



# **Univerzitet Singidunum**

## **Tehnički fakultet**

### **Candy Store**

**-Projektni rad-**

**Predmet:** Klijentske Veb Aplikacije

**Profesor:**

Mlađan Jovanović

**Asistent:**

Nikola Savanović

**Studenti:**

Jovana Kržanović 2018/200030

Božidar Mladenović 2018/240254

Beograd, 2021 godine

# Sadržaj

1. Uvod .....	3
2. Implementacija .....	3
3. Prikaz implementacije korisničkog interfejsa .....	3
4. Prikaz korisničkog interfejsa .....	7
5. Zaključak .....	14

# 1. Uvod

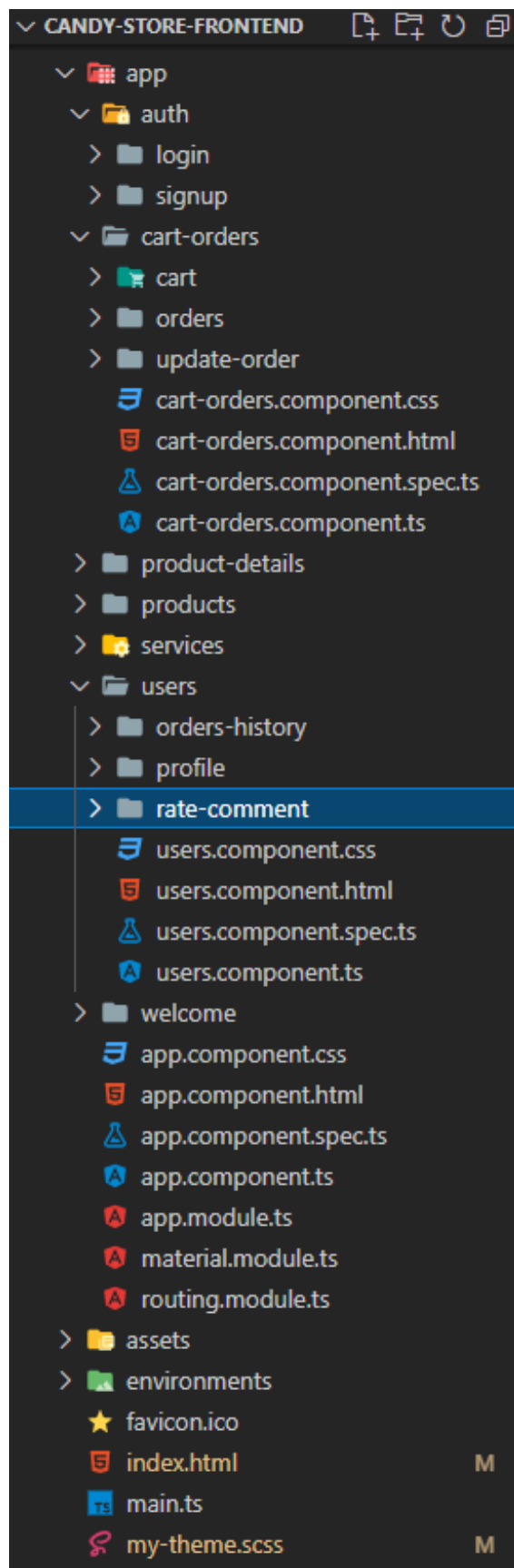
„Candy Store” je veb aplikacija koja se bavi prodajom slatkiša. Običan korisnik može da pregleda, filtrira, sortira i pretražuje slatkiše po navedenim kriterijumima. Ostalim funkcionalnostima sajta imaju pristup samo prijavljeni korisnici, i to su: odabir i dodavanje slatkiša u korpu, pregledanje sadržaja korpe, kao i brisanje stavki iz nje, pristup stranici porudžbine koje imaju status čekanja (pending), pristup profilnoj stranici gde se mogu pregledati i ažurirati podaci profila, stranici istorije porudžbina na kojoj se nalaze porudžbine koje imaju status otkazane (canceled) i završene (completed), gde se završene porudžbine mogu oceniti i komentarisati.

## 2. Implemantacija

Pored korisničkog interfejsa, aplikacija poseduje i pozadinsku logiku koja je pisana uz pomoć Spring Boot Framework-a, koji je zasnovan na Javi i MongoDB baze podataka. Što se tiče korisničkog interfejsa, korišćen je Angular Framework verzije 11 sa dodatkom Angular Material-a, koji omogućava konzistentan i intuitivan dizajn. Aplikacija se sastoji iz 14 komponenti (uključujući i root komponentu) i 5 servisa, od kojih je jedan auth guard koji zabranjuje neprijavljenim korisnicima pristup određenim stranicama, dok ostali služe za komunikaciju sa backend-om putem http client-a.

## 3. Prikaz implemantacije korisničkog interfejsa

Na slici 1. je prikazana struktura aplikacije. „Login” i „signup” komponente, kao što im i sam naziv kaže, u njima se nalazi kod za registraciju i prijavljivanje korisnika na aplikaciju. „Cart-orders” komponenta je roditeljska komponenta u kojoj se nalaze komponente „cart”, „orders” i „update-order”. Što se tiče „cart” komponente, tu se nalazi kod za celu logiku korpe. Zatim „orders” komponenta koja je zadužena za sve funkcionalnosti koje se tiču porudžbina koje su na čekanju i „update-order” komponenta koja je u stvari mat dialog u kom se nalazi logika vezana za ažuriranje tih porudžbina. „Product-details” komponenta sadrži informacije o proizvodu, komentarima, ocenama i formu za dodavanje tog proizvoda u korpu. Products komponenta je zadužena za prikazivanje svih proizvoda, u njoj se nalaze sve funkcionalnosti filtriranja, sortiranja, pretrage i paginacije. „Users” komponenta je roditeljska komponenta u kojoj se nalaze komponente. „Profile”, „orders-history” i „rate-comment”. U „profile” komponenti se nalazi logika vezana za pregledanje i ažuriranje korisničkih informacija, „orders-history” komponenta u kojoj je moguće pregledati i brisati završene i otkazane porudžbine i kod završenih porudžbina postoji dugme za ocenjivanje i komentarisanje koje otvara mat dialog komponentu „rate-comment”. „Welcome” komponenta je početna stranica aplikacije i ona je više skoncentrisana na dizajn nego na funkcionalnosti. I na kraju, „App” komponenta je roditeljska komponenta svim gore navedenim komponentama. App module je modul u kome se importuju sve komponente i svi moduli koji su potrebni za rad same aplikacije, material module je modul koji služi za importovanje angular material komponenti i routing module koji služi za rutiranje u ovoj aplikaciji. Što se tiče svih servisa, njihove funkcionalnosti su spomenute u prethodnom delu.



