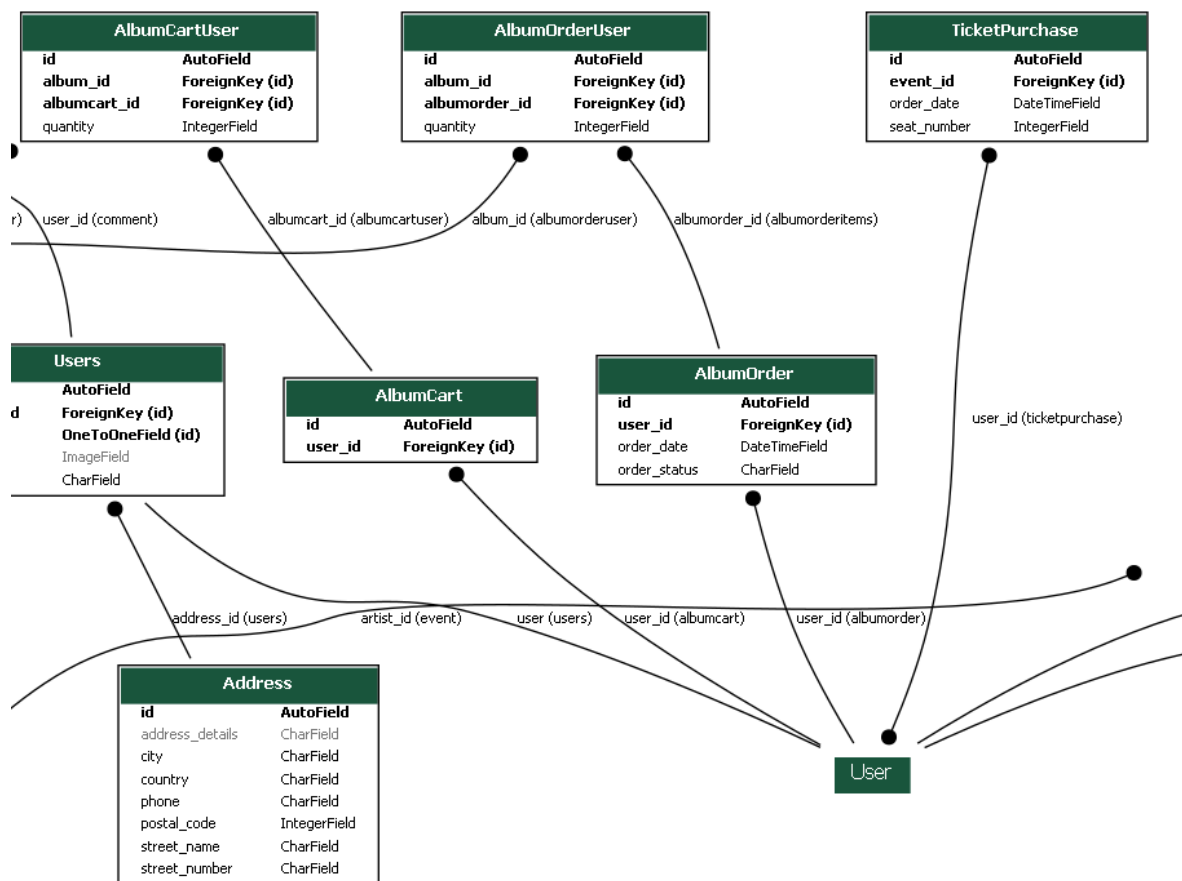
W3C (World Wide Web Consortium) 1996. godine prvi put spominje ideju stilskih obrazaca za oblikovanje HTML dokumenta – CSS (Cascading Style Sheets). CSS je stilski jezik kojim se opisuje izgled i točno definira kako prikazati HTML dokument i njegove elemente, što je omogućilo web dizajnerima odvajanje strukture i oblika dokumenata. Separacija sadržaja dokumenta i njegovog dizajna je omogućila ne samo jednako formatiranje više dokumenata, već i prikaz istog HTML dokumenta pomoću različitih stilova za različite potrebe (desktop računalo, mobilni uređaj, itd.) što je u današnje vrijeme jako korisno.

S vremenom se ukazala potreba za stvaranjem interaktivnog sadržaja u HTML-u, te je 1995. godine došlo do razvoja *JavaScript*-a. JavaScript je skriptni jezik, što bi značilo da se kôd izvršava izravno bez prevođenja u binarni kôd. Namijenjen je prvenstveno razvoju interaktivnih web stranica, a podržavaju ga gotovo svi današnji preglednici. Omogućava izvršavanje radnji u inače statičnim HTML dokumentima, kao što su interakcija s korisnikom, dinamično stvaranje HTML elemenata itd.

3. Modeli i baza podataka

Model baze podataka prikazan je E-V dijagramom na slici 2. Model Albums povezan je s modelima izvođača (engl. *artist*), žanra (engl. *genre*), izdavačke kuće (engl. *recordlabel*), komentara (engl. *comments*) i narudžbi (engl. *albumorderuser*), slično kao i model Events koji je također povezan s modelom izvođača, lokacijom (engl. *venue*) i kupljenim kartama (engl. *ticketpurchase*).





Nakon stvaranja aplikacije, prvi korak u razvoju iste je stvaranje modela. Django modeli se definiraju kao klase unutar kojih se nalaze atributi, tj. polja. Svako polje mora imati definiran tip podatka, a neki od njih traže i dodatne argumente, na osnovu kojih Django stvara tablice u bazi podataka bez dodanog komuniciranja s bazom podataka putem SQL-a. Modeli se nalaze u skripti *models.py* koja se definira posebno za svaku aplikaciju, njene klase odgovaraju tablicama u bazi podataka, a varijable odgovaraju stupcima.

Za klasu korisnika korišten je Djangov model User koji je preuzet iz ugrađenog `django.contrib.auth` modula i predstavlja osnovu ugrađenog Django autentikacijskog sustava. Model User nije potrebno implementirati, on dolazi sa nekoliko predefiniranih varijabli:

- korisničko ime (engl. *username*)
- lozinka (engl. *password*)
- e-mail adresa korisnika (engl. *email*)
- ime korisnika (engl. *first_name*)
- prezime korisnika (engl. *last_name*)

S obzirom da su za ovaj projekt bili potrebni dodatni atributi osim onih u modelu User, stvoren je dodatni model Users u kojem su definirana polja za spol korisnika, adresu korisnika i profilnu sliku korisnika, kao što je vidljivo na slici 3. S obzirom na vezu „*jedan naprema jedan*“ (engl. *OneToOne*) između `django.contrib.auth` predefinirane klase i Users klase, svaki dodatno definirani atribut je zapravo dodijeljen modelu User.

```

male = 'male'
female = 'female'
rather_not_say = 'rather_not_say'

GENDER_CHOICES = (
    (male, 'Male'),
    (female, 'Female'),
    (rather_not_say, 'Rather Not Say')
)

class Users (models.Model):
    user = models.OneToOneField(User, on_delete=models.CASCADE, null=True)
    gender = models.CharField(max_length=20, choices=GENDER_CHOICES,
                              default=rather_not_say)
    address_id = models.ForeignKey(Address, on_delete=models.DO_NOTHING,
                                    null=True, blank=True)
    avatar = models.ImageField(upload_to='users/', null=True, blank=True)

    def __str__(self):
        return self.user.username

```

Slika 3. Dodatni model Users i OneToOne veza s Django modelom User

Polja za adresu korisnika i profilnu sliku (engl. *avatar*) nisu obavezna, što je određeno naredbom `null=True` i `blank=True`. Atribut za spol korisnika je polje u koje će se upisati neka od vrijednosti „male“, „female“ ili „rather_not_say“ koju korisnik odabere u padajućem izborniku prilikom registracije. Sama forma i funkcija za registraciju korisnika će biti pobliže opisana u sljedećem poglavlju.

