

WEB APLIKACIJA ZA EVIDENCIJU I UKLANJANJE ONEČIŠĆENJA U GRADSKIM I VANGRADSKIM PODRUČJIMA

Jakobljević, Bruno

Undergraduate thesis / Završni rad

2024

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

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:228:112290>

Rights / Prava: [In copyright](#)/[Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-03-01**



Repository / Repozitorij:

[Repository of University Department of Professional Studies](#)



UNIVERSITY OF SPLIT



SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Stručni prijediplomski studij Informacijska tehnologija

TEO VUKELIĆ

ZAVRŠNI RAD

**IZRADA WEB TRGOVINE GLAZBENIH
INSTRUMENATA I OPREME**

Split, rujan 2024.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Stručni prijediplomski studij Informacijska tehnologija

Predmet: Sigurnost računala i podataka

Z A V R Š N I R A D

Kandidat: Teo Vukelić

Naslov rada: Izrada web trgovine glazbenih instrumenata i opreme

Mentor: Lada Sartori, viši predavač

Split, rujan 2024.

Sadržaj

SAŽETAK	1
SUMMARY	2
1. UVOD	3
2. KORIŠTENE TEHNOLOGIJE I ALATI	4
2.1. Django.....	4
2.2. PostgreSQL	5
2.2.1. Povezivanje Djanga s PostgreSQL	5
2.3. HTML, Bootstrap	5
2.4. jQuery, AJAX	6
3. BAZA PODATAKA I MODELI	7
4. OPIS APLIKACIJE	10
4.1. Kreiranje projekta i instalacija paketa	10
4.2. Navigacijska traka	11
4.3. Prikaz i mogućnosti aplikacije za gosta.....	12
4.4. Prikaz i mogućnosti aplikacije za prijavljenog korisnika	15
4.5. Prikaz i mogućnosti aplikacije za administratora	22
5. Zaključak	27
Literatura	28

SAŽETAK

Cilj ovog rada je izrada web trgovine za glazbene instrumente i opremu. Trgovina na jednom mjestu omogućava jednostavan pregled ponude i kupnju instrumenata i ostale opreme, kao i sve potrebne informacije o pojedinim proizvodima, njihovoj dostupnosti, cijenama i sniženjima. Implementirane su sljedeće uloge: administrator, prijavljeni korisnik i gost. Administrator stranice može uređivati proizvode i odlučivati o sniženjima, te upravljati korisničkim računima. Gosti mogu slobodno pretraživati sadržaj web trgovine te obavljati kupnju, dok prijavljeni korisnici mogu uređivati svoj profil, pregledavati narudžbe te ostavljati recenzije.

Poslužiteljski dio aplikacije napravljen je u razvojnom okviru Django. Za korisničko sučelje korišteni su HTML i Bootstrap biblioteka. Za pohranu podataka korištena je baza podataka PostgreSQL.

Ključne riječi: Django, PostgreSQL, web aplikacija, web trgovina

SUMMARY

Development of web shop for musical instruments and equipment

The goal of this thesis is the creation of a web store for musical instruments and equipment. Such a shop enables a simple overview of the offer and the purchase of instruments and other equipment, all in one place, as well as all the necessary information about individual products, their availability, prices and discounts. The following roles are implemented: administrator, logged-in user and guest. The administrator can edit products, make decisions about discounts and manage user accounts. The guests can freely search the contents of the web store and make purchases, while the logged-in users can edit their own profile, view their orders and write reviews.

The backend part of the application was created with the framework Django. HTML and the Bootstrap library were used for the user interface. The PostgreSQL database was used for data storage.

Keywords: Django, PostgreSQL, web application, web store

1. UVOD

Kako je internet postao dostupan velikom broju populacije populariziralo se i obavljanje kupovine putem internetskih trgovina, zahvaljujući jednostavnom i brzom obliku kupovine. Kupci više ne moraju trošiti vrijeme na odlazak u trgovine, samo da bi saznali da njihov traženi proizvod uopće nije dostupan. Ideja ove aplikacije je dovesti glazbu u dom potencijalnih kupaca. S obzirom na nepostojeće radno vrijeme, kupci mogu u bilo koje doba dana istraživati ponudu, pratiti aktualne cijene i popuste, dobiti informacije o traženim instrumentima te obavljati kupnju, a ukoliko nisu zadovoljni određenim proizvodima mogu ih vratiti u određenom vremenskom roku.

Administrator odlučuje o sadržaju stranice, ponudi i cijeni proizvoda. Ima pregled svih poslanih i neposlanih narudžbi, kao i uvid u recenzije koje može brisati ukoliko procijeni da se radi o neprimjerenom sadržaju, te sukladno tome kupac bude upozoren putem pripadajuće elektroničke pošte. Korisnik ima uvid u dostupnost proizvoda, njihove cijene i detalje o proizvodima, koje može pretraživati putem tražilice ili ih filtrirati preko pripadajućih kategorija. Kao gost može dodavati proizvode u košaricu, obavljati kupnju, a po završetku narudžbe dobije potvrdu o istoj putem elektroničke pošte. Ukoliko korisnik odluči napraviti račun i prijaviti se, njegovi osobni podaci bit će pohranjeni, koje će također moći uređivati ukoliko bude htio, a u krajnjoj mjeri može i izbrisati svoj profil. Može pratiti je li njegova narudžba obrađena i poslana ili nije. Ima i mogućnost ostavljanja recenzija i ocjena, ali isključivo samo za one proizvode koje je kupio.

U prvom poglavlju opisane su osnovne funkcionalnosti aplikacije. Drugo poglavlje opisuje tehnologije koje su bile korištene za izradu ovog rada. U trećem poglavlju je opisana baza podataka sa njenim modelima. U četvrtom poglavlju opisan je praktični dio aplikacije, te instalacija potrebnih alata. Peti i zadnji dio odnosi se na zaključak samog rada, te je također naveden popis literature.

2. KORIŠTENE TEHNOLOGIJE I ALATI

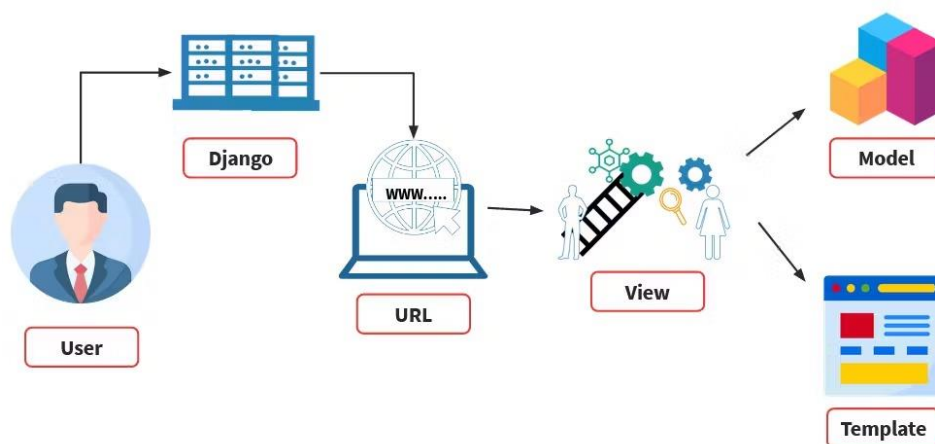
2.1. Django

Django je razvojni okvir napisan u programskom jeziku Python. To je okvir visoke razine, koji je jednostavan i lagan za korištenje, čemu pridonose brojni dostupni paketi i biblioteke (engl. *library*), što naravno pozitivno utječe na samu brzinu izrade aplikacija. Django koristi MVT (engl. *Model View Template*) arhitekturu. Sastoji se od modela, prikaza i predloška (slika 1).

