



**UNIVERZITET U ZENICI**  
**POLITEHNIČKI FAKULTET**



**MINTY**  
**DOKUMENTACIJA**

**Predmet: Web Dizajn**

**Student: Zijad Doglod**

**Indeks: 185**

**Studij: Softversko Inženjerstvo**

**Zenica, maj 2022**

# **SADRŽAJ**

|  |    |
|--|----|
| I. UVOD.....                                     | 3  |
| II. IZGLED STRANICE .....                        | 4  |
| III. ADMINISTRATORSKI POGLED NA STRANICU .....   | 10 |
| IV. IZGLED STRANICE NA RAZLIČITIM UREĐAJIMA..... | 16 |

## I. UVOD

Projekat Minty iz predmeta Web Dizajn predstavlja „Online delivery food“ gdje krajnji korisnici mogu da vide koja je hrana u ponudi te istu naruče. Administratori imaju privilegije da hranu u aplikaciji uređuju, brišu ili dodaju, zavisno od potreba same firme kojoj će biti dostavljena ova aplikacija.

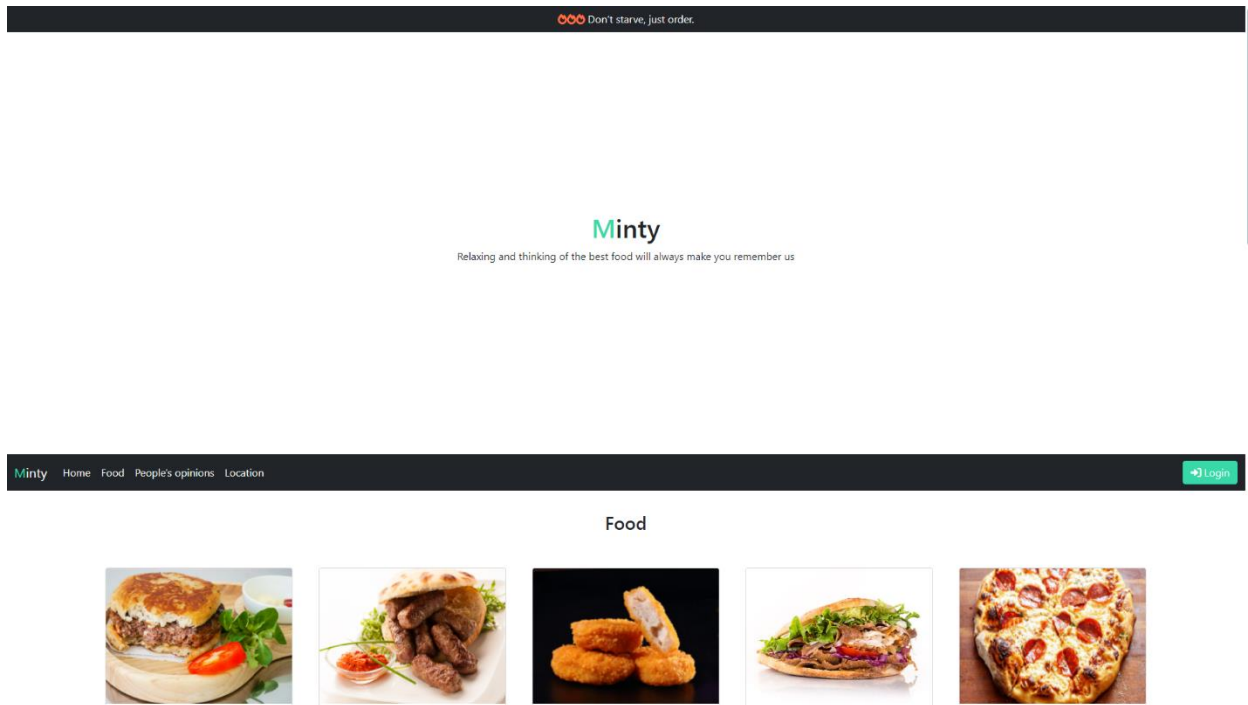
Za potrebe ovog projekta su korištene sljedeće tehnologije:

- HTML5
- CSS3
- JavaScript
- Bootstrap 5
- FontAwesome

U projektu postoji samo jedan HTML fajl i u njemu je predefinisana verzija HTML-a, a to je HTML5. CSS3 je korišten za neke sekcije u kojima nije bilo moguće izvesti zamišljeno sa Bootstrap 5, a također korištene su i ikone iz FontAwesome čisto radi estetskog izgleda. U nastavku dokumentacije će biti detaljno objašnjeno šta je urađeno. Struktura fajlova je bazirana na 3 fajla u kojima je urađeno sve, a to su HTML, CSS i JS fajl. Također tu se nalazi i folder sa slikama koje su upotrebljene na samom projektu.

Zbog nemogućnosti mijenjanja, dodavanja ili brisanja hrane direktno sa API-a, u JavaScript kodu su vršene određene improvizacije kako bi se postigao efekat koji je prvobitno trebao biti urađen. Nakon refresh-ovanja stranice, sve kartice koje su bile uređene se vraćaju na zadane postavke, one koje su obrisane se vraćaju, a one koje su dodane nestaju.

