Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektronsko poslovanje

REACT E-BOOK

Seminarski rad iz Elektronskog poslovanja

Nastavnik: prof. dr Aleksandra Labus

Student: Sofija Milinović 2019/1025

Beograd, 2021.

Sadržaj:

[1.UVOD 3](#_Toc64831134)

[2. PREGLED LITERATURE 4](#_Toc64831135)

[3. PREGLED POSTOJEĆIH REŠENJA 5](#_Toc64831136)

[4. RAZVOJ APLIKACIJE 6](#_Toc64831137)

[5. ZAKLJUČAK 18](#_Toc64831138)

[6. REFERENCE 19](#_Toc64831139)

# 1.UVOD

Tema koja je odabrana za izradu ovog seminarskog rada je izrada sajta za onlajn (eng. online) naručivanje knjiga, u JavaScript biblioteci React (eng. Library). Glavni razlog zbog koga je odabran upravo React, jeste mogućnost, da se do sada stečena znanja na kursu elektronsko poslovanje, u vezi sa JavaScript-om, a zatim i osnova koje je dobijena po pitanju React-a, dodatno proširi i na kraju dobije svoju realnu primenu. Takođe, jedan od razloga iz kojeg je odabran React, a ne okvir Angular (eng. Framework), upravo jeste dodatna sloboda, kada je reč o samoj izradi koda.

U narednim poglavljima biće reči o samoj aplikaciji i procesu implementacije, što će biti propraćeno svim neophodnim objašnjenjima i izdvojenim najbitnijim delovima koda.

Drugo poglavlje će dati detaljnije informacije o celokupnoj tehnologiji koja se koristi u konkretnom okruženju koje je korišćeno za izradu i ostalim neophodnim bibliotekama i serverima ili biblitekama koje su omugućile jednostavniju i efikasniju izradu.

Treće poglavlje se odnosi na istraživanje već postojećih mobilnih ili web aplikacija sa sličnom ili istom tematikom koja je gore navedena kao tema izrade ovog seminarskog rada.

Četvrto poglavlje je centralno, jer će u njemu, kako je već prethodno u tekstu napomenuto biti dat detaljan opis same implementacije, kao i objašnjenja svih funkcionalnosti koje su obezbeđene sa pogleda korisničke strane.

Peto poglavlje sadrži zaključak sa ključnim saznanjima i mogućnostima za dodatno unapređenje aplikacije.

Šesto poglavlje je namenjeno za prikaz celokupne literature i svih izvora korišćenih prilikom izrade projektnog zadatka.

# 2. PREGLED LITERATURE

Za React bi se najopštije moglo reći da je to JavaScript biblioteka (eng. Library) za izradu korisničkog interfejsa. React se koristi za izradu aplikacija na jednoj stranici (eng. Single-page applications) ili mobilnih aplikacija. Glavna svrha, na kojoj se zasniva rad u React-u jeste upravljanje *stanjem* (eng. state) i prikazivanjem (eng. rendering) tog stanja u DOM-u (Document Object Model). Za rad u Reactu neophodna je upotreba pomoćnih biblioteka, među kojima je potrebno izdvojiti biblioteku za rutiranje - *React Route*, koja je neophodna za proces rutiranja (eng. routing) kako bi se omogućila izrada *Single-page aplikacije*.

Prilikom izrade aplikacije u React-u neophodno je korišćenje Node.js servera – JavaScript platforma koja služi za izvršavanje koda napisanog u JavaScript-u. Takođe za izradu aplikacije i rad potrebno je odabrati određeno razvojno okruženje (eng. development environment), a prilikom pravljenja aplikacije o kojoj je reč korišćen je Visual Studio Code, koji prilikom kreiranja React aplikacije sadrži određene biblioteke i ekstenzije o kojima će biti reč u nastavku.

Kako bi se na efikasan način obezbedio lep dizajn aplikacije korišćena je najpoznatija CSS i JavaScript biblioteka (eng. library) – Bootstrap.

# 3. PREGLED POSTOJEĆIH REŠENJA

Poznato je da postoji određen broj sajtova i knjižara koji svoju prodaju plasiraju putem interneta. Za primer je uzeta web stranica knjižare “ Vulkan”. Ova stranica je po nameni veoma slična aplikaciji koja je napravljena pri izradi ovog projekta. Baš kao i napravljena web aplikacija ima mogućnost da korisnicima omogući kupovinu omiljenih knjiga putem interneta. Međutim, postoje određene razlike, u pogledu funkcionalnosti, po kojima se značajno razlikuje. Glavna od njih je postojanje realne baze podataka, gde korisnici mogu poručivati knjige koje se trenutno nalaze u ponudi, koja ažurira podatke vezane za popuste i promenu asortimana. Takođe, obezbeđena je stvarna mogućnost kupovine knjiga i komunikacije sa korisnicima, koja uključuje forme za unos ličnih podataka korisnika i drugih informacija vezanih za porudžbinu. U pogledu funkcionalnosti, nudi još mogućnost sortiranja knjiga, pored filtriranja, ali za razliku od aplikacije o kojoj je reč, Vulkanov sajt omogućava filtriranje isključivo klikom na dugme za pretragu, dok je u E-Book aplikaciji filtriranje zasnovano na dinamičkoj promeni kako se podaci unose, a to je jedna od prednosti. Može se zaključiti da se ova aplikacija može dalje unapređivati i poboljšavati svoje funkcionalnosti, kako bi dobila realnu primenu.