Modeli predstavljaju izvor podataka u bazi podataka. To su Python klase koje nasljeđuju `django.db.models.Model`. Svaki atribut modela predstavlja jedno polje u tablici u bazi podataka, te možemo definirati tip podatka, duljinu polja, odrediti je li polje obavezno itd. Django automatski generira pristup bazi podataka koristeći ORM (engl. *Object-Relational Mapping*). ORM omogućava programerima da preko Django kôda pristupaju bazi podataka bez potrebe da pišu SQL (engl. *Structured Query Language*) upite.

Prikaz (eng. *view*), koji se odnosi na suštinu same aplikacije, preko URL-a (engl. *Uniform Resource Locator*) primi korisnikov upit, koji se obradi u definiranoj funkciji, te se vraća korisniku kao predložak (engl. *template*).

Predložak je uglavnom HTML (engl. *HyperText Markup Language*) datoteka i odgovoran je za prikazivanje podataka korisnicima, podataka koje je mu je *view* proslijedio. Na kraju korisnik vidi generiranu web stranicu u svom web pregledniku.



Slika 1: Arhitektura MVT [1]

2.2. PostgreSQL

PostgreSQL je sustav otvorenog kôda za upravljanje relacijskim bazama podataka (engl. RDBMS – *Relational Database Management System*). Podržava upite SQL, te isto tako i upite JSON (engl. *JavaScript Object Notation*). Poznat je po svojoj robusnosti, sigurnosti, skalabilnosti i bogatim značajkama. Zbog svoje stabilnosti i fleksibilnosti idealan je izbor za raznovrsne aplikacije. Podržava sve poznate operacijske sustave i brojne programske jezike, kao što su: Python, JavaScript, Ruby, Perl i drugi. Također podržava svojstva ACID (engl. *Atomicity, Consistency, Isolation, Durability*), što osigurava integritet podataka i u slučaju nepredviđenih kvarova.

2.2.1. Povezivanje Djanga s PostgreSQL

Prvo se u konzoli treba instalirati `psycopg2` modul: `pip install psycopg2`. Nakon toga u prethodno kreiranom Djangovom projektu unutar `settings.py` datoteke treba promijeniti zadanu bazu podataka, koja je inače SQLite, tako da PostgreSQL bude korišten (ispis 1) [2].