Slika 1. (prikaz strukture aplikacije)

```

1 <div fxFlex fxLayout="column">
2   <div fxFlex fxLayoutAlign="space-around center" id="search-cart-div">
3
4     <mat-form-field color='accent'>
5       <input type="text" ngModel matInput placeholder="Search" name="search"
6         search #searchInput="ngModel" (keyup) = "search(searchInput)">
7     </mat-form-field>
8
9     <mat-icon matBadge="{{cartNumber}}" *ngIf="username != null" matBadgeColor="accent" id="cartIcon" class="material-icons-outlined" fxLayoutAlign="center center" routerLink="/cart">
10      shopping_cart
11    </mat-icon>
12  </div>
13
14  <div fxFlex fxLayout="row">
15    <div class="sideFilter">
16      <div fxLayout="column" fxLayoutAlign="center center" fxLayoutGap="30px">
17        <mat-form-field appearance="fill" id="sort">
18          <mat-label>Sort By</mat-label>
19          <mat-select (selectionChange)="onSortChange($event)">
20            <mat-option value="name-asc">Name ASC</mat-option>
21            <mat-option value="name-dsc">Name DESC</mat-option>
22            <mat-option value="price-asc">Price ASC</mat-option>
23            <mat-option value="price-dsc">Price DESC</mat-option>
24            <mat-option value="date-asc">Date ASC</mat-option>
25            <mat-option value="date-dsc">Date DESC</mat-option>
26          </mat-select>
27        </mat-form-field>
28
29        <div class="check">
30          <h4>Price: {{value}} €</h4>
31          <mat-slider color="accent" (change)="onPriceChange($event)"
32            thumbLabel
33            step="0.1"
34            min="0"
35            max="50"
36            [(ngModel)]="value"
37            aria-label="units"
38            value>
39          </mat-slider>
40        </div>
41
42        <div class="check">
43          <h4>Categories:</h4>
44          <mat-radio-group
45            class="radio-group"
46            [(ngModel)]="categories"
47            (change)="onCategoryChange($event)">
48            <p><mat-radio-button value="all" name="all">All Categories</mat-radio-button></p>

```

Slika 2. (Izgled dela HTML fajla „products“ komponente)

```

1 import { Component, OnInit } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { from } from 'rxjs';
4 import { Products } from '../services/products.service.service';
5 import { sortBy } from 'sort-by-typescript';
6
7 @Component({
8   selector: 'app-products',
9   templateUrl: './products.component.html',
10  styleUrls: ['./products.component.css']
11 })
12 export class ProductsComponent implements OnInit {
13
14   constructor(private productService: Products, private router: Router ) { }
15
16   data: any;
17   copyData: any;
18   p: number = 1;
19   value: any;
20   cartNumber: string;
21   rating: any;
22   categories: any;
23   username: string;
24
25   ngOnInit(): void {
26     this.cartNumber = localStorage.getItem("cartNumber");
27     this.username = localStorage.getItem("username");
28     this.findAll();
29   }
30
31   onSortChange(sortType: any){
32
33     //SORT
34     if (sortType.value == ""){
35       this.findAll();
36     }
37     if (sortType.value == "name-asc"){
38
39       this.data = this.data.sort(sortBy("name"));
40     }
41     if (sortType.value == "name-dsc"){
42
43       this.data = this.data.sort(sortBy("-name"));
44     }
45     if (sortType.value == "price-asc"){
46
47       this.data = this.data.sort(sortBy("price"));
48     }
49     if (sortType.value == "price-dsc"){

```

Slika 3. (Prikaz dela TypeScript fajla „products“ komponente)

```

mat-card{
  margin: 15px 5px;
}

.main{
  margin-top: 8px;
}

#search-cart-div{
  width: 100%;
  background-color: #d6d6d6;
}

mat-form-field{
  width: 80%;
  text-align: center;
  font-size: 20px;
}

.sideFilter{
  width: 20%;
  background-color: #f3f3f3;
}

#cartIcon{
  font-size: 40px;
  color: #e04a78;
  cursor: pointer;
  padding: 27px;
  transition: 0.3s;
  border-radius: 100px;
}

#cartIcon:hover{
  background-color: #e04a78;
  border-radius: 100px;
  color: #f3f3f3;
  padding: 27px;
  transition: 0.3s;
}

#pagination{
  background-color: #f3f3f3;
  width: 100%;
  height: 8vh;
}

.check{
  width: 90%;
}

```

Slika 4. (Prikaz dela CSS fajla „products“ komponente)

```

@Injectables({
  providedIn: 'root'
})
export class Products {
  constructor(private http: HttpClient, private router: Router) {}

  public findAll(): Observable<HttpResponse<any>> {
    return this.http.get<any>("http://localhost:8080/candies/all");
  }

  public findAllByName(search: string): Observable<HttpResponse<any>> {
    return this.http.get<any>("http://localhost:8080/candies/search/" + search);
  }

  public findCandyById(id: string): Observable<HttpResponse<any>> {
    return this.http.get<any>("http://localhost:8080/candies/candy/" + id);
  }

  public findAllByCandiesId(id: string): Observable<HttpResponse<any>> {
    return this.http.get<any>("http://localhost:8080/comments/all-comments/" + id);
  }

  public insertComment(model: Comment): Observable<HttpResponse<any>> {
    return this.http.post<any>("http://localhost:8080/comments/insert", model);
  }

  public updateQuantity(model: Candies): Observable<HttpResponse<any>> {
    return this.http.post<any>("http://localhost:8080/candies/update-quantity", model);
  }

  public cartDeleteQuantity(model: QuanDeleteCart): Observable<HttpResponse<any>> {
    return this.http.post<any>("http://localhost:8080/candies/cart-delete-quantity", model);
  }

  public updateStars(model: UpdatingStars): Observable<HttpResponse<any>> {
    return this.http.post<any>("http://localhost:8080/candies/update-stars", model);
  }

  public cancelOrderBackQuantity(items: any): Observable<HttpResponse<any>> {
    return this.http.post<any>("http://localhost:8080/candies/cancel-order-quantity", items);
  }

  showCandy(id: string): any {
    this.router.navigate(['candies/candy/' + id]);
  }
}