4. Opis praktičnog rada aplikacije

4.1. Registracija

Kao što je opisano u prošlom poglavlju, kako bi se korisnik uspješno registrirao, mora popuniti polja – korisničko ime, e-mail adresu, lozinku, ponovljenu lozinku, a polja spola korisnika i profilne slike su opcionalna. U datoteci *forms.py* aplikacije „accounts“ definirane su dvije forme – *registrationForm* i *addFields* vidljive na slici 4. Forma *registrationForm* je tipa *UserCreationForm* koja podatke dohvaća iz Django modela *User* koji je korišten za registraciju i prijavu korisnika, dok je forma *addFields* tipa *ModelForm* koja dohvaća podatke iz baze podataka kreirane od strane korisnika.

```
class registrationForm(UserCreationForm):
    class Meta:
        model = User
        fields = ("first_name", "last_name", "username", "email", "password1",
                  "password2")

class addFields(forms.ModelForm):
    gender = forms.ChoiceField(choices=GENDER_CHOICES)
    avatar = forms.ImageField(required=False)
    class Meta:
        model = Users
        fields = ("gender", "avatar",)
```

Slika 4. Forme aplikacije *accounts*

U datoteci *views.py* aplikacije *accounts* nalazi se funkcija *register* vidljiva na slici 5 koja vrši registraciju novog korisnika. Funkcija šalje obje forme na stranicu za registraciju korisnika, a ako se radi o *POST* request-u, dohvaća podatke iz obje forme i vrši validaciju podataka. Ako je forma validna, stvara novog korisnika i sprema sve upisane podatke o korisniku u bazu podataka. Slika 6 prikazuje kako forma za registraciju izgleda u aplikaciji.

```

def register(request):
    if request.method == "POST":
        u_form = registrationForm(request.POST)
        p_form = addFields(request.POST, request.FILES)
        if u_form.is_valid() and p_form.is_valid():
            user = u_form.save()#dohvacanje podataka iz UserCreationForm-a
            details = p_form.save(commit=False)#dohvacanje podataka iz forme sa
gender i slikom
            details.user=user#postavljanje usera iz UserCreationForm u korisnika
            details.save()
            return redirect ('accounts:login')
        else:
            u_form = registrationForm()
            p_form = addFields()
    return render(request, 'register.html', {'u_form':u_form, 'p_form':p_form})

```

Slika 5. Funkcija za registraciju korisnika

New Customer

All fields marked with a * are mandatory.

USERNAME *	<input type="text"/>
EMAIL ADDRESS *	<input type="text"/>
PASSWORD * (6 CHARACTERS OR LONGER)	<input type="password"/>
RETYPE PASSWORD *	<input type="password"/>
GENDER	Male <input type="button" value="v"/>
FIRST NAME *	<input type="text"/>
LAST NAME *	<input type="text"/>
AVATAR	<input type="button" value="Choose File"/> <input type="button" value="No file chosen"/>

Register

Slika 6. Forma za registraciju novog korisnika u aplikaciji

4.2. Prijava

Nakon uspješne registracije, korisnik se mora prijaviti kako bi mu se omogućile funkcije aplikacije koje su rezervirane samo za registrirane i prijavljene korisnike. Korisnik se prijavljuje e-mail adresom i lozinkom koje je upisao prilikom registracije (slika 8). Za autentikaciju korisnika korištena je `AuthenticationForm` iz paketa `django.contrib.auth.forms`. Dohvaćaju se podaci koje je korisnik upisao u formu za prijavu i ako validacija bude uspješna, korisnika se prijavljuje u aplikaciju funkcijom `login` uvezenom iz `django.contrib.auth`. Funkcija za prijavu prikazana je na slici 7.

```
def login_fun(request):
    prev_url = request.META.get('HTTP_REFERER')
    if request.method == 'POST':
        form = AuthenticationForm(data=request.POST)#validacija podataka
        if form.is_valid():
            next_url = request.POST.get('next_url')
            #log in the user
            user = form.get_user()#dohvacanje upisanih podataka (usera)
            login(request, user)#logiranje usera
            return redirect (next_url)
        else:
            messages.warning(request, "Login error!")
            context = {
                'prev_url': prev_url,
                'form': AuthenticationForm(),
            }
    else:
        context = {
            'prev_url': prev_url,
            'form': AuthenticationForm(),
        }
    return render(request, 'login.html', context)
```

Slika 7. Funkcija za prijavu korisnika

Existing Customer

Username

Password

Login

[Forgot your password?](#)

New Customer

Click to provide us with your details

Register

Slika 8. Forma za prijavu korisnika na stranici

Iz istog paketa `django.contrib.auth` uvezena je i funkcija `logout` (slika 9.) koju koristimo za odjavu korisnika. Nakon odjave, korisnika preusmjeravamo na početnu stranicu.

```
@login_required
def logout_fun(request):
    logout(request)
    return redirect('base')
```

Slika 9. Funkcija za odjavu korisnika sa stranice

4.3. Navigacija i početna stranica

Navigacija stranice se za sve korisnike sastoji se od poveznica koje vode na početnu stranicu, stranicu s albumima, stranicu s eventovima te od padajućeg izbornika. Ukoliko korisnik aplikacije nije prijavljen, u padajućem izborniku nalaze se linkovi za prijavu i registraciju (slika 10).



ALBUMS EVENTS LOGIN OR REGISTER

Slika 10. Navigacija za gosta

Ako prijavljeni korisnik ima ulogu zaposlenika (engl. *staff*), u padajućem izborniku nalaze se poveznice za stranicu za upravljanje narudžbama te stranicu za upravljanje korisnicima (slika 11).



Slika 11. Navigacija za ulogu „staff“

Ukoliko prijavljeni korisnik nema takvu ulogu, u padajućem izborniku pronaći će poveznicu za pristup profilu, poveznicu za pregled svih svojih narudžbi te poveznicu za odjavu sa stranice (slika 12). Nakon padajućeg izbornika, prijavljeni korisnik ima i ikonu “košarice” koja vodi na poveznicu na kojoj može pregledati proizvode koje je označio za kupnju te ukupni iznos narudžbe.