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'MUSIC_WEB_SHOP',
        'USER': 'teo_vukelic',
        'PASSWORD': '',
        'PORT': '5432',
    }
}
```

Ispis 1: Povezivanje Djanga s PostgreSQL

2.3. HTML, Bootstrap

HTML (engl. *HyperText Markup Language*) je standardni jezik za izradu i strukturiranje web stranica. Za definiranje strukture i sadržaja koriste se oznake koje govore web pregledniku što će se prikazati na web stranici. Svaki HTML dokument ima tri osnovne oznake: `html`, `head` (zaglavlje) i `body` (tijelo). Oznaka `html` govori web pregledniku da je riječ o HTML dokumentu. Zaglavlje sadrži metapodatke `<meta>`, naslov stranice `<title>`, skripte `<script>`, stilove `<style>`. Tijelo se odnosi na sadržaj dokumenta koji

web preglednik prikazuje korisniku. Ostale najčešće korištene oznake su: `<div>` za grupiranje sadržaja, `<h>` za naslove, `<a>` za hiperlinkove, `` za prikazivanje slika, `<p>` za označavanje odlomka teksta [3].

Bootstrap je popularan razvojni okvir za izradu responzivih web stranica i web aplikacija. Pruža unaprijed definirane HTML, CSS i JavaScript komponente koje omogućuju brzu i jednostavnu izradu korisničkog sučelja. Namijenjen je i početnicima i iskusnijim programerima. Omogućuje prilagodbu izgleda stranice različitim veličinama zaslona, od mobilnih uređaja do desktop računala. Responzivan dizajn omogućuje korisnicima besprijekorno iskustvo na različitim uređajima. Dolazi s unaprijed definiranim komponentama, kao što su: navigacijska traka, gumbi, forme, kartice, padajući izbornici. Također nudi mnoge besplatne i plaćene predloške koji se mogu koristiti za izradu web stranica. Optimiziran je za rad na većini web preglednika. Zbog svoje jednostavnosti, prilagodljivosti i opsežne podrške jedan je od najkorištenijih okvira u izradi web aplikacija [4].

2.4. jQuery, AJAX

jQuery je JavaScript biblioteka koja uvelike pojednostavljuje pisanje JavaScript kôda te rukovanje operacijama Ajax na web stranicama. Svrha jQuerya je pojednostavljivanje uobičajenih zadataka koji zahtijevaju mnogo linija JavaScript kôda na način da se ti zadaci obave uz pisanje što manje kôda. Omogućava manipuliranje HTML/DOM-om (engl. *Document Object Model*), tj. olakšava dodavanje, brisanje elementa na web stranici. jQuery je također kompatibilan sa svim najpoznatijim web preglednicima [5].

AJAX (engl. *Asynchronous JavaScript and XML*) omogućava dinamičku razmjenu podataka između klijenta i poslužitelja, bez potrebe da se cijela web stranica ponovno učitava. JavaScript preko AJAX-a šalje zahtjev poslužitelju, koji nakon što obradi zahtjev vrati podatke klijentu, uglavnom u formatu JSON ili XML (engl. *Extensible Markup Language*), te se tada podatci mogu prikazati na web stranici bez ponovnog učitavanja cijele stranice, što utječe na poboljšano korisničko iskustvo jer stranica ostaje responzivna i interaktivna.

3. BAZA PODATAKA I MODELI

Modeli predstavljaju izvor podataka u bazi podataka. Unutar njih se definiraju atributi koji predstavljaju stupce u tablicama baze podataka. Django omogućava pristup i rad s bazom podataka bez potrebe za pisanjem SQL upita. Neki od primjera manipulacije podacima su: `products = Product.objects.all()`, za dohvaćanje svih proizvoda iz baze, `products = Product.objects.filter(is_sale=True)`, za dohvaćanje svih proizvoda na sniženju. Nakon kreiranja modela u `models.py` ili nakon nekakvih značajnih promjena u samim modelima, potrebno je stvoriti migracije koje će se primijeniti na bazu podataka.

Pri izradi aplikacije definirano je 8 modela: `User`, `Category`, `Product`, `ProductReview`, `Profile`, `Order`, `OrderItem` i `ShippingAddress`. Odnos između njih prikazan je na slici 2. U nastavku je opisan svaki od modela.

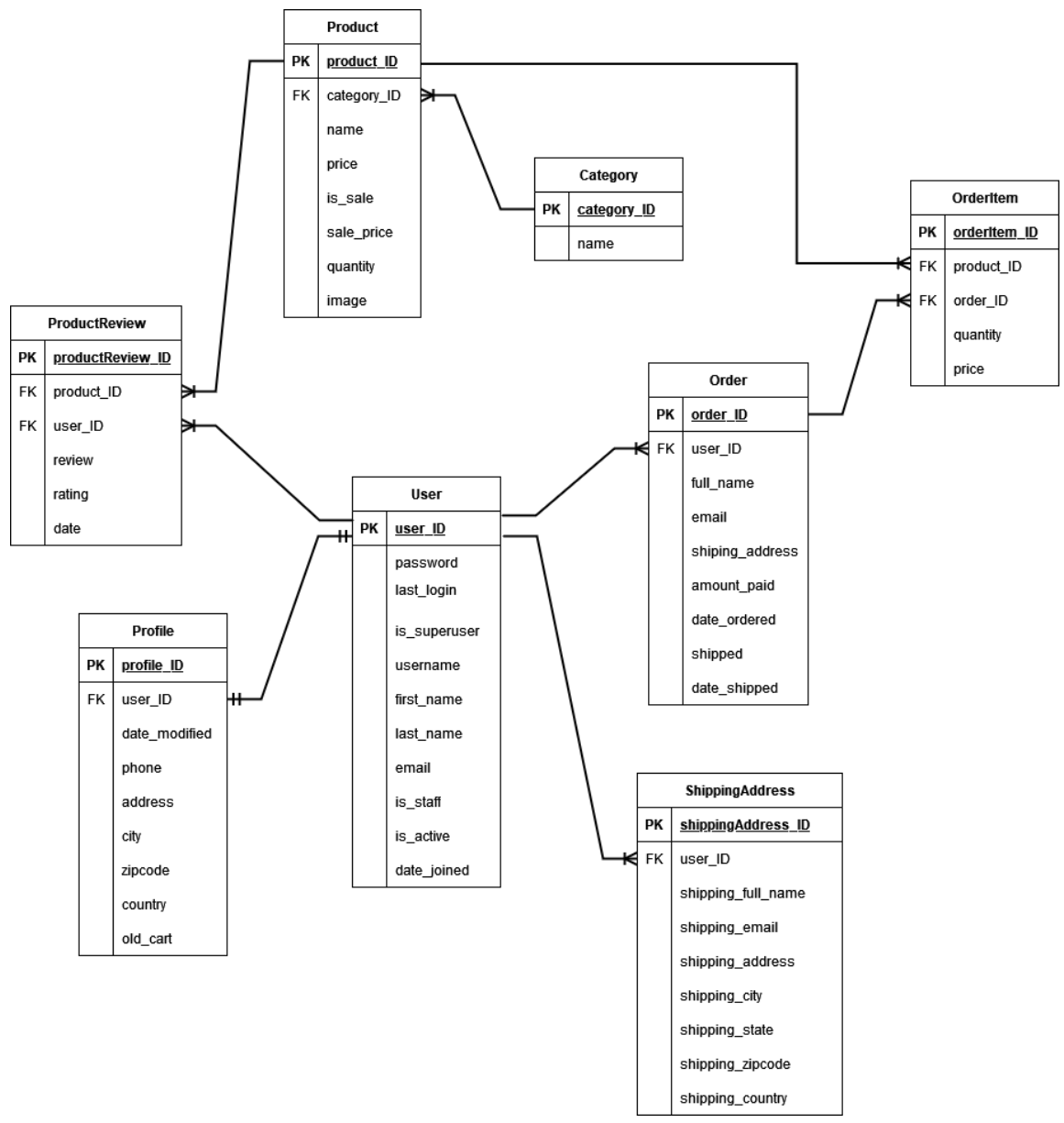
- `User` je već ugrađeni Django model. Sadrži osnovne podatke o korisniku, kao što su: korisničko ime, lozinka, ime i prezime, elektronička adresa, zapis o posljednjoj prijavi, informacija je li korisnik administrator ili ne. `User` s tablicama `Order`, `ProductReview` i `ShippingAddress` ostvaruje relaciju jedan-prema-više (korisnik može imati više narudžbi, te adresa za dostavu i više puta može recenzirati različite proizvode), dok s tablicom `Profile` ima relaciju jedan-prema-jedan (korisnik može imati samo jedan profil).
- `Category` sadrži nazive svih kategorija proizvoda. S tablicom `Product` ima relaciju jedan-prema-više (jedna kategorija može sadržavati više proizvoda).
- `Product` sadrži informacije o nazivu proizvoda, kojoj kategoriji pripada, cijeni, potencijalnoj akcijskoj cijeni, količini na zalih, te pripadajuću sliku proizvoda. S tablicama `ProductReview` i `OrderItem` ima relaciju jedan-prema-više (proizvod može biti recenziran više puta, dok s tablicom `Category` ima relaciju više-prema-jedan (više proizvoda može pripadati jednoj kategoriji).
- `ProductReview` sadrži informacije o svim recenzijama i ocjenama za pojedine proizvode, te o korisnicima koji su recenzirali proizvode. S tablicama `Product` i `User` ima relacije više-prema-jedan (više recenzija može pripadati jednom proizvodu i korisniku).

- `Profile` sadrži podatke o korisnikovom telefonu, adresi, državi, mjestu, poštanskom broju. S tablicom `User` ima relaciju jedan-prema-jedan (jedan profil za jednog korisnika).
- `Order` sadrži podatke o kupcu koji je obavio narudžbu, njegovo ime i prezime, elektroničku adresu, ukupni iznos narudžbe, datum narudžbe, status narudžbe, te datum isporuke. S tablicom `User` ostvariva relaciju više-prema-jedan (jedan korisnik može imati više narudžbi).
- `OrderItem` je među tablica između tablice `Order` i `Product`. S obje tablice ima vezu više-prema-jedan, što znači da jedan proizvod može postojati na više narudžbi, te da na jednoj narudžbi može biti više proizvoda. Tablica `OrderItem` sadrži podatke o količini i cijeni kupljenog proizvoda.
- `ShippingAddress` sadrži slične informacije kao i tablica `Profil`, ali su ove informacije isključivo vezane za određenu narudžbu, dok su podaci navedeni u `Profilu` više osobni, nisu povezani sa narudžbama i nisu namijenjeni čestim promjenama. Također omogućava korisniku da ima više adresa dostave.

