Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/05BE01Ljahovic

APLIKACIJA ZA PRAĆENJE CENA PO MARKETIMA

APPLICATION FOR PRICE TRACKING BY MARKET

Aleksandar Ljahović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj članak treba da demonstrira probleme koji se javljaju tokom implementacije aplikacije za praćenje cena po marketima i način na koji su oni rešeni. Takođe, pojasniće se i tehnologije u kojima je implementirana aplikacija i pojmovi eleketronske trgovine i elektronskog poslovanja.*

Ključne reči: *e-trgovina, ASP.NET Web API, Angular*

Abstract – *This article should demonstrate the problems that appear during implementation of the application for price tracking by markets and the way they are solved. Also, the technologies used for application implementation and terms of e-commerce and e-business will be explained.*

Keywords: *e-commerce, ASP.NET Web API, Angular*

1. UVOD

U članku se opisuje napravljena aplikacija koja prikazuje najnižu cenu proizvoda u svim prodavnicama koje prodaju taj proizvod, značaju takvih aplikacija i način na koji je ona implementirana.

U prvoj celini opisan je pojam elektronskog poslovanja, zašto je ovakvo poslovanje bitno i šta doprinosi preduzećima. Takođe, opisan je i defnisan pojam elektronske trgovine. Na kraju ove celine opisana je aplikacija koja je razvijana, funkcionalnosti koje podržava i ograničenja koja su uvedena.

Kroz drugu celinu se opisuju tehnologije, radni okviri i prakse koje su korišćene pri implementaciji aplikacije.

U trećoj celini opisan je način na koji je apliakcija implementirana. U ovoj celini opisana je implementacija onoga što korisnik vidi, informacije koje su mu od interesa, informacije koje sme da vidi u zavisnosti od uloge koju ima, mogućnost unosa, izmene i brisanja podataka i slično.

2. OPIS REŠAVANOG PROBLEMA

Pod pojmom elektronsko poslovanje (e-poslovanje, eng. ***E-business***) podrazumeva se kupovina i prodaja na internetu, organizovanje poslovne komunikacije prema klijentu, organizacija poslovanja firme u mrežnom okruženju, briga o klijentu,...

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kupusinac, van. prof.

Pod elektronskim poslovanjem podrazumeva se obavljanje poslovnih procesa uz primenu elektronske tehnologije. Elektronska tehnologija podrazumeva kombinovanu upotrebu informacionih tehnologija i telekomunikacija. Ova vrsta tehnologije omogućava slanje velikog broja informacija, na velike daljine u kratkom vremenskom periodu. To omogućava preduzeću, koje u svom poslovanju koristi elektronsku tehnologiju, da ostvari značajne uštede u troškovima poslovanja, efikasnije obavlja svoje zadatke i, samim tim, bude konkurentnije na tržištu. E-poslovanje omogućava stvaranje potpuno novog modela poslovanja, prilagođavanje kupcu, neprekidan pristup svetskom tržištu, efikasnije poslovanje, smanjivanje troškova itd.

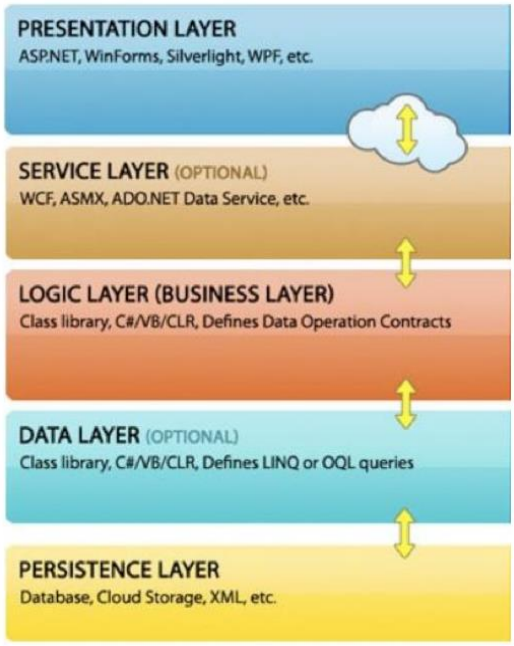
Elektronska trgovina (E-trgovina) je multidisciplinarni koncept, koji utiče na načine na koje se odvijaju interakcije i pregovori sa klijentima, načine na koje se obavljaju plaćanja, kao i odnose sa dobavljačima. E-trgovina se može posmatrati sa šireg i užeg stanovišta pa tako šira definicija obuhvata razmenu poslovnih informacija, održavanje poslovnih odnosa i vođenje poslovnih transakcija sredstvima telekomunikacionih mreža, a uža definicija obuhvata kupovinu i prodaju robe, usluga i informacija putem mreže. Prednosti elektornske trgovine u odnosu na tradicionalnu su mnogobrojne i značajne su kako za kupce i prodavce, tako i za čitavo društvo. Elektornoska trgovina smatra se globalnim fenomenom i kao takva podržava nekoliko modela. Dobra stvar kod e-trgovine je mogućnosti korišćenja jednog ili kombinovanje više modela kako bi poslovanje bilo uspešnije. Poslovni modeli, ili vrste, e-trgovine mogu se generalno svrstati nekoliko kategorija: biznis prema biznisu (B2B), biznis prema potrošaču (B2C), potrošač prema potrošaču (C2C), biznis prema vladi (B2G), vlada prema biznisu (G2B), vlada prema potrošaču (G2C).

