

Odsek Visoka ICT škola,  
Akademija tehničko-umetničkih strukovnih studija,  
Beograd

# BerixLibrary

Smer: Internet tehnologije

Modul: Web programiranje

Predmet: Praktikum iz objektnog web programiranja

Berisavac Aleksa 32/18

## Sadržaj

1. Uvod .....	2
1.1 Korišćeni programski jezici/tehnologije .....	2
1.2 Opis funkcionalnosti .....	3
1.3. Napomene za pregledanje.....	3
2. Slike stranica i opisi funkcionalnosti.....	4
2.1 Home.....	4
2.2 Books.....	4
2.3. Cart.....	5
2.4. Checkout.....	7
2.5. 404 – Not Found.....	8
2.6. Modeli .....	8
2.7. Struktura tsBusinessLayer-a .....	10

## 1. Uvod

### 1.1 Korišćeni programski jezici/tehnologije

Za potrebe sajta korišćene su sledeće tehnologije:

- HTML, CSS
- JavaScript (TypeScript)
- Angular 16
- Material Design
- VS Code

## 1.2 Opis funkcionalnosti

Domen koji aplikacija pokušava da reši je **onlajn prodavnica knjiga**. Korisnici mogu da razgledaju proizvode i dodaju ih u korpu. Potom mogu da izvrše kupovinu, mada kako je ceo projekat radjen na frontu gde ne mogu da se edituju fajlovi nisam mogao da simuliram potrebne dto jsone i modele za ovu funkcionalnost. Svi podaci na sajtu su generisani od strane raznih online AI-jeva, od slika pa do naziva zanrova za knjige.

Funkcionalnosti sajta se mogu svrstati u sledeće kategorije:

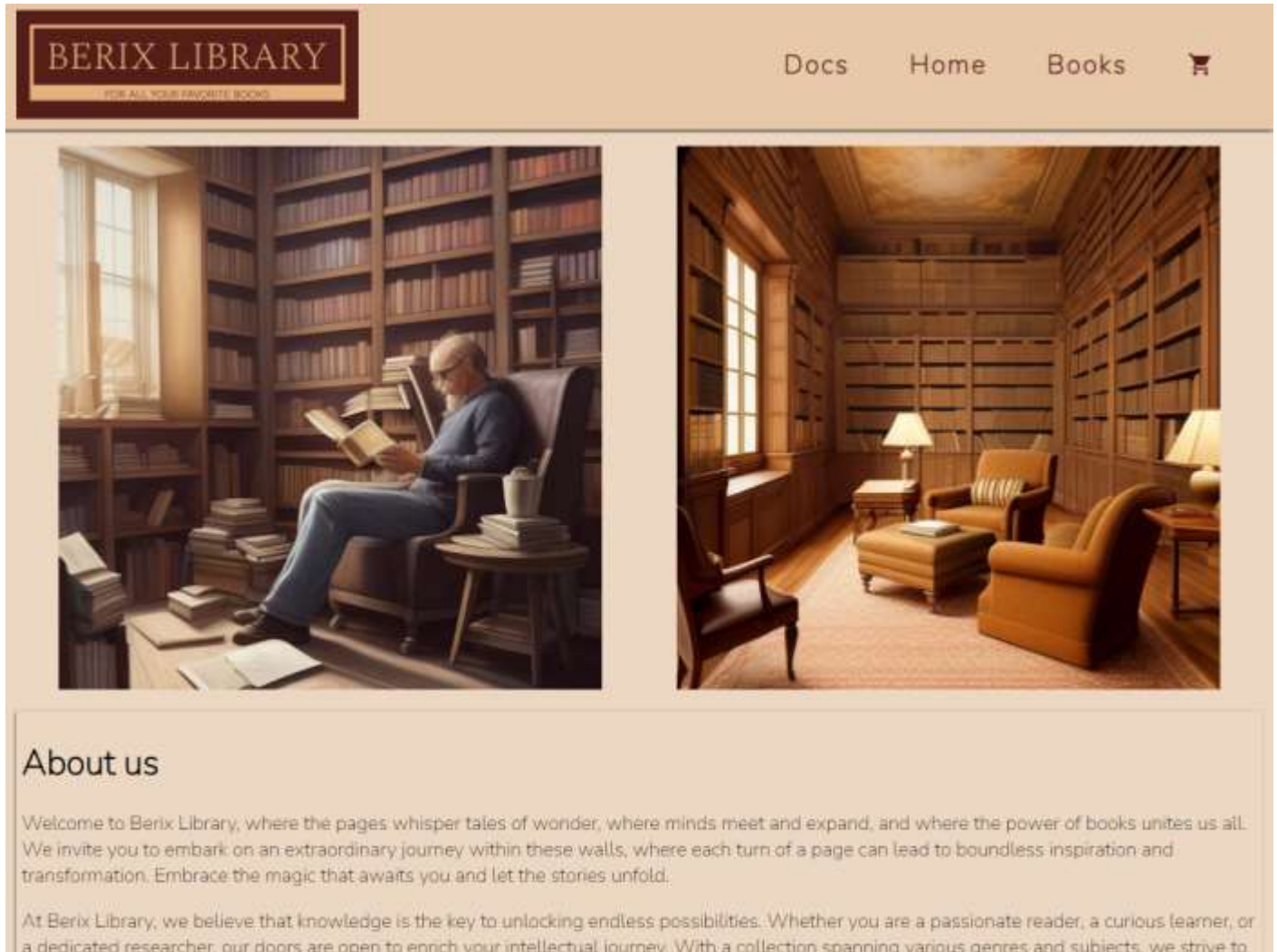
- **Skoro pa celokupan backend simuliran samo pomocu typescript-a**, u folderu `tsBusinessLayer` nalaze se svi potrebni interfejsi, klase, modeli i tipovi koji sluze kao spona izmedju podataka (json-a) i front aplikacije. Ovim sam zeleo da demonstriram OOP pristup rada na frontu. Spona izmedju komponenti i ovog sloja jesu u 99% slucajeva servisi koji koriste potrebne modele za crud operacije.
- **Reponzivan dizajn**
- **Pocetna stranica**, dinamicki ucitava tekstualne podatke za prikaz.
- **Rutiranje**, ovim se postize funkcionalnost SPA.
- **Galerija knjiga sa dinamickim ispisom pojedinih knjiga i njenih detalja**.
- **Korpa** koja smešta podatke u komponentu, sa ukupnom sumom i opcijama za brisanje i izmenu količine.
- **Checkout Forma**, za završavanje jednog ciklusa koriscenja sajta. Napravljena kao modal koji je svoja komponenta.