## II. IZGLED STRANICE



*Slika 2.1 Izgled stranice nakon učitavanja*

Na slici 2.1 je prikazan news feed na vrhu stranice koji se sastoji od nekoliko poruka koje se izmjenjuju svakih 5 sekundi. Zatim se nalazi logo Minty stranice sa dodanom porukom ispod te nakon toga navigation bar koji sadrži button-e koji vode do određenog dijela stranice i button-a koji je namijenjen administratorima stranice. U navigation bar-u se nalaze button-i koji vode do početka stranice, sekcije za hranu, sekcije gdje su ljudi dali svoja mišljenja o Minty-u te lokacija Minty-a u gradu Zenica.

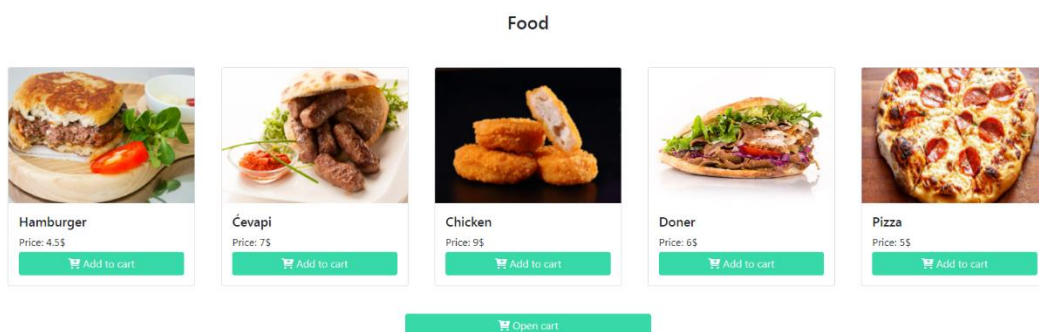
```

1 let counterForNewsMessage = 0;
2
3 document.querySelector('#header__news-feed').innerHTML =
4
5 <i class="fa-solid fa-fire news-feed__i"></i><i class="fa-solid fa-fire ne
6 ws-feed__i"></i><i class="fa-solid fa-fire news-feed__i"></i>
7 &nbsp;The food at your doorstep.';
8
9 setInterval(() => {
10   const message = document.querySelector('#header__news-feed');
11
12   const messages = [
13     '&nbsp;The food at your doorstep.',
14     '&nbsp;Why starve when you have&nbsp;<strong><span style="color: #38d9a9">M</
15 span><span class="text-light">inty</span></strong>.',
16     '&nbsp;Don't starve, just order.',
17     '&nbsp;We will make your mood.',
18     '&nbsp;Hunger gives stress; we give food.',
19     '&nbsp;Delivering happiness.'
20   ];
21
22   message.innerHTML = `
23 <i class="fa-solid fa-fire news-feed__i"></i><i class="fa-solid fa-fire ne
24 ws-feed__i"></i><i class="fa-solid fa-fire news-feed__i"></i>
25
26   + messages[(counterForNewsMessage =
27     (counterForNewsMessage+1) % messages.length)];
28 }, 5000);

```

Slika 2.2. JavaScript kod koji prikazuje poruke u news-feed svakih 5 sekundi

Na slici 2.2 se nalazi JavaScript kod koji svakih 5 sekundi promjeni poruku u news feed-u. Na početku se nalazi predefinisana poruka koja se prikaže kada se tek stranica učita. Nakon toga slijedi interval koji izmjenjuje poruke u news feed sekciji.



Slika 2.3. Izgled sekcije „Food“

Na slici 2.3 je prikazana sekcija „Food“ u kojoj korisnik može da doda hranu, po želji, u korpu. Nakon toga, korisnik popunjava formu koja se prikaže, nakon što se pritisne button „Open cart“. Nakon što korisnik popuni podatke i pritisne button „Order food“ ispiše mu se poruka da je uspješno naručio hranu.

Order food

Name: Hamburger Price: 4.5

Name: Doner Price: 6

Price: 10.5\$

Full name:

Zijad Doglod

Address:

Fakultetska 1

Phone number:

0603240384

Order food

*Slika 2.4. Izgled forme koju korisnik popunjava prilikom naručivanja hrane*

```

1  const putFoodInOrder = (food) => {
2    const foodId = food.parentElement.parentElement.id;
3    const foodCard = document.querySelector(`[id = '${foodId}']`);
4    const foodName = foodCard.children[1].firstElementChild.innerText;
5    const foodPrice = foodCard.children[1].children[1].children[0].
      innerText;
6    const foodOrder = document.querySelector('#modal-body__ordered-foods'
7    );
8    const priceContainer = document.querySelector('
9    #modal-body__ordered-food-price');
10   let price = priceContainer.innerText;
11
12   price = parseFloat(price);
13
14   popup('Food ${foodName} is successfully added to cart !');
15
16   let order = `
17     <li class="list-group-item d-flex justify-content-between align-i
18     tems-center">
19       Name: ${foodName} Price: ${foodPrice}
20     </li>
21   `;
22
23   price += parseFloat(foodPrice);
24   priceContainer.innerText = price;
25
26   foodOrder.innerHTML += order;
27 }
28
29 const orderFood = () => {
30   const priceContainer = document.querySelector('
31   #modal-body__ordered-food-price').innerText;
32
33   if(priceContainer == 0) {
34     errorPopup('Unable to order food because cart is empty !');
35     document.querySelector('#modal-body__customer-form').reset();
36     document.querySelector('#order-food-close-button').click();
37   } else {
38     popup('
39     You have successfully ordered food. It will be with you in a few minutes
40     ! Stay with Minty :)
41   ');
42     document.querySelector('#modal-body__ordered-foods').innerHTML =
43     '';
44     document.querySelector('#modal-body__ordered-food-price').
45     innerHTML = '0';
46     document.querySelector('#modal-body__customer-form').reset();
47     document.querySelector('#order-food-close-button').click();
48   }
49 }

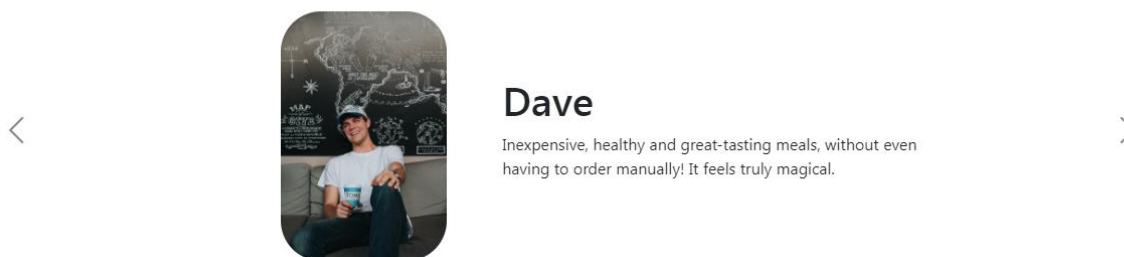
```