```

Slika 5. (Prikaz dela products.service TypeScript fajla)

```

1 import { NgModule } from '@angular/core';
2 import { MatButtonModule } from '@angular/material/button';
3 import { MatToolbarModule } from '@angular/material/toolbar';
4 import { MatTabsModule } from '@angular/material/tabs';
5 import { MatFormFieldModule } from '@angular/material/form-field';
6 import { MatInputModule } from '@angular/material/input';
7 import { MatDatepickerModule } from '@angular/material/datepicker';
8 import { MatNativeDateModule } from '@angular/material/core';
9 import { MatCheckboxModule } from '@angular/material/checkbox';
10 import { MatCardModule } from '@angular/material/card';
11 import { MatGridListModule } from '@angular/material/grid-list';
12 import { MatTableModule } from '@angular/material/table';
13 import { MatSnackBarModule } from '@angular/material/snack-bar';
14 import { MatIconModule } from '@angular/material/icon';
15 import { MatBadgeModule } from '@angular/material/badge';
16 import { MatSelectModule } from '@angular/material/select';
17 import { MatSliderModule } from '@angular/material/slider';
18 import { MatRadioModule } from '@angular/material/radio';
19 import { MatDialogModule } from '@angular/material/dialog';
20
21
22 @NgModule({
23   imports: [
24     MatButtonModule,
25     MatToolbarModule,
26     MatTabsModule,
27     MatFormFieldModule,
28     MatInputModule,
29     MatDatepickerModule,
30     MatNativeDateModule,
31     MatCheckboxModule,
32     MatCardModule,
33     MatGridListModule,
34     MatTableModule,
35     MatSnackBarModule,
36     MatIconModule,
37     MatBadgeModule,
38     MatSelectModule,
39     MatSliderModule,
40     MatRadioModule,
41     MatDialogModule
42   ],
43   exports: [
44     MatButtonModule,
45     MatToolbarModule,

```

Slika 6. (Prikaz dela material.module TypeScript fajla)

```

import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import { LoginComponent } from '../auth/login/login.component';
import { SignupComponent } from '../auth/signup/signup.component';
import { CartOrdersComponent } from '../cart-orders/cart-orders.component';
import { ProductDetailsComponent } from '../product-details/product-details.component';
import { ProductsComponent } from '../products/products.component';
import { UsersComponent } from '../users/users.component';
import { WelcomeComponent } from '../welcome/welcome.component';

const route: Routes = [
  {path: '', component: WelcomeComponent},
  {path: 'signup', component: SignupComponent},
  {path: 'login', component: LoginComponent},
  {path: 'products', component: ProductsComponent},
  {path: 'cart', component: CartOrdersComponent},
  {path: 'candies/candy/:id', component: ProductDetailsComponent},
  {path: 'users', component: UsersComponent}
]

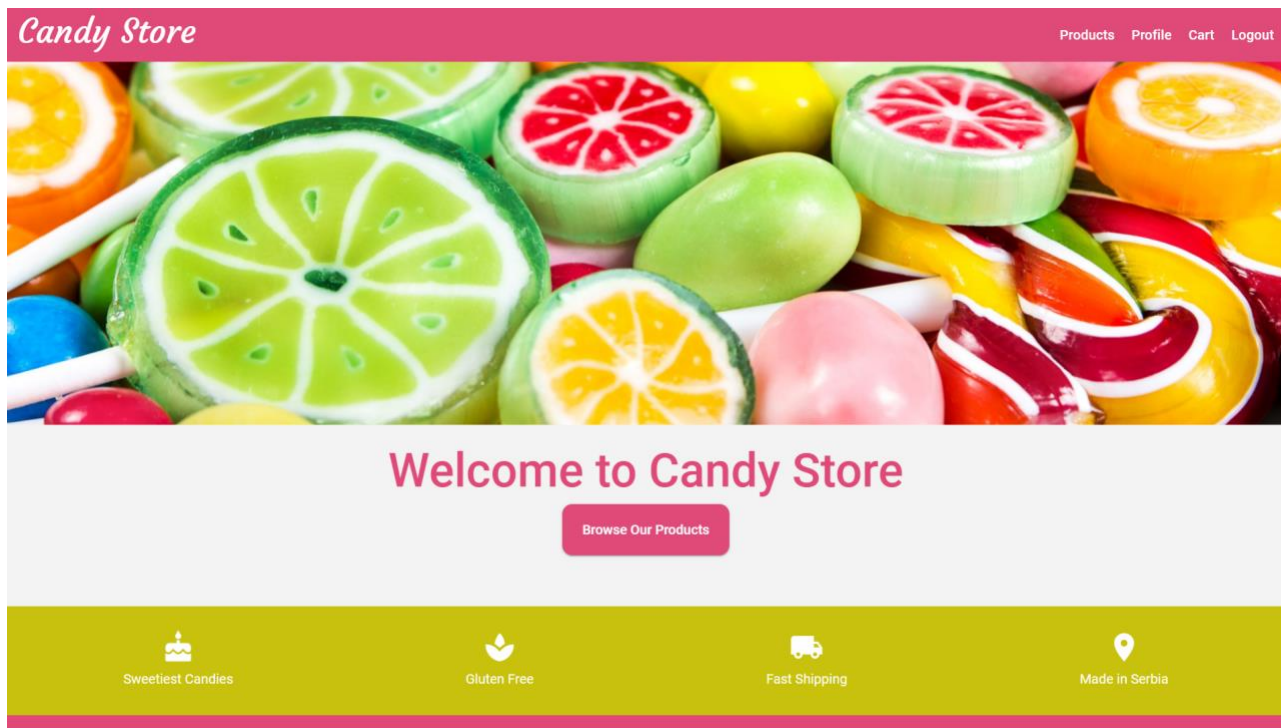
@NgModule({
  imports: [
    RouterModule.forRoot(route)
  ],
  exports: [
    RouterModule
  ]
})

export class RoutingModule {}

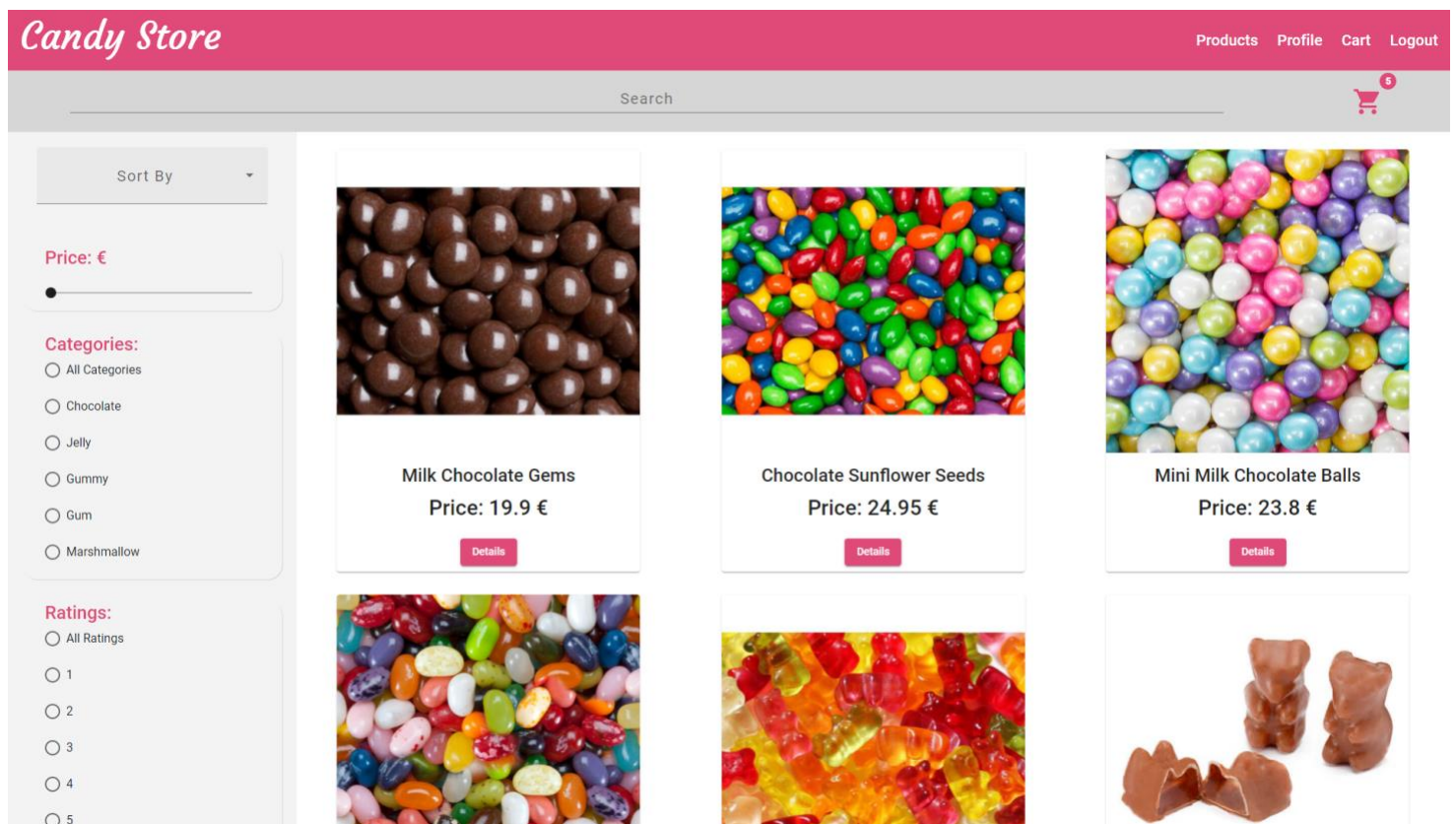
```