# 4. RAZVOJ APLIKACIJE

Kao što je navedeno na početku dokumenta, tema projekta je izrada web sajta za onlajn (eng. online) prodaju knjiga, pod nazivom “E-Book”.

Osnovne funkcionalnosti i korisnički zahtevi biće detaljnije opisani u nastavku.

Sajt se sastoji od 3 stranice, početne stranice, na kojoj su date osnovne informacije o samoj prodavnici i uslugama koje pruža. Zatim centralna stranica koja sadrži pregled svih knjiga koje se nalaze u ponudi i jednu od glavnih funkcionalnosti koje aplikacija obezbeđuje korisnicima, a to je mogućnost filtriranja knjiga. Ova funkcionalnost je omogućena formom koja je smeštena na stranici sa knjigama i koja dozvoljava korisnicima da pretražuju knjige po nekoliko različitih kriterijuma. Treća stranica je takođe važna za efikasno funkcionisanje aplikacije i lakše snalaženje korisnika kada su u pitanju narudžbine koje su izvršili. Na njoj se nalazi “korpa”, која daje uvid u knjige koje je korisnik kliknuo da želi da kupi, kao i celokupan iznos napravljene porudžbine. Na stranici se nalazi i dugme “Buy”, koje simulira kupovinu knjiga i korisnik dobija informaciju o uspešnoj kupovini, a “korisnička korpa” ostaje prazna i ispisuje se informacija o tome.

Kako bi se krenulo sa izradom kreiranjem React aplikacije potrebno je da pokrenemo *Command Prompt*, tako što ćemo ukucati *cmd.* Nakon što je prozor otvoren, potrebno je izvršiti određena podešavanja kako bismo kreirali aplikaciju i pristupili joj. Potrebno je najpre ukucati *npm install –g create-react-app*, ukoliko ranije nije izvršeno ovo podešavanje. Npm ( Node Package Manager) - je osnovni menadžer paket Node.js-a koji sadrži stotine paketa i automatski je sadržan u instalaciji Node.js-a, pomaže u instaliranju različitih modula koji su potrebni za web razvoj. Zatim, potrebno je uneti sledeću komandu kako bi aplikacija bila kreirana: *npx create-react app naziv*. Npx (Node Package Execute) - predstavlja pokretač npm paketa, koji izvršava bilo koji paket iz npm registra. Ukoliko je instaliran npm verzije 5.2.0 i novije, npx dolazi u sklopu njega. Možete proveriti da li je npx već instaliran ili ne, pokretanjem komande *npx –v*. Nakon nekoliko minuta instalacija će biti završena, tada je potrebno pristupiti fajlu preko *cd* komande, a da bi aplikacija bila otvorena u *Visual Studio Codu*, izvršena je komanda *code .* Kao što je ranije napomenuto text editor koji je izabran je gore navedeni. Izvršavanjem komande *npm start*, pokreće se kreirana aplikacija na lokalnom serveru na portu 3000. Otvorena je podrazumevana (eng. default) React aplikacija koja sadrži React logo.

Vraćanjem u Visual Studio Code, primećuje se određen broj izgenerisanih foldera koji se pojavljuju prilikom kreiranja svake React aplikacije. Centralni deo aplikacije smešten je u fajlu *index.html* koji se nalazi u okviru *public* foldera. Neophodno je da ovaj html fajl sadrži barem jedan html dokument u kome će se renderovati sadržaj kreirane aplikacije. Konvencionalna vrednost za taj element je smeštena u div tagu pod nazivom *id=“root”*. Zatim u src folderu se nalazi još jedan značajan fajl za rad React aplikacije, a to je *Index.js.* Ovaj fajl je karakterističan po tome što sadrži ***render*** metodu koja prima dva atributa, prvi je App komponenta, koja zapravo predstvlja jedan html element i o kojoj će više reči biti u nastavku, a druga se odnosi na html elemnt u kome će prvi atribut biti smešten u okviru gore pomenutog *root* elementa.

U src folderu je takodje smeštena i App komponenta, koja sadrži određeni izgenerisani kod koji je potrebno izbrisati i pisati novi. Rad sa komponentama je osnova rada u React-u i zato ovo nije jedina komponenta koja je korišćena prilikom izrade. Iz tog razloga, kako bi sam kod bio pregledniji u okviru src foldera, je kreiran novi folder pod nazivom *components* u kome se nalaze folderi sa svim komponentama koje su korišćene u projektu.

Kako je već ranije nagovešteno, svaka React aplikacija će biti sastavljena od delova koji se nazivaju komponentama. Komponente služe kako bi se znatno olakšao razvoj korisničkog interfejsa i omogućavaju rad sa komponentama samostalno i na kraju njihovo spajanje u nadređenu komponentu koja predstavlja konačan korisnički interfejs, a to je upravo komponenta App.

React koristi dve vrste komponenti - funkcionalne i klasne.

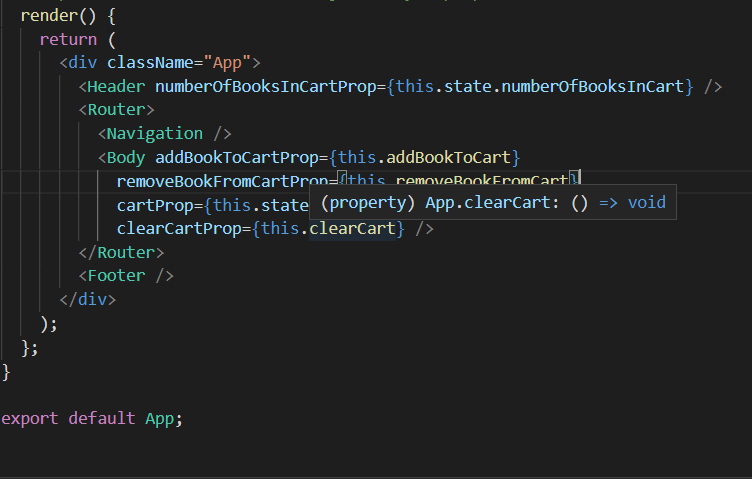
*Funkcionalne komponente* – predstvljaju JavaScripte funkcije koje kao parameter primaju *props* (u funkcionalnim komponentama nije nužno da stoje pod nazivom “props”) i vraćaju određeni *jsx* kod.

