

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 5111

**INTERNETSKA PRODAVAONICA
MOBILNIH UREĐAJA**

Bruno Kantura

Zagreb, lipanj 2017.

Sadržaj

1. Uvod	1
2. Internetska prodavaonica.....	2
2.1 Povijest internetskog poslovanja	2
2.2 Karakteristične značajke internetskih prodavaonica	2
3. Tehnologija	4
3.1 Web-aplikacija	4
3.2 Ruby	5
3.3 Ruby on Rails	6
3.4 HTML, CSS i JavaScript (jQuery)	7
3.5 SQLite i PostgreSQL	8
4. Baza podataka	9
5. Korisničko sučelje prodavaonice	11
6. Administratorsko sučelje prodavaonice	17
7. Implementacija internetske prodavaonice.....	21
7.1 Autorizacija i autentikacija	21
7.2 Košarica	22
7.3 Dinamičko filtriranje	24
7.4 Stripe	25
7.5 Rails Admin	27
7.6 Statistički grafovi	29
8. Zaključak	30
9. Literatura	31

1. Uvod

Internetska prodavaonica (eng. *webshop*) omogućuje svojim posjetiteljima tj. kupcima naručivanje proizvoda putem Interneta. Prodaja putem Interneta smatra se trenutno najisplativijom vrstom trgovine zbog logističke jednostavnosti i malih troškova pokretanja i vođenja.

Internetska prodavaonica najčešće je ostvarena kao web-aplikacija kojom korisnici pristupaju putem internetskog preglednika. Danas su sve češće i prodavaonice ostvarene mobilnim aplikacijama kao alternativa webu, ali uobičajeno mobilne prodavaonice nastaju kod popularnijih web prodavaonica. Uobičajeni slijed kupovine u internetskim prodavaonicama sastoji se od pretraživanja, dodavanja u košaricu, pregleda narudžbe, plaćanja i dostave. Današnje internetske prodavaonice podržavaju sve više načina plaćanja, dok su i dalje najpopularnije kreditne kartice. Česta je i integracija posredničkih rješenja za plaćanje pomoću kojih je moguće u minimalnom roku pouzdano obavljati transakcije. U ovome radu koristi se posredničko rješenje Stripe, no samo u testnom okruženju. Nakon što se izvrši naplata, dužnost je zaposlenika prodavaonice poslati proizvod poštom kupcu i osigurati njegovo zadovoljstvo kupnjom (za uspješno poslovanje). Sve se više ljudi odlučuje na kupnju u internetskim prodavaonicama kako bi uštedjeli vrijeme i lakše pregledali bogati asortiman.

Mobilni uređaji odabrani su kao glavni proizvod prodavaonice zbog karakterističnih značajki po kojima se mogu filtrirati.

Ovaj rad za izradu internetske prodavaonice mobilnih uređaja koristi Ruby on Rails, programski okvir za izradu web-aplikacija napisan u programskom jeziku Ruby. Ruby on Rails jedan je od najpopularnijih programskih okvira (za web-aplikacije) i vrlo je dobro dokumentiran.

2. Internetska prodavaonica

2.1 Povijest internetskog poslovanja

Prvi čin koji bi se mogao nazvati internetskim poslovanjem dogodio se početkom 1970.-ih godina kada su studenti računalne znanosti američkih fakulteta Stanford i MIT dogovarali prodaju kanabisa pomoću ARPANET-a, prve mreže koja implementira protokol TCP/IP za razmjenu paketa i u skoroj budućnosti postaje temelj današnjeg Interneta.

Od tada nastaju brojne ideje i sustavi za internetsko poslovanje, no Internet je nedovoljno razvijen i nesiguran za javno poslovanje, a razvijena rješenja uglavnom koriste tvrtke. Sve do 11. kolovoza 1994. godine kada je prodan prvi proizvod (audio CD engleskog glazbenika „Sting“) na internetskoj trgovini „NetMarket“ koristeći kreditnu karticu. To je prva maloprodajna transakcija preko Interneta koja je koristila dovoljno napredan softver za enkripciju podataka kako bi se garantirala njihova privatnost. Nakon toga slijede vino, čokolada i cvijeće kao jedne od prvih maloprodajnih kategorija koje su drastično ubrzale rast internetskog poslovanja. Statističari utvrđuju da je ključ uspješnog internetskog poslovanja baš u odabiru asortimana za prodaju. Uglavnom se radi o proizvodima koje kupci kupuju bez potrebe da ih prije isprobaju, dodirnu ili osjete.

Danas se u internetskim prodavaonicama može naći širi asortiman nego u konvencionalnim trgovinama, sve od hrane pa do automobila i plovila.

2.2 Karakteristične značajke internetskih prodavaonica

Gotovo svaka internetska prodavaonica sastoji se od nekih karakterističnih značajki koje dijele sve internetske prodavaonice.

Za početak, neophodno je pretraživanje i/ili kategorizacija proizvoda za lakšu navigaciju i pronalazak željenih artikala. Internetska prodavaonica obuhvaćena ovim radom sadrži pretraživanje i filtriranje mobilnih uređaja prema karakterističnim značajkama.

Nakon što korisnik pronađe proizvod za koji je zainteresiran uobičajeno je da ga doda u košaricu. Košarica predstavlja košaricu za kupovinu u konvencionalnim trgovinama u koju se dodaju proizvodi koji se žele kupiti. Vrlo je teško pronaći internetsku prodavaonicu bez nekog oblika internetske košarice.

Neophodno je da korisnik navede adresu na kojoj želi preuzeti narudžbu, ali postoje i prodavaonice koje omogućuju preuzimanje narudžbe u poslovnica.

Provedba plaćanja sljedeći je korak narudžbe kojeg implementiraju sve internetske prodavaonice. Danas postoji širok izbor plaćanja kako bi se privuklo što više potencijalnih kupaca. Plaćanje u većini prodavaonica koristi sigurnosne enkripcijske sustave kako bi se zaštitili korisnički podaci i izbjegla krađa informacija o karticama. Postoje iznimke koje svakako treba izbjegavati jer se najčešće radi o neprovjerenim prodavačima.

Nakon izvršene narudžbe uobičajen je pregled narudžbe i informacije o trenutnom stanju. Često prodavaonice svaku narudžbu šalju uz broj za praćenje pošiljke koji se zatim ažurira za narudžbu te korisnik može pratiti lokaciju njegovog paketa.

Nakon što preuzme narudžbu, uobičajeno je da kupac ima pravo ostaviti svoj dojam za svaki naručeni proizvod. Ocjenjivanje i opis proizvoda nakon što ga prihvati pomaže svim budućim kupcima u odabiru te olakšava kupnju pošto je kod internetskih prodavaonica nemoguće isprobati proizvode prije plaćanja.

Prednosti kupovine internetskom poslovnicom

- Dostupnost – Gotovo svaki proizvod, bilo kojeg proizvođača može se naći u nekoj prodavaonici bilo kada. U svakom dobu dana korisnik može pregledavati asortimane najudaljenijih trgovina i planirati kupovinu.
- Ušteda vremena i novaca – Često je dovoljno samo upisati naziv proizvoda koji se traži i već se može pronaći. Za usporedbu cijena otvara se novi prozorčić u pregledniku s drugom prodavaonicom i uz jednako pretraživanje kupac bira najpovoljniji izbor.
- Promocije i popusti – Pretplatom na email novosti pojedinih prodavaonica stižu email poruke o eventualnim popustima na određene proizvode. Na taj način, dovoljno je otvoriti email sandučić za najbolje ponude.

Nedostaci kupovine internetskom poslovnicom

- Trošak dostave – Iako se rastom internetske prodaje smanjuju cijene poštanskih usluga i dalje se za veće i/ili teže proizvode treba izdvojiti značajan iznos koji čini internetsku kupovinu određenih proizvoda neisplativom.
- Čekanje – Uobičajeno je naručivanje iz udaljenijih prodavaonica (često i iz drugih država) pa je za pretpostaviti da se za udaljenije narudžbe čeka tjednima.
- Kupovanje prije isprobavanja – Kupnjom na Internetu ne mogu se isprobati proizvodi prije plaćanja i dostave tako da nisu rijetki slučajevi nezadovoljstva nakon prvog korištenja proizvoda.

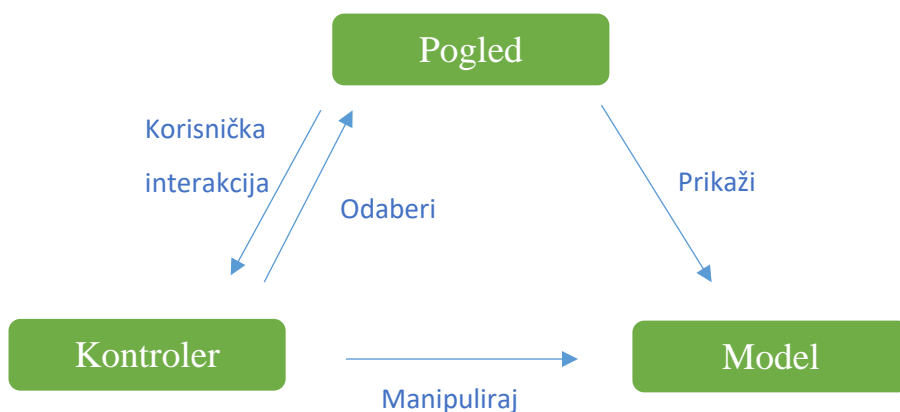
3. Tehnologija

Internetska poslovnica implementirana je kao web-aplikacija koristeći programski okvir za razvoj web-aplikacija Ruby On Rails, razvijen u programskom jeziku Ruby. Uz navedeni programski okvir, koriste se još različite tehnologije karakteristične za specifične dijelove u izradi web-aplikacija.

3.1 Web-aplikacija

Web-aplikacija je klijentsko-poslužiteljska aplikacija u kojoj klijent koristi internetski preglednik kako bi joj pristupio. Takav oblik aplikacija ujedno je i najrašireniji baš zbog svoje dostupnosti u bilo koje vrijeme s bilo kojeg mjesta (tj. uređaja) koji sadrži internetski preglednik. Web-aplikacije široke su primjene i u poslovnom svijetu gdje se koriste kako bi bitni podaci bili sigurni u oblaku i imali mogućnost paralelnog pristupa s raznih lokacija bez potrebe za dodatnim korisničkim softverom.