Slika 7. (Prikaz routing.module TypeScript fajla)

## 4. Prikaz korisničkog interfejsa



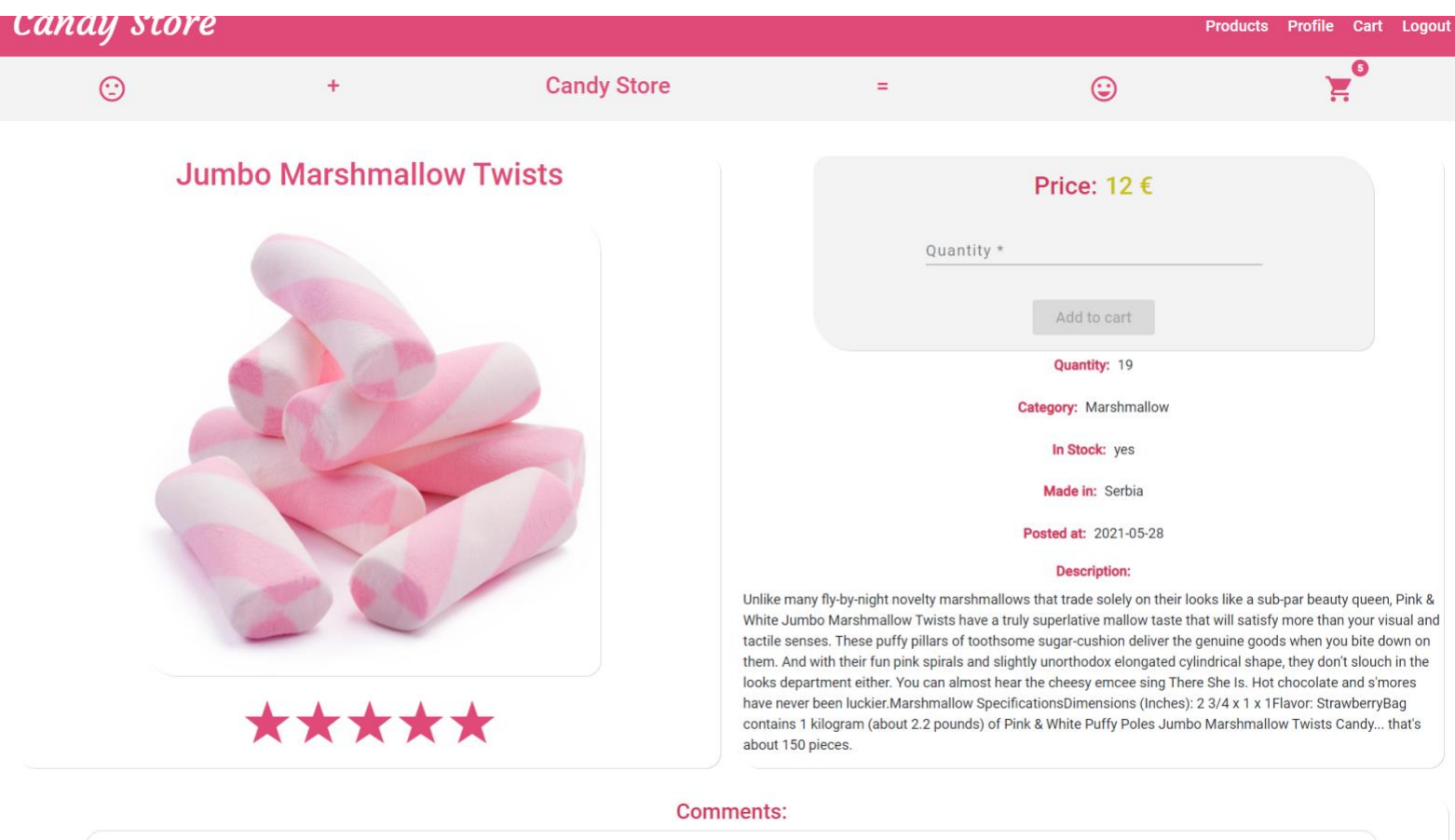
Slika 8. (Izgled „welcome“ stranice)