*Slika 2.5. JavaScript kod koji uzima podatke o korisniku i ubačenoj hrani u korpu*

Zatim se nalazi sekcija ljudi koji su dali svoje mišljenje o Minty-u. Tu je iskorišten Bootstrap 5 slider i ručno kreirane kartice koje, na lijep i jednostavan način, prikazuju slike ljudi,

njihova imena i mišljenja. Na slici 2.6. je prikazana sekcija „People's opinion“.

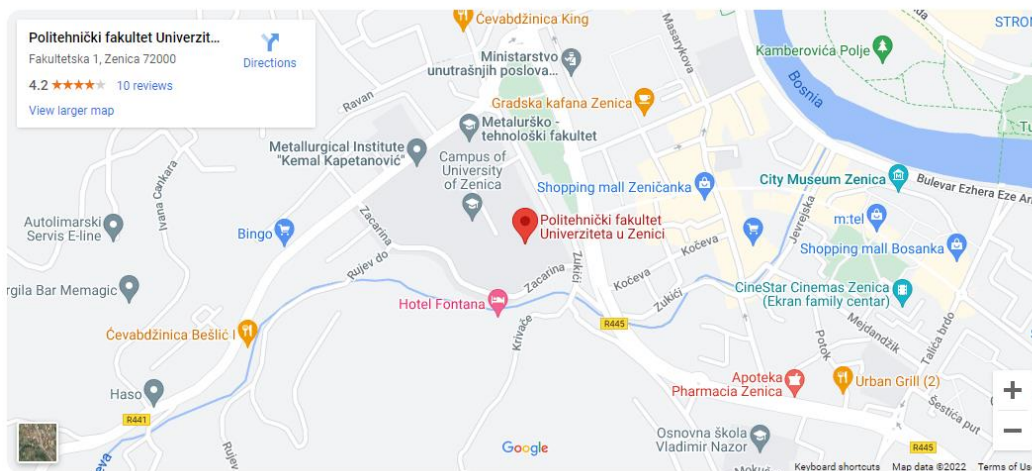
What other thinks about us ?



Slika 2.6. Sekcija „People's opinions“

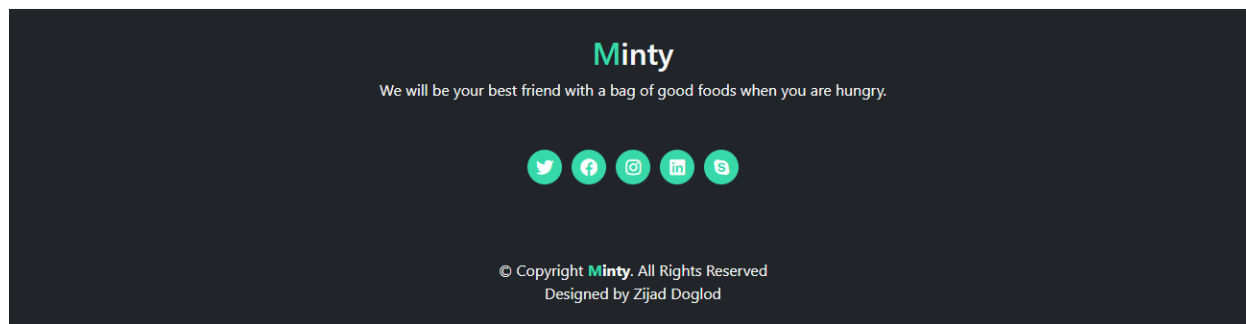
Na slici 2.7. je prikazana mapa lokacije Politehničkog fakulteta iz razloga što Minty u stvarnom svijetu ne postoji te ova sekcija daje slikovit prikaz kako bi to u stvarnosti izgledalo.