## 1.3. Napomene za pregledanje

- **tsBusinessLayer**, u cilju demonstriranja svih mogucnosti koje nam typescript daje i ucenja istih, odlucio sam se da odradim jako dubok i imerzivan sloj za manipulaciju sa podacima. Ovo mozda odskace od strukture koja je koriscena na predavanjima i vebama, ali mislim da izuzetno demonstrira jako bitan deo ovog kursa: Typescript. Ovde je iskorisceno skoro sve osim enuma (za koju nisam uspeo da nadjem primenu na sajtu ove sadrzine). Naravno neki interfejsi koji su bitni samo za rad servisa i komponenti se NE nalaze u ovom folderu, vec u odgovorajucem folderu unutar [/src/app](#).
- Zbog vise utrosenog vremena za gornju tacku, nisam stigao u potpunosti da implementiram sve funkcionalnosti koje sam zamislio. U narednom tekstu sam naveo za svaki deo sta je uradjeno, i sta nije uradjeno i zbog cega nije.

## 2. Slike stranica i opisi funkcionalnosti

### 2.1 Home



Na ovoj stranici se nalazi i navigacija, koja se zbog jako malog broja elemenata ne učitava ni iz kakvog json objekta. Ona koristi rutiranje, i responzivna je.

Tekst koji se ispisuje na početnoj stranici se učitava dinamički iz json-a, u vidu niza paragrafa.

### 2.2 Books

## Our Selection Of Books



Whispers of the Forgotten

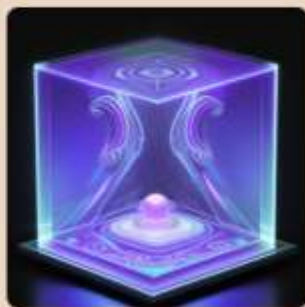
Price:

96.76\$



Authors:

1. Kendyll Peterson
2. Esthela Preston



The Enigma's Echo

Price:

92.67\$



Authors:

1. Koralyann Hahn
2. Kourtlyn Perkins
3. Veralyann Foley



Shadows of Redemption

Price:

56.6\$



Authors:

1. Zahvia Harrison
2. Vanessa Moran
3. Oluwadamilola Anderson



The Serpent's Symphony

Price:

106.75\$



Authors:

1. Zayan Noble
2. Lisanna Acevedo



Na ovoj stranici se mogu videti sve knjige. Knjige se ucitavaju preko servisa koji komunicira sa tsBusinessLayer-om, preko odgovarajucih modela, koji uzimaju podatke iz json-a. Potom se knjige ispisuju u komponenti dinamicki. Tu se takodje nalazi dugme za dodavanje u korpu. Nakon klika na ovo dugme stvori se jos jedno dugme koje služi za uklanjanje jednog tog item-a iz korpe. Ono sto je ostalo neimplementirano zbog nedostatka vremena jesu parcijalne stranice za autora I knjigu, a sto se tice dizajna da se iskoriste slideUp I slideDown za autore kako bi bilo preglednije. Svi neophodni podaci za ove funkcionalnosti su naravno tu, knjige imaju svoje zanrove, opise I jos dosta toga.