```
class Product(models.Model):
    name = models.CharField(max_length=50)
    price = models.DecimalField(default=0, decimal_places=2,
max_digits=7)
    category = models.ForeignKey(Category,
on_delete=models.CASCADE, default=1)
    image = models.ImageField(upload_to='uploads/product/')
    is_sale = models.BooleanField(default=False)
    sale_price = models.DecimalField(default=0, null=True,
decimal_places=2, max_digits=7)
    quantity = models.PositiveBigIntegerField(default=1)
```

Ispis 2: Primjer modela `Product`

U Ispisu 2 vidi se na koji način je definiran model `Product`. Za atribut `price` je određen maksimalni broj znamenki te broj decimala, za `image` je navedena putanja direktorija gdje su slike pohranjene, za `quantity` će zadana količina proizvoda biti jedan ukoliko nije drugačije određeno, a `category` predstavlja vezu između proizvoda i kategorije gdje je određeno ako se kategorija izbrise s njome će biti izbrisani i svi pripadajući proizvodi.



Slika 2: E-R Dijagram

4. OPIS APLIKACIJE

4.1. Kreiranje projekta i instalacija paketa

Za početak se kreira direktorij unutar kojeg treba kreirati virtualno okruženje koje služi tome da sve promjene i paketi instalirani u projektu budu isključivo vezani za taj projekt. Nakon toga se instalira Django, kreira projekt i aplikacija. Postupak instalacije prikazan je na ispisu 3.

```
python -m venv virtualno_okruzenje
virtualno_okruzenje\Scripts\activate
pip install django
django_admin startproject web_shop
cd web_shop
python manage.py runserver
```

Ispis 3: Instalacija Djanga i kreiranje projekta za aplikaciju

Za kreiranje administratora koristi se naredba `python manage.py createsuperuser`, nakon koje treba unijeti korisničko ime i lozinku.

Kako bi bilo moguće upravljati slikama proizvoda treba instalirati paket Pillow, koji je Python biblioteka. Za generiranje PDF datoteka potreban je paket ReportLab. Za Stripe plaćanje je također potrebno instalirati odgovarajući paket. Naredbe za instalaciju sva ova tri paketa prikazane su u ispisu 4.

```
pip install Pillow
pip install reportlab
pip install stripe
```

Ispis 4: Instalacija preostalih paketa

Nakon što su instalirani svi potrebni paketi vidljivi u ispisu 3 i 4, aplikacija je spremna za korištenje.

4.2. Navigacijska traka

Navigacijska traka omogućava korisnicima jednostavno i brzo kretanje kroz web stranicu. Nalazi se na samom vrhu svake stranice i sadrži poveznice putem kojih se pristupa određenim stranicama (slika 3). U navigacijskoj traci poveznice su prikazane ovisno o korisnikovoj ulozi, je li korisnik prijavljen ili ne, te je li administrator. Svi imaju mogućnost pretraživanja proizvoda preko tražilice, te je svima vidljiva „Košarica“ gdje mogu pregledavati sadržaj košarice i obavljati kupnju.

Gostu se prikazuju sljedeće poveznice:

- Početna (vodi korisnika na početnu stranicu gdje su prikazani svi proizvodi, te se na samoj stranici mogu filtrirati sniženi proizvodi)
- Kategorije (otvara se padajući izbornik sa svim kategorijama proizvoda)
- Akcija (prikažu se svi sniženi proizvodi)
- Prijava (otvara se padajući izbornik sa opcijom prijave ili registracije)

Prijavljeni korisnik, za razliku od gosta, preko navigacijske trake može uređivati svoj profil, pratiti svoje narudžbe, te se može odjaviti.

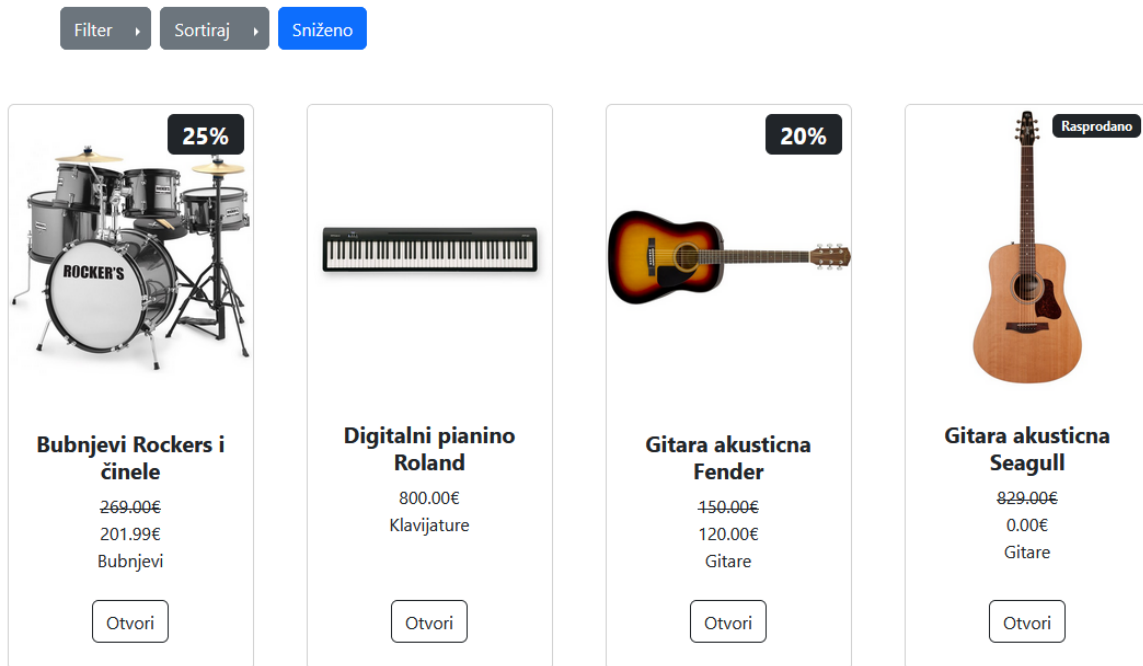
Administrator, uz sve navedene opcije na navigacijskoj traci, može pristupiti i popisu svih korisničkih računa kojima može upravljati.



Slika 3: Primjer izgleda navigacijske trake za prijavljenog korisnika

4.3. Prikaz i mogućnosti aplikacije za gosta

Gost ima najmanje ovlasti u aplikaciji. Na početnoj stranici (slika 4) su prikazani svi proizvodi koji se mogu pregledavati, te filtrirati po cijeni, nazivu i datumu. Također se mogu filtrirati i sniženi proizvodi. Sniženi proizvodi na samoj slici imaju istaknut postotak sniženja. Na jednak način je i za rasprodane proizvode istaknuto da su rasprodani. Proizvode je moguće sortirati po cijeni, nazivu i datumu.



Slika 4: Pregled proizvoda na početnoj stranici

Također preko navigacijske trake i kartice „Kategorije“ se preko padajućeg izbornika može pristupiti svim kategorijama. U navigacijskoj traci se nalazi i tražilica koja omogućava da se na brz način pronađu željeni proizvodi (ispis 5).