Aplikacija za praćenje cena po marketima u osnovi treba da omogući korisnicima pregled cena po raznim marketima za neki proizvod i da mu izdvoji najpovoljniju cenu za taj proizvod. Aplikaciju mogu koristiti 3 tipa korisnika (administratori, kupci i posmatrači). Zadatak administratora je održavanje sajta. Ovaj korisnik radi održavanja sajta ima mogućnost dodavanja novih i brisanja postojećih proizvoda, prodavnica i kategorija proizvoda. Kupac je korisnik koji pored mogućnosti da vidi cene proizvoda po marketima, kao i najbolju cenu na početnoj stranici može da dodaje i uklanja proizvode iz korpe i prati najnižu cenu za odabrane proizvode. Proizvode iz korpe kupac može kupiti ili ih izbaciti iz nje. Posmatrač je korisnik aplikacije koji nema mogućnost rada sa korpom (dodavati, brisati, pregledati niti kupovati

proizvode), ali može pratiti cene po marketima. Administratori i kupci se moraju logovati kako bi izvršavali dodatne aktivnosti. Takođe, oni imaju mogućnost izmene svojih profila. Početna stranica sadrži nekoliko izlistanih proizovda i mogućnost promene strane kako bi se prikazali neki drugi proizvodi. Na početnoj stranici je moguće i pretraživanje proizvoda po nazivu, kao i filtriranje po kategoriji kojoj pripadaju.

3. OPIS KORIŠĆENIH TEHNOLOGIJA I ALATA

Aplikacija koja prati cene po marketima organizovana je po višeslojnoj arhitekturi. Ovaj stil arhitekture se najčešće svede na troslojnu arhitekturu koja obuhvata prezentacioni sloj, sloj poslovne logike i sloj podataka.



Slika 1. *Višeslojna arhitektura aplikacija.*

Višeslojna arhitektura ima mnogo prednosti zbog čega se, iako nastala početkom 90-ih, i dalje najčešće koristi u razvijanju aplikacija. Prezentacioni sloj je zadužen za komunikaciju sa krajnjim korisnikom. U njemu se izrađuje deo aplikacije koji korisnik treba da vidi. Servisni sloj služi za razmenu podataka između prezentacionog i sloja poslovne logike.

Ovaj sloj je opcioni pošto se često spaja sa slojem poslovne logike. Neki od zadataka ovog sloja su obrada zahteva, obaveštavanje o greškama i vođenje računa o sigurnosti sistema. Sloj poslovne logike implementira poslovne procese i poslovne komponente aplikacije. Zadužen je za obradu i validiranje svih zahteva sistema, sadrži logiku i poslovna pravila koja održavaju aplikaciju u celini. Sloj za pristup podacima ima zadatak da ostatku sistema prikupi zahtevane podatke, da ih izmeni ili upiše u skladište podataka koje aplikacija koristi. Samo ovaj sloj treba da ima pristup skladištu i da zna njegovu strukturu podataka. Sloj za perzistenciju predstavlja skladište u koje se smeštaju podaci koji su potrebni sistemu za funkcionisanje. U ova skladišta podataka moguće je upisati, izbaciti, izmeniti i iščitati podatke u zavisnosti šta je zahtavano od sloja iznad. Ovaj sloj podrazumeva bazu podataka, cloud, XML i slično.