Slika 9. (Izgled „products“ stranice)








Na slici 9. su prikazani svi proizvodi, zatim sa leve strane je prikazan sidebar, u kome se nalazi select polje za sortiranje koje sortira po određenom kriterijumu, mat-slider koji služi za filtriranje proizvoda po ceni i dve mat-radio grupe u kojim se nalaze radio buttoni koji služe za filtriranje proizvoda po kategoriji i oceni. Zatim iznad njih se nalazi search input polje za pretragu proizvoda koji ima keyup event listener što znači da se pri svakom otkucanom slovu pretraga filtrira i ikonica korpe sa malim kružićem koji prikazuje koliko ima proizvoda u korpi. Svaki od proizvoda sadrži i dugme za preusmeravanje na product-details stranicu (slika 10).



Slika 10. (Prikaz „product-details“ stranice)

Na slici 10. su prikazane informacije o jednom proizvodu, kao i njegova ocena i komentari, takođe se nalazi forma koja sadrži input polje količina i dugme za dodavanje u korpu. Proizvod se može dodati u korpu samo od strane prijavljenih korisnika, ako je taj proizvod na stanju, tj. ako mu je količina veća od 0 i nije moguće dodati veću količinu u korpu od navedene.



Id	Image	Name	Quantity	Price	Action
2		Chocolate Sunflower Seeds	1	24.95	<button>Delete</button>
3		Jelly Belly Beans	1	10.5	<button>Delete</button>
5		Mini Milk Chocolate Balls	1	23.8	<button>Delete</button>
1		Milk Chocolate Gems	1	19.9	<button>Delete</button>
4		Mini Milk Chocolate Balls	1	23.8	<button>Delete</button>




**Total Price:****102.95 €**

Payment Met... ▾

Checkout

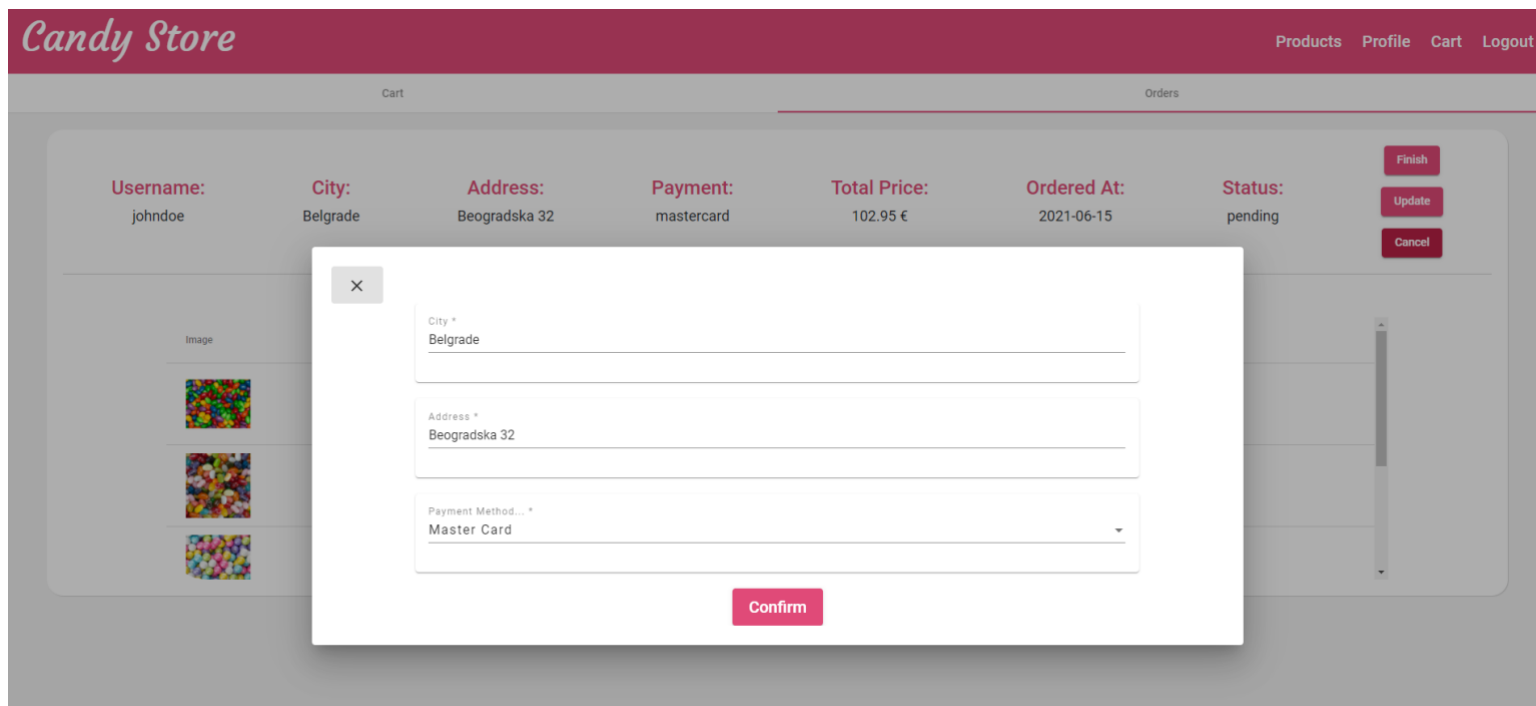
Slika 11. (Prikaz „cart“ stranice)