```
def search(request):
    if request.method == "POST":
        searched = request.POST['searched']
        if len(searched) < 3:
            messages.error(request, "Unesite najmanje 3 slova za pretragu")
            return redirect("search")

        searched_result =
        Product.objects.filter(name__icontains=searched)
```



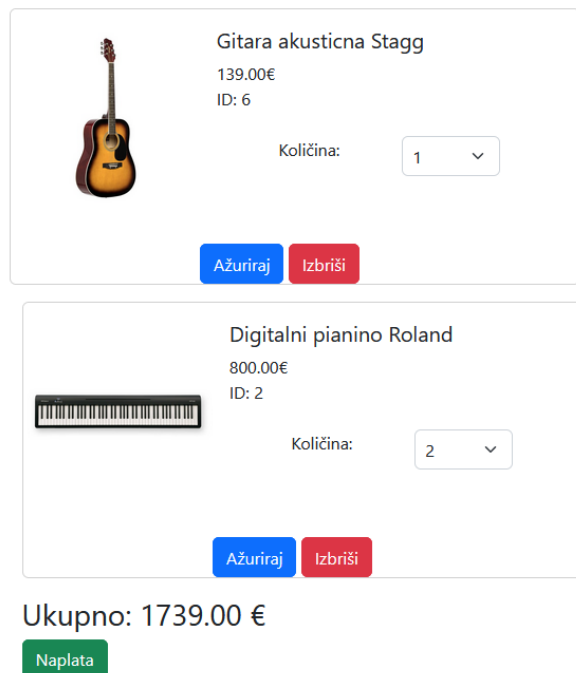
```

    if not searched_result:
        messages.success(request, "Taj proizvod ne postoji")
        return render(request, "search.html",
            {'searched':searched})
    else:
        return render(request, "search.html",
            {'searched':searched,
            'searched_result':searched_result})
    else:
        return render(request, "search.html", {})

```

Ispis 5: Kôd za pretraživanje proizvoda preko tražilice

Klikom na proizvod otvara se detaljniji pregled proizvoda, gdje korisnik može vidjeti dostupnu količinu, ocjenu i recenzije proizvoda, ukoliko je proizvod bio ocjenjivan. Korisnik ne mora biti registriran da bi mogao kupovati. Dovoljno je samo odabrati željeni proizvod ili proizvode te ih dodati u košaricu. Sadržaj košarice može pregledati preko kartice „Košarica“ koja se nalazi u navigacijskoj traci. Na samoj kartici prikazan je broj artikala u košarici. Unutar košarice moguće je promijeniti količinu pojedinog artikla, a ponuđena količina odgovara količini na zalihama. Naravno proizvodi se mogu i izbrisati iz košarice. Pri završetku kupnje zalihe se automatski smanjuju ovisno o kupljenoj količini (slika 5).



Slika 5: Izgled košarice

Korisnik na početnoj stranici aplikacije ima opciju prijaviti se ili registrirati ukoliko nema svoj korisnički račun. U navigacijskoj traci se nalazi kartica „Prijava“ klikom na koju se otvara padajući izbornik s opcijama za prijavu ili registraciju. Ako korisnik želi uživati veće ovlasti mora se prvo registrirati. Prilikom registracije otvara se forma koju mora popuniti odgovarajućim podacima, kao što su: korisničko ime, ime, prezime, elektronička pošta, lozinka i ponovljena lozinka (slika 6). Zbog sigurnosti lozinka se kriptira pomoću sigurnosnog CSRF (engl. *Cross-Site Request Forgery*) tokena. Kad korisnik ispuni formu, Django provjeri CSRF token kako bi se uvjerio da je zahtjev došao iz forme na samoj web stranici, a ne sa neke zlonamjerne stranice. Ukoliko su svi podaci validni i ispravno uneseni korisniku će biti dodijeljen pripadajući račun, te će biti automatski preusmjeren na novu stranicu gdje može unijeti dodatne osobne podatke, te podatke vezane za samu dostavu. Ukoliko ne želi ne mora odmah unositi dodatne podatke o sebi. Podatke može uvijek unijeti kasnije.

Korisničko ime
150 znakova ili manje. Slova, znamenke i @/./+/-/_.

Ime

Prezime

E-mail

Lozinka

- Lozinka ne smije biti previše slična vašim osobnim podacima.
- Najmanje 8 znakova.
- Lozinka ne smije biti sastavljena isključivo od znamenaka.

Potvrdite lozinku
Potvrdite lozinku.

Registrirajte se

Slika 6: Registracijska forma

4.4. Prikaz i mogućnosti aplikacije za prijavljenog korisnika

Nakon registracije korisnik se može prijaviti preko poveznice „Prijava“ koja se nalazi u navigacijskoj traci, klikom na koju se otvara forma za prijavu. Potrebno je unijeti korisničko ime i lozinku. Kad korisnik unese potrebne podatke aplikacija onda provjeri da li korisnik ima kreiran korisnički račun, te ako ima korisnik bude prijavljen. Na vrhu stranice iskoči poruka o uspješnoj ili neuspješnoj prijavi. Ukoliko je uspješno prijavljen bit će preusmjeren na početnu stranicu, a ukoliko prijava nije bila uspješna ostaje na stranici sa formom za prijavu. Provjera o postojećem korisniku vrši se preko Django ugrađene funkcije `authenticate()`. Kôd za prijavu korisnika prikazan je u ispisu 6.