Većina današnjih aplikacija na poslužiteljskoj strani koristi arhitekturu *Model-View-Controller* (MVC) koja razdvaja samu aplikaciju u tri glavne logičke komponente: model (eng. *model*), pogled (eng. *view*) i kontroler (eng. *controller*). Svaka od tih komponenti građena je tako da pokriva određene razvojne aspekte aplikacije.



Slika 1. Simbolički prikaz MVC-a

Modeli su komponenta aplikacije koja implementira logiku za podatke u aplikaciji. Objekti modela vrlo često dohvaćaju i spremaju stanje modela u bazu podataka. Npr. objekt dohvati informacije iz baze podataka, obrađuje ih, te zatim obrađene informacije sprema u bazu podataka.

Pogledi su komponenta koja prikazuje korisniku aplikacijsko sučelje. Uglavnom se sučelje sastoji od podataka sakupljenih iz modela. Npr. objekt izvađen iz baze podataka prikazuje se korisniku koji ima mogućnost uređivanja njegovih atributa, stoga se sučelje sastoji od ulaznih polja koja je moguće ispuniti.

Kontroleri su komponenta koja obrađuje korisničku interakciju, radi s modelom, te odabire pogled koji će se prikazati korisniku.

MVC arhitektura olakšava posao programeru tako da razdvaja navedene komponente te ubrzava održavanje i pronalaženje grešaka. Također, dozvoljava nezavisni rad na dijelu aplikacije od strane više programera te tako ubrzava sam proces razvitka aplikacije.

3.2 Ruby

Ruby je dinamički, reflektivni, objektno-orijentirani programski jezik opće namjene. Dizajnirao i razvio ga je sredinom 1990.-ih japanski programer i računalni znanstvenik Yukihiro Matsumoto. Prema autoru, Ruby je inspiriran programskim jezicima Perl, Ada, Lisp, Eiffel i Smalltalk kako bi stvorio novi jezik koji uravnotežuje funkcijsko i imperativno programiranje. Ruby je danas smješten među 10 najpopularnijih i najbrže rastućih programskih jezika na svijetu, dok je većina tog uspjeha pribilježena upravo programskom okviru za izradu web-aplikacija – Ruby On Rails.

U Ruby programskom jeziku sve se smatra objektom, stoga se svakom djeliću informacije ili koda mogu se dodati svojstva i akcije. Isječak koda koji prikazuje izraziti objektno-orijentirani pristup jezika Ruby prikazan je u nastavku.

```
5.times { print "We *love* Ruby - it's outrageous!" }
```

Isječak 1. Primjer akcije nad brojem

U brojnim programskim jezicima, brojevi i ostali primitivni tipovi podataka nisu objekti, dok Ruby daje mogućnost metoda i objektnih varijabli nad svim tipovima.

Ruby je smatran fleksibilnim jezikom jer svakom korisniku dopušta slobodno mijenjanje

njegovih dijelova. Osnovni dijelovi jezika Ruby mogu se brisati i mijenjati po želji. Na primjer, moguće je zbrajanje dvaju brojeva izmijeniti te umjesto pretpostavljenog operatora (+) koristiti metodu nad brojem za što je potrebno dodati kod klasi *Numeric* koju nasljeđuju brojevi.

Ruby ima jednostavnu sintaksu u usporedbi s drugim jezicima zbog brojnih ugrađenih funkcionalnosti i metoda koje olakšavaju korištenje početnim i naprednim korisnicima. Radi se na visokoj razini apstrakcije kako bi se povećala produktivnost, te se smatra laganim za naučiti. S druge strane, zbog svih mogućnosti i modificiranja koje pruža programeru, pronalazi primjenu u raznim područjima te je dugotrajan proces potpunog razumijevanja i savladavanja jezika kod naprednijih značajki.

3.3 Ruby on Rails

Ruby on Rails, ili kraće Rails je programski okvir za izradu web-aplikacija napisan u programskom jeziku Ruby. Najveća razlika u usporedbi s ostalim programskim okvirima za razvoj web-aplikacija je brzina i lakoća kojom je moguće izraditi aplikacije. Rails se koncentrira na principe *Convention over Configuration* (CoC) i *Don't Repeat Yourself* (DRY). *Convention over Configuration* (hrv. Konvencija iznad konfiguracije) znači da razvojni programer treba samo napraviti nekonvencionalne aspekte aplikacije, te generalno vodi k manjoj količini koda i ponavljanja. *Don't repeat Yourself* (hrv. Ne ponavljaj se) znači da svaka informacija mora imati jedan i samo jedan prikaz unutar sustava. Ne ponavljajući jednake informacije više puta, kod postaje lakše održiv, nadogradiv i manje podložan greškama.

Rails je, tehnički, biblioteka za Ruby koja proširuje i dodaje nove funkcionalnosti namijenjene za izradu web-aplikacija. U Ruby programskom jeziku, sve takve biblioteke koje sadrže maskirane funkcionalnosti nazivaju se simboličkim imenom *gem*. *Gem* je, dakle, kod napisan u jeziku Ruby, te je izveden za ponovnu i vanjsku upotrebu bez potrebe poznavanja samog izvornog koda.

Rails kombinira jezik Ruby s ostalim tehnologijama za izradu web sadržaja (HTML, CSS i JavaScript) za izradu web-aplikacije koja se pokreće na web-serveru. Pošto se izvršava na web-serveru Rails se smatra poslužiteljskim dijelom aplikacije (engl. *back-end*), dok je dio koji klijent vidi u svom internetskom pregledniku klijentski dio aplikacije (eng. *front-end*).

Rails sadrži alate koji olakšavaju karakteristične zadatke izrade web-aplikacija, te omogućava automatsko generiranje potrebnih datoteka za migracije, modele, kontrolere i poglede.

3.4 HTML, CSS i JavaScript (jQuery)

HTML, CSS i JavaScript služe za oblikovanje, uređivanje i omogućavanje interakcije korisničkog sučelja koje korisnik vidi prilikom korištenja aplikacije u svom internetskom pregledniku.

HTML (eng. *Hypertext Markup Language*) je vrsta jezika za označavanje podataka koji se koristi za izradu web-stranica. HTML opisuje strukturu stranice pomoću različitih raspoloživih elemenata. Elementima se mogu dodavati klase i razni drugi atributi kako bi se na specifične elemente primijenili razni stilovi pomoću stilskog jezika CSS.

CSS (eng. *Cascading Style Sheets*) se, dakle, koristi za opis prezentacije dokumenta napisanog pomoću HTML jezika. CSS je prvenstveno dizajniran kako bi odvojio sadržaj i njegovu prezentaciju (npr. boja, vrsta slova, oblik). Takvo odvajanje olakšava pristup, povećava fleksibilnost i kontrolu te dozvoljava da više HTML dokumenata dijele isti format pisan u CSS jeziku.

JavaScript je programski jezik visoke razine koji ima široku primjenu, ali zajedno s jezicima HTML i CSS upotpunjuje tri temeljne web-tehnologije. Pošto ima široku primjenu, postoje i popularni programski okviri pisani u JavaScriptu (baš kao Rails), ali njegovo korištenje je ipak najzastupljenije za omogućavanje interakcije korisnika, ponašanje određenih elemenata u HTML dokumentu i dinamičko učitavanje sadržaja (Ajax). Pomoću JavaScripta moguće je slati zahtjeve poslužitelju i obrađivati ih bez osvježavanja i tipičnog zahtjeva preglednika.

jQuery je JavaScript biblioteka dizajnirana da olakša skriptiranje HTML dokumenta. Sadrži brojne funkcionalnosti i metode baš za manipulaciju strukture dokumenta i postavljanja događaja na elemente koji se aktiviraju po potrebi i korisničkom interakcijom.

3.5 SQLite i PostgreSQL

Relacijska baza podataka tip je baze podataka temeljen na relacijskom modelu podataka. U relacijskom modelu podaci se organiziraju u skup relacija. Relacija je skup n-torki (n-torka – skup parova oblika *atribut:vrijednostAtributa*) s istim atributima koji poprimaju vrijednosti iz istih domena (definiranih relacijskom shemom). Ciljevi relacijskog modela su osigurati visok stupanj nezavisnosti podataka, postaviti temelje za rješavanje problema semantike, konzistentnosti i redundancije podataka te omogućiti razvoj jezika za upravljanje podataka temeljenih na operacijama nad skupovima (npr. jezik SQL).

Relacijska baza podataka koristi SUBP – sustav za upravljanje bazom podataka koji omogućava definiciju, pretraživanje, ažuriranje i administraciju iste (eng. RDBMS – *Relational database management system*). SQLite i PostgreSQL primjeri su takvih sustava te koriste jezik SQL za ostvarivanje svojih funkcionalnosti.