Minty's location



Slika 2.7. Sekcija „Minty's location“



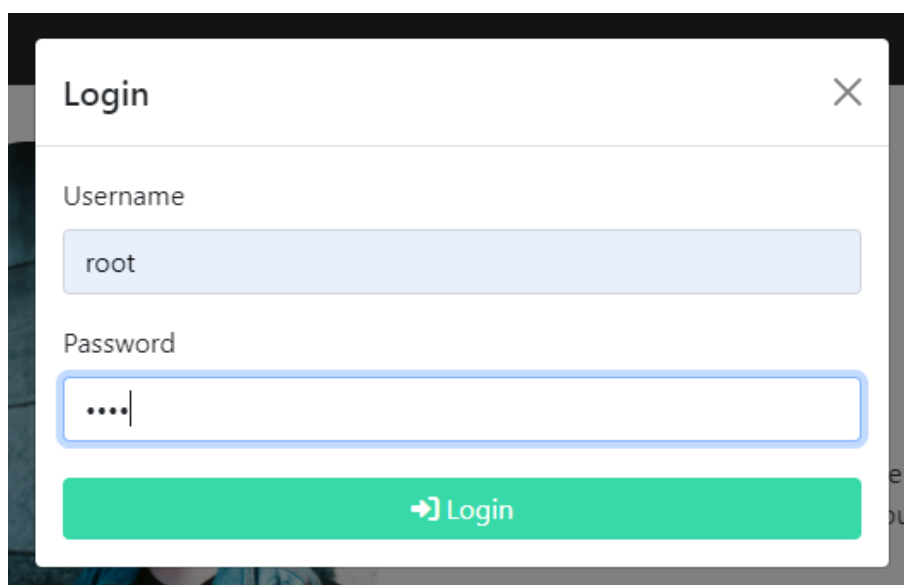


*Slika 2.8. Sekcija footer*

Na slici 2.8 je prikazan ručno izrađeni footer sa logom Minty stranice, porukom, linkovima koji bi trebali da vode do određene stranice i na samom kraju Minty copyright i autorska prava.

### III. ADMINISTRATORSKI POGLED NA STRANICU

Da bi se Administrator prijavio na stranicu mora da popuni formu koja se prikaže nakon što se pritisne button „Login“ u navigation baru, što je prikazano na slici 2.1. Nakon što popuni formu sa predefinisanim username-om i password-om, administratoru će se prikazati administratorske privilegije na stranici. Razlog predefinisaniog username-a i password-a je taj što API nema mogućnost dodavanja, brisanja ili eventualnog uređivanja korisnika te nije moguće odraditi login ili eventualno prijavu na stranicu. Predefinisani username je root, kao i password. Na slici 3.1. je prikazana login forma.

The image shows a web application login form. It has a title bar with the word "Login" and a close button (X). Below the title bar, there are two input fields. The first is labeled "Username" and contains the text "root". The second is labeled "Password" and contains masked characters "....". Below these fields is a large green button with a right-pointing arrow and the text "Login".

*Slika 3.1. Login forma*

Za potrebe login forme je iskorištena funkcija `loginCheck()` koja uzima podatke iz forme i provjerava ih. Ako su ispravni poziva se druga funkcija `rootControls()` koja prikazuje administratorske button-e na stranici, a skriva buttone koji su namijenjeni običnim korisnicima, a ako nisu ispravni, onda se ispiše poruka ispod button-a „Login“ da je username ili password pogrešan. Na slici 3.2. su prikazane upravo te 2 funkcije koje su namijenjene za tu svrhu.

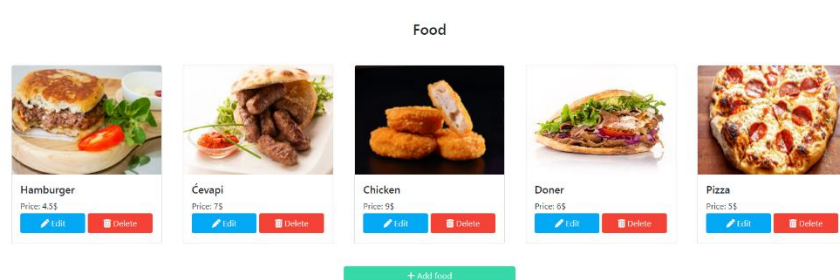
```

1  const loginCheck = () => {
2      const loginUsername = document.querySelector('#username').value;
3      const loginPassword = document.querySelector('#password').value;
4
5      if (loginUsername === 'root') {
6          if (loginPassword === 'root') {
7              document.querySelector('#login-form-close-button').click();
8              rootControls();
9              popup('You have successfully logged in !');
10             document.querySelector('#modal-body__login-form').reset();
11         } else {
12             const warningMessage = document.querySelector('#warning-message');
13             warningMessage.style.display = 'block';
14             warningMessage.innerText = 'Invalid password.';
15         }
16     } else {
17         const warningMessage = document.querySelector('#warning-message');
18         warningMessage.style.display = 'block';
19         warningMessage.innerText = 'Invalid username';
20     }
21 }
22
23 const rootControls = () => {
24     const cards = document.querySelectorAll('.cards__root-buttons');
25     const orderFoodButtons = document.querySelectorAll('.card-body__add-to-cart');
26     document.querySelector('#food__root-button').style.display = 'block';
27     document.querySelector('#navbar-collapse__login-button').style.display = 'none';
28     document.querySelector('#navbar-collapse__logout-button').style.display = 'inline-block';
29     document.querySelector('#food__order-button').style.display = 'none';
30
31     for (let i = 0; i < orderFoodButtons.length; i++) {
32         orderFoodButtons[i].style.display = 'none';
33     }
34
35     for (let i = 0; i < cards.length; i++) {
36         cards[i].style.display = 'inline-block';
37     }
38 }