Na slici 11. prikazana je „cart“ stranica na kojoj se vidi tabela proizvoda koji su dodati u korpu i koji mogu da se obrišu. Kada sa obrišu, količina obrisanog proizvoda će se vratiti nazad u bazu. Sa desne strane se nalazi ukupna cena proizvoda i forma u kojoj se nalazi select polje za odabir načina plaćanja i dugme checkout koje pravi porudžbinu koja će se prikazati na „orders“ stranici gde se nalaze porudžbine na čekanju (slika 12).

<b>Username:</b> john doe	<b>City:</b> Belgrade	<b>Address:</b> Beogradska 32	<b>Payment:</b> mastercard	<b>Total Price:</b> 102.95 €	<b>Ordered At:</b> 2021-06-15	<b>Status:</b> pending	<b>Finish</b> <b>Update</b> <b>Cancel</b>
<b>Items:</b>							
	Chocolate Sunflower Seeds	1	24.95				
	Jelly Belly Beans	1	10.5				
	Mini Milk Chocolate Balls	1	23.8				

Slika 12. (Prikaz „orders“ stranice)

Na slici 12. prikazana je „orders“ stranica na kojoj se nalaze porudžbine koje su u stanju čekanja, gde postoje tri dugmeta „finish“, „update“ i „cancel“. Kada se klikne na dugme „finish“ porudžbina menja status u completed i prelazi na stranicu „orders-history“, tako i dugme „cancel“ kada se klikne porudžbina dobija status canceled, takođe se prebacuje na drugu stranicu i svakoj stavci porudžbine se vraća količina u bazu. Što se tiče „update“ dugmeta, ono otvara mat-dialog „update-order“ (slika 13).






Slika 13. (Prikaz mat-dialoga „update-order“)

Na slici 13. prikazan je mat-dialog „update-order“ koji sadrži tri mat-form-field polja uz pomoć kojih može da se promeni grad, adresa i način plaćanja porudžbine.



<b>Username:</b>	<b>City:</b>	<b>Address:</b>	<b>Payment:</b>	<b>Total Price:</b>	<b>Ordered At:</b>	<b>Status:</b>	<a href="#">Delete Order</a>
johndoe	Belgrade	Beogradska 32	paypal	45.7 €	2021-06-14	completed	

## Items:

Image	Name	Quantity	Price	Action
	Ice Cream Cones	1	13.8	<a href="#">Rate and Comment</a>
	Jumbo Marshmallow Twists	1	12	<a href="#">Rate and Comment</a>
	Gold Gummy Bears	1	19.9	<a href="#">Rate and Comment</a>

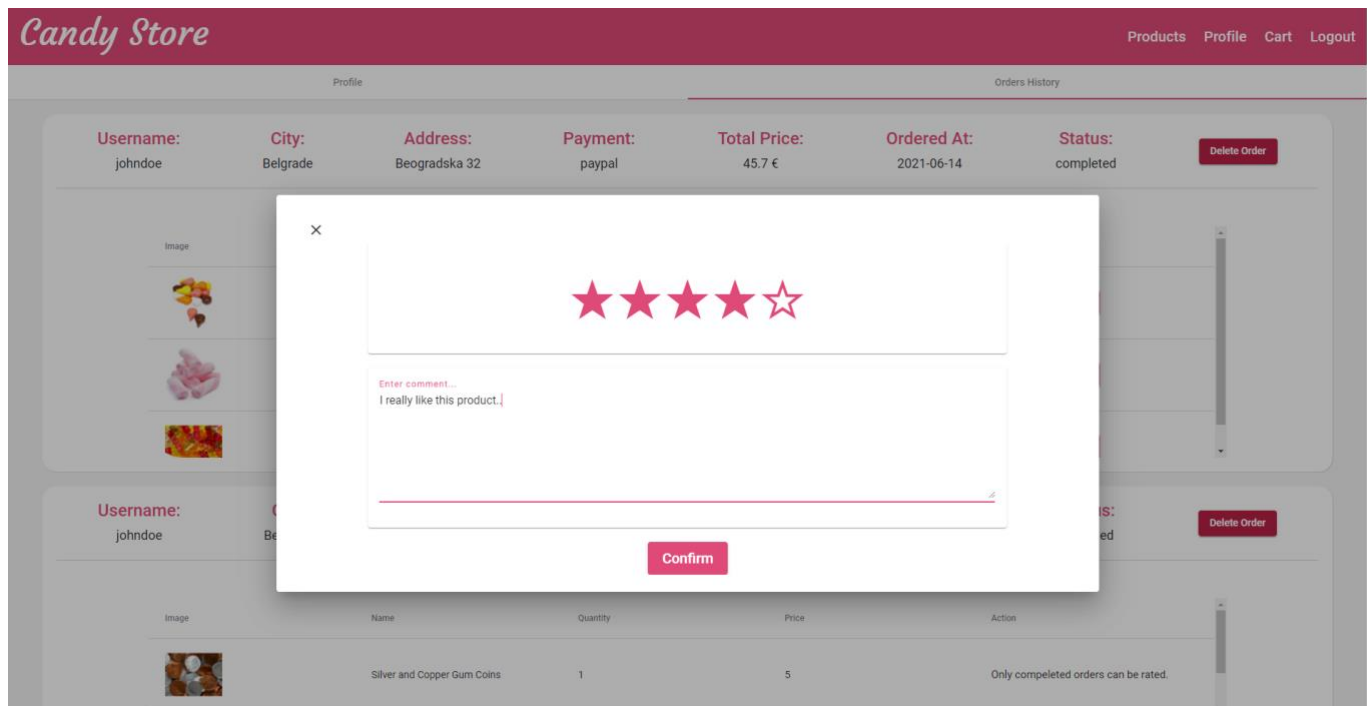
<b>Username:</b>	<b>City:</b>	<b>Address:</b>	<b>Payment:</b>	<b>Total Price:</b>	<b>Ordered At:</b>	<b>Status:</b>	<a href="#">Delete Order</a>
johndoe	Belgrade	Beogradska 32	visa	50 €	2021-06-14	canceled	

## Items:

Image	Name	Quantity	Price	Action
	Silver and Copper Gum Coins	1	5	Only completed orders can be rated.
	Silver and Copper Gum Coins	1	5	Only completed orders can be rated.