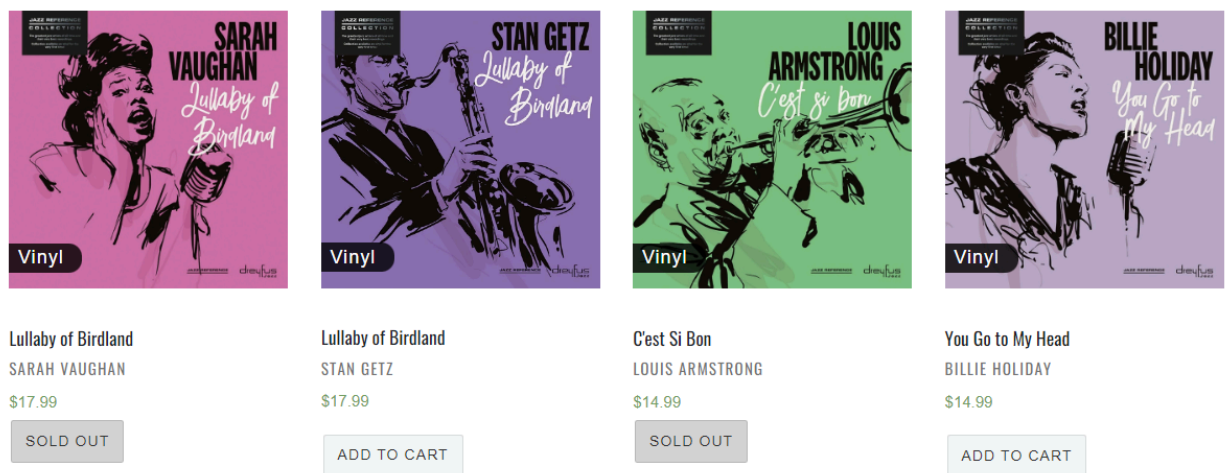


Slika 12. Navigacija za prijavljenog korisnika

5. Funkcionalnosti aplikacije za goste

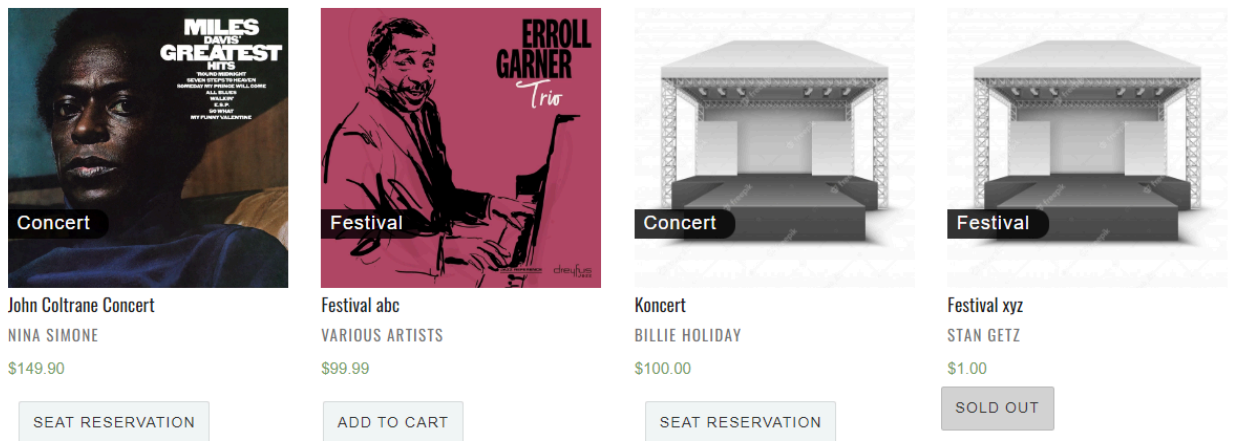
5.1. Pregled albuma i eventova

Na stranici s albumima korisnik vidi prikaz svih albuma – onih koji su u prodaji kao i onih kojih trenutno nema u zalihama kao što je prikazano na slici 13. Ako albuma nema u zalihama, onemogućeno je dodavanje istog u košaricu, ali je klikom na fotografiju albuma moguć pristup stranici s detaljima o albumu. Korisnik može sortirati albume uzlazno i silazno po cijeni te abecednom poretku. Korisnik može i filtrirati albume po izvođaču, formatu albuma, cijeni i žanru te može pretraživati albume po imenu.



Slika 13. Albumi u zalihama i van zalihe

Na stranici s *eventovima* vidljiva su sva događanja s budućim datumom – koncerti i festivali (slika 14). Ako postoji još slobodnih karata, korisnik karte za festivale može odmah dodati u košaricu, a za koncerte korisnik prvo treba odabrati sjedalo na kojem želi sjediti. Sami proces odabira sjedala će biti detaljnije opisan u zasebnom poglavlju. Koncerte je također moguće sortirati uzlazno i silazno po cijeni te abecednom poretku, kao i filtrirati po izvođaču i kategoriji (koncert ili festival) te pretražiti po imenu.



Slika 14. Koncerti i festivali

Kao što je već spomenuto, korisnik može filtrirati albume i događaje. U skripti *filters.py* aplikacije JazzBluesApp definirani su filteri koji su korišteni za filtriranje podataka modela Albums i Events. Uzmimo za primjer model Album – korisnik može filtrirati podatke po izvođaču, imenu, cijeni, formatu albuma i žanru albuma, kao što je vidljivo na slici 15. Ime albuma (engl. *album_name*) je označeno kao *CharFilter*, tako da se u bazi traži ime albuma koje sadrži upisani string. Za cijenu albuma je određen *NumberFilter*, ali se korisniku prikazuju dva polja za upis cijene – polje za minimalnu (engl. *album_price__gt*) i maksimalnu (engl. *album_price__lt*) cijenu, što bi značilo da će se u bazi podataka tražiti album čija je cijena veća od minimalne cijene ili manja od maksimalne cijene. Polja nisu obavezna, dakle ako korisnik upiše samo minimalnu cijenu, ispisat će mu se svi albumi prodajne cijene veće od upisane. Ista logika vrijedi i za upisanu samo maksimalnu cijenu. Format, žanr i izvođača albuma korisnik može odabrati iz padajućih izbornika, a moguće je odabrati i više od jednog izvođača u istom filtriranju.

```

class AlbumFilter(django_filters.FilterSet):
    album_name = django_filters.CharFilter(field_name='album_name',
                                          lookup_expr='icontains')
    album_price__gt = django_filters.NumberFilter(field_name='album_price',
                                                  lookup_expr='gt')
    album_price__lt = django_filters.NumberFilter(field_name='album_price',
                                                  lookup_expr='lt')

    album_format = django_filters.ChoiceFilter(
        choices = ALBUM_FORMAT_CHOICES,
        empty_label = "Select album format ... "
    )

    genre_id = django_filters.ModelChoiceFilter(
        queryset = Genre.objects.all(),
        widget = (),
        label = "Genre",
        empty_label = "Select genre ... ",
    )

    artist_id = django_filters.ModelMultipleChoiceFilter(
        queryset = Artist.objects.all(),
        widget = forms.SelectMultiple(attrs=
            {'id': 'artistMultiple', 'class': 'artistMultiple'}),
        label = "Artist",
    )
    class Meta:
        model = Album
        fields= ['artist_id', 'genre_id', 'album_name', 'album_format',
                'album_price']

```

Slika 15. Filteri za model Album

Pritiskom na sliku ili naziv nekog od izlistanih albuma, korisnik je preusmjeren na stranicu s detaljima o odabranom albumu. Osim samih detalja o albumu, korisnik može vidjeti ocjenu albuma koja je izračunata kao prosjek ocjena koje su upisane od strane registriranih korisnika. Kao i na stranici prikaza svih albuma, ako odabranog albuma nema u zalihama, onemogućeno je dodavanje istog u košaricu (slika 16).



Saxophone Colossus

SONNY ROLLINS

★ ★ ☆ ☆ ☆ (5 customer reviews)

\$14.99

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

SOLD OUT

Categories: Clothing, Hoodies

DESCRIPTION

REVIEWS (5)

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

Slika 16. Stranica s detaljima o albumu

Unutar *tab-a* „*description*“ vidljivi su detalji o albumu, a pritiskom na „*reviews*“, korisnik može vidjeti sve recenzije odabranog albuma koje su upisali registrirani korisnici (slika 17).

DESCRIPTION

REVIEWS (5)

5 reviews for Saxophone Colossus



skomic – Aug. 19, 2021, 8:41 a.m.