```

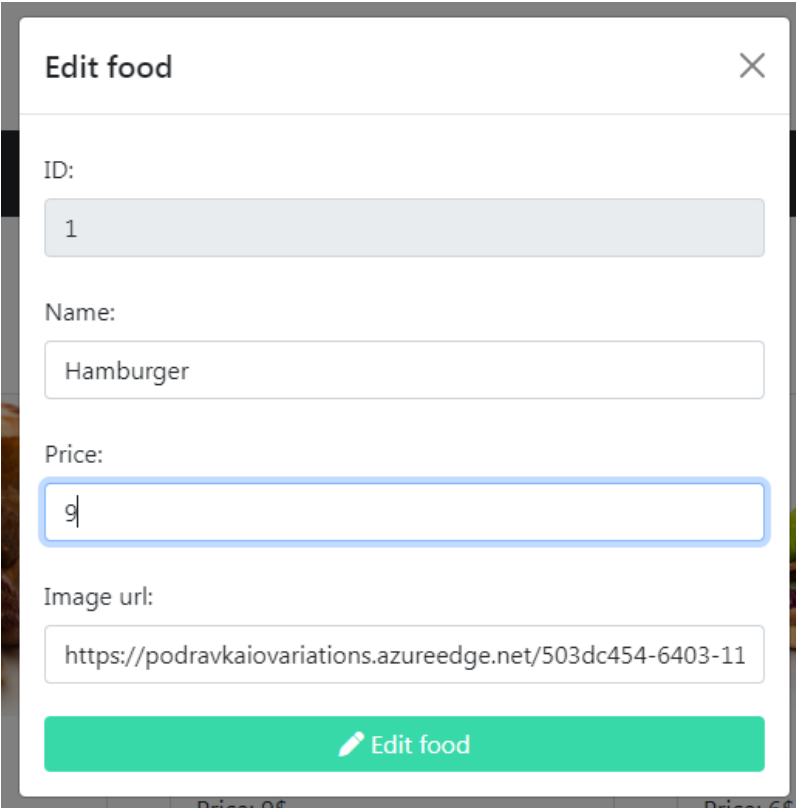
Slika 3.2. Funkcije koje su namijenjene za svrhu login-a i prikazivanja administratorskih button-a na stranici

Administratorske privilegije su prikazane samo u sekciji „Food“ gdje postoji mogućnost da se hrana uredi, obriše ili doda. Na slici 3.3. je dat pogled na stranicu od strane administratora.



Slika 3.3. Administratorski pogled na dio stranice pod nazivom „Food“

Kao što je prikazano na slici 3.3., na svakoj kartici se nalaze 2 button-a za uređivanje konkretne kartice ili njeno eventualno brisanje. Zatim se ispod kartica nalazi button „Add food“ koji ima namjenu da doda novu hranu u ponudu. Kada korisnik pritisne button „Edit“ otvori mu se forma sa popunjenim podacima o konkretnoj kartici, kao npr. id, ime hrane, cijena i slika. Administratoru je dato na pravo da mijenja samo ime, cijenu ili sliku hrane, dok ID nema pravo mijenjati kako se ne bi gubio izvorni ID zbog daljeg mijenjanja informacija o hrani na kartici. Nakon što administrator klikne button za uređivanje hrane, podaci se šalju na API te ako API vrati status code 204, informacije o hrani će se promijeniti na kartici, u suprotnom neće. Cjeloukupan proces je prikazan sa slikama 3.4. i 3.5.



The image shows a web application form titled "Edit food". The form is a modal window with a close button (X) in the top right corner. It contains the following fields:

- ID:** A text input field containing the value "1".
- Name:** A text input field containing the value "Hamburger".
- Price:** A text input field containing the value "9". This field is highlighted with a blue border.
- Image url:** A text input field containing the URL "https://podravkaiovariations.azureedge.net/503dc454-6403-11".

At the bottom of the form is a green button with a pencil icon and the text "Edit food".

Slika 3.4. Forma za uređivanje hrane

```

1  const fillEditData = (foodId) => {
2    const food = document.querySelector(`[id = '${foodId}']`);
3    const foodFormId = document.querySelector('#food-edit-id');
4    const foodFormName = document.querySelector('#food-edit-name');
5    const foodFormImage = document.querySelector('#food-edit-imageUrl');
6    const foodFormPrice = document.querySelector('#food-edit-price');
7
8    foodFormId.value = food.id;
9    foodFormName.value = food.children[1].children[0].innerText;
10   foodFormImage.value = food.children[0].src;
11   foodFormPrice.value = food.children[1].children[1].children[0].
    innerText;
12 }
13
14 const editFoodForm = document.querySelector('#edit-food-form');
15 let foodId = 0;
16 editFoodForm.addEventListener('show.bs.modal', (event) => {
17   let editRootButton = event.relatedTarget;
18   foodId = editRootButton.getAttribute('data-bs-whatever');
19 });
20
21 const editFood = () => {
22   const foodName = document.querySelector('#food-edit-name').value;
23   const foodPrice = document.querySelector('#food-edit-price').value;
24   const foodImageUrl = document.querySelector('#food-edit-imageUrl').
    value;
25
26   document.querySelector('#edit-food-close-button').click();
27   document.querySelector('#modal-body__edit-food-form').reset();
28
29   fetch('https://ptf-web-dizajn-2022.azurewebsites.net/api/Food', {
30     method: 'PUT',
31     headers: new Headers({'content-type': 'application/json'}),
32     body: JSON.stringify({
33       id: foodId,
34       name: foodName,
35       price: foodPrice,
36       imageUrl: foodImageUrl
37     })
38   })
39   .then(res => {
40     console.log('Status code: ${res.status}');
41
42     if (res.ok) {
43       let foodCard = document.querySelector(`[id = '${foodId}']`);
44       foodCard.children[1].firstElementChild.innerText = foodName;
45       foodCard.children[1].children[1].children[0].innerText =
        foodPrice;
46       foodCard.children[0].src = foodImageUrl;
47
48       popup('Food with ID: ${foodId} is successfully edited !');
49     } else {
50       errorPopup('Unable to edit food with ID: ${foodId} !');
51     }
52   })
53 }
54