### 2.3. Cart

# Your Shopping Cart:

There are currently no items in your cart. [You can go to the books page and see what you like.](#)

BERIX LIBRARY


FOR ALL YOUR FAVORITE BOOKS

[Docs](#)

[Home](#)

[Books](#)

 | 6

Your Shopping Cart: 259\$  [checkout](#)

Art	Title	Price	Total Price	Quantity	Remove	Add
	Whispers of the Forgotten	96.76\$	96.76\$	#1		
	Whispers in the Mist	44.19\$	132.57\$	#3		
	The Labyrinth's Secret	16.09\$	16.09\$	#1		
						

Na ovoj stranici se vidi sadržaj korpe korisnika (ako nema proizvoda u korpi onda se izdaje takva poruka). Ukupna vrednost se izračunava i prikazuje.

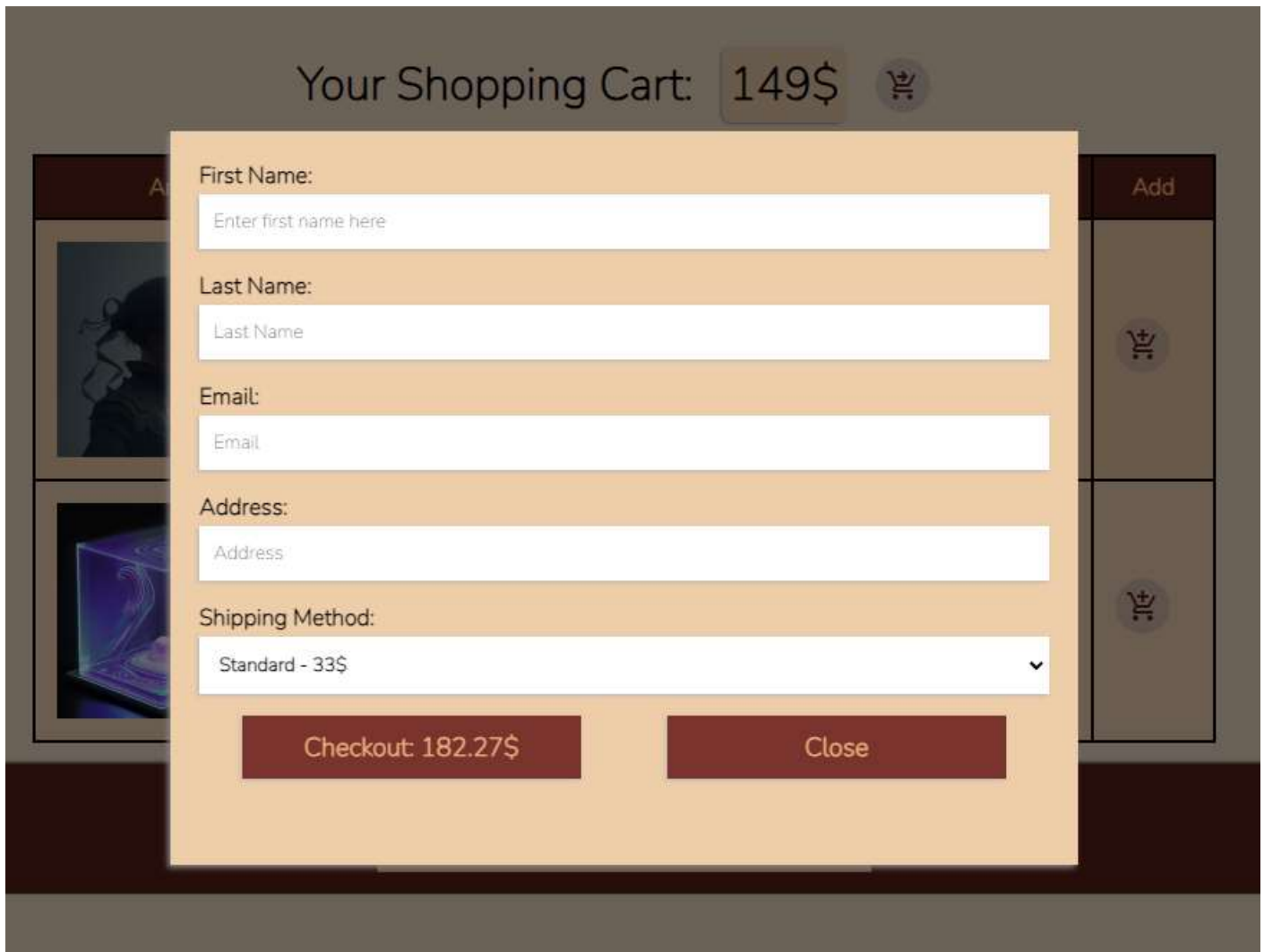
Korpa je napravljena kao servis koji se koristi na vise mesta, od kojih su neki : navigacija (gde se pored ikonice korpe vidi broj itema trenutno u njoj), sama komponenta korpa, checkout, prikaz knjiga u books ruti..