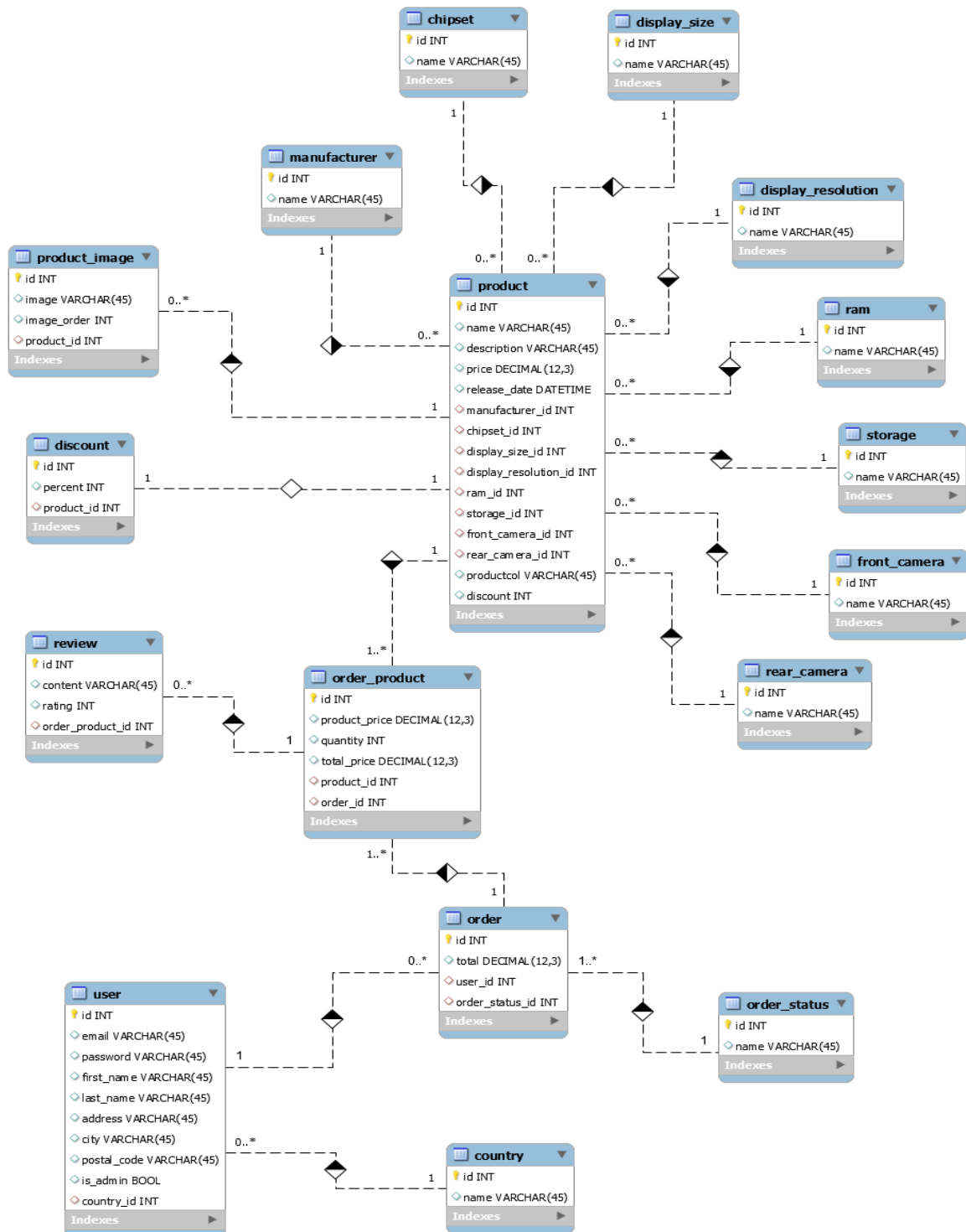
SQLite je SUBP koji se razlikuje od ostalih po tome što nema zasebni serverski proces. SQLite čita i piše direktno u obične podatkovne datoteke. Potpuna relacijska baza podataka s više tablica, indeksa i okidača nalazi se u samo jednoj datoteci na tvrdom disku. Ta datoteka baze podataka je platformno neovisna, te se može slobodno dijeliti među sistemima s različitim arhitekturama.

PostgreSQL je napredni SUBP koji je sposoban savladavati mnoge zadatke vrlo efikasno. Podrška za paralelnost je ostvarena bez zaključavanja kod čitanja i PostgreSQL pruža podršku za programabilnost i nadograđivanje proizvoljnim procedurama. Takve procedure se rade za olakšavanje ponavljajućih, zahtjevnih i često potrebnih operacija baze podataka.

Rails pruža podršku za oba tipa upravljanja baze podataka, a internetska prodavaonica opisana ovim radom koristi SQLite u razvojnom okruženju te PostgreSQL u produkcijskom okruženju. SQLite je dobar izbor za razvojno okruženje jer nitko osim razvojnog programera ne treba pristup bazi i nisu potrebne napredne funkcionalnosti PostgreSQL-a. Osim toga, vrlo je brzo postavljanje i početak programiranja kod korištenja SQLite baze i ne zahtijeva ništa osim jedne datoteke na disku.

4. Baza podataka

Model baze podataka internetske prodavaonice prikazan je na slici 2., a opis slijedi u nastavku. Model osim tablica entiteta prikazuje i veze (eng. *relationship*) između njih na koje je potrebno obratiti pozornost (ER model – eng. *entity-relationship model*).



Slika 2. ER model baze podataka internetske prodavaonice mobilnih uređaja

Baza podataka internetske prodavaonice mobilnih uređaja temelji se na tablicama proizvoda (eng. *product*), narudžbe (eng. *order*) i korisnika (eng. *user*). Kod proizvoda koriste se strani ključevi za povezivanje s pojedinim karakteristikama. Tablice karakteristika sastoje se od proizvođača (eng. *manufacturer*), čipseta (eng. *chipset*), veličini i razlučivosti ekrana (eng. *display size/resolution*), RAM memorije, memorije za pohranu podataka (eng. *storage*), te razlučivosti prednje (eng. *front camera*) i zadnje (eng. *back camera*) kamere. Referenciranje stranim ključem omogućava administratoru lakše stvaranje novih uređaja (tj. proizvoda) tako da aplikacija pri odabiru pojedine karakteristike ponudi sve trenutne zapise te karakteristike u bazi podataka. Nadalje, znatno se ubrzava filtriranje po pojedinim karakteristikama, štedi memorijski prostor te olakšava i unapređuje izrada statistika u odnosu na spremanje karakteristika u tablici proizvoda.

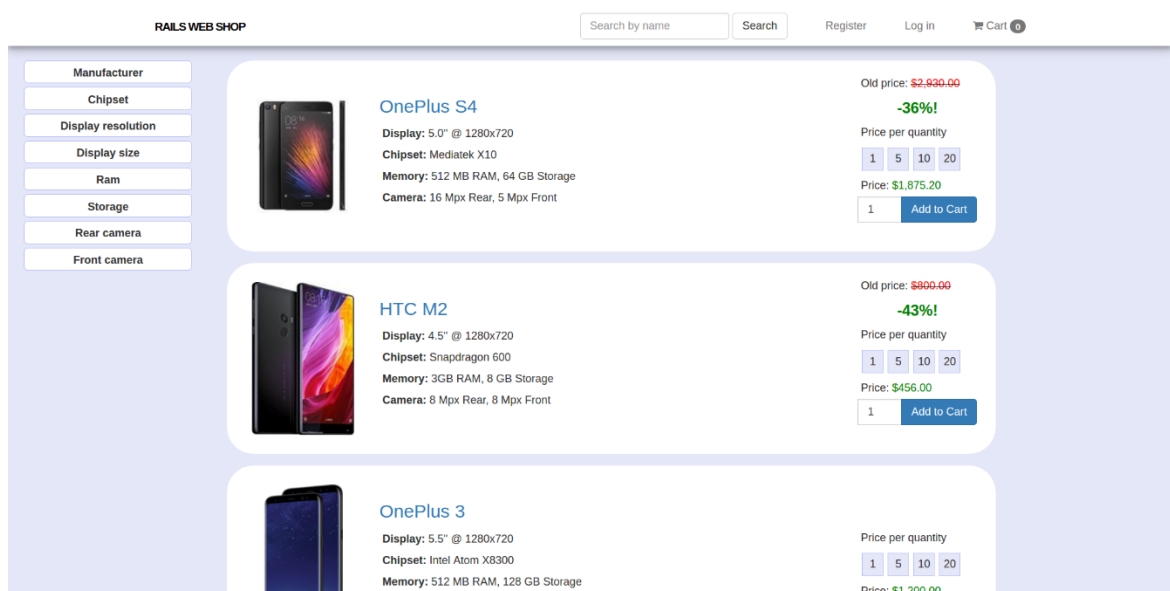
Svaki proizvod može imati 0 ili više pridijeljenih slika pri čemu svaka sadrži atribut „image_order“ kojim se određuje redni broj prikaza slike. Svaka slika također sadrži atribut „image“ koji sprema poveznicu na lokaciju slike u poslužitelju.

Narudžbe i proizvodi povezani su *many to many* vezom, što u slobodnom prijevodu znači da svaka narudžba može sadržavati više proizvoda, ali i svaki proizvod može se nalaziti u više narudžbi. Kod relacijskih baza podataka nužno je definirati novu tablicu „order_product“ kao takvu vezu. U toj tablici također se pohranjuju informacije poput cijene proizvoda u toj narudžbi, količina pojedinog proizvoda u narudžbi te izračunata ukupna cijena proizvoda (količina * cijena pojedinačno). Nužno je kopirati ove podatke u ovu tablicu jer velika je vjerojatnost da će se u budućnosti cijene proizvoda mijenjati, a da podaci nisu kopirani u vrijeme kupnje, sve prethodne narudžbe s proizvodom kojem se mijenja cijena bi bile izmijenjene (iako je plaćen točan iznos). Slično vrijedi i za tablicu narudžbi koja sadrži atribut za ukupnu cijenu narudžbe. Tablica „order_product“ također je povezana tablicom „review“ (hrv. osvrt) koja predstavlja osvrte korisnika. Takva kombinacija je važeća jer korisnik ima pravo svakom narudžbom ostaviti jedan osvrt za svaki naručeni proizvod, što upravo reprezentira „order_product“ (proizvod u narudžbi).

Kod korisnika se u bazi podataka evidentiraju više-manje standardni podaci od kojih je dobro istaknuti atribut „is_admin“ koji izvršuje podjelu između običnih korisnika i korisnika s administratorskim pravima (više o autorizaciji u poglavlju 7.1).