```
def login_user(request):
    if request.method == "POST":
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=username,
password=password)
        if user:
            login(request, user)
            messages.success(request, ("Prijavljeni ste"))
            return redirect('home')
        else:
            messages.success(request, ("Neuspješna prijava,
pokušajte opet"))
            return redirect('login')
    else:
        return render(request, 'login.html', {})
```

Ispis 6: Prikaz kôda za prijavu korisnika

Prijavljeni korisnik će za razliku od gosta u navigacijskoj traci imati poveznice „Profil“, gdje će moći uređivati ili brisati svoj profil, te poveznicu „Narudžbe“ gdje će moći pratiti status svojih narudžbi. Klikom na karticu „Narudžbe“ otvara se padajući izbornik gdje su kroz dvije poveznice „Poslane narudžbe“ i „Neposlane narudžbe“ filtrirane narudžbe ovisno o statusu. Klikom na „Poslane narudžbe“ otvara se tablični prikaz podataka vezanih za narudžbe, kao što su korisničko ime, broj narudžbe, iznos računa, elektronička pošta i datum isporuke, slika 7. Na isti način može pregledati i neposlane narudžbe.

Poslane narudžbe

Korisnik	Narudžba	Cijena	Email	Poslano
ante	60	1600.00 €	anteantic@gmail.com	Sept. 1, 2024, 3:07 p.m.
ante	37	800.00 €	anteantic@gmail.com	Sept. 1, 2024, 3:06 p.m.
ante	46	576.99 €	anteantic@gmail.com	Sept. 1, 2024, 12:18 a.m.
ante	54	120.00 €	anteantic@gmail.com	Sept. 1, 2024, 12:17 a.m.
ante	53	120.00 €	anteantic@gmail.com	Aug. 19, 2024, 12:59 p.m.

Slika 7: Prikaz pregleda poslanih narudžbi

Radi lakše preglednosti narudžbe su poredane kronološki, od najnovije ka najstarijoj. Svaka pojedina narudžba sadrži poveznicu koja otvara stranicu sa pregledom narudžbe, a na dnu se nalazi botun za generiranje PDF datoteke na kojoj su ispisani korisnikovi podaci, kupljeni proizvodi te ukupni iznos narudžbe. Kod za generiranje PDF datoteke prikazan je u ispisu 7.

```
def orders_pdf(request, pk):
    buf = io.BytesIO()
    c = canvas.Canvas(buf, pagesize=letter, bottomup=0)
    text_object = c.beginText()
    text_object.setTextOrigin(inch, inch)
    text_object.setFont("Verdana", 14)

    order = Order.objects.get(id=pk)
    items = OrderItem.objects.filter(order=pk)

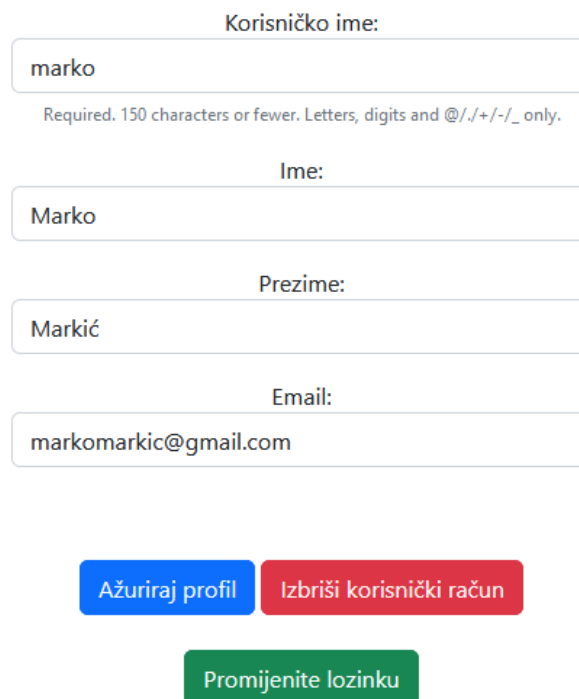
    lines = []
    lines.append(f"Narudžba: {str(pk)}")
    lines.append(f"Korisnik: {str(order.full_name)}")
    for item in items:
        temp = item.product.name + " - Količina: " +
str(item.quantity) + " - Cijena: " + str(item.price)
        lines.append(temp)

    for line in lines:
        text_object.textLine(line)
```

```
c.drawText(text_object)
c.showPage()
c.save()
buf.seek(0)
return FileResponse(buf, as_attachment=True,
filename='orders.pdf')
```

Ispis 7: Prikaz kôda za generiranje PDF datoteke

Registrirani korisnik također može uređivati svoj profil preko poveznice „Profil“ u navigacijskoj traci. Otvari se padajući izbornik sa poveznicama „Korisnički profil“ i „Korisnički podaci“. Otvaranjem prve poveznice pristupa osnovnim podacima kao što su: korisničko ime, ime, prezime i elektronička pošta (slika 8). Na toj stranici se nalaze botuni za uređivanje i brisanje računa, te za promjenu lozinke.



Korisničko ime:

Required. 150 characters or fewer. Letters, digits and @/./+/_ only.

Ime:

Prezime:

Email:

Ažuriraj profil Izbriši korisnički račun

Promijenite lozinku

Slika 8: Stranica za uređivanje i brisanje računa

Ako korisnik želi promijeniti lozinku klikom na botun „Promijenite lozinku“ otvara se stranica s formom za promjenu lozinke. Korisnik će biti upozoren ukoliko je lozinka prekratka ili se nova lozinka ne podudara s ponovljenom novom lozinkom. Nakon uspješno promijenjene lozinke korisnika se vraća na prethodnu stranicu gdje može nastaviti uređivati

svoj račun. Nakon promjene lozinke korisnik bude automatski ponovno prijavljen na stranicu, bez potrebe da se sam prijavljuje sa novim podacima.

```
class ChangePasswordForm(SetPasswordForm):
    class Meta:
        model = User
        fields = ['new_password1', 'new_password2']
    def __init__(self, *args, **kwargs):
        super(ChangePasswordForm, self).__init__(*args,
**kwargs)

        self.fields['new_password1'].widget.attrs['class'] =
'form-control'
        self.fields['new_password1'].widget.attrs['placeholder'
] = 'Lozinka'
        self.fields['new_password1'].widget.attrs['style'] =
'max-width: 500px;'
        self.fields['new_password1'].label = ''
        self.fields['new_password1'].help_text = '<ul
class="form-text text-muted small"><li>Lozinka ne smije biti
previše slična vašim osobnim podacima.</li><li>Najmanje 8
znakova.</li><li>Lozinka ne smije biti sastavljena isključivo
od znamenaka.</li></ul>'

        self.fields['new_password2'].widget.attrs['class'] =
'form-control'
        self.fields['new_password2'].widget.attrs['placeholder'
] = 'Potvrdite lozinku'
        self.fields['new_password2'].widget.attrs['style'] =
'max-width: 500px;'
        self.fields['new_password2'].label = ''
        self.fields['new_password2'].help_text = '<span
class="form-text text-muted"><small>Potvrdite
lozinku.</small></span>'
```

Ispis 8: Kôd forme za promjenu lozinke

Ispis 8 prikazuje kôd forme za promjenu lozinke.

Za ažuriranje profila treba promijenti jedan od podataka koje je unio prilikom registracije te nakon toga odabrati „Ažuriraj profil“. Po završetku uređivanja profila bit će preusmjeren na početnu stranicu. Kôd za promjenu podataka profila prikazana je u ispisu 9.