U implementaciji prezentacionog sloja korišćen je Angular radni okvir. Angular je JavaScript radni okvir (eng. framework) otvorenog koda koji se koristi za razvoj klijentskih web aplikacija. Za izradu aplikacija u Angular-u najčešće se koristi TypeScript. To je jezik otvoreng koda koji predstavlja nadskup JavaScript-a. JavaScript program je i TypeScript program, dok obrnuto ne mora da važi. TypeScript se kompajlira u JavaScript i dizajniran je za razvoj velikih aplikacija. Angular projekat moguće je kreirati ručno, koristeći šablon projekta ili Angular CLI. Za razliku od prve dve opcije, Angular CLI omogućava ne samo brzo kreiranje projekta nego i komande za njegov brzi razvoj. Umesto da programer razmišlja o uvezivanju novih elemenata u projektu, ove komande to automatski rade.

Veći deo razvoja u Angular aplikacijama vrši se u komponentama. Komponente kontrolišu prikaz jednog dela ekrana. U osnovi, one su zapravo klase koje preko metoda i svojstava (eng. ***properties***) API-ja komuniciraju sa prikazom. Za grupisanje Angular komponenti, servisa, direktiva i ***pipes***-ova koji se odnose na aplikaciju koriste se moduli. Moduli mogu uvoziti funkcionalnosti koje su izvezene od drugih modula, kao i izvoziti svoje funkcionalnosti koje mogu koristiti drugi moduli. Servisi u Angular-u se koriste za pristup metodama i svojstvima drugih komponenti u celom projektu. Pomoću njih je moguće da istim podacima pristupa i manipuliše više komponenti u projektu. Servisi se mogu koristiti i da bi se komponente rasteretile pa se na primer validacije korisničkog unosa rade u njima, a komponentama signalizira ukoliko unos nije validan. Za dinamičko menjanje stranice i sadržaja na njoj, u Angular-u, se koristi rutiranje. Kod rutiranja povezuju se putanje (URL-ovi) sa odgovarajućim komponentama koje se učitavaju kada je njihova putanja pozvana. Vezivanje podataka (eng. ***Data binding***) omogućava komunikaciju između komponenti i ***template***-a tj. HTML-a. Njegova uloga je da neki sadržaj na HTML stranici učini dinamičkim.

U radu sa bazom aplikacija se oslanja na MySQL koji je višenitni i višekorisnički SQL sistem za upravljanje bazama podataka. Sistem radi kao server, obezbeđujući višekorisnički interfejs za pristup bazi podataka. Biblioteke za pristup bazi podateka MySQL postoje za većinu programiskih jezika, čiji oblik zavisi od datog programskog jezika. ER (***entity-relationship***) dijagrami pokazuju odnose između entiteta. Najčešće se koriste za organizovanje podataka unutar baze podataka ili informacionih sistema.

Za razvoj sloja poslone logike aplikacije korišćen je .NET Core. .NET Core na različitim platformama i platformama sa otvorenim kodom obezbeđuje lagani programerski model i fleksibilnost za rad na velikom broju razvojnih alata OS platformi. .NET Core se odnosi na više tehnologija koje obuhvataju .NET Core, ASP.NET Core i Entity Framework Core. On je kreiran da bi se .NET mogao koristiti na više platformi i u okruženjima koja imaju veća ograničenja. ASP.NET Core je ***web*** radni okvir kreiran od strane Microsoft-a za izradu ***web*** aplikacija, mikroservisa i API-ja. ASP.NET Web API je radni okvir koji olakšava izgradnju HTTP servisa koji dopiru do širokog kruga klijenata, uključujući

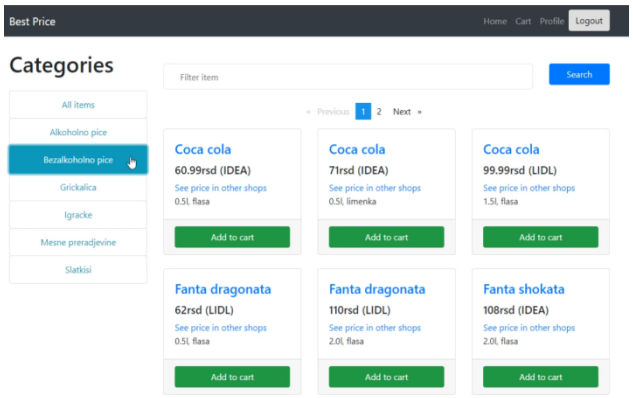
pretraživače i mobilne uređaje. Web API se posmatra kao HTTP interfejs koji od klijentske aplikacije prihvata HTTP zahtev i obavlja neke operacije nad podacima. Entity Framework (EF) Core je proširena verzija popularnog Entity Framework-a, tehnologije za pristup podacima. EF Core je verzija sa otvorenim kodom koja je podržana na više platformi. EF Core može da posluži kao ***object-relation mapper*** (ORM), omogućavajući .NET programerima da rade sa bazom podataka koristeći .NET objekte i eliminišu potrebu za većinom koda za pristup podacima koji obično treba da napišu.