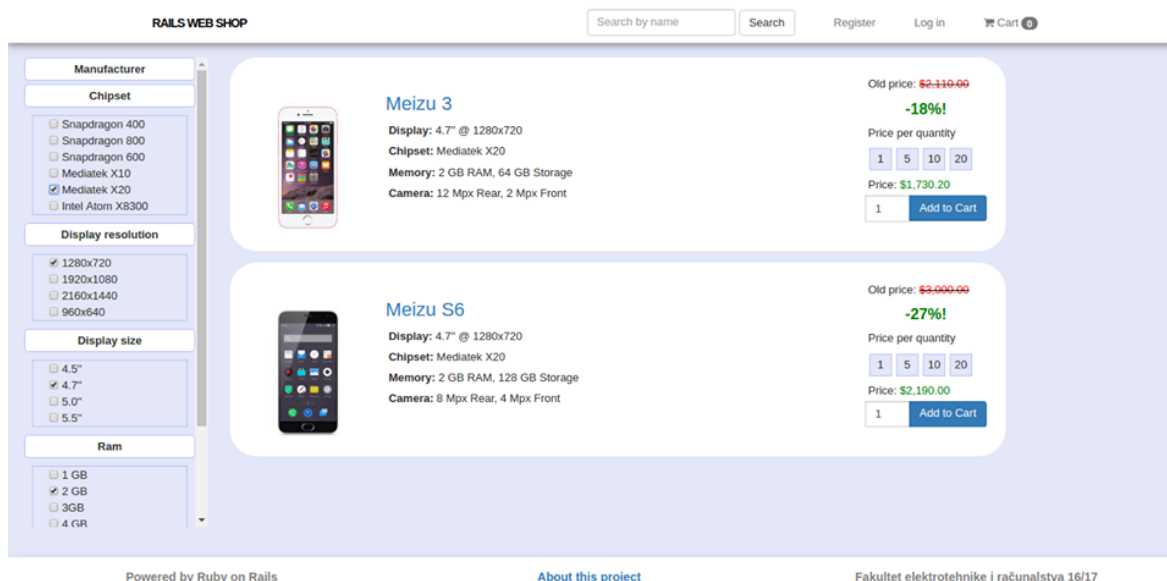
5. Korisničko sučelje prodavaonice

Korisnik pristupom prodavaonici dolazi do glavnog asortimana mobilnih uređaja gdje se također nalazi mogućnost filtriranja uređaja. Na vrhu stranice korisnik će pronaći navigacijsku traku koja pruža različite mogućnosti i koja je raspoloživa na svim stranicama internetske prodavaonice. Moguće je pretraživati prodavaonicu prema imenima proizvođača i mobilnih uređaja, dostupne su funkcionalnosti registracije i prijave u prodavaonicu, te pregled trenutne kupovine (tj. pregled statusa košarice) odabirom na košaricu. Također je klikom miša na logo prodavaonice moguće posjetiti početnu stranicu s mobilnim uređajima.



Slika 3. Početna stranica s navigacijskom trakom

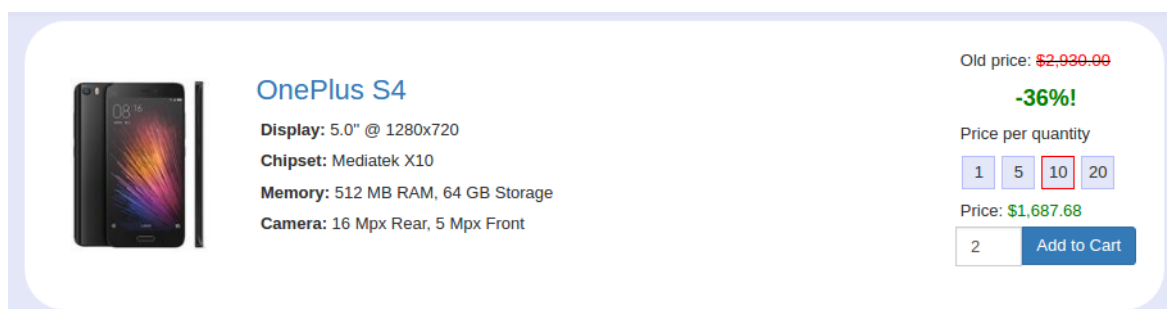
Filtriranje mobilnih uređaja moguće je po proizvođaču, čipsetu, rezoluciji i dijagonali ekrana, količini RAM memorije i pohrane, te rezoluciji prednje i stražnje kamere. Klikom na bilo koju od tih opcija filtriranja, ladica s filterima se proširuje te ispisuje raspoložive filtere. Korisnik zatim označuje koji filteri ga zanimaju te se stranica osvježava dinamički (implementacija opisana u narednim poglavljima). Rezultat filtriranja je prikazan na slici 4.



Slika 4. Odabir filtera i rezultat filtriranja (2 uređaja)

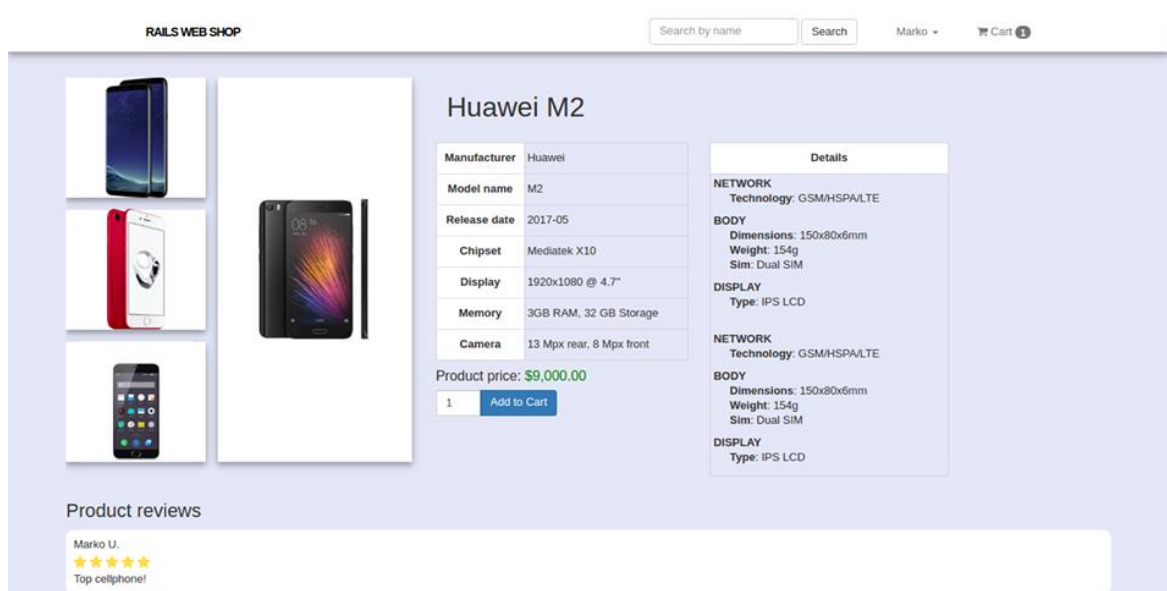
Filtriranjem se također može nazvati i pretraživanje po imenu koje uzima korisnički unos kao filter po imenu proizvođača i imenu uređaja. Na primjer, kada bi na slici 4. još u polje *Search* korisnik unio „S6“ ili „3“ preostao bi samo jedan uređaj čije ime odgovara traženom pojmu.

Prikaz svakog pojedinog uređaja sastoji se od slike uređaja, imena proizvođača i uređaja, te ostalim osnovnim karakteristikama po kojima se izvršava filtriranje. Na desnoj strani prikaza uređaja nalazi se iznos popusta (ako popust postoji za taj uređaj), te cijena uređaja prije popusta. Zatim korisnik može pritiskom na polja za količinu vidjeti cijenu koja će biti uzeta u obzir ako se odluči za kupovinu određene količine. Različiti popusti postoje za različite količine (od 5 do 20%). Na prikazu uređaja također se nalazi i polje za unos količine koju korisnik želi dodati u košaricu, te i gumb kojim se proizvod dodaje u košaricu u navedenoj količini.



Slika 5. Prikaz mobilnog uređaja

Korisnik klikom na ime uređaja dolazi do detaljnog prikaza uređaja s dodatnim informacijama. Ovdje se nalaze sve detaljne karakteristike uređaja koje bi mogle zanimati korisnika zajedno s galerijom slika koje korisnik ima mogućnost pregledavati za svaki uređaj (Slika 6.). Klikom na bilo koju od raspoloživih slika iz galerije, otvara se prozorčić s povećanom slikom (Slika 7.) te je moguć dinamički pregled čitave galerije pojedinog uređaja. Na dnu stranice nalaze se osvrti korisnika za odabrani uređaj, te njihova ocjena. Korisnici tj. kupci koji se odluče za kupovinu u poslovnici nakon primitka uređaja imaju pravo izraziti javno mišljenje te budućim korisnicima olakšati izbor kod kupovine. Ovdje se također nalazi gumb za dodavanje u košaricu koji funkcionira jednako kao i na listi svih uređaja.

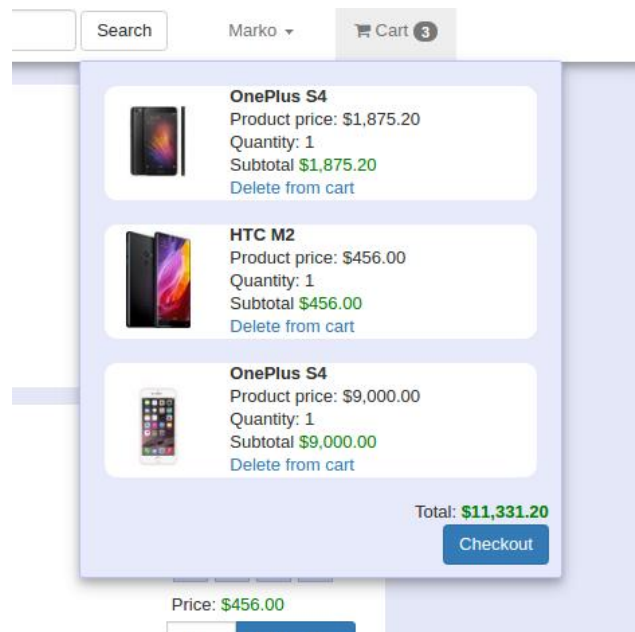


Slika 6. Detaljan prikaz uređaja



Slika 7. Otvorena slika iz galerije

Kao što je ranije spomenuto, korisnik u svakom trenu ima pristup navigacijskoj traci, a time i svojoj košarici. Klikom na košaricu otvara se prozorčić koji prikazuje sve uređaje koje je korisnik dodao u košaricu zajedno sa individualnim cijenama, te ukupnim iznosom uređaja iz košarice. Korisnik može brisati uređaje iz košarice ili, ako je zadovoljan izborom, može nastaviti svoju kupovinu te pripremiti narudžbu za idući korak.