*Klasne komponente* – ove komponente su nešto složenije od funkcionalnih i sadrže *state objekat* (eng. state object), čije atribute možemo koristiti u render funkciji, odnosno za prikazivanje komponenti. Kada se bilo koji atribut iz state objekta promeni, pomoću *setState* funkcije, React će pronaći sve html elemente u kojima se pojavljuje taj atribut, i “osvežiće” taj element sa postavljenom novom vrednošću. Upravo zbog ovog *state objekta* ova biblioteka nosi naziv React, jer html reaguje na promenu stanja, odnosno kada se stanje promeni, menja se i korisnički interfejs.

Glavna komponenta, koja se kako je ranije pomenuto poziva u samoj *render* metodi u okviru *index.js* fajla je ***App.js*** ima sledeći sadržaj:

U okviru render metode pozivaju se klasne komponente *Header*, *Navigation*, *Body* i *Footer.* Bitno je naglasiti da su komponente *Navigation i Body* uokvirene Router komponentom, što je važo za proces rutiranja (eng. routing), kako bi se obezbedilo funkcionisanje aplikacije kao *Single Page aplikacije*.

Na slici 1 prikazano je rutiranje stranice.



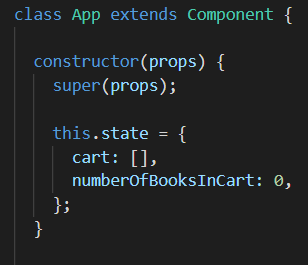
**Slika 1 – Rutiranje**

Druga bitna funkcionalnost ove komponente je kreiranje state objekta, koji omogućava upravljanje korisničkom “korpom”, razlog zbog kog je smeštena upravo u App komponenti je, jer se promene koje se dešavaju u okviru Cart komponente, odražavaju na komponente *Header* (koja sadrži ikonicu koja prati trenutno stanje o broju artikala u korpi) i *Body* koja je povezana upravo sa *Cart*-om

Pomenuti state objekat sastoji se iz dva atributa, a to su:

**cart:[ ]** – koji predstavlja niz u kome su smeštene knjige koje se trenutno nalaze u korisničkoj “korpi” i drugi, **numberOfBookInCart** koji menja vrednost u zavisnosti od trenutnog stanja u “ korpi”.

##### Na slici 2 je prikazan state objekat App komponente



**Slika 2 - state App komponente**

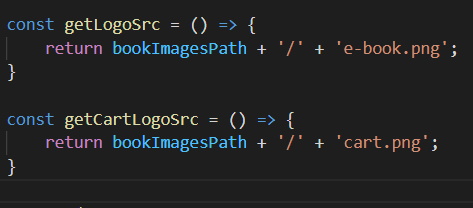
Takođe ova komponenta sadrži i dve funkcije:

*addBookToCart* – koja dodaje novu knjigu u korpu ili povećava količinu ukoliko se knjiga već nalazi u korpi.

*RemoveBookFromCart* – jednstavno uklanja knjigu iz korpe po želji korisnika

Sledeća komponenta jeste **Header** i ona sadrži logo aplikacije kao i ikonicu koja simulira punjenje “korpe”, a pored toga bitan je detalj o kome je ranije bilo nagoveštaja, de se u ovoj komponenti menja vrednost broja knjiga, na osnovu promene state-a iz App.js komponente kada se knjiga doda ili ukloni iz korpe.

##### Na slici 3 date su funkcije za prikaz ikonica iz Header-a

****

**Slika 3 - prikazivanje ikonica za logo i korisničku korp**

Sledeća komponenta jeste **Navigation**. Ona predstavlja funkcionalnu komponentu, koja u svojoj return metodi poziva komponentu **NavigationItem** kojoj prosleđuje putanju preko atributa *link* i vrednost atributa *text* koja će biti prikaza na stranici.

