****

**Food&Fitness**

**Predmet: Klijent Server Sistemi**

**Profesor: Student:**

**dr Mirko Kosanović Marina Stojanović**

**Miloš Kosanović Rer 55/17**

***Februar 2021***

SADRŽAJ

[1. Uvod - 3 -](#_Toc512514140)

[2. Instalacija i podešavanje projekta - 3 -](#_Toc512514141)

[2.1. Instaliranje modula - 3 -](#_Toc512514142)

2.2. Pokretanje aplikacije……………………………………………………………………- 4 -

[3. Arhitektura aplikacije - 4 -](#_Toc512514143)

[3.1 Serverski deo - 5 -](#_Toc512514144)

[3.2 Klijentski deo - 5 -](#_Toc512514145)

[3.3 Baza podataka - 5 -](#_Toc512514146)

[3.4 Komunikacija - 6 -](#_Toc512514147)

[4. Rad aplikacije - 7 -](#_Toc512514148)

[4.1 Opis implementacije - 7 -](#_Toc512514149)

[4.2 Opis funkcionalnosti – korisničko uputstvo - 9 -](#_Toc512514150)

[5. Literatura - 11 -](#_Toc512514151)

# 1. Uvod

# U ovom projektu obrađena je izrada web aplikacije pod nazivom *„Food&Fitness“* . Tehnologije koje su korišćene na klijentskoj strani su HTML5, CSS, JavaScript sa Jquery bibliotekom, Bootstrap framework, FontAwesome za ikonice i Google Fonts za fontove, dok su na serverskoj strani korišćeni NodeJS sa Express framework-om. Za prikaz html-a je korišćen EJS templejtski jezik. Za pamćenje podataka o namirnicama korišćena je MongoDB baza podataka. Alati koji su korišćeni prilikom izrade aplikacije su Google Chrome pretraživač̌ i Visual Code Studio.

# Kada pristupimo sajtu, možemo videti tri stranice od kojih se sastoji sajt. Na stranici „Namirnice“ možemo dodati neke naše namirnice ili pretražiti neke već postojeće proizvode koje nas zanimaju. Pored stranice „Namirnice“ imamo jos dve stranice „Dnevnik“ gde možemo da unosemo obroke koje smo u toku dana jeli, i stranica „Kalkulator“ gde možemo izračunati broj kalorija koje su nam potrebne da unesemo.

# 2. Instalacija i podešavanje projekta

Da bismo pokrenuli nodejs aplikaciju potrebno je da instaliramo NodeJs, *a* za ovaj projekatkoristimo i MongoDB, ali nam za bazu nije potrebna instalacija zato što se ona nalazi na cloud-u. Jako je bitno da nam je dostupan internet prilikom pokretanja programa, jer aplikacija koristi MongoDB Atlas (nerelacionu cloud bazu) kao i Bootstrap i CSS.Active CDN linkove.

## 2.1. Instaliranje modula