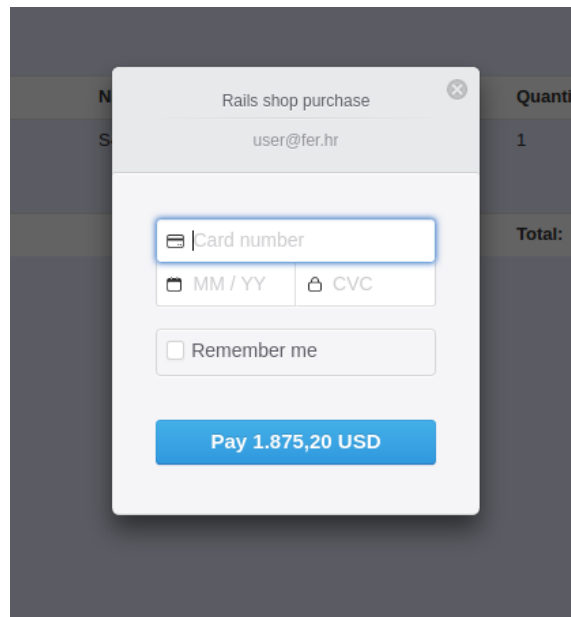


Slika 8. Prikaz košarice

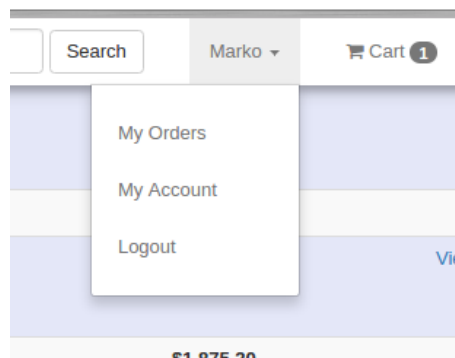
Do sada, sve prikazane funkcionalnosti i korisnička sučelja dostupna su svima, uključujući i neregistrirane korisnike. Idući korak, nakon košarice, je pregled trenutne narudžbe te mogućnost plaćanja kreditnom karticom. Ovom koraku narudžbe mogu pristupiti samo registrirani i prijavljeni korisnici. U suprotnom, aplikacija preusmjerava neprijavljenog korisnika na stranicu za prijavu gdje se može odlučiti za registraciju ili prijavu.

Klikom na „Pay with card“ otvara se Stripe (više u poglavlju 7.4) prozorčić u kojem korisnik unosi svoje podatke s kreditne kartice te provodi plaćanje.

Ako korisnik nije do sad podesio informacije adresi na koju želi primiti narudžbu, aplikacija će ga preusmjeriti na uređivanje vlastitog profila gdje može podesiti osobne podatke uključujući državu, grad, adresu i poštanski broj. U suprotnom, ako te informacije već postoje narudžba će biti zaprimljena te će korisnik moći vidjeti zaprimljenu narudžbu u izborniku svojeg računa.



Slika 9. Provedba plaćanja

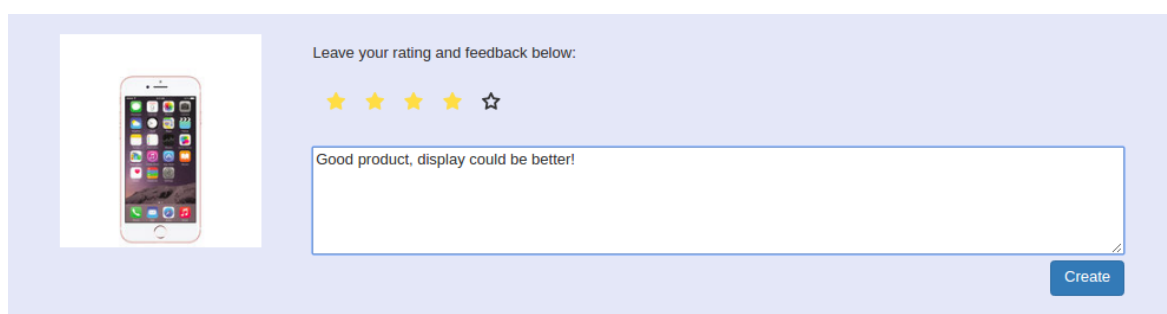


Slika 10. Korisničke opcije

Izbornik korisničkog računa prikazan je na slici 10., te se sastoji od pregleda korisničkih narudžbi (eng. *My Orders*), pregled i uređivanje korisničkog profila (eng. *My Account*) i odjave iz poslovnice klikom na gumb *Logout*. Ovaj izbornik prikazuje se samo registriranim i prijavljenim korisnicima, dok se ostalima na mjestu izbornika nalaze poveznice za registraciju i prijavu.

Klikom na pregled narudžbi otvara se stranica s popisom svih narudžbi trenutnog korisnika. Na toj stranici korisnik može pregledavati status svake narudžbe odnosno vidjeti vrijeme zadnje promjene statusa (npr. kad je narudžba poslana) pojedine narudžbe. Također, za svaku narudžbu postoji detaljni ispis na kojem je moguće vidjeti koji svi proizvodi su naručeni tom narudžbom. Ukoliko je narudžba uspješno provedena, te je administrator to potvrdio mijenjanjem statusa u izvršeno, korisnik dobiva opciju za

ostavljanje osvrta za svaki uređaj koji je uključen u narudžbu. Osvrt se ostavlja klikom na *Leave feedback* (hrv. ostavi osvrt), unutar detaljnog ispisa proizvoda narudžbe. Zatim se otvara nova stranica na kojoj korisnik ocjenjuje svoje zadovoljstvo kupljenim uređajem ocjenama od 1 (najlošije) do 5 (najbolje) simbolički označavajući zvjezdice. Na toj stranici korisnik ima mogućnost riječima izraziti svoje (ne)zadovoljstvo tim uređajem što će budući korisnici moći pogledati kao što je ranije prikazano na slici 6.



Leave your rating and feedback below:

★ ★ ★ ★ ☆

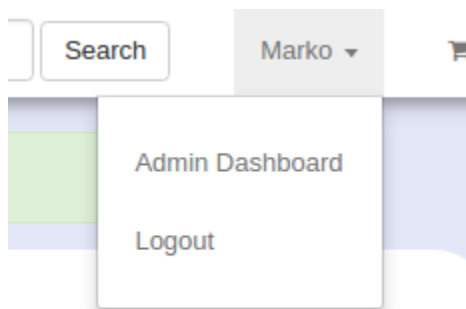
Good product, display could be better!

Create

Slika 11. Dodavanje osvrta

6. Administratorsko sučelje prodavaonice

Osim običnih korisnika aplikacije (tj. kupaca), prodavaonica ima svoje administratore. Administratori imaju mnoge mogućnosti koje su centralizirane na administratorskoj upravljačkoj ploči. Administratorskoj ploči imaju pristup samo korisnici koji su administratori, te u tom slučaju umjesto korisničkog izbornika prikazanog na slici 10. će imati samo poveznicu na administratorsku upravljačku ploču i odjavu iz prodavaonice.



Slika 12. Administratorski izbornik u navigacijskoj traci

Upravljačka ploča sastoji se od glavne stranice s prikazanim bitnim informacijama o nedavnim narudžbama i od izbornika za upravljanje zapisima u bazi podataka.

Preko izbornika se mogu odabrati narudžbe, popusti, uređaji i korisnici. Nakon odabira otvara se ispis svih zapisa iz baze podataka odabranog skupa. Na tom ispisu također postoje opcije kod svakog zapisa za njegovo uređivanje pri čemu se otvara obrazac koji je prikazan na slici 13. te se ispunjuje i kod dodavanja novog zapisa u bazu prodavaonice. Na taj način, uređivanjem zapisa, administrator ima mogućnost mijenjati pojedine podatke o bilo kojem zapisu iz navedenih skupova što uključuje neophodno mijenjanje statusa narudžbi po potrebi. Do obrasca za novi zapis može se doći klikom na *Add new* (hrv. dodaj novi) prilikom ispisa pojedinog skupa.

Products

New

List

Add new

Export

Full name

Optional.

Name

Optional.

Manufacturer

+ Add a new Manufacturer

Edit this Manufacturer

Required.

Price

Optional.

Description

A Normal text Bold Italic Underline Small " " Bulleted Numbered Bulleted Numbered Link Image

Slika 13. Dio obrasca za novi uređaj

Na glavnoj stranici postoje 4 moguća prikaza.

Prvi prikaz (*Placed*, hrv. predano) ispisuje sve predane i plaćene narudžbe kupaca koje su spremne za dostavu. U toj tablici administrator vidi iznos narudžbe te osnovne informacije o kupcu, kao i prečicu do uređivanja svake narudžbe. Administratorov je posao pripremiti uređaje i isporučiti ih poštanskoj službi, a zatim ažurirati status narudžbe u *Shipped* (hrv. poslano).