Bootstrap je radni okvir namenjen izradi ***front-end*** dela ***web*** aplikacija. On je besplatan radni okvir otvorenog koda koji je baziran na HTML-u, CSS-u i JavaScript-u. Ovaj radni okvir olakšava izgradnju ***web*** stranica jer koristi globalne CSS postavke i osnovne HTML elemente, stilizovane i poboljšane proširenim klasama. Stranice napravljene u Bootstrap-u dostupne su na različitim platformama, za sve veličine ekrana. Dizajn ***web*** stranica je potpuno dinamičan i prilagodljiv (RWD - Responsive Web Design).

4. OPIS REŠENJA PROBLEMA

Implementacija aplikacije za praćenje cena u marketima i pronalaženje najniže cene bazirana je na troslojnoj arhitekturi. Za skladištenje podataka korišćena je MySQL baza podataka, a za manipulisanje bazom alat MySQL Workbench. Ovim alatom napravljena je šema sa tabelama, kolonama i relacijama između tabela. Relacijama između tabela postignuta su neka ograničenja: Svaka prodavnica može imati više proizvoda i svaki prozvod pripada tačno jednoj prodavnici. Dakle, isti proizvod u različitim prodavnicama posmatra se kao različit entitet. Ukoliko dođe do uklanjanja prodavnice iz baze, uklanjaju se i svi proizvodi koje je ona sadržala što je obezbeđeno opcijom kaskadnog brisanja.

Proizvod se svrstava u jednu kategoriju, dok jednoj kategoriji može pripadati više proizvoda. Korisnik može da bude ili administrator aplikacije ili kupac (registrovani korisnik). Administratorski nalog nema pravo kupovine pa mu ne treba ni korpa. Dakle, korisnik može a ne mora da ima svoju korpu u zavisnosti od uloge koju ima u aplikaciji, Korpa može biti prazna ali može sadržati i više proizvoda.



Slika 2. *Početna stranica aplikacije za praćenje cena po marketima*

Sprega između sloja poslovne logike i sloja za pristup podacima je generičko skladište čiji je zadatak da poveže zahteve kontrolera sa bazom. Dakle, ovo skladište kada dobije zahtev od kontrolera treba da ga prilagodi i prosledi ka bazi tj. sloju za pristup podacima. Ovaj deo se najčešće implementira upotrebom ***repository*** šablona i može se posmatrati i kao deo sloja za pristup podacima. Interfejs IRepositoryBase koristi se kao osnova za implementaciju repository šablona i sadrži metode koje će koristiti sva konkretna skladišta. Konkretna skladišta se odnose na neki model ili tabelu u bazi podataka.

U konkretnim skladištima implementiraju se samo metode koje su od interesa za kontrolere. Kako se u ovim skladištima implementiraju kompleksnije metode kontroleri ostavljaju čitkiji i lakši za održavanje. Za razliku od čitanja iz baze, kod zahteva za manipulisanje bazom neophodno je uraditi i čuvanje tih izmena pa se kod kreiranja, izmena i brisanja mora eksplicitno pozvati čuvanje izmena u bazi. Postoje kontroleri kojima su potrebne informacija ili treba da manipulišu sa više entiteta iz baze. Da se u takvim kontrolerima ne bi morala instancirati sva, potrebna, konkretna skladišta, napravljen je omotač koji omogućava pristup svim skladištima.

Ovaj omotač kontroleri dobijaju ***dependency injection***-om i koriste njegov interfejs za pristup konkretnom skladištu. Kao što je ranije rečeno, kontroleri u sloju poslovne logike imaju zadatak da obrade zahteve klijenta pristigle preko prezentacionog sloja, a zatim da preko istog vrati odgovor. Svaka metoda koja se isporučuje klijentu ima rutu koju je neophodno pogoditi da bi joj se pristupilo. Sve metode jednog Web API kontrolera imaju zajednički deo rute i deo rute koji se odnosi samo na tu metodu. Metode ovih kontrolera imaju i dekorator HTTP akcije koja se izvršava. Kada kontroler obradi pristigli zahtev, kroz interfejs IActionResult vraća odgovor klijentu. Ovaj interfejst očekuje odgovor u formatu status odgovora uz odgovarajuću poruku (objekat ili tekstualni opis). Kontroleri su, takođe, odgovorni i za validaciju modela. Kada sa prezentacionog sloja stigne zahtev za kreiranje ili izmenu nekog entiteta, kontroler će pre obrade tog zahteva proveriti validnost pristiglog objekta sa odgovarajućim modelom. Neretko se dešava da model kreiran u sloju za pristup podacima ne sadrži sve informacije koje kontroler treba da vrati prezentacionom sloju. U ovakvim situacijama pravi se DTO model.

U implementaciji prezentacionog sloja fokus je na prikazu svih funkionalnosti korisnicima koji će koristiti aplikaciju, ali i pribavljanju neophodnih podataka sa servera. Prezentacioni sloj implementiran je korišćenjem Angular framework-a pa su za pribavljenje podataka zaduženi Angular servisi. Servisi imaju zadatak da preko kontrolera sloja poslovne logike komuniciraju sa serverskim delom aplikacije tako što će slati zahteve koje je korisnik inicirao ili zahteve za pribavljanje informacija koje su prezentacionom sloju od interesa. Kao što je opisano u kratkoj specifikaciji projekta aplikacija je namenjana za korišćenje različitih tipova korisnika. Različiti korisnici imaju različite uloge što znači i mogućnosti u radu sa aplikacijom.Veći deo početne strane aplikacije se ne menja u zavisnosti od korisnika. Svi korisnici imaju mogućnost da pretražuju proizvode po nazivu, kao i da ih filtriraju po kategoriji kojoj pripadaju. Na početnoj stranici prikazano je 6 proizvoda i mogućnost

listanja strana korišćenjem broja strane na koju korisnik želi da ode ili Previous/Next opcija za izlistavanje prehtnodne/sledeće strane proizvoda. Jedna od retkih razlika na početnoj strani je opcija dodavanja proizvoda u korpu. Korisnici sa ulogom kupca imaju mogućnost dodavanja, dok je ostalim dugme onemogućeno. Na klik „See price in other shops“ prikazuje se modalni dijalog u kojem su izlistane cene tog proizvoda u svim prodavnicama, sortirane po najpovoljnijoj. Modalni dijalog sadrži ***header*** (prikazuje informaciju o nazivu proizvoda), telo u kojem su izlistane prodavnice i cene za taj proizvod i ***footer*** na kojem se nalazi dugme za napuštanje dijaloga.

Otvaranjem stranice Manage administratoru su izlistane sve kategorije koje postoje u sistemu (bazi padataka). Administrator može da obriše kategoriju samo pod uslovom da je prazna (ni jedan proizvod joj ne pripada). Takođe, on može i da dodaje nove kategorije, a dovoljno je samo da unese njihov naziv. Osim kategorija, admin može da vidi sve prodavnice i sve proizvode. Za uklanjanje prodavnice iz sistema ne postoji ograničenje za proizvode kao kod kategorija (sme biti vezana za proizvode) već se zajedno sa njom uklanjaju i svi njeni proizvodi. Brisanje pojedinačnih proizvoda iz prodavnice je takođe moguće bez ikakvih ograničenja. Admin pri dodavanju prodavnice pored naziva unosi i adresu na kojoj se ona nalazi. Pri dodavanju proizvoda admin, između ostalog, mora odabrati kategoriju i prodavnicu kojima proizvod pripada.

Stranica Cart tabelarno prikazuje sve proizvode koje je kupac dodao u korpu. U tabeli su osim naziva proizvada prikazane i najniže cene i prodavnica u kojoj je ta cena aktuelna. Takođe, pored svakog proizvoda nalazi se dugme „Cancel“ koje omogućava korisniku da iz korpe izbaci taj proizvod. Pored pojedinačnog izbacivanja proizvoda, kupac na „Cancel All“ dugme može da izbaci sve proizvode tj. da isprazni korpu.

Kupcu je takođe prikazana informacija o ukupnoj, najnižoj, vrednosti svih odabranih proizvoda. Za kraj, kupac ima mogućnost kupovine proizvoda pritiskom na dugme „Buy“. Admin i kupac otvaranjem stranice Profile mogu videti neke od podataka koje su uneli kao što su korisničko ime, ime i prezime, ***email*** adresa, broj mobilnog i odabrana slika. Oni mogu da promene svoju sliku odaberom druge slike sa svog računara. Osim slike, korisnici mogu menjati i druge podatke (ime, prezime, broj mobilnog). Takođe, ukoliko su napravili izmene, a žele da ih poništе, postoji dugme „Reset“ kojim se poništavaju sve unete izmene. Admini i kupci mogu na stranici Profile menjati i svoju lozinku tako što će u polja „Password“ i „Verify“ uneti novu lozinku (lozinka se mora poklapati u oba polje kako bi bila prihvaćena). Pored nabrojanih stranica, postoji i Login stranica preko koje se admini i kupci autentifikuju na sistemu. Kako bi se ulogovali, neophodno je da unesu korisničko ime ili ***email*** u prvo polje i odgovarajuću lozinku u drugo polje.

5. ZAKLJUČAK

U implementaciji aplikacije korišćene su tehnologije koje su jedne od najpopularnijih u svetu. Angular polako gubi bitku sa React-om koji je dosta jednostavniji pa se sve više koristi, međutim Angular je sve bolji iz verzije u verziju i ima mnoštvo prednosti u odnosu na ostale radne okvire za izradu front-end dela aplikacije. MySQL vodi borbu sa Oracle i Microsoft SQL Server, a ASP.NET je i uz porast popularnosti i dalje iza PHP.

Pri implementiranju su korišćene dobre prakse kao što su primena višeslojne arhitekture i podela nadležnosti, primenjivani su dizajn šabloni, pravljeni omotači (eng. ***wrappers***) itd. Ovakve aplikacije mogu mnogo pomoći kupcima da povoljnije izvrše kupovinu. Umesto da pamte cene po prodavnicama ili ulaze na razne sajtove prodavnica da bi uporedili cene, ova aplikacija ima sve cene na jednom mestu i prikazuje im najpovoljniju. U aplikaciji je implementirana „potrošačka korpa“ koja omogućava da se skupe svi proizvodi na jednom mestu, a zatim izvrši kupovina. Naravno, dodati proizvodi se mogu i izbaciti iz korpe ukoliko se kupac predomisli.

Dodavanje, izmena i brisanje prodavnica, proizvoda i kategorija u sistemu implementirano je tako da administrator mora ručno unositi informacije o njima. U daljem razvoju bi implemetiranje servisa koji sam dobavlja ove infromacije značajno poboljšalo i ubrzalo rad aplikacije. Takođe, izdvajanje administratora da sajt održava iz druge aplikacije bi bilo znatno sigurnije od eventualnih napada malicioznih lica. Korisnicima koji nisu registrovani je ipak omogućeno da vide najniže cene, što ih može zainteresovati i privući da se registruju i izvršavaju kupovinu preko sajta.

Logovanje je implementirano tako da korisnik može uneti ili ***username*** ili ***email***, koji su jedinstveni u sistemu, što im omogućava da se prijave sa onim što lakše pamte. Takođe, u aplikaciji su implementirane opcije filtriranja proizvoda po kategorijama i pretraga proizvoda po nazivu što značajno ubrzava traženje željenog proizvoda, a dodate su i stranice po kojima su proizvodi raspoređeni radi bolje preglednosti sajta.

6. LITERATURA

[1] Кончар Ј: „Електронска трговина“, Универзитет у Новом Саду, Економски факултет Суботица, Суботица, 2003

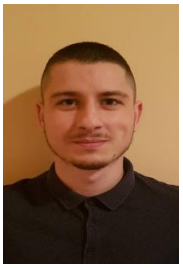
[2] https://www.eyerys.com/articles/types-e-commerce-models

[3] https://www.c-sharpcorner.com/article/learn-about-angular-lifecycle-hooks

[4] https://docs.microsoft.com/en-us/dotnet/core/about

[5] https://docs.microsoft.com/en-us/ef/core/

Kratka biografija:



Aleksandar Ljahović rođen je 19.05.1994. godine u Sremskoj Mitrovici. Završio je srednju ekonomsku školu 9. maj u Sremskoj Mitrovici 2013. godine. Fakultet tehničih nauka u Novom Sadu je upisao 2013. godine. Ispunio je sve obaveze i položio je sve ispite predviđene studijskim programom.