```

def update_user(request):
    if request.user.is_authenticated:
        if request.POST.get("Delete user"):
            User.objects.get(id=request.POST["Delete
user"]).delete()
            messages.success(request, ("Račun izbrisan"))
            return redirect("home")
        else:
            current_user =
User.objects.get(id=request.user.id)
            user_form = UpdateUserForm(request.POST or None,
instance=current_user)
            if user_form.is_valid():
                user_form.save()
                login(request, current_user)
                messages.success(request, "Korisnički podaci
ažurirani")
                return redirect('home')
            return render(request, "update_user.html",
{'user_form':user_form})
        else:
            messages.success(request, "Morate biti prijavljeni za
pristup")
            return redirect('home')

```

Ispis 9: Prikaz kôda za ažuriranje i brisanje korisničkog računa

Prilikom registracije, nakon što se kreira korisnički račun, korisnika se preusmjeri na stranicu gdje može unijeti osobne podatke te podatke vezane za dostavu. Te podatke može odmah unijeti ili kasnije. Ako odluči to napraviti kasnije to može učiniti preko poveznice „Profil“ u navigacijskoj traci, gdje nakon toga odabire „Korisnički podaci“. Prikaz formi za unos podataka dan je na slici 9.

Telefon/Mobitel:

095123456789

Adresa:

Ulica Ante Antića

Grad:

Grad Ante Antića

Poštanski broj:

21000

Država:

Hrvatska

Detalji dostave

Ime i prezime:

Ante Antić

Email:

anteantic@gmail.com

Adresa za dostavu:

Ulica Ante Antića 2

Grad:

Grad Ante Antića 2

Poštanski broj:

25000

Država:

Hrvatska

[Ažurirajte profil](#)

Slika 9: Forma za unos korisničkih podataka

Podatci vezani za dostavu su odvojeni od osobnih podataka unesenih poviše zbog toga što su češće podložniji promjenama i što adresa za dostavu ne mora nužno biti adresa prebivališta. Prilikom obavljanja kupovine, netom prije završetka kupovine korisniku će se prikazati pregled narudžbe i obrazac za unos podataka o dostavi. Ukoliko je te podatke unio ranije ili tijekom registracije, obrazac će se automatski popuniti tim podacima. Nakon toga završava narudžbu plaćanjem preko Stripe [6] platforme gdje unosi kartične podatke (slika 10).

Platite karticom

Adresa e-pošte

anteantic@gmail.com

Podaci o kartici

4242 4242 4242 4242

VISA

05 / 28

123

123

Ime vlasnika kartice

Ante Antić

Zemlja ili regija

Hrvatska



Plati



Powered by **stripe**

Uvjeti

Privatnost

Slika 10: Prikaz Stripe plaćanja

Kada kupac završi kupnju, ukoliko želi na stranici može ocjenjivati i recenzirati kupljene proizvode. Ne može recenzirati one proizvode koje nije kupio jer nije ni ispravno da ocjenjuje nešto što ne posjeduje, te može ostaviti samo jednu recenziju za pojedini proizvod. Na stranici kupljenog proizvoda će se stvoriti botun za recenziranje koji korisnika vodi na formu gdje u tekstualnom okviru upisiva svoj komentar, a ispod okvira odabire ocjenu u obliku zvjezdica. Recenzija ne mora sadržavati komentar, ali mora biti ocijenjena, od 1 do 5, da bi se prikazala na stranici proizvoda. Korisnik će moći vidjeti sve recenzije za pojedini proizvod, ukoliko postoje, a svoje će recenzije moći brisati. Izgled forme za recenziju i ocjenjivanje prikazan je na slici 11.

Recenzija

Recenzija:

Ocjena: ★★★★★ ▾

Slika 11: Forma za recenziranje proizvoda

4.5. Prikaz i mogućnosti aplikacije za administratora

Administrator upravlja sadržajem stranice, dodaje proizvode u pripadajuće kategorije, uređuje ih i briše. Također upravlja korisničkim računima, kao i svim narudžbama. On nema zasebno administratorsko sučelje već su njegove funkcionalnosti u obliku raznih botuna raspoređene po cijeloj stranici. U navigacijskoj traci pak može preko poveznice „Korisnici“ pristupiti svim korisničkim računima koje tu može uređivati ili brisati (slika 12).

Korisnici

User ID	Ime	Prezime	Email	Adresa	
16	Stipe	Stipić	stipestipic@gmail.com		<input type="button" value="Uredi"/> <input type="button" value="Ukloni"/>
14	Marko	Markić	markomarkic@gmail.com		<input type="button" value="Uredi"/> <input type="button" value="Ukloni"/>
4	Ante	Antić	anteantic@gmail.com	Ulica Ante Antića	<input type="button" value="Uredi"/> <input type="button" value="Ukloni"/>
10	Teo	Vukelić	teo.vukelic.ynwa@gmail.com	Ulica 11	Administrator
5	Pero	Perić	peroperic@gmail.com	Ulica Pere Perića	<input type="button" value="Uredi"/> <input type="button" value="Ukloni"/>
12	Ana	Anić	teo.vukelic.zavrzni@gmail.com	Ulica Ane Anić	<input type="button" value="Uredi"/> <input type="button" value="Ukloni"/>

Slika 12: Prikaz svih korisnika

Ako korisnici prilikom ispunjavanja svog profila nisu unijeli određene podatke oni u tablici neće biti prikazani. Administrator klikom na botun „Uredi“ dobiva uvid u osobne podatke svakog korisnika koje ili na vlastitu inicijativu ili na inicijativu korisnika može uređivati. Po potrebi administrator može i brisati korisničke račune. Kôd za brisanje prikazan je u ispisu 10.

```
def admin_web(request):
    if request.method == 'POST':
        if request.POST.get("Delete user"):
            User.objects.get(id=request.POST["Delete
user"]).delete()
            return redirect("admin_web")
```

Ispis 10: Kôd za administratorsko brisanje korisničkog računa

Na početnoj stranici poviše pregleda svih proizvoda administratoru će biti prikazan botun za dodavanje proizvoda, klikom na koji se otvara forma gdje se unose podatci vezani za proizvod, kao što su: ime, cijena, količina, pripadajuća kategorija i slika. Akcijske cijene i popuste može definirati preko iste forme (slika 13).

The image shows a web form for adding a product. It consists of several input fields and a submit button. The fields are: 'Ime' (text input), 'Cijena' (text input), 'Kategorija' (dropdown menu), 'Dodaj sliku: Pregledaj ...' (file upload button), 'Akcijska cijena' (checkbox), 'Akcijska cijena' (text input), 'Količina' (text input), 'Opis' (text input), and a 'Dodaj' (submit) button at the bottom.

Slika 13: Dodavanje proizvoda

Ukoliko želi urediti ili izbrisati određeni proizvod morat će otići na stranicu samog proizvoda gdje će preko botuna moći izvršiti željene radnje. Ispis koda za promjenu proizvoda prikazan je na ispisu 11.