Placed Orders

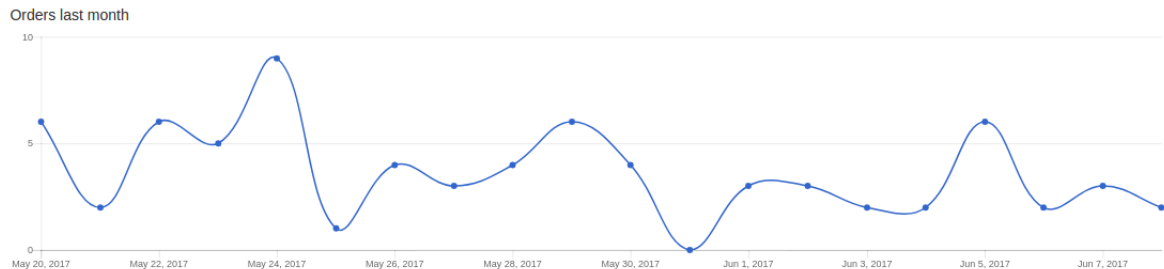
Placed					
Shipped	Reviews	Statistics			
ID	Time placed		User	Total	
151	2017-06-18 10:53:36		user@fer.hr	\$2,340.00	Edit
41	2017-06-08 16:02:11		user6@fer.hr	\$10,000.00	Edit
54	2017-06-07 16:02:12		user0@fer.hr	\$4,510.00	Edit
125	2017-06-04 16:02:17		user17@fer.hr	\$35,220.00	Edit
138	2017-06-03 16:02:17		user5@fer.hr	\$1,600.00	Edit
148	2017-05-28 16:02:18		user16@fer.hr	\$2,400.00	Edit

Slika 14. Prikaz tablice *Placed*

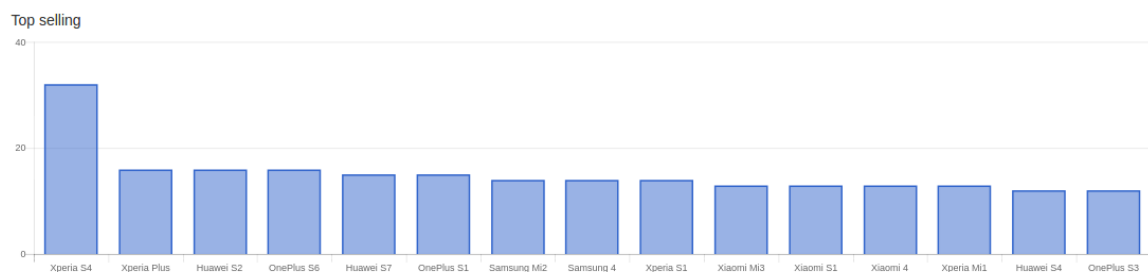
Drugi prikaz (*Shipped*) prikazuje narudžbe koje su isporučene te čekaju isporuku. Ovaj prikaz ne razlikuje se od ranije navedenog osim što administrator prilikom provjere isporuke s poštanskom službom mijenja status u *Completed* (hrv. završeno) čime je narudžba završena.

Treći prikaz (*Reviews*, hrv. osvrti) ispisuje sve negativno ocijenjene (ocjene 1 i 2) osvrte neovisno o uređajima ili narudžbama. U tablici se nalazi i prečica do uređivanja gdje administrator može ažurirati sadržaj osvrta ako je korisnik ostavio lažne informacije ili je problem riješen s korisnikom. Dužnost je administratora razriješiti sve moguće probleme s uređajima ili narudžbama.

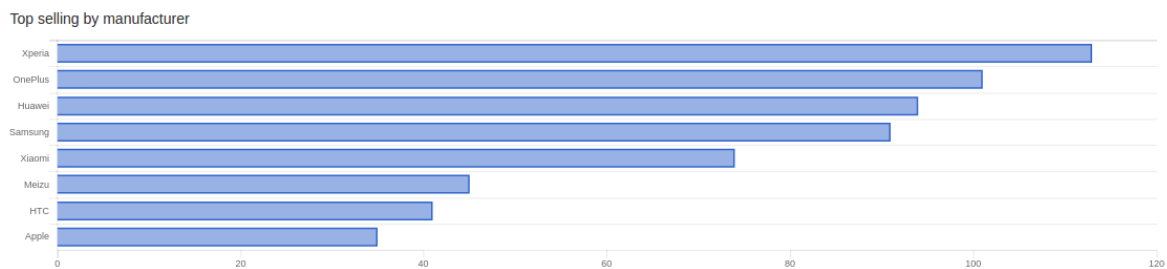
Posljednji prikaz (*Statistics*, hrv. statistike) sadrži grafove koji prikazuju poslovanje prodavaonice. Mogu se vidjeti dva grafa s količinom narudžbi u vremenskim razdobljima od tjedan i mjesec dana. Slijede grafovi o količini najprodavanijih uređaja, najprodavanijih uređaja po proizvođaču, najmanje prodavani uređaji, te države s najvećim brojem narudžbi.



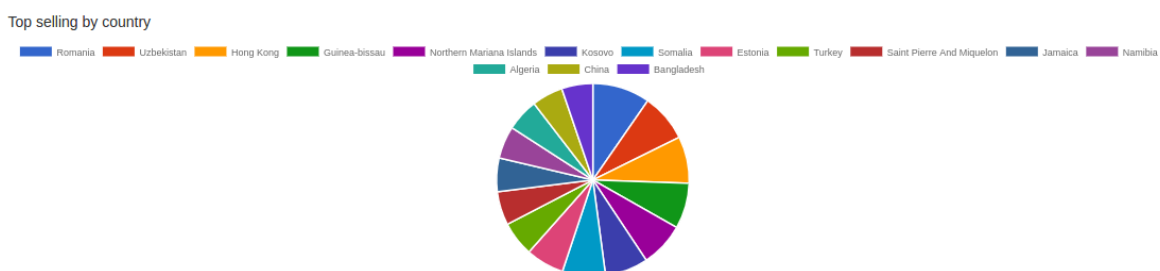
Slika 15. Narudžbe protekli mjesec



Slika 16. Najprodavaniji uređaji



Slika 17. Najprodavaniji uređaji po proizvođaču



Slika 18. Najprodavaniji uređaji po državi

7. Implementacija internetske prodavaonice

7.1 Autorizacija i autentikacija

Za potrebe jednostavne autorizacije i autentikacije koju zahtjeva ova internetska prodavaonica korišten je *gem* za autentikaciju imena Devise. Devise je daleko najkorištenije i najfleksibilnije rješenje autentikacije okvira Rails. Korištenjem *gema* Devise generiraju se kontroleri i pogledi za registraciju i prijavu Devise modela. Svaki kontroler i pogled je u potpunosti otvorenog tipa, te je moguće mijenjati kod uz bogatu dokumentaciju sa službenih stranica.

Generirani Devise modeli razlikuju se od običnih Rails modela po tome što imaju konfiguracijske opcije za Devise koje označavaju koji se modeli mogu stvarati registracijom i koristiti u prijavi korisnika.

Po pretpostavljenim postavkama, Devise modeli koriste samo email ili korisničko ime i lozinku kod registracije. Kako bi dodali dodatne parametre za stvaranje modela, potrebno je dodati akcije koje se izvršavaju prije registracije i ažuriranja korisničkog profila. Isječak 2. prikazuje kako je dovoljno kao argumente navesti imena svih željenih parametara koji se podudaraju sa identifikacijskim ključem u HTML kodu pogleda (email i lozinka su uključeni).

```
before_action :configure_sign_up_params, only: [:create]
before_action :configure_update_params, only: [:update]

protected
def configure_sign_up_params
  devise_parameter_sanitizer.permit(:sign_up, keys: [:first_name,
:last_name])
end

def configure_update_params
  devise_parameter_sanitizer.permit(:account_update, keys: [:first_name,
:last_name, :country_id, :postal_code, :address, :city] )
end
```

Isječak 2. Prilagodba parametara kod registracije i uređenja profila korisnika

Pogledi za registraciju, prijavu i uređivanje Devise modela moraju se podudarati s parametrima koji su dozvoljeni. Pogledi su automatski generirani za pretpostavljene parametre te je potrebno dodati željene parametre.

Nakon stvaranja modela i prilagođavanja kontrolera i pogleda kako bi zadovoljili potrebe prodavaonice, Devise pruža nekoliko korisnih pomoćnih metoda i filtera za kontrolere.

- `before_action :authenticate_user!`

Filter se provodi prije odabranih akcija kontrolera (u ovom slučaju svih) koji provjerava da li je korisnik (model imena *user*) prijavljen prije nego što izvrši bilo koju akciju.

- `user_signed_in?`

Pomoćna metoda kontrolera koja vraća *true* ako je korisnik (model imena *user*) prijavljen odnosno *false* ako korisnik nije prijavljen.

- `current_user`

Pomoćna metoda kontrolera koja dohvaća trenutno prijavljenog korisnika (model imena *user*) u objekt te dozvoljava uobičajenu manipulaciju njime.

Autorizacija se koristi samo kod pristupa administratorskoj upravljačkoj ploči poslovnice te administratori koriste isti model kao i korisnici. Administratori svoje dodatne ovlasti ostvaruju preko atributa „*is_admin*“ koji je *true* za sve administratore, a *false* za korisnike. Jednostavna provjera da li je korisnik administrator ostvaruje se filterom prije svih akcija u administratorskom kontroleru. Drugi način izvedbe hijerarhije autorizacije mogao se ostvariti generiranjem novog modela za administratore te tada administrator ne bi bio i korisnik (u logičkom smislu). U tom slučaju filter bi tražio trenutno prijavljenog administratora a ignorirao korisnika.

7.2 Košarica

Kod implementacije košarice za prednarudžbe prednjači jQuery kako bi korisničko iskustvo u dodavanju i brisanju proizvoda iz košarice bilo što ugodnije. Kako postoje 2 osnovne akcije kod proizvoda u košarici (dodavanje i brisanje), kod svakog treba naznačiti da se ne koristi standardna MVC arhitektura nego da se zahtjevi obrađuju putem eksternih (eng. *remote*) JavaScript zahtjeva i osvježavaju sadržaj košarice bez potrebe za ponovnim učitavanjem čitave stranice. Kod dodavanja proizvoda koristimo formu pošto imamo parametar za količinu proizvoda koju želimo dodati, dok je kod brisanja dovoljno na poveznici označiti da se radi o *remote* zahtjevu. Kada se koristi *remote* zahtjev, Rails traži JavaScript datoteku odgovarajućeg imena u direktoriju s pogledima za pripadajući model. Najbitniji model košarice je već opisan (u opisu baze podataka) „Order product“ koji predstavlja vezu između proizvoda i narudžbe (količinu, cijenu, osvrt). Kod dodavanja u košaricu, stvara se novi „Order product“ koji je povezan s dodanim proizvodom i

trenutačnom narudžbom. Trenutačna narudžba pohranjena je u korisničkoj sesiji odnosno u bazi podataka ako je korisnik prijavljen. Narudžba se sprema u bazu kako bi se po ponovnoj prijavi korisnika mogla obnoviti te korisnik ne bi ponovno trebao dodavati ranije dodane proizvode. Maksimalan broj narudžbi u tijeku (narudžbi u košarici) je 1, te takva vrsta narudžbe (preciznije, prednarudžba) koristi se isključivo za proizvode u košarici i nije vidljiva u popisu provedenih korisničkih narudžbi. Brisanjem iz košarice, briše se „Order product“ entitet. Rails za *remote* zahtjeve koristi poseban oblik JavaScripta s ugrađenom podrškom i za Ruby. Tako je moguće kombinirati dva jezika za što širu primjenu.