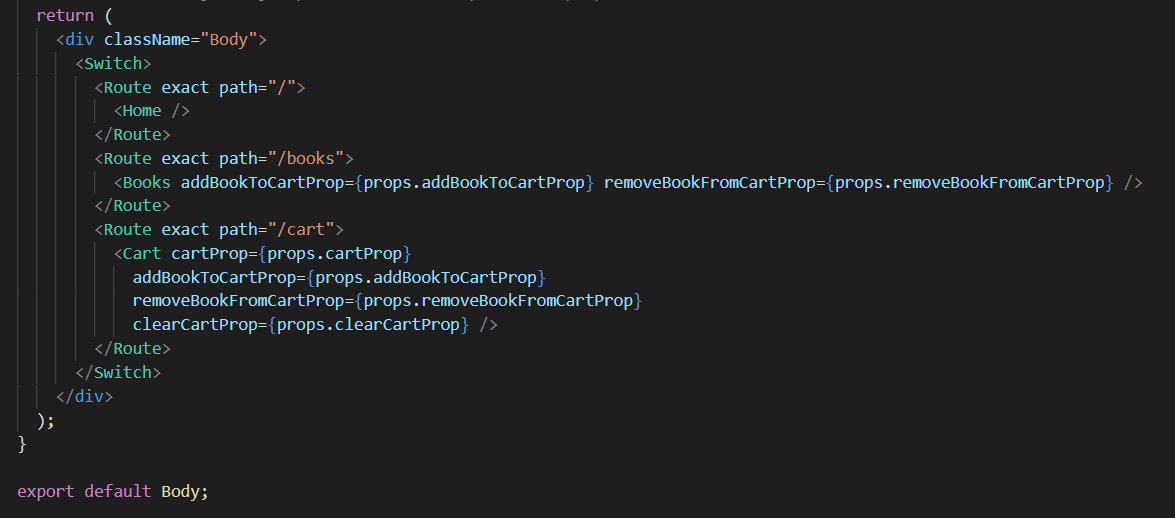
Na slici 4 vidi se poziv NavigationItem komponente i dodeljivanje vrednosti atributima



**Slika 4 - pozivanje NavigationItem komponente**

Komponenta **Body** se sastoji od komponenata *Home*, *Books* i *Cart,* ona takođe sadrži Route komponentu, a još važnije je prisustvo komponente *Switch*, koja obezbeđuje da se samo jedna od ruta izrenderuje, na osnovu putanje postavljene u okviru *Navigation* komponente

##### Na slici 5 prikazan je rad Switch komponente



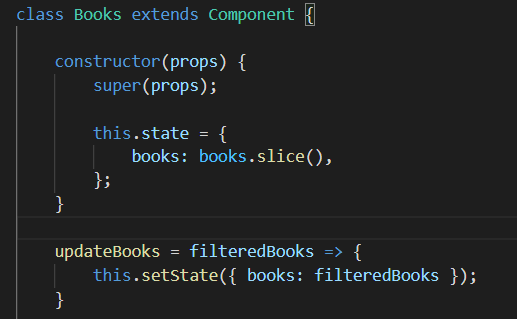
**Slika 5 - Switch komponenta**

**Home** komponenta ne sadrži nikakve funkcionalnosti, već se sastoji od slike i teksta koji daju osnovne informacije o onlajn (eng. online) knjižari.

Pre nego što se pređe na objašnjavanje ostalih komponenti, potrebno je pomenuti fajl **data.js** u kome se nalaze svi statički podaci koji oponašaju podatke iz baze podataka, između ostalog tu se nalazi i statički niz pod nazivom *books*, u kome su smeštene knjige sa svim podacima.

Sada se dolazi do komponente **Books**. Ova komponenta sadrži takođe jedan state objekat, u kome se nalazi atribut koji predstavlja niz knjiga i ovaj niz na početku dobija vrednost svih knjiga iz niza. Komponenta Books koristi Filter komponentu koja služi kako bi se promenila vrednost niza iz state-a, nakon filtriranja. Takođe koristi i Book komponentu za prikaz svake knjige i dodavanje, odnosno uklanjanje te knjige iz “korpe”.

Na slici 6 vidi se kako funkcija setState podešava vrednost state-a, nakon filtriranja

****

**Slika 6 - menjanje niza knjiga iz state-a**

Naredna komponenta pod nazivom Book sadrži sve funkcionalnosti koje se odnose na pojedinačnu knjigu. I Book komponenta sadrži jedan state objekat u kome se nalazi atribut pod nazivom showBookButtons i inicijalno ima vrednost postavljenu na false. Svrha toga je da se omogući da prelaskom mišem preko pojedinačne knjige, ta knjiga se uokviri i pojave se dugmići koji omogućavaju korisniku da dodaje/uklanja knjigu iz “korpe”. Iz tog razloga napravljena je funkcija *toggleBookButtons* koja pomoću funkcije *setState*, renderuje vrednost state-a, na osnovu toga da li je njen parametar trenutno postavljen na true ili false, u zavisnosti da li je mišem pređeno preko slike.

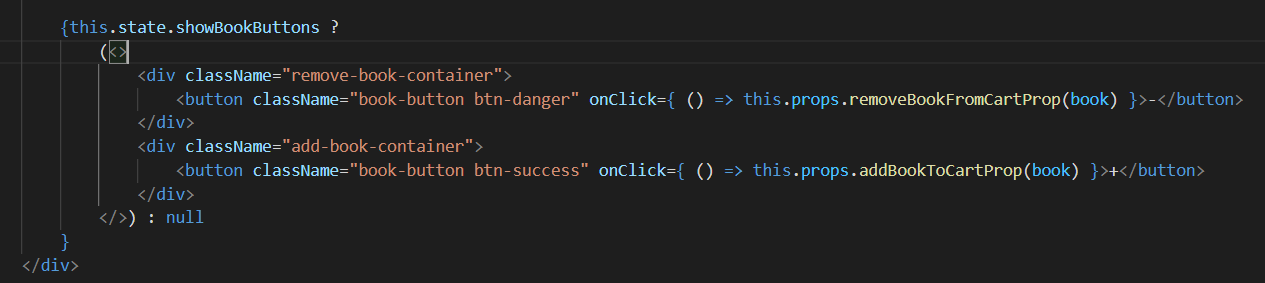
Na slici 7 prikazan je deo koda koji se odnosi na označavanje knjige 

**Slika 7 - prelazak mišem preko knjige**

U nastavku koda nailazi se na ternarni operator, čiji iskaz označava da ukoliko je neka knjiga trenutno označena, odnosno vrednost state atributa je podešena na true

Prikazuju se dugmići za dodavanje i uklanjanje iz korpe.