★ ★ ★ ☆ ☆

Komentar
Komentar



skomic – Aug. 19, 2021, 8:41 a.m.

★ ☆ ☆ ☆ ☆

Bla bla
Bla



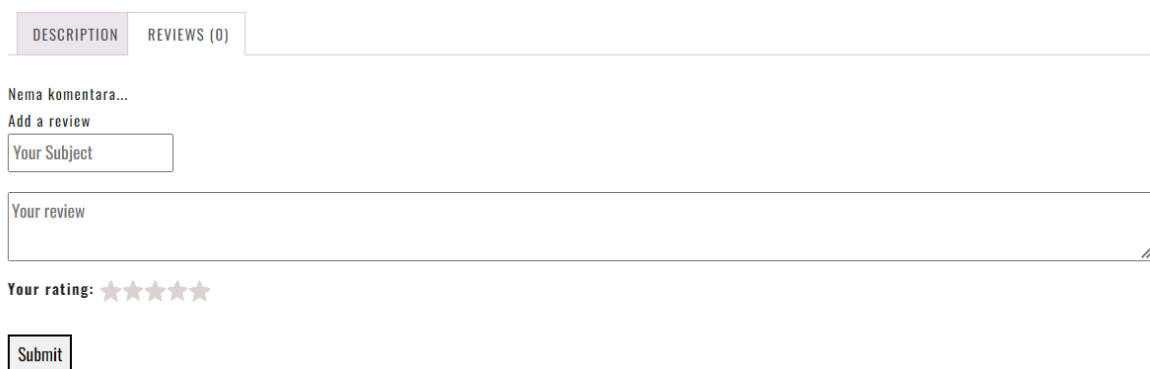
skomic – July 16, 2022, 10:13 a.m.

★ ☆ ☆ ☆ ☆

test review
test test

Slika 17. Recenzije odabranog albuma

Korisnik može i sam ocijeniti album ako je registriran i prijavljen na stranicu. Ako nije prijavljen u tom trenutku, forma za upis recenzije albuma se ne prikazuje, već je ispisana napomena da se treba prijaviti ukoliko želi ostaviti recenziju. Ako je korisnik prijavljen, prikazuje mu se forma vidljiva na slici 18, koja sadrži polje za naslov recenzije u kojem bi korisnik trebao u nekoliko riječi opisati album, polje za recenziju u kojoj bi mogao iznijeti svoje osjećaje i razmišljanja o albumu detaljnije, te pet zvjezdica kojima može album ocijeniti ocjenama od 1 do 5.

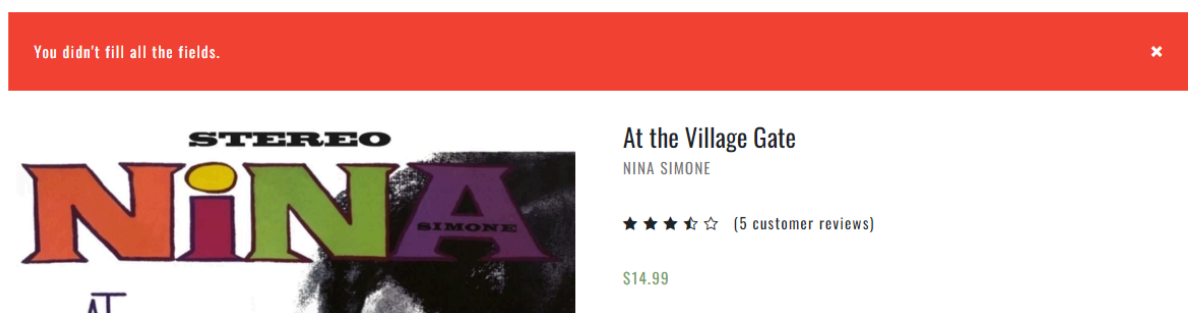


The screenshot shows a web interface for adding a review. At the top, there are two tabs: 'DESCRIPTION' and 'REVIEWS (0)'. Below the tabs, it says 'Nema komentara...' and 'Add a review'. There is a text input field labeled 'Your Subject'. Below that is a larger text area labeled 'Your review'. At the bottom, there is a rating section labeled 'Your rating:' followed by five stars. A 'Submit' button is located at the bottom left.

Slika 18. Forma za upis recenzije prijavljenog korisnika

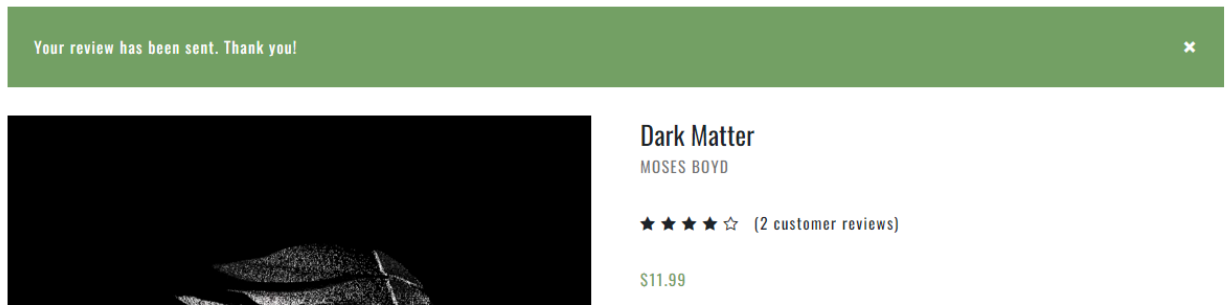
Ako korisnik ne popuni sva obavezna polja prilikom upisa recenzije albuma, ispisuje se upozorenje crvene boje vidljivo na slici 19 koje naglašava da nisu popunjena sva polja forme, a recenzija nije spremljena. Kada korisnik pravilno upiše sva polja, tekstualna recenzija i ocjena se spremaju u bazu podataka, što je korisniku vidljivo na način prikazan na slici 20. Korisnikova recenzija je nadalje vidljiva svim korisnicima koji budu pregledavali album te ulazi u računanje prosječne ocjene albuma.

Home / Albums / At the Village Gate



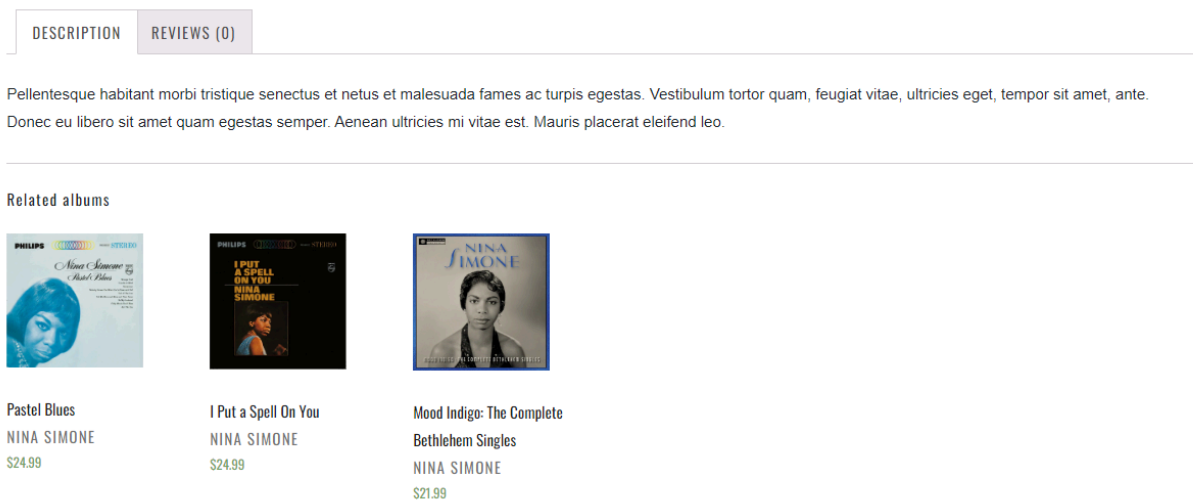
The screenshot shows the album page for 'At the Village Gate' by Nina Simone. At the top, there is a red error message bar that says 'You didn't fill all the fields.' with a close button. Below the error bar, there is a large image of the album cover. To the right of the image, the album title 'At the Village Gate' and the artist 'NINA SIMONE' are displayed. Below the title, there is a rating of 4 stars out of 5, with the text '(5 customer reviews)'. At the bottom right, the price '\$14.99' is shown.