Isječkom 3. prikazano je osvježavanje sadržaja košarice. Prvim retkom u element klase „shopping-cart“ (odnosno košaricu) učitava se novi HTML kod koji je definiran Rails naredbom za prikaz odsječka HTML koda naziva „shopping_cart“ odnosno predefinirana struktura košarice u kojoj su kontrolerom osvježeni podaci o proizvodima. Drugi redak obnavlja prikaz količine proizvoda u košarici u navigacijskoj traci. Identičan kod koristi se i za osvježavanje prikaza kod brisanja proizvoda iz košarice.

```
$("#shopping-cart").html("<%= escape_javascript(  
render 'carts/shopping_cart' ) %>");  
$("#product-count").html("<%= current_order.product_count %>");
```

Isječak 3. Osvježavanje sadržaja košarice

S druge strane, neki dio aplikacije mora brisati i stvarati nove „Order product“ objekte i to je upravo – kontroler. Kao i kod MVC karakterističnih zahtjeva, *remote* zahtjeve obrađuje nadležni kontroler. Kod kontrolera je poprilično jednostavan te barata trenutnom narudžbom iz košarice kojoj stvara nove „Order product“ objekte i sprema ih u bazu ili već postojeće briše. Isječak 4. prikazuje kako se dohvaća trenutna narudžba koju prikazuje košarica.

U navedenom isječku prikazuje se postupak traženja narudžbe u sesiji te dohvat iste ako postoji. U suprotnom, pregledava se trenutno prijavljeni korisnik i dohvaća se njegova posljednja narudžba. Ako ta narudžba predstavlja nedovršenu narudžbu (status 1), dohvaća se iz baze i korisnik u novoj sesiji ima pristup staroj košarici. U suprotnom vraća se nova narudžba.

```

def current_order
  if !session[:order_id].nil?
    Order.find(session[:order_id])
  elsif !current_user.nil?
    last_order = current_user.orders.last
    unless last_order.nil?
      if last_order.order_status.id == 1
        return last_order
      end
    end
    Order.new(order_status_id: 1)
  else
    Order.new(order_status_id: 1)
  end
end
end

```

Isječak 4. Dohvat trenutne narudžbe

7.3 Dinamičko filtriranje

Dinamičko filtriranje iduća je i posljednja funkcionalnost koja koristi *remote* način osvježavanja sadržaja. Prvo se dinamički popunjava odjeljak koji sadržava sve filtere kojem je dovoljno samo u polje modela upisati imena modela (npr. *manufacturer*, *chipset*, *ram* itd.) koje se želi uključiti u tablicu filtera. Iz baze se dohvaćaju svi zapisi tog modela te se popunjava HTML dokument listama koje sadrže nazive zapisa i element za korisnički unos odnosno kvačicu.

Korisnik klikom na kvačicu aktivira jQuery događaj prikazan isječkom 5. koji se nalazi u HTML dokumentu početne, indeksne stranice. Nakon aktivacije događaja provjerava se da li je kvačica postala aktivna ili je deaktivirana. U slučaju da je kvačica deaktivirana, pretražuju se elementi HTML dokumenta tražeći skriveno polje istog imena kao i kvačice. Skriveno polje omogućava zadržavanje željenih filtera prilikom mijenjanja stranica te se deaktivacijom briše kao što isječak prikazuje.

```

$(function() {
  $('input[type=checkbox]').change(function() {
    if (!$(this).is(':checked')) {
      var lf_value = $(this).attr('value');
      $("input").filter(function () {
        return this.value === lf_value &&
          $(this).attr('type') === 'hidden';
      }).remove();
    }
    $.get($('#product_filter').attr('action'),
      $('#product_filter').serialize(), null, 'script');
  });});

```

Isječak 5. Događaj nad kvačicama za filtriranje

Nakon deaktivacije kvačice i brisanja povezanog skrivenog polja ili nakon aktivacije kvačice slijedi *remote* zahtjev nad formom za filtriranje proizvoda (ona sadrži sve ranije spomenute kvačice) kojem se kontroleru šalju serijalizirane postavljene kvačice u obliku parametara. U kontroleru se nad svakim parametrom (postavljenom kvačicom) provodi dio koda definiran u pripadnom modelu koristeći Rails „scoping“ prikazan narednim isječkom.

```
scope :with_name, -> (name) { joins(:manufacturer)
  .where("products.name || ' ' || manufacturers.name
        || ' ' || products.name
        LIKE ?", "%#{name.downcase}%") }

scope :with_manufacturer, -> (manufacturer) { joins(:manufacturer)
  .where('manufacturers.name' => manufacturer) }

scope :with_chipset, -> (chipset) { joins(:chipset)
  .where('chipsets.name' => chipset) }
```

Isječak 6. Rails *scope* isječci

Svaki od zapisa u prethodnom isječku predstavlja tzv. *scope* odnosno pretragu baze podataka kratkim lambda izrazom koristeći pri tom predane parametre koje korisnik postavlja aktivacijom kvačica. Postoji po jedan *scope* za svaki model po kojem se filtrira (*manufacturer*, *chipset*, *ram* itd.), te također za pretraživanje po nazivu proizvoda što je također prikazano na početku isječka 6. Uglavnom se radi o prirodnom spajanju određenih tablica baze s postavljenim uvjetom na ime. Isječak 6. nalazi se u modelu proizvoda (model *product*) te se spajanje metodom *joins* poziva nad tablicom *product*.

7.4 Stripe

Za korištenje simboličkog plaćanja u sklopu ovog rada korištena je obrada plaćanja imena Stripe istoimene američke tvrtke. Nakon kratke integracije u aplikaciju, Stripe omogućava brzo plaćanje kreditnim i debitnim karticama. Stripe podržava većinu najpopularnijih programskih okvira za izradu web-aplikacija, te je u sklopu Rails okvira dostupno kao *gem*.

Za početak korištenja Stripe plaćanja u Rails okviru, potrebno je instalirati istoimeni *gem* i napraviti novi kontroler koji će obrađivati naplatu.

```

def create
  if current_user.address.blank? ||
    current_user.postal_code.blank? ||
    current_user.city.blank? ||
    current_user.country.blank?
    flash[:error] =
      "You have to setup your shipping
        information before placing an order."
    redirect_to(edit_user_registration_path)
  else
    @order = Order.find(params[:order_id])

    customer = Stripe::Customer.create(
      :email => current_user.email,
      :source => params[:stripeToken]
    )

    charge = Stripe::Charge.create(
      :customer => customer.id,
      :amount => (@order.total * 100).to_i,
      :description => 'Rails web shop purchase',
      :currency => 'usd'
    )

    @order.update_attributes(user_id: current_user.id,
                          order_status_id: 2)
    session.delete(:order_id)
    flash[:success] = "Your order has been placed successfully!"
    redirect_to(root_path)
  end
end

```

Isječak 7. Glavni sadržaj kontrolera za obradu plaćanja

Kontroler prikazan isječkom koristi dvije predefinirane Stripe funkcije koje stvaraju kupca i naplatu. Kod stvaranja kupca poželjno je koristiti email trenutnog korisnika prodavaonice kako bi se izbjegli problemi s identifikacijom korisnika plaćenih narudžbi. Kod naplate pretvara se iznos u najmanju jedinicu (u primjeru se radi o američkim centima) i poželjno je dodati opis plaćanja koji je kasnije vidljiv na Stripe upravljačkoj ploči. Prije samog stvaranja Stripe kupca i naplate provjerava se da je korisnik ima podešene informacije o adresi dostave, a nakon Stripe plaćanja ažurira se kupac i status narudžbe za potrebe prodavaonice.

Korisničko sučelje kod plaćanja potpuna je odgovornost JavaScript skripte *checkout.js* kojoj je potrebno predati informacije koje želimo od korisnika kao argumente ili predefinirati neke parametre narudžbe (npr. email u isječku 8.).

```

<%= form_tag charges_path do %>
  <article>
    <% if flash[:error].present? %>
      <div id="error_explanation">

```

```

        <p><%= flash[:error] %></p>
    </div>
    <% end %>
</article>
<%= hidden_field_tag "order_id", @order.id %>
<div class="pull-right">
    <script src="https://checkout.stripe.com/checkout.js" class="stripe-
button"
        data-email="<%= current_user.email %>"
        data-key="<%= Rails.configuration.stripe[:publishable_key] %>"
        data-description="Rails shop purchase"
        data-amount="<%= @order.total * 100 %>"
        data-zip-code="true"
        data-locale="auto"
    >
    </script>
</div>
<% end %>

```

Isječak 8. Implementacija korisničkog sučelja za Stripe plaćanje

Uz još ponešto generiranja tajnih ključeva, registracije na službenoj stranici Stripe tvrtke i podešavanja Rails aplikacije za Stripe, provedba plaćanja spremna je za korištenje. U poglavlju koje se bavi korisničkim sučeljem moguće je vidjeti formu za plaćanje gdje se za razvojno, test okruženje koristi fiktivna kartica kako bi se vrlo jednostavno moglo provoditi demonstrativno plaćanje. Nakon provedbe plaćanja, sve provedene transakcije dostupne su na Stripe upravljačkoj ploči gdje postoji mnogo opcija za uspješno poslovanje. Kad bi internetska prodavaonica počela koristiti produkcijsko okruženje i bila spremna za stvarne naplate, korištenje Stripe plaćanja ne bi bilo besplatno, te bi se od svake transakcije postotak iznosa narudžbe izdvojio Stripe tvrtci. Za više detalja pogledati Stripe uvjete korištenja na službenim stranicama tvrtke navedenim u popisu literature.