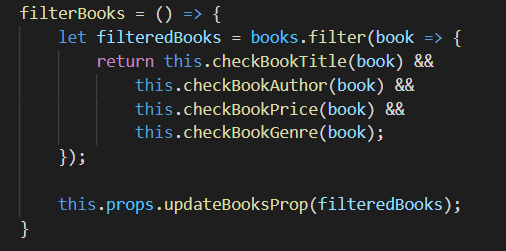
##### Na slici 8 prikazano je dodavanje dugmića



**Slika 8 - prikazivanje dugmića za dodavanje i uklanjanje**

Sledeća komponenta je veoma značajna za efikasno funkcionisanje, jer daje mogućnost korisnicima za efikasnu pretragu. U komponenti se nalaze četiri različite funkcije koje filtriraju knjige po četiri različita kriterijuma, a to su *naslov, ime autora, raspon cena* i *žanr.* Odnosno u pitanju su funkcije *checkBookTitle*, *checkBookAuthor*, *checkBookGenre i checkBookPrice.* Svaka od ovih funkcija kao parameter prima knjigu, nad kojom proverava da li se vrednost iz odgovarajućih input polja poklapaju sa vrednostia knjiga iz niza. U tu svrhu napravljena je funkcija pod nazivom *filterBooks* koja prolazi kroz statički niz, poziva gore navedene funkcije i vrši proveru na način prethodno definisan.

##### Na slici 9 prikazana je funkcija koja vrši filtriranje knjiga



**Slika 9 - Filtriranje knjiga**

Poslednja komponenta u okviru Body foldera, jeste komponenta Cart koja se odnosi na funkcionalnosti vezane za potrošačku korpu. State objekat je zastupljen i u ovoj komponenti i njegov atribut ima boolean vrednost inicijalno postavljenu na false, pod nazivom *purchased.* Svrha ovog atributa je da da informaciju o tome da li je kupovina obavljena, kada korisnik klikne na dugme Buy.

Za to je upotrebljena funkcija *onBuy* koja se sastoji of funkcija setState koja kako je već poznato renderuje vrednost state-a, tako što vrednost atributa *purchased* postavlja na true i poziva se funkcija za brisanje sadržaja “korpe”, i takođe generičke funkcije *setTimeout* koja omogućava da se nakon 5 sekundi skloni obaveštenje o uspešnoj kupovini i vrednost atributa *purchased* ponovo postaje false.

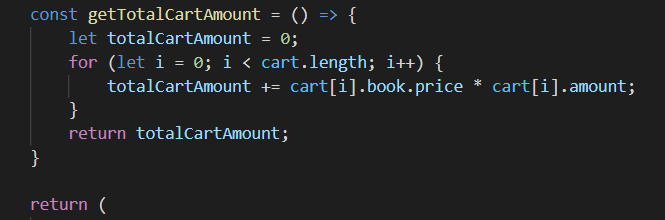
Na slici 10 prikazan je rad prethodno opisane funkcije



**Slika 10 - prikaz promene statusa kupovine**

Još jedna funkcija koju je potrebno pomenuti je funkcija pod nazivom getTotalCartAmount, koja prolazi kroz niz knjiga koje se nalaze u korpi i izračunava ukupnu cenu ostvarene kupovine.

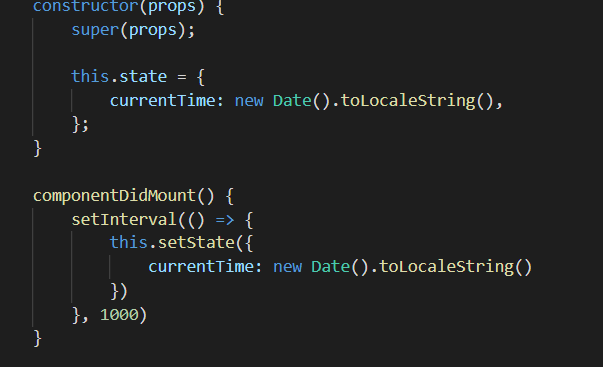
Na slici 11 je prikazana funkcija za izračunavanje ukupne vrednosti kupovine



**Slika 11 - funkcija getTotalCartAmount**

Poslednja komponenta koja je korišćena u izradi aplikacije je **Footer** komponenta, koja sadrži funkcionalnost u vezi sa ispisom datuma koji se menja u realnom vremenu.