```

Slika 3.5. Cjeloukupan kod u JavaScript-u koji radi postupak uređivanja hrane

Brisanje hrane se izvodi jednostavnim klikom na button „Delete“. Nakon klika, ID hrane se šalje na API te ako API vrati status code 204, kartica se briše sa stranice, u suprotnom kartica ostaje tu sa prikazanom porukom da nije moguće obrisati hranu. Na slici 3.6. je prikazana funkcija koja briše hranu sa stranice.

```

1 const deleteFood = (foodId) => {
2   fetch(`https://ptf-web-dizajn-2022.azurewebsites.net/api/Food/${foodId}`)
3     .then(res => {
4       console.log('Status code: ${res.status}');
5
6       if (res.ok) {
7         let foodCard = document.querySelector(`[id = '${foodId}']`);
8         foodCard.remove();
9         popup('Food with ID: ${foodId} is successfully deleted !');
10      } else {
11        errorPopup('Unable to delete food with ID: ${foodId}');
12      }
13    })
14  }
15 }
16 }

```

Slika 3.6. Funkcija koja briše hranu sa stranice

Dodavanje hrane na stranicu se izvodi jednostavnim popunjavanjem forme koja se otvori kada se klikne button „Add food“. Kada se forma popuni i klikne button „Add food“, podaci o dodanoj hrani se šalju na API te ako API vrati status code 201, kartica sa tom hranom će biti dodana na stranicu, u suprotnom neće. Na slici 3.7. je prikazan forma kako izgleda dodavanje hrane na stranicu, dok je na slici 3.8. prikazan kod koji je namijenjen za tu svrhu.