Slika 19. Upozorenje kad nisu popunjena sva polja pri upisu recenzije



Slika 20. Obavijest da je recenzija uspješno spremljena

Na dnu stranice detalja, odmah ispod polja u kojem je prikazan opis albuma i recenzije, prikazani su povezani albumi (engl. *related albums*) tj. ostali albumi istog izvođača, ukoliko postoje takvi u ponudi (slika 21).



Slika 21. Ostali albumi izvođača odabranog albuma

5.2. Košarica

Košarica prikazuje sve albume i karte za festivale koje je prijavljeni korisnik prilikom pregleda ponude aplikacije dodao u nju i koje planira kupiti (slika 22). Količinu pojedinog artikla u košarici je moguće smanjiti i povećati ako ima dovoljno albuma i slobodnih festivalskih karata u zalihama.

Cart

	Django Reinhardt: Echoes of France CD Album 6 x £6.99	<div>-</div> <div>+</div>
	Sarah Vaughan: Lullaby of Birdland CD Album 3 x £6.99	<div>-</div> <div>+</div>
Total £62.91		

Checkout

Slika 22. Košarica

Ako je odabran broj albuma ili karata u košarici jednak dostupnoj količini u zalihama, a korisnik pokuša povećati broj, prikaže se obavijest da veća količina odabranog artikla nije dostupna, kao što je vidljivo na slici 23.

No more albums available!



Cart

	Django Reinhardt: Echoes of France CD Album 6 x £6.99	<div>-</div> <div>+</div>
	Sarah Vaughan: Lullaby of Birdland CD Album 3 x £6.99	<div>-</div> <div>+</div>
Total £62.91		

Checkout

Ispod liste albuma koje korisnik želi kupiti nalazi se ukupni iznos koji će korisnik morati platiti ukoliko se odluči na kupovinu svih navedenih artikala. Ako je sve u redu s narudžbom, korisnik klikom na gumb „*Checkout*“ nastavlja na stranicu za upis adrese za dostavu i plaćanje narudžbe, koja će biti pobliže objašnjena u sljedećem poglavlju.

5.3. Kupovina

Zadnja stranica narudžbe sadrži finalnu listu artikala koje je korisnik odlučio kupiti bez mogućnosti uređivanja iste (slika 24). Ispod liste artikala i ukupnog iznosa kojeg treba platiti, nalaze se podaci za slanje i plaćanje narudžbe. Ako korisnik već ima adresu za dostavu povezanu s njegovim profilom, ista se ispisuje kao adresa za dostavu (engl. *shipping address*) uz mogućnost mijenjanja postojeće adrese na drugu adresu. Ako je ovo prva narudžba prijavljenog korisnika i nema adresu na koju mu se do sada vršila dostava, korisnik mora upisati adresu na koju će mu narudžba biti dostavljena. Desno od podataka za dostavu nalaze se podaci za plaćanje (engl. *payment*) gdje korisnik mora unijeti ime i prezime s kartice, broj kartice i datum isteka kartice kako bi plaćanje bilo provedeno.

Checkout

	Django Reinhardt: Echoes of France CD Album	6 x £6.99
	Sarah Vaughan: Lullaby of Birdland CD Album	3 x £6.99
Total £62.91		

Shipping address

Fra Bonina 4B
21000 Split
Croatia
0989558527
[Edit adress](#)

Payment

Name on card:	John Smith
Card Number:	XXXX XXXX XXXX XXXX
Exp. date:	XX / XX

Place order

Slika 24. Stranica za plaćanje košarice i dovršetak narudžbe

5.4. Odabir sjedećih mjesta za koncerte

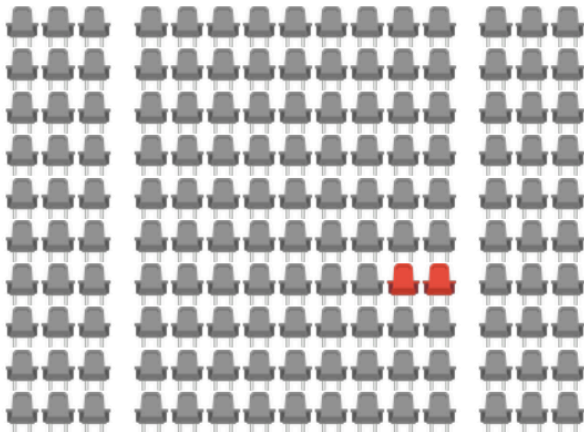
Ako je korisnik odlučio kupiti kartu za koncert, prije plaćanja karte mora odabrati sjedalo koje želi rezervirati. S obzirom na podatke koji su uneseni o lokaciji koncerta (engl. *venue*), formira se izgled dvorane s brojem sjedala koje odgovara onom upisanom prilikom unošenja novog događaja. Na prikazu dvorane siva sjedala predstavljaju slobodna sjedala, a crvena sjedala predstavljaju već zauzeta sjedala tj. prodane karte. Klikom na siva sjedala, korisnik odabire koja sjedala želi rezervirati i ona se prikazuju u plavoj boji, kao što je vidljivo na slici 25. Nakon označavanja svih sjedala koje korisnik želi rezervirati, klikom na kupovinu karte (engl. *ticket purchase*) vodi ga se na stranicu za plaćanje karata.



Slika 25. Prikaz dvorane prije i tijekom izbora sjedala

Sljedeća stranica slična je stranici za plaćanje narudžbe albuma. Na vrhu stranice korisnik vidi odabrana sjedala na prikazu dvorane, ispod toga su prikazani podaci o koncertu – ime koncerta, vrijeme održavanja koncerta i broj odabranih sjedala a na kraju se nalaze polja u koja korisnik treba upisati podatke za uplatu. Klikom na gumb za dovršavanje narudžbe (engl. *place order*) kupovina karte je završena. Sada je to mjesto označeno kao prodano i nije dostupno za daljnju kupovinu ostalim korisnicima (slika 26). Karte za festivale se prodaju kao stajaće karte, pa rezervacija mjesta za njih nije potrebna.

Ticket Purchase



Ticket information

Concert: John Coltrane Concert
Date: Sept. 3, 2022, 2:47 p.m.
Seat: 101, 102,

Payment

Name on card: John Smith

Card Number: xxxx xxxx xxxx xxxx

Exp. date: xx / xx

Place order

Slika 26. Prikaz stranice za kupovinu i plaćanje karte za koncert

5.5. Pregled narudžbi albuma i karata

Korisnik na stranici „My orders“ može vidjeti listu svih svojih dosadašnjih narudžbi albuma i kupljenih karata za festivale i koncerte. Za sve narudžbe albuma, ispisan je ID broj narudžbe, status narudžbe, datum i vrijeme kada je narudžba napravljena te ukupni iznos narudžbe, kao je prikazano na slici 27. Odmah ispod narudžbi, korisnik može vidjeti sve karte za koncerte i festivale koje je kupio.