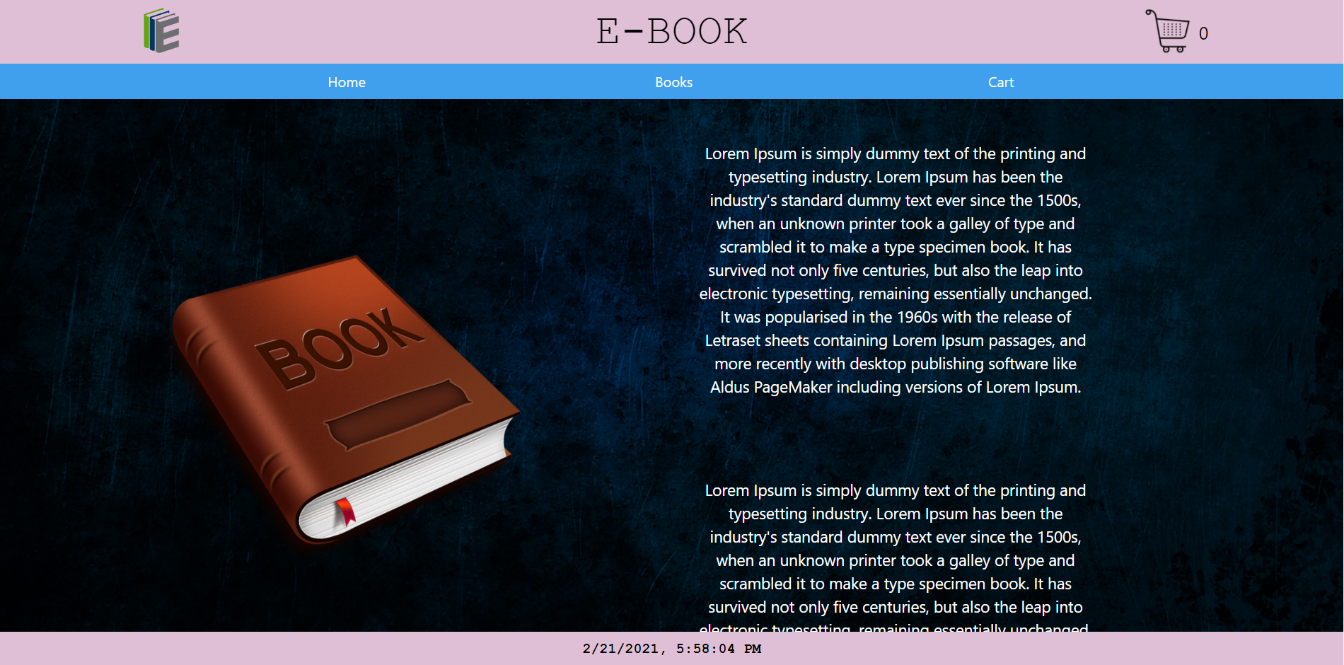
Na slici 12 prikazana je funkcija za podešavanje trenutnog datuma



**Slika 12 - trenutni datum**

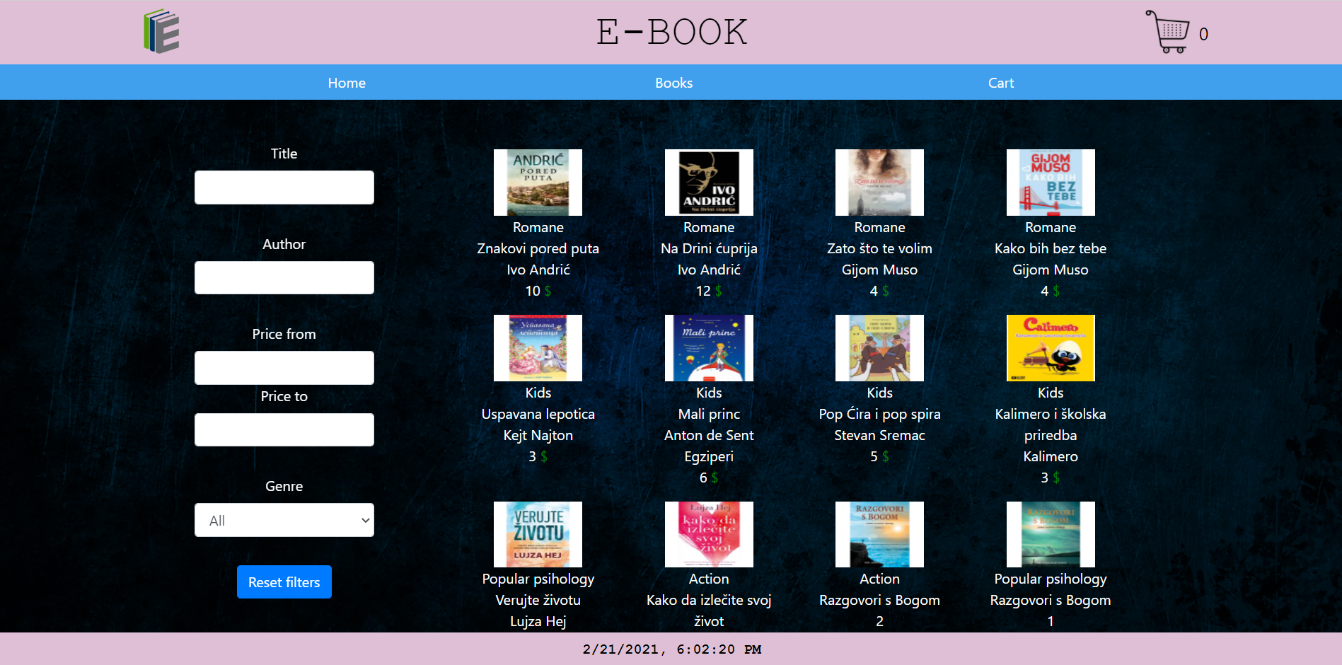
U nastavku će biti dat prikaz snimaka ekrana (eng. prints screen) aplikacije na strani korisnika i prikaz funkcionalnosti ranije opisanih.