Slika 3.7. Forma za dodavanje hrane

```

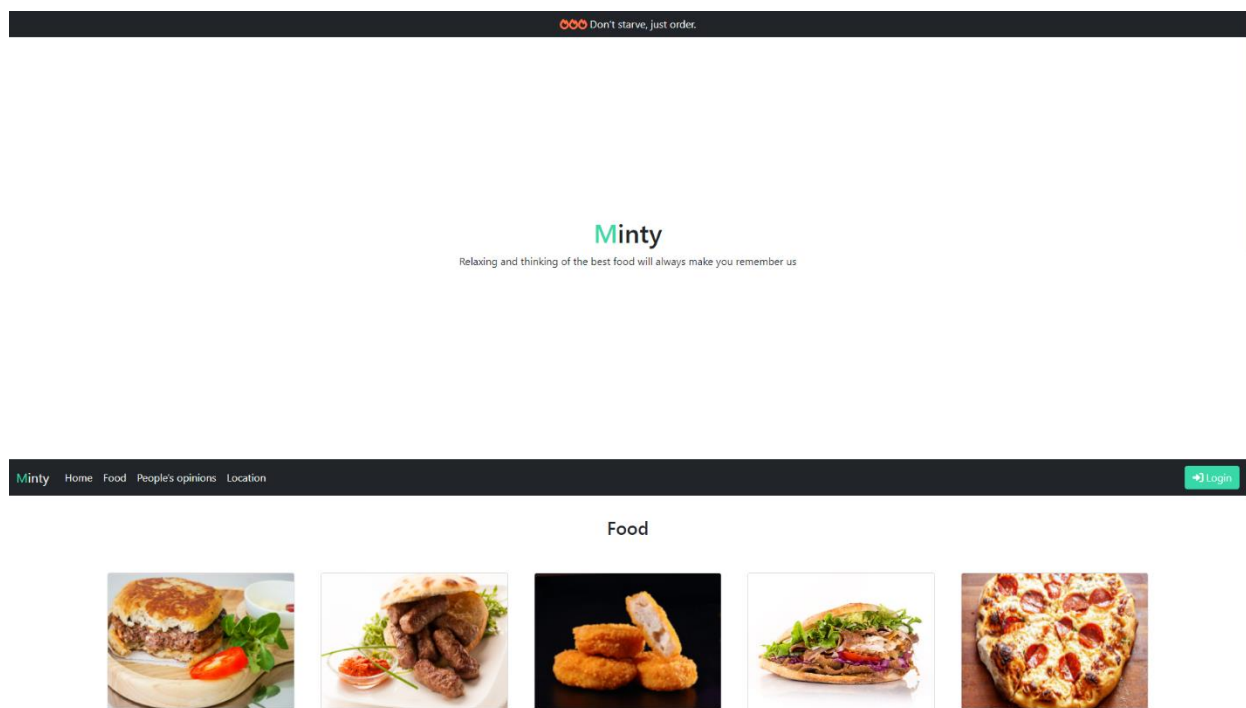
1  const addFood = () => {
2      const foodCards = document.querySelector('#food_cards');
3      let lastFoodId = 0;
4
5      for (let i = 0; i < foodCards.children.length; i++) {
6          if (!document.querySelector(`[id = '${i}']`)) {
7              lastFoodId = i;
8              break;
9          } else {
10             lastFoodId = foodCards.children.length;
11         }
12     }
13
14     const foodAddedName = document.querySelector('#food-add-name').value;
15     const foodAddedPrice = document.querySelector('#food-add-price').
value;
16     const foodAddedImageUrl = document.querySelector('#food-add-imageUrl'
).value;
17
18     document.querySelector('#add-food-close-button').click();
19     document.querySelector('#modal-body__add-food-form').reset();
20
21     fetch('https://ptf-web-dizajn-2022.azurewebsites.net/api/Food', {
22         method: 'POST',
23         headers: new Headers({'content-type': 'application/json'}),
24         body: JSON.stringify({
25             id: lastFoodId,
26             name: foodAddedName,
27             price: foodAddedPrice,
28             imageUrl: foodAddedImageUrl
29         })
30     })
31     .then(res => {
32         console.log(`Status code: ${res.status}`);
33
34         if (res.ok) {
35             let card = `
36             <div class="card" style="width: 18rem; height: auto; marg
in: 0 20px; margin-top: 20px;" id="
${lastFoodId}>
37                 
38                 <div class="card-body">
39                     <h5 class="card-title text-dark" id="card-body__n
ame">
${foodAddedName}</h5>
40                     <div>Price: <p class="card-text text-dark" id="ca
rd-body__price">
${foodAddedPrice}</p></div>
41                     <a type="button" class="btn cards__root-buttons t
ext-light" data-bs-toggle="modal" data-bs-target="#edit-food-form" onclik
k="fillEditData(
${lastFoodId})" data-bs-whatever="${lastFoodId}
" id="cards__root-buttons-edit"><i class="fa-solid fa-pen"></i> Edit</a>
42                     <button href="" class="btn cards__root-buttons" o
nclick="deleteFood(
${lastFoodId}
)" id="cards__root-buttons-delete"> <span class="text-light"><i class="fa
-solid fa-trash-can"></i> Delete</span></button>
43                     <button href="" class="btn btn-main card-body__ad
d-to-cart" onclick="putFoodInOrder(this)" style="display: none;"><span cl
ass="text-light"><i class="fa-solid fa-cart-arrow-down"></i> Add to cart
</span></button>
44                 </div>
45             </div>
46             `;
47             foodCards.innerHTML += card;
48
49             const cards = document.querySelectorAll('.cards__root-buttons
');
50
51             for (let i = 0; i < cards.length; i++) {
52                 cards[i].style.display = 'inline-block';
53             }
54
55             popup(`Food with ID: ${foodId} is successfully added !`);
56         } else {
57             errorPopup('Unable to add food !');
58         }
59     })
60 }

```

Slika 3.8. JavaScript kod koji je zadužen za dodavanje hrane na stranicu

## IV. IZGLED STRANICE NA RAZLIČITIM UREĐAJIMA

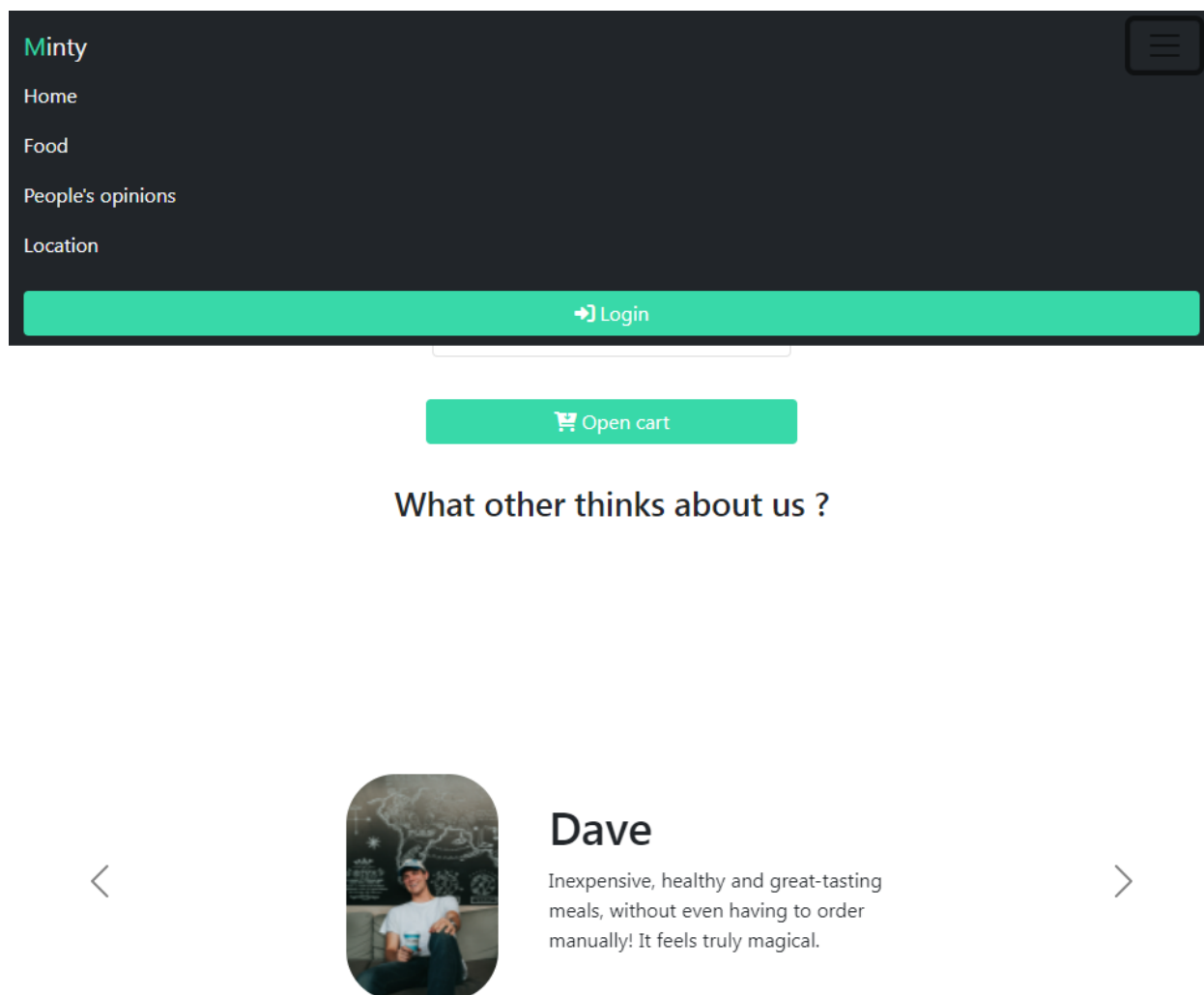
Slika 4.1. prikazuje kako stranica izgleda na desktop računarima.