My Orders

Album orders

#132: Processing	
Order date: 29.05.2022	£32.98
#131: Processing	
Order date: 20.08.2021	£55.98
#130: Shipped	
Order date: 19.08.2021	£36.97
#128: Canceled	
Order date: 19.08.2021	£42.98

Slika 27. Prikaz svih narudžbi korisnika

5.5.1. Pregled kupljenih albuma

Klikom na broj ili status narudžbe albuma na stranici „My Orders“, korisnik je preusmjeren na stranicu s detaljima o odabranoj narudžbi gdje može vidjeti točno koje albume je naručio i u kojoj količini te ukupni iznos narudžbe kao što je prikazano na slici 28.

Order #130

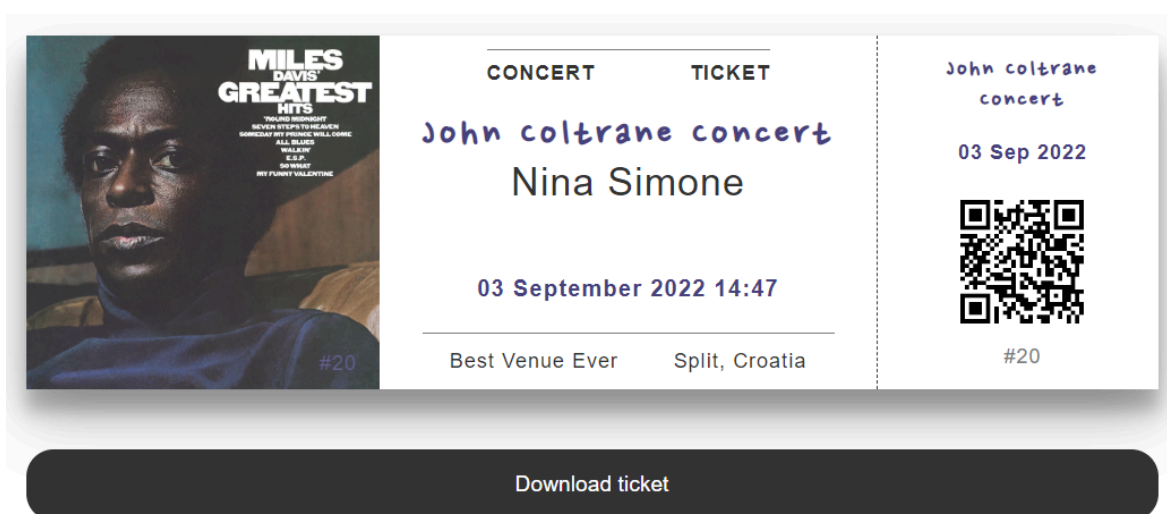
	Ella Fitzgerald: Love for Sale Release Date 18 February 2022 2 x £14.99	£29.98
	Stan Getz: Lullaby of Birdland Release Date 18 February 2022 1 x £6.99	£6.99

Total £36.97

Slika 28. Detalji narudžbe korisnika

5.5.2. Pregled i printanje kupljenih karata

Klikom na neki od navedenih događaja na stranici „My orders“, korisnik je preusmjeren na stranicu s detaljima o događaju za kojeg je kupljena karta. Odmah ispod detalja o događaju, nalazi se prikaz kupljene karte za koncert ili festival. Na kartama su navedeni osnovni podaci o događaju kao što su – ime događaja i izvođača te vrijeme i lokacija održavanja događaja (slika 29). Kartu korisnik može preuzeti u .pdf obliku pritiskom na „Download Ticket“ kako bi je mogao isprintati.



Slika 29. Prikaz kupljene karte

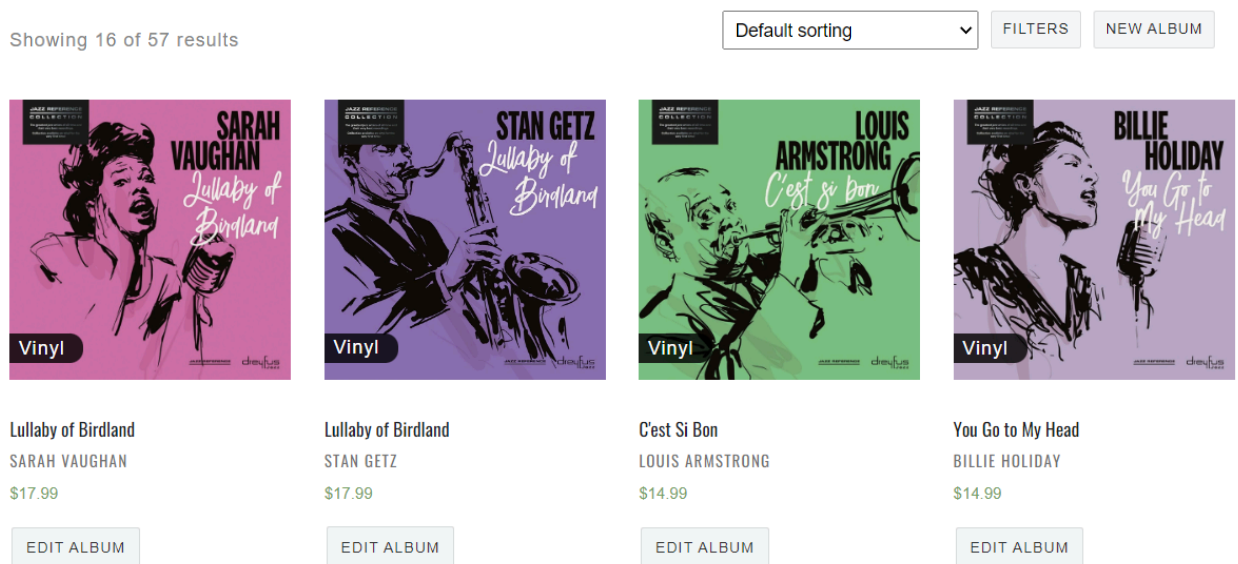
Za preuzimanje karte u .pdf formatu korištena je vanjska biblioteka HTML2canvas[6] koja uzima HTML i CSS svojstva određenog elementa stranice i prikazuje ih u istom obliku na tzv. platnu (engl. *canvas*) što je u našem slučaju .pdf dokument.

6. Funkcionalnosti aplikacije za ulogu „staff“

6.1. Upravljanje albumima i eventovima

Na stranici s albumima, uz prikaz, filtriranje i mijenjanje poretka albuma, korisnik s ulogom *staff* ima pristup dodatnim opcijama – unos novog albuma i uređivanje postojećeg albuma, kao što je vidljivo na slici 30. Ono što je takvom korisniku onemogućeno je dodavanje albuma u košaricu s obzirom da korisnik registriran kao *staff* nema opciju punjenja i pregleda košarice te kupovine artikala iz ponude web aplikacije.

Uz sve spomenute opcije koje korisnik može koristiti i na stranici s albumima, korisnik s ulogom *staff* na stranici s *eventovima* može dodati novi koncert i novi festival, te može urediti postojeće festivale i koncerte kao što je prikazano na slici 31. Dodavanje karata u košaricu mu je onemogućeno. Umjesto gumba koji bi dodavao artikl ili kartu u košaricu, gumb „Edit album“ i „Edit event“ korisnika vodi na stranicu za uređivanje albuma ili *eventa* koja će biti pobliže objašnjena u sljedećem poglavlju.