7.5 Rails Admin

Rails Admin je gotovo rješenje u Rails okviru za administraciju web-aplikacija. Odmah pri integraciji *gema* Rails Admin u aplikaciju, na pretpostavljenom putu (/admin) moguće je pokrenuti Rails Admin sučelje koje očitava bazu podataka aplikacije i pruža osnovne administrativne funkcionalnosti poput; pregleda, dodavanja i uređivanja zapisa, izvoza podataka u tabličnom obliku i pregleda osnovnih numeričkih statistika o broju zapisa pojedinih modela u bazi podataka.

Internetska prodavaonica mobilnih uređaja koristi uređeno sučelje gdje su izbačeni svi modeli kojima nije potrebno administriranje, ostvarena je podrška za dodavanje slika kod

svakog uređaja, podrška za napredni uređivač teksta kod opisa pojedinog uređaja (uvlačenja, liste, debljina, oblik i veličina teksta itd.), kod preostalih modela izbačeni su atributi koje nije potrebno administrirati, dodane su nove upravljačke ploče za upravljanje narudžbama i za statistike, promijenjena imena modelima za lakšu navigaciju i još mnogi detalji. Samo sučelje opisano je i vidljivo u poglavlju 6.

```
config.model Product do

  object_label_method do
    :full_name
  end

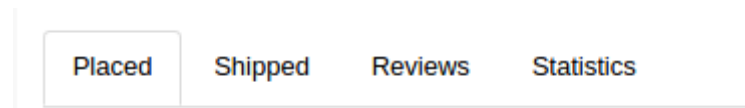
  list do
    field :id
    field :full_name
    field :description
    field :price
  end

  edit do
    field :name
    field :manufacturer
    field :price
    field :description, :wysihtml5 do
      config_options toolbar: { fa: true },
                     html: true,
                     parserRules: { tags: { p:1 } }
    end
    field :release_date
    field :chipset
    field :display_resolution
    field :display_size
    field :ram
    field :storage
    field :rear_camera
    field :front_camera
    field :product_images, :simple_has_many do
    end
  end
end
```

Isječak 9. Modifikacija administracije modela *product*

Modifikacija pojedinog modela vrši se u konfiguracijskoj datoteci za Rails Admin pri čemu se za svaki model koji želimo modificirati radi kod iz isječka 9. Na početku isječka postavlja se ime za svaki zapis pomoću metode `:full_name` koja uključuje i ime proizvođača kao prefiks imena. Zatim se u *list* bloku navode atributi koji se žele prikazati u ispisu svih zapisa tog modela. U *edit* bloku navode se svi atributi koje administrator može uređivati kod svakog zapisa. Ista forma koristi se i za stvaranje novih zapisa.

Za dodavanje novih upravljačkih ploča potrebno je napraviti kontroler i pogled za svaku po pretpostavljenim Rails Admin postavkama. Nakon predefiniranog Rails Admin koda, moguće je kontroler koristiti kao i svaki drugi. Pogledi su istog imena kao i kontroleri te nije potreban nikakav predefinirani kod. Kada su napravljeni kontroler i pogled, na putanji za administraciju može se vidjeti ažurirane upravljačke ploče zajedno s upravo napravljenom.



Slika 19. Upravljačke ploče

7.6 Statistički grafovi

Kod crtanja statističkih grafova na administratorskoj ploči korištena je JavaScript biblioteka za iscrtavanje grafova naziva Chartkick. Chartkick je vrlo lak za korištenje i odradi gotovo čitav posao sam. Sve što je potrebno uraditi za crtanje većinu grafova je proslijediti *hash* zapis s točkama grafova u obliku (ime: vrijednost). Moguće je proslijediti više oblika *hash* zapisa koje Chartkick prikazuje na drugačije načine, ali to nije obuhvaćeno ovim radom.

Hash zapis moguće je dobiti direktno koristeći Rails naredbe za baratanje bazom podataka i direktno proslijediti na iscrtavanje grafa. Kod složenijih pretraživanja u kontroleru se izvršavaju sva potrebna spajanja i manipulacije kako bi krajnja varijabla koja se predaje metodi za iscrtavanje sadržavala samo ispravne *hash* vrijednosti.

```
<h4> Top selling </h4>
<%= column_chart @top_selling %>
```

Isječak 10. Prikaz korištenja Chartkick biblioteke

Isječkom 10. prikazuje se kako Chartkick nakon minimalnog podešavanja ne zahtijeva ništa drugo od programera osim da izvrši filtriranje sebi po volji, prilagodi ulaznu varijablu na odgovarajući oblik te pozove odabranu metodu. Grafovi se mogu vidjeti u poglavlju 6. koje prikazuje administratorsko sučelje prodavaonice.

8. Zaključak

Ovim radom prvenstveno je bio cilj upoznati se s Ruby on Rails programskim okvirom kroz izradu rudimentarne internetske poslovnice, odnosno naučiti osnove izrade web-aplikacija te modeliranje i upravljanje relacijskom bazom podataka.

Izrada je počela modelom baze podataka koji je zatim implementiran koristeći Ruby on Rails. Dodana je podrška za autorizaciju i autentikaciju korisnika i administratora te filtriranje proizvoda. Filtriranje koristi jQuery za osvježavanje sadržaja bez osvježavanja stranice. Iako je korisno koristiti jQuery u mnogim situacijama, u ovom slučaju ispostavilo se da dobitak ostvaren dinamičkim osvježavanjem nije značajan (ne radi se o složenim pretraživanjima baze i rezultat čini gotovo čitav sadržaj stranice), a kod implementacije je utrošeno više vremena nego što bi bilo utrošeno za klasično osvježavanje. Bilo je potrebno implementirati dodavanje i brisanje skrivenih parametara kod svakog osvježavanja kako se filteri ne bi gubili promjenom stranice. Također je kod promjene stranice trebalo voditi računa o ispravnom prikazu odsječka s filterima. Zaključak je da treba pronaći uravnoteženost između dobitka (u ovom slučaju korisničko iskustvo) i uloženog vremena. Implementirana je jQuery košarica, a nakon toga i narudžbe su postale funkcionalne. Odabrana je Rails Admin administracija kao rješenje za administraciju te je prilagođeno zahtjevima prodavaonice. Naposljetku, implementiran je Stripe za provedbu simboličkog plaćanja.

Rails pruža mnoge gotove funkcionalnosti putem *gemova* koje znatno ubrzavaju izradu web-aplikacija. No iako je ubrzan proces izrade, kod učestalog korištenja gotovih rješenja postoji mogućnost da se naiđe na više problema nego prednosti. U ovome radu primjer toj situaciji je Rails Admin administracija koju je bilo potrebno gotovo u potpunosti transformirati kako bi se uklopila u administrativnu ulogu internetske poslovnice. Na samom kraju, puno je više vremena i truda uloženo za izmjene nego što bi bilo potrebno za izradu čitavog novog administrativnog rješenja kojemu bi otpočetak bila poznata svrha i funkcionalnost.

Unatoč takvoj grešci uzrokovanom neiskustvom, Rails ostavlja impresivan dojam jednostavnog (iako to i nije), ali moćnog programskog okvira za web-aplikacije. Vrlo rijetko se na novonastale probleme troši značajna količina vremena te postoji vrlo čista struktura koda koja garantira preglednost i lako snalaženje.

9. Literatura

1. About Ruby, <https://www.ruby-lang.org/en/about/>
2. Ruby on Rails, https://en.wikipedia.org/wiki/Ruby_on_Rails, 05.06.2017.
3. MVC framework introduction,
https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm, 2017.
4. MVC Architecture, https://developer.chrome.com/apps/app_frameworks
5. Ruby on Rails, <http://rubyonrails.org/>
6. What is Ruby on Rails?, <http://railsapps.github.io/what-is-ruby-rails.html>, 11.11.2013.
7. Getting started with Rails, http://guides.rubyonrails.org/getting_started.html
8. jQuery, <https://en.wikipedia.org/wiki/JQuery>, 29.05.2017.
9. JavaScript, <https://en.wikipedia.org/wiki/JavaScript>, 05.06.2017.
10. Devise, <https://github.com/plataformatec/devise>
11. RailsAdmin, https://github.com/sferik/rails_admin
12. Chartkick Rails, <https://github.com/ankane/chartkick>
13. Stripe, <https://stripe.com/>

Sažetak

Internetska prodavaonica mobilnih uređaja

Internetska prodavaonica omogućuje svojim posjetiteljima tj. kupcima naručivanje proizvoda putem Interneta. Prodavaonica je ostvarena kao web-aplikacija u programskom okviru Ruby on Rails. Korisnik prilikom posjeta prodavaonici ima pristup standardnim funkcionalnostima koje uglavnom dijele internetske prodavaonice (pregled proizvoda, košarica, kupovina, filtriranje, itd.). Prodavaonicom upravlja administrator koji ima pregled nad svim narudžbama, proizvodima, popustima i korisnicima. Njegov je posao mijenjati status narudžbi, brinuti za zadovoljstvo korisnika te voditi računa o uspješnosti poslovanja prodavaonice.

Ključne riječi: web-aplikacija;web;aplikacija;prodavaonica.

Summary

Mobile Devices Web Shop

Web shop allows its customers to directly buy products online. Web shop is implemented as a Ruby on Rails web-application. Web shop customers have access to usual features that most conventional web stores implement (such as shopping cart, product filters, product purchase and more). Web shop is operated by an administrator who has total control over orders, products, discounts and users. Administrator's job is to keep track of order status of each order in progress and update it as and when needed, to provide customer service and to monitor shop's statistics and take necessary actions (such as discounting the least selling products).

Keywords: webshop;web;shop;web-app;rails;ruby.