Na slici 13 prikazana je početna stranica



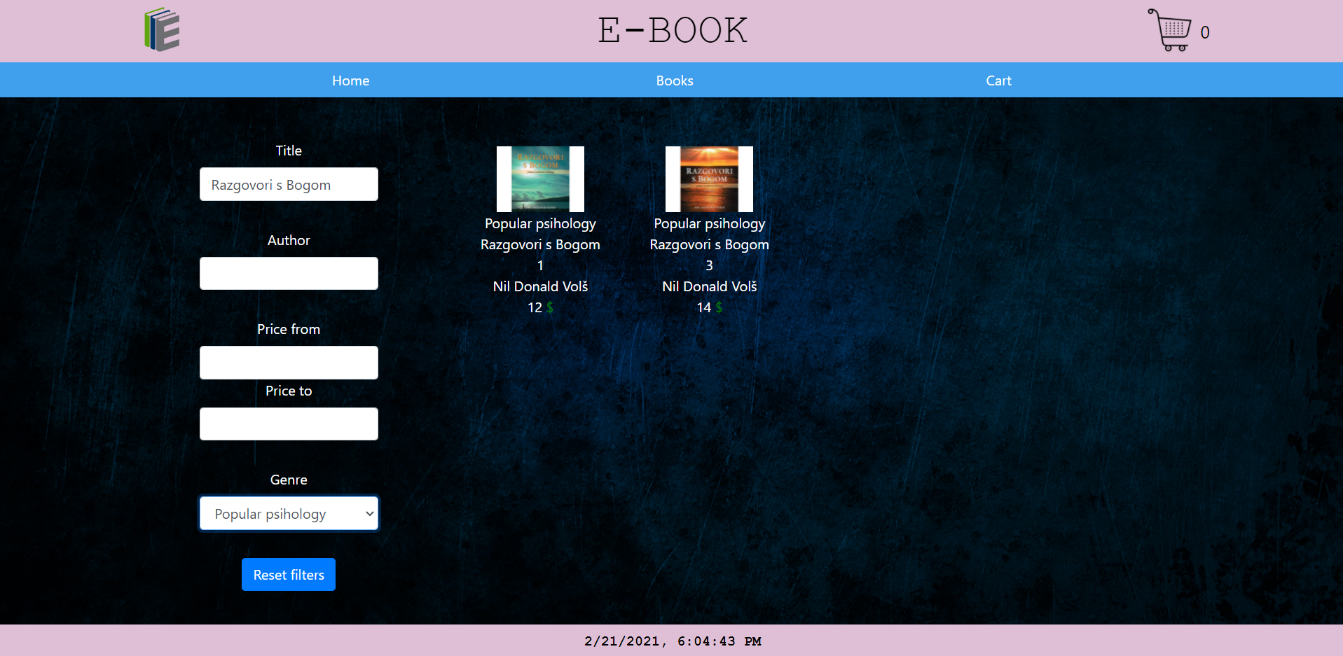
**Slika 13 - Početna stranica**

Na slici 14 prikazana je glavna stranica za prikaz knjiga

****

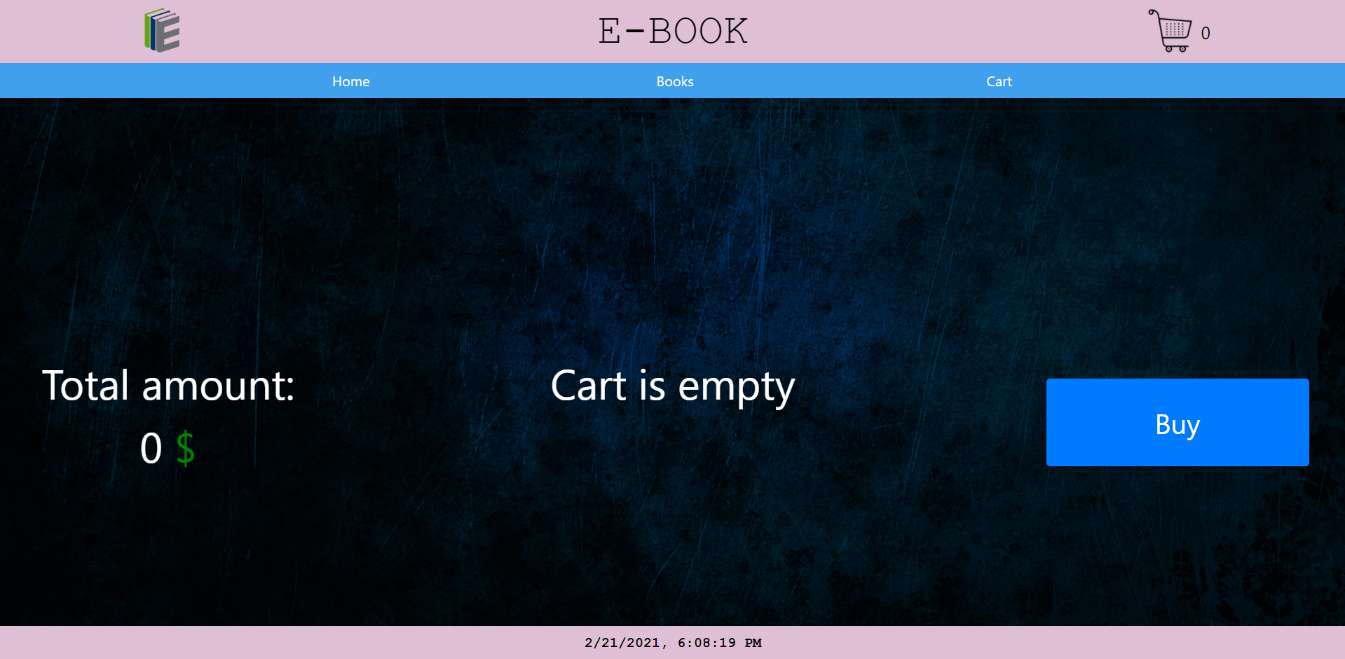
**Slika 14 - stranica za prikaz knjiga**