Slika 30. Prikaz albuma i dodatnih opcija za korisnika s ulogom *staff*

Showing 16 of 19 results

Default sorting

FILTERS

NEW FESTIVAL

NEW CONCERT



John Coltrane Concert

NINA SIMONE

\$149.90

EDIT EVENT

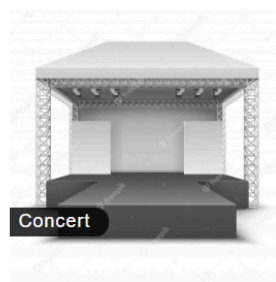


Festival abc

VARIOUS ARTISTS

\$99.99

EDIT EVENT



Koncert

BILLIE HOLIDAY

\$100.00

EDIT EVENT



Festival xyz

STAN GETZ

\$1.00

EDIT EVENT

Slika 31. Prikaz *eventova* i dodatnih polja za korisnika s ulogom *staff*

6.1.1. Unos novog albuma/koncerta/festivala

Korisnik sa ulogom *staff* ima mogućnost dodavanja novog albuma u prodaju te upis novih koncerata i festivala s brojem slobodnih mjesta ili karata. Kako bismo omogućili korisniku upis u bazu podataka, postoje forme koje su definirane u datoteci *forms.py*. Stvaranjem forme kreiramo pomoćnu klasu koja dopušta stvaranje Django forme za već postojeći model podataka.

Uzmimo za primjer model „Album“. Na slici 32 prikazana je klasa modela „Album“ sa svim poljima koje sadrži, a na slici 33 prikazana je forma za taj isti model.


```

vinyl = 'vinyl'
cd_album = 'cd_album'
cd_box_set = 'cd_box_set'

ALBUM_FORMAT_CHOICES = (
    (vinyl, 'Vinyl 12" Album'),
    (cd_album, 'CD Album'),
    (cd_box_set, 'CD Box Set')
)

class Album (models.Model):
    artist_id = models.ForeignKey(Artist, on_delete=models.CASCADE, null=True)
    genre_id = models.ForeignKey(Genre, on_delete=models.CASCADE, default=1)
    album_name = models.CharField(max_length=200)
    date_released = models.DateField(default=date.today)
    album_format = models.CharField(max_length=20, choices=ALBUM_FORMAT_CHOICES,
                                     default=None)
    album_price = models.DecimalField(max_digits=10, decimal_places=2, null=True)
    in_stock = models.PositiveIntegerField()
    number_of_tracks = models.IntegerField()
    number_of_discs = models.IntegerField(default=1)
    record_label_id = models.ForeignKey(RecordLabel, on_delete=models.CASCADE,
                                       null=True)
    album_cover = models.ImageField(upload_to="albums/", blank=True, null=True)

    def __str__(self):
        return self.album_name

```

Slika 32. Model album

Iako Django dopušta preoblikovanje formi u nekoj mjeri, u ovom slučaju, forma za upis novog ima sva polja navedena u modelu Album (slika 33). Bitno je da se naziv polja (npr. `album_name`), tip podatka (npr. `CharField`) i maksimalna veličina unosa (npr. `max_length=200`) slažu s podacima upisanim u klasi istog modela. U formama je moguće koristiti *widget* u kojem se navodi kako će se podatak unositi, kao u polju `date_released` gdje je određeno da će *widget* biti padajući izbornik s kalendarom kako bi zaposlenik mogao unijeti podatka o datumu izdavanja albuma. Polja koja imaju predefinirane ili određene inicijalne vrijednosti u modelu se mogu izostaviti iz forme. Izgled forme za upis novog albuma u aplikaciji prikazan je na slici 34.

```

class NewAlbumForm(forms.ModelForm):
    artist_id = forms.ModelChoiceField(queryset=Artist.objects.all())
    genre_id = forms.ModelChoiceField(queryset=Genre.objects.all())
    album_name = forms.CharField()
    date_released = forms.DateField(widget=AdminDateWidget(
        attrs={'type': 'date'}))
    album_format = forms.ChoiceField(choices=ALBUM_FORMAT_CHOICES)
    album_price = forms.DecimalField()
    in_stock = forms.IntegerField()
    number_of_tracks = forms.IntegerField()
    number_of_discs = forms.IntegerField()
    recordLabel_id = forms.ModelChoiceField(queryset=RecordLabel.objects.all())
    album_cover = forms.ImageField(required=False)

class Meta:
    model = Album
    fields = '__all__'

```

Slika 33. Forma za novi album u *forms.py*

The screenshot shows a web form titled "New Album". On the left, there is a list of labels for each field, each followed by a red asterisk indicating a required field: ARTIST *, RECORD LABEL *, GENRE *, ALBUM NAME *, DATE RELEASED *, ALBUM FORMAT *, PRICE *, ADDING TO STOCK *, NUMBER OF TRACKS *, NUMBER OF DISCS *, and ALBUM COVER. The corresponding input fields are on the right: three dropdown menus for ARTIST, RECORD LABEL, and GENRE; a text input for ALBUM NAME; a date picker for DATE RELEASED showing "dd----yyyy"; a dropdown menu for ALBUM FORMAT with "Vinyl 12" Album" selected; text inputs for PRICE, ADDING TO STOCK, NUMBER OF TRACKS, and NUMBER OF DISCS; and a file upload area for ALBUM COVER with a "Choose File" button and the text "No file chosen". At the bottom left, there is a dark grey button labeled "Save".

Slika 34. Stranica za unos novog albuma u aplikaciji

Nakon unosa svih podataka od strane korisnika, funkcija `newAlbum` prikazana na slici 35 dohvaća podatke upisane u formu i vrši validaciju podataka. Ako je forma

popunjena kako treba biti i ako validacija prođe bez greške, u bazu podataka se spremaju podaci o novom albumu s prvim sljedećim slobodnim ID brojem a *staff* korisnika se preusmjerava na stranicu s albumima.

```
@login_required
@staff_member_required
def newAlbum(request):
    if request.method == 'POST':
        albumForm = NewAlbumForm(request.POST)
        if albumForm.is_valid():
            albumForm.save()
            return redirect('JazzBluesApp:albums')
    albumForm = NewAlbumForm()
    context = {
        'form' : albumForm,
    }
    return render(request, 'new_album.html', context)
```

Slika 35. Funkcija za stvaranje novog albuma.

6.1.2. Uređivanje postojećeg albuma/koncerta/festivala

Kao što je već prethodno spomenuto, korisnik s ulogom *staff* ima mogućnost uređivanja svih postojećih artikala u ponudi. Uzmimo za primjer uređivanje albuma. Klikom na „*Edit album*“ na stranici sa svim albumima, korisnik je preusmjeren na stranicu za uređivanje informacija o albumu vidljivu na slici 36. U polju svakog atributa modela nalazi se već upisana vrijednost tog atributa odabranog albuma. Vrijednost polja je moguće izmijeniti i spremiti nove podatke koji će „pregaziti“ postojeće podatke u bazi podataka.