Slika 4.1. Izgled stranice na desktop računarima

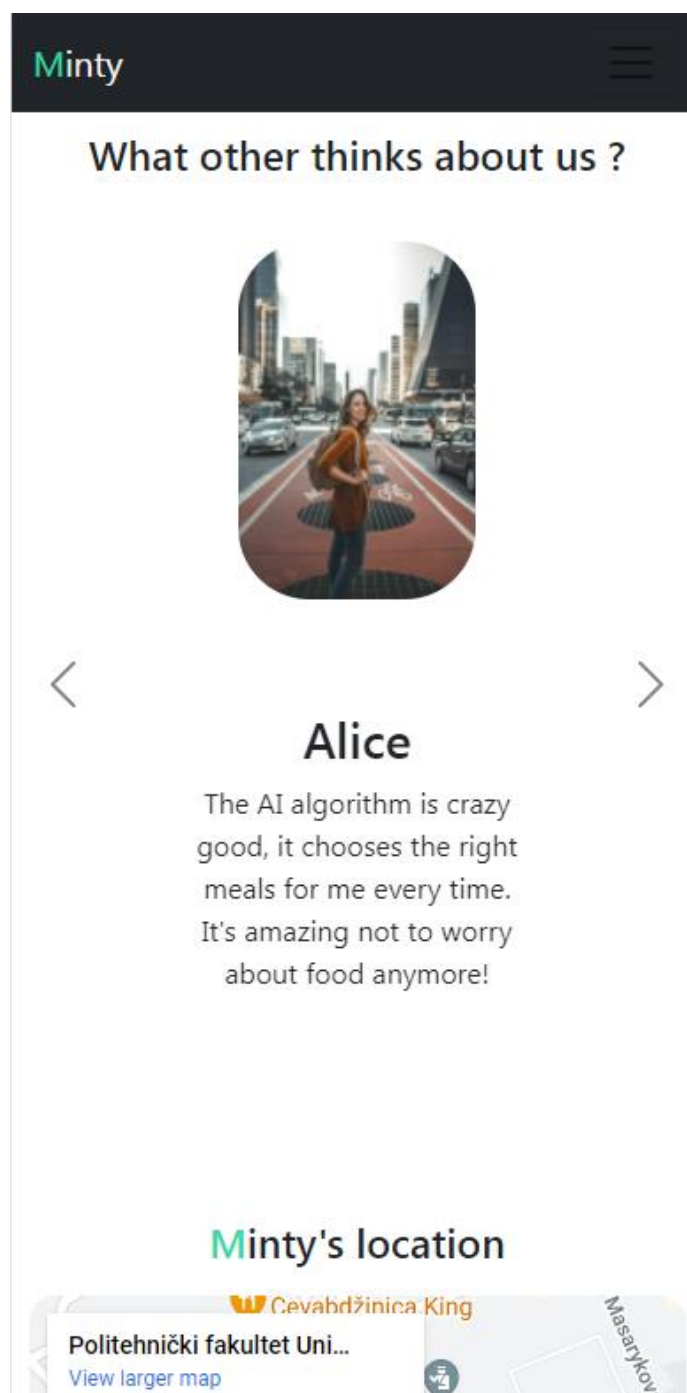


Slika 4.2. prikazuje kako stranica izgleda na tablet uređajima



*Slika 4.2. Izgled stranice na tablet uređajima*

Slika 4.3. Prikazuje kako stranica izgleda na mobilnim uređajima



Slika 4.3. Izgled stranice na mobilnim uređajima