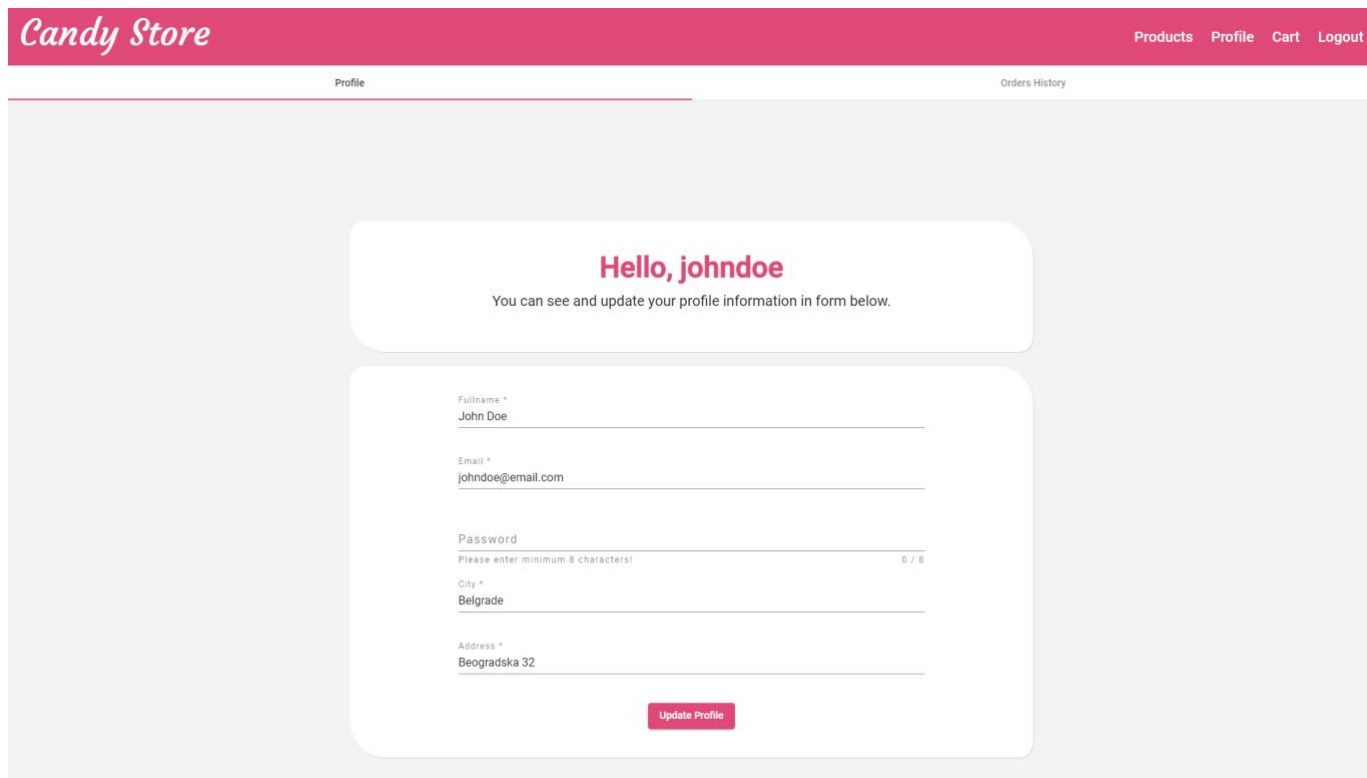
Slika 14. (Prikaz „orders-history“ stranice)

Na slici 14. prikazana je stranica „orders-history“ gde se nalaze porudžbine koje imaju status canceled i completed. Na ovoj stranici se takođe nalazi i delete order dugme koje služi da obriše porudžbinu iz istorije, bilo da je completed ili canceled, dok porudžbine sa statusom complete imaju funkcionalnost da se u njima ocenjuju i komentarišu sve stavke porudžbine, naravno po izboru korisnika. Što se tiče dugmeta za ocenjivanje i komentarisanje, klikom na njega otvara se mat dialog u kome se nalazi jedna forma uz pomoć koje možemo da selektujemo broj zvezdica i jedan text area input uz pomoć kog možemo da unesemo komentar (slika 15).



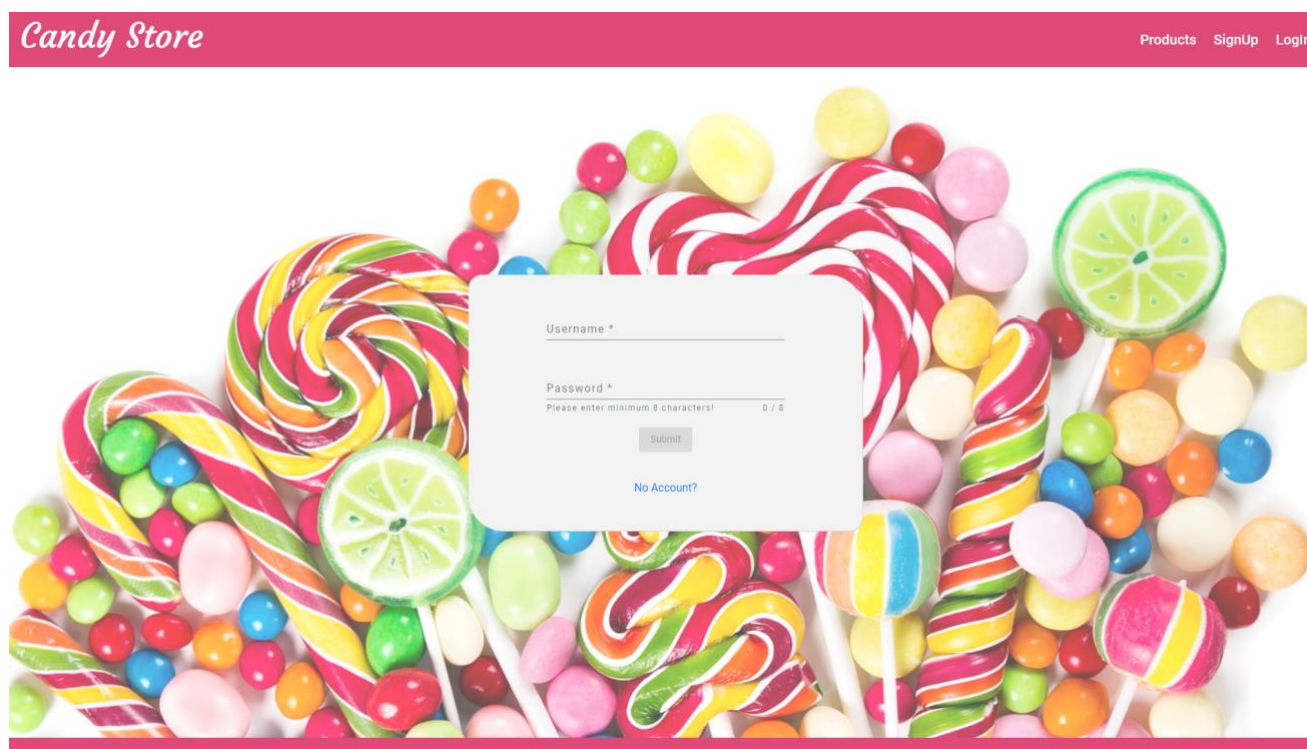
Slika 15. (Izgled mat-dialoga za ocenjivanje i komentarisanje)