Edit Ko-ko

ARTIST *

Duke Ellington

+

RECORD LABEL *

Dreyfus Jazz

+

GENRE *

Jazz

▼

ALBUM NAME *

Ko-ko

DATE RELEASED *

19-Oct-2018

ALBUM FORMAT *

Vinyl 12" Album

▼

PRICE *

17.99

ADDING TO STOCK *

0

NUMBER OF TRACKS *

14

NUMBER OF DISCS *

1

ALBUM COVER

☐ CLEAR

Change:

Choose File

No file chosen

Save changes

Delete

Slika 36. Stranica za uređivanje postojećeg albuma

Dohvat podataka koji su prikazani u poljima odvija se u funkciji `albumEdit` vidljivoj na slici 37. Dohvaća se album iz baze podataka pomoći ID-a koji je proslijeđen funkciji kao parametar i stvara se rječnik (engl. *dictionary*) u kojeg se spremaju podaci svih polja albuma kojeg korisnik želi urediti. Instanca se šalje u *template* kao dio forme i tako se vrijednost svakog polja ispisuje u njegovom polju za upis. Vrijednosti se mogu mijenjati, a klikom na „Save changes“ inicijalni podaci o albumu se „prebrišu“ novounesenim vrijednostima.

```

@login_required
@staff_member_required
def albumEdit(request, album_id):
    prev_url = request.META.get('HTTP_REFERER')
    album = Album.objects.all().filter(id=album_id)
    instanca = album.first()
    data = {
        "artist_id": album[0].artist_id,
        "genre_id": album[0].genre_id,
        "album_name": album[0].album_name,
        "date_released": album[0].date_released,
        "album_format": album[0].album_format,
        "album_price": album[0].album_price,
        "in_stock": album[0].in_stock,
        "number_of_tracks": album[0].number_of_tracks,
        "number_of_discs": album[0].number_of_discs,
        "recordLabel_id": album[0].recordLabel_id,
        "album_cover": album[0].album_cover,
    }
    if request.method == 'POST':
        albumForm = NewAlbumForm(request.POST, instance=instanca)
        if albumForm.is_valid():
            next_url = request.POST.get('next_url')
            albumForm.save()
            return redirect(next_url)
    else:
        albumForm = NewAlbumForm(initial=data)
    context = {
        'prev_url': prev_url,
        'artist': album[0].artist_id,
        'recordLabel': album[0].recordLabel_id,
        'albumForm': albumForm,
        'album': album,
    }
    return render(request, 'album_edit.html', context)

```

Slika 37. Funkcija za uređivanje albuma u *views.py*

6.1.3. Brisanje albuma/koncerta/festivala

Na stranici za uređivanje albuma, desno od gumba „Save changes“ nalazi se gumb „Delete“ koji se koristi za brisanje izabranog albuma. Nakon uspješnog brisanja albuma, korisnik sa ulogom *staff* preusmjeren je na stranicu sa svim albumima, kao što je vidljivo u kôdu funkcije `albumDelete` na slici 38.

```

@login_required
@staff_member_required
def albumDelete(request, album_id):
    try:
        album = Album.objects.get(id=album_id)
        album.delete()
        return redirect('JazzBluesApp:albums')
    except:
        return redirect('JazzBluesApp:albumEdit', album_id=album_id)

```

Slika 38. Funkcija za brisanje albuma

6.2. Upravljanje narudžbama

Korisnik sa ulogom *staff* ima pristup stranici sa prikazom svih narudžbi koje su napravili registrirani korisnici (slika 39). Osim ID-a narudžbe, ispisan je status narudžbe, korisničko ime korisnika koji je napravio narudžbu i datum narudžbe. Klikom na narudžbu, korisnik je preusmjeren na stranicu s detaljima o narudžbi koja izgleda slično stranici s detaljima o narudžbi za kupca, samo što korisnik sa ulogom *staff* ima opciju mijenjanja statusa narudžbe. Dostupni statusi narudžbe su:

- u obradi (engl. *processing*)
- poslana (engl. *shipped*)
- dostavljena (engl. *delivered*)
- otkazana (engl. *canceled*)

User orders

Album orders

#140: Processing		
skomic		19.07.2022

#139: Processing		
skomic		19.07.2022

#138: Processing		
skomic		19.07.2022

#137: Processing		
skomic		19.07.2022



#136: Delivered		
adlaka		18.07.2022

#135: Processing		
adlaka		12.07.2022

Slika 39. Stranica sa prikazom svih narudžbi

Nakon što korisnik uspješno plati narudžbu, ona se automatski sprema u bazu podataka sa statusom *processing*, što je određeno u datoteci *models.py* pri definiranju klase `AlbumOrder` s *default* vrijednosti. Ostale opcije su za korisnika sa ulogom *staff* dostupne u padajućem izborniku kao što je prikazano na slici 40.

Order #130

	Ella Fitzgerald: Love for Sale Release Date 18 February 2022	
	2 × £14.99	£29.98
	Stan Getz: Lullaby of Birdland Release Date 18 February 2022	
	1 × £6.99	£6.99
Total £36.97		
Shipped <input type="button" value="v"/>		
Change order status		

Slika 40. Detalji narudžbe za *staff* korisnika

7. Zaključak

U ovom radu predstavljena je web aplikacija za kupnju albuma i karata iz svijeta jazz & blues glazbe izrađena korištenjem razvojnog okvira Django pisanog u programskom jeziku Python.

Cilj izrade ove aplikacije bio je približiti sadržaj jazz & blues glazbe njenim ljubiteljima zato što navedeni žanrovi nisu toliko zastupljeni u postojećim glazbenim web shopovima. Ideja je bila objediniti sve informacije i novosti te omogućiti korisnicima brzo i jednostavno pregledavanje takvog sadržaja. Korisnicima je omogućen pregled raznih formata albuma i vrsta događaja, kao i kupovina karata sa sjedećim mjestima za koncerte. Korisnici zaposlenici mogu jednostavno dodati nove artikle u ponudu i ukloniti postojeće i to sve korištenjem Django formi koje su uvelike pojednostavile implementaciju upisa novih artikala u bazu podataka od strane zaposlenika. Registrirani korisnici mogu i ocjenjivati artikle i na taj način razmjenjivati mišljenja o albumima te ostavljati prijedloge drugim korisnicima.

Odabrana tema završnog rada ostavlja dosta prostora za nadogradnje jer je sama po sebi opširna. Neka od mogućih proširenja bi svakako bilo dodavanje liste pjesama koje se nalaze na albumima te ispis istih na stranici s detaljima o albumu, a nadogradnje bi bile moguće i u dijelu naplate narudžbi. Nadalje, korisnicima bi bila korisna i mogućnost dodavanja artikala u favorite, kako bi mogli spremati albume koji im se sviđaju, a ne pretraživati stranicu svaki put prilikom vršenja narudžbe. Favorite bi implementirala kao zasebni model s vezom s korisnikom i artiklima. Ono što stranici nedostaje, što imaju sve modernije stranice, je responzivnost. Stranica je prilagođena samo za računala i ostale uređaje velikih ekrana. Nadogradnje su dobrodošle i u dizajnu stranice – korištenjem nekih drugih tehnologija postigao bi se moderniji izgled stranice, sama izrada aplikacije bi isto bila puno jednostavnija a i korištenje aplikacije bi bilo ugodnije iskustvo.

Literatura

[1] „Django Introduction“, MDN Web Docs Group

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction>, (posjećeno 10.07.2022.)

[2] „What is PostgreSQL“, The PostgreSQL Global Development Group,

<https://www.postgresql.org/about/>, (posjećeno 10.07.2022.)

[3] Duckett J.: „Html & Css“, John Wiley & Sons, Inc.

[4] Freeman E., Robson E.: Head First Javascript Programming: A Brain-Friendly Guide

<https://en.wikipedia.org/wiki/JavaScript>

[6] „HTML2canvas“, Niklas von Herten

<https://html2canvas.hertzen.com/dist/html2canvas.js>