Ovde je takodje moguće uklanjati i dodavati proizvode.

Klikom na dugme za checkout otvara se modal sa formom za unos podataka za dostavu. Nije mi radio bas kako sam hteo Material Modal sa svojim inputima pa sam napravio svoje. Ukoliko se ta forma popuni ispravno korpa se čisti i

korisnik se obavestava o pozitivnom ishodu narudžbine. U suportnom dobija poruku da nije lepo popunio podatke. Zbog manjka vremena nisam previše vremena posvetio validaciji.

## 2.4. Checkout

The image shows a web application interface for a shopping cart checkout. At the top, it says "Your Shopping Cart: 149\$" with a shopping cart icon. Below this, there is a modal form with the following fields: "First Name:" with a placeholder "Enter first name here", "Last Name:" with a placeholder "Last Name", "Email:" with a placeholder "Email", "Address:" with a placeholder "Address", and "Shipping Method:" with a dropdown menu showing "Standard - 33\$". At the bottom of the form, there are two buttons: "Checkout: 182.27\$" and "Close". The background shows a blurred view of a shopping cart with items and an "Add" button.

U ovoj komponenti se unose podaci o korisniku. Ideja je bila da se ovi podaci cuvaju, ali posto na frontu ne moze da se edituje json fajl zbog security-ja to nije odradjeno. Svakako je predvidjeno kao da jeste, I u svrhu simulacije I završavanja ovog flow-a ne moze se naruciti ako se ne unesu bar neki podaci.

Na dugmetu za checkout pise ukupna cena sa selektovanom postarinom.

Na klik na checkout, posle uspesne validacije, formira se promenljiva tipa FormData, I appenduju se svi potrebni podaci u nju. Sa njom se nista ne radi iz gore navedenih razloga, ali tu je u svrhu demonstriranja format a promenljive koja bi se slala na server.

Jos jednom ponavljam, celokupna struktura podataka sto se tice ordera I orderInvoice-a se moze videti u dto sloju unutar tsBusinessLayer foldera. Jedino sto nije implementirano jesu modeli za Usera I za Order, iz gore navedenih razloga.

## 2.5. 404 – Not Found



Pri pokušaju pristupanja stranici koja nije definisana u ruter modelu prikazuje se 404 komponenta.

## 2.6. Modeli

Primer modela za knjigu:



```

declare var require: any;

import { BookPriceDTO } from "../dtos/BookPriceDTO";
import { IEntityGet } from "../interfaces/common/IEntityGet";
import { IEntityGetAll } from "../interfaces/common/IEntityGetAll";
import { TBookJSON } from "../types/JSON/TBookJson";
import { TBookPriceJson } from "../types/JSON/TBookPriceJson";

export class BookPriceModel implements IEntityGetAll, IEntityGet{
    private bookPrices: Array<BookPriceDTO> = [];

    constructor(){
        let bookPrices: Array<TBookPriceJson> = require("src/assets/data/bookPrice.json");
        this.bookPrices = this.convertAuthorFromJsonToDTO(bookPrices);
    }

    public getAll<BookPriceDTO>(): Array<BookPriceDTO> {
        return this.bookPrices as Array<BookPriceDTO>;
    }

    public get<BookPriceDTO>(id: number): BookPriceDTO {BookPriceDTO
        return this.bookPrices.filter(x=>x.id==id)[0] as BookPriceDTO;
    }

    public getBookPricesByBook(book: TBookJSON): Array<BookPriceDTO>{
        let bookPriceIdsArray = book.BookPrices;

        let bookPrices: Array<BookPriceDTO> = [];
        for(let bookPriceId of bookPriceIdsArray){
            bookPrices.push(this.get(bookPriceId))
        }

        return bookPrices.sort((x,y)=>{return x.CreatedAt>y.CreatedAt? -1: 1});
    }

    private convertAuthorFromJsonToDTO(bookPrices: Array<TBookPriceJson>): Array<BookPriceDTO>{
        let bookPricesDTO: Array<BookPriceDTO> = [];
        for(let bookPrice of bookPrices){
            bookPricesDTO.push({
                "id": bookPrice.id,
                "Price": bookPrice.Price,
                "CreatedAt": new Date(`${bookPrice.CreatedAt}`)
            })
        }

        return bookPricesDTO;
    }
}

```

## 2.7. Struktura tsBusinessLayer-a