```
def edit_product(request, id):
    if request.method == "POST":
        form = ProductEditForm(request.POST, request.FILES)
        if form.is_valid():
            product = Product.objects.get(id=id)
            image = product.image
            for key,value in form.cleaned_data.items():
                setattr(product, key, value)
            if not product.image:
                product.image=image
            product.save()
            messages.success(request, ("Uspješno ažurirano"))
            return redirect("home")
        else:
            messages.success(request, ("Neuspješno, pokušajte
opet"))
            return render(request, "edit_product.html",
{"form": form})
    else:
        try:
            product = Product.objects.get(id=id)
            form =
ProductEditForm(initial=forms.models.model_to_dict(product))
        except Product.DoesNotExist:
            return redirect("home")
    return render(request, 'edit_product.html', {"form": form})
```

Ispis 11: Kôd za uređivanje proizvoda

Administrator na jednak način kao i prijavljeni korisnik može pregledavati popis narudžbi, poslanih i neposlanih, samo što za razliku od prijavljenog korisnika on ima uvid u narudžbe svih korisnika. Sadržaj stranice detaljnije je opisan na stranici 16 (slika 7). U trenutku kada korisnik završi narudžbu, administratoru će se ona pojaviti u popisu neposlanih narudžbi (slika 14) koju će onda administrator moći označiti kao poslanu kad je obradi. Nakon toga narudžba će biti vidljiva u poveznici „Poslane narudžbe“.

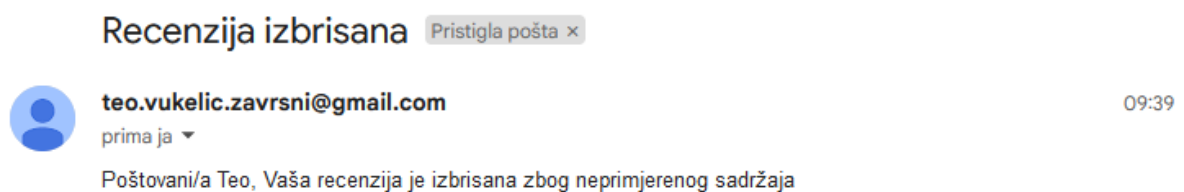
Neposlane narudžbe

Korisnik	Narudžba	Cijena	Email	Datum narudžbe	Status
None	69	201.99 €		Sept. 2, 2024, 2:55 p.m.	Označi poslano
None	64	800.00 €	gost.gost@gmail.com	Sept. 2, 2024, 3:32 a.m.	Označi poslano
ante	63	800.00 €	anteantic@gmail.com	Sept. 2, 2024, 3:29 a.m.	Označi poslano
ante	62	800.00 €	anteantic@gmail.com	Sept. 1, 2024, 11:09 p.m.	Označi poslano
marko	61	437.99 €	markomarkic@gmail.com	Sept. 1, 2024, 1:41 a.m.	Označi poslano

Slika 14: Prikaz neposlanih narudžbi za administratora

Na slici u prvom retku vidimo narudžbu korisnika koji je izbrisao svoj račun nakon čega su izbrisani njegovi korisnički podatci te zbog toga nisu prikazani u tablici, nego je prikazana narudžba vezana za tog korisnika. U drugom retku je prikazana narudžba koju je obavio gost, a ostale narudžbe su vezane za registrirane korisnike.

Na stranici pojedinog proizvoda će u donjem dijelu stranice biti prikazane sve recenzije za taj proizvod i ukupna ocjena. Ukoliko administrator procijeni da je sadržaj određene recenzije neprimjeren može je izbrisati. Pored svake recenzije će biti prikazan botun za brisanje recenzije. Nakon što se recenzija izbriše prikazat će se skočna poruka o uspješnom brisanju. Također nakon brisanja, korisnik čija je recenzija izbrisana će biti upozoren putem elektroničke pošte (slika 15).



Slika 15: Obavijest korisniku o izbrisanoj recenziji putem elektroničke pošte

Dio koda koji obavlja brisanje recenzije i slanje elektroničke pošte prikazan je u ispisu 12.

```
elif request.POST.get("Delete review"):\n    recenzija =\n    ProductReview.objects.get(id=request.POST["Delete review"])\n    if request.user.is_superuser:\n        send_mail("Recenzija izbrisana", "Recenzija izbrisana\n        zbog neprimjerenog sadržaja", settings.EMAIL_HOST_USER,\n        [recenzija.user.email])\n    ProductReview.objects.get(id=request.POST["Delete\n    review"]).delete()\n    messages.success(request, "Review deleted")\n    return redirect(f"/product/{id}")
```

Ispis 12: Isječak koda za brisanje recenzije i slanja elektroničke pošte

5. Zaključak

U ovom završnom radu opisana je izrada web aplikacije kako bi se olakšala prodaja glazbenih instrumenata i opreme. Kupci mogu na jednostavan način pretraživati i pregledavati proizvode putem tražilice, raznih kategorija, filtera i akcijske ponude. S obzirom na sve brži razvoj tehnologije i sve veću osviještenost kupaca o prednostima ovakvog oblika trgovanja, cilj ove aplikacije je bio omogućiti kupcima i ljubiteljima glazbe da na učinkovitiji i moderniji način dođu do željenih proizvoda i tako nastave ili započnu svoje glazbeno putovanje. Kupcima je nakon kupovine omogućeno ostavljanje recenzija čime se daje do znanja da je njihovo mišljenje bitno te da svojim iskustvima mogu pomoći budućim kupcima u donošenju odluka o kupovini pojedinih proizvoda.

Za izradu poslužiteljske strane aplikacije korišteni su Python i njegov razvojni okvir Django koji se zbog svoje jednostavnosti i unaprijed ugrađenih funkcija pokazao kao odličan alat, dok su za klijentsku stranu korišteni HTML i Bootstrap koji je zbog unaprijed pripremljenih predložaka i komponenata omogućio olakšano stiliziranje izgleda stranice.

Iako je aplikacija jednostavna i intuitivna to također ostavlja prostora za dodatna poboljšanja i unaprjeđenja. Potencijalne nadogradnje bi uključivale: praćenje i prikazivanje najprodavanijih proizvoda, praćenje najčešćih kupaca i nagrađivanje vjernosti kuponima za popuste, mogućnost filtriranja proizvoda po brendu te dodavanje podkategorija.

Literatura

[1] Django, „Introduction to Django“, <https://aiprobably.hashnode.dev/introduction-to-django>, (posjećeno 13.08.2024.)

[2] PostgreSQL, „How to use PostgreSQL with Django“, <https://www.enterprisedb.com/postgres-tutorials/how-use-postgresql-django>, (posjećeno 15.8.2024.)

[3] HTML, „HTML Introduction“, https://www.w3schools.com/html/html_intro.asp (posjećeno 17.8.2024.)

[4] Bootstrap, „Bootstrap themes, templates and UI tools to help you started your next project“, <https://startbootstrap.com/template/shop-homepage> (posjećeno 19.8.2024.)

[5] jQuery, „jQuery Tutorial“, <https://www.w3schools.com/jquery/default.asp>, (posjećeno 23.8.2024.)

[6] Stripe, „Stripe-hosted page“, <https://docs.stripe.com/checkout/quickstart?lang=python> (posjećeno 27.8.2024.)