Kao što je već napomenuto, ovde korisnik bira broj zvezdica koji želi da dodeli proizvodu i u input polje unosi željeni komentar ukoliko to želi. Klikom na dugme submit, ocene i komentar se dodeljuju tom proizvodu, i oni mogu da se vide na „product-details“ stranici za taj proizvod. U gornjem levom uglu imaju i opciju close za zatvaranje mat dialoga, tako da ne mora ništa da unese ako ne želi.

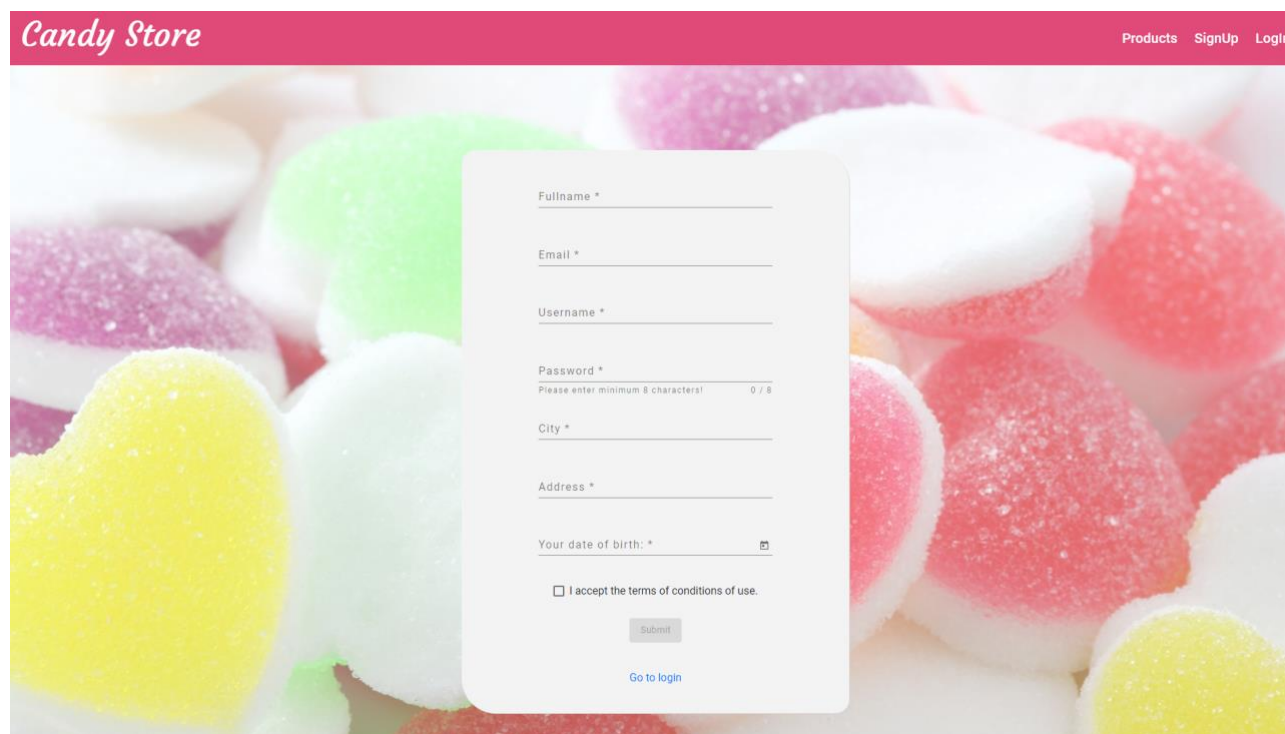


Slika 16. (Izgled „profile“ stranice)

Na stranici „profile“ korisnici mogu da vise svoje podatke i da ih ažuriraju (slika 16).



Slika 17. (Prikaz „login“ stranice)



Slika 18. (Prikaz „register“ stranice)

Na slikama 17. i 18. prikazane su „login“ i „register“ stranice i kao što se vidi na tim slikama, na njima se nalaze forme za unošenje podataka. Da bi se korisnik ulogovao, mora da poseduje korisnički nalog koji kreira na register stranici i naravno, potrebno je da unese ispravne kredencijale za logovanje.

Još nešto što nije spomenuto, je da se navbar menja u zavisnosti da li je korisnik ulogovan ili ne.



Slika 19. (izgled navbar-a za neulogovane korisnike)



Slika 20. (izgled navbar-a za ulogovane korisnike)

## 5. Zaključak

Cilj izrade ovog projekta je bila primena znanja i veština stečenih na predmetu „Klijentske Veb Aplikacije“. Iako su ispoštovani svi projektni zahtevi i aplikacija sadrži i stranu pozadinske logike, ne može se primenjivati u industriji jer je potrebno poraditi na samoj sigurnosti i optimizaciji aplikacije, kao i dodavanju admin panela, koji bi služio admin korisnicima da održavaju celu aplikaciju.