Na slici 15 prikazana je stranica sa knjigama nakon što je izvršeno filtriranje po kriterijumu *Title* и *Genre*

****

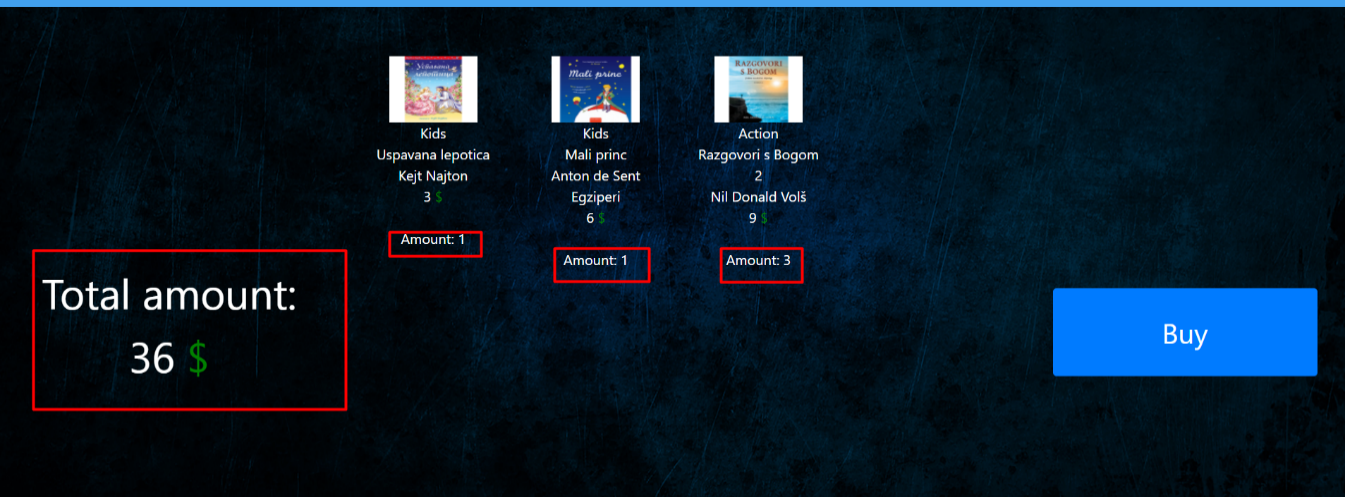
**Slika 15 - Filtrirane knjige**

Na slici 16 prikazan je izgled Cart stranice kada nema knjiga koje su dodate u korpu i kada je korpa ispražnjena



**Slika 16 - Prazna korpa**

Na slici 17 prikazana je korisnička “korpa” kada postoji spremna narudžbina sa ukupnim iznosom narudžbine i brojem artikala.



**Slika 17 - Korisnička korpa sa narudžbinom**

# 5. ZAKLJUČAK

Kao krajnji rezultat projekta o kome je reč u ovom dokumentu, napravljena je aplikacija, odnosno web stranica koja simulira onlajn kupovinu knjiga. Namera ovog projekta je da na interesantan način, prikaže kako se može napraviti jednostavna stranica sa najvažnijim funkcionalnostima i pruži osnovu za dalji rad na složenijim projektima, koji se na bilo koji način bave nekom vrstom interakcije sa korisnikom. Bitna strana aplikacije je što je obradila određene funkcionalnosti poput filtriranja koje mogu naći svoju primenu u daljem razvoju bilo kojih drugih aplikacija pisanih u Reactu, kao i to što je kroz jasne i jednostavne primere dovoljno pažnje usmereno na najvažnije karakteristike React-a. Pre svega u pitanju je upotreba komponenti, kako klasnih tako i funkcionalnih, i naravno upotreba state objekta na mnogo načina, što ima za cilj da što bolje objasni način korišćenja ovog neizostavnog elementa svake React aplikacije.

Ono što se u aplikaciji može pronaći kao zamerka je to što je ona pokaznog tipa i ne sadrži realnu bazu podataka, kao i neke unapređene funkcionalnosti, kao što bi moglo biti i sortiranje. Ova aplikacija bi se, dakle, mogla unaprediti uvođenjem dodatnih funkcionalnosti i njena celokupna izrada kao backend aplikacije.

# 6. REFERENCE

[1] Getting Started, <https://reactjs.org/docs/getting-started.html>, poslednja poseta stranici 19.02.2021.

[2] Tutorial: Intro to React, <https://reactjs.org/tutorial/tutorial.html>, poslednja poseta stranici 19.02.2021.

[3]ReactJs-Components,<https://www.geeksforgeeks.org/reactjs-components/>, poslednja poseta stranici 19.02.2021.

[4] Programming with Mosh. (16. Jul 2018). React Tutorial for Beginners [React js], <https://www.youtube.com/watch?v=Ke90Tje7VS0&fbclid=IwAR0G0fizAAftRyu9rse1wYb6ViQY2n4GCvFnMTxFoZpNndm9epMoCwZX264&ab_channel=ProgrammingwithMosh>, poslednja poseta 16.02.2021.

[5] Programming with Mosh. (24. Apr 2018). JavaScript Tutorial for Beginners: Learn JavaScript in 1 Hour [2020], <https://www.youtube.com/watch?v=W6NZfCO5SIk&fbclid=IwAR1WSo5GQ8A43gkkTUfj8HYBlx9Wy6xcWciwBbjIed8uN5i4jUWhvlX2OII&ab_channel=ProgrammingwithMosh>, poslednja poseta 16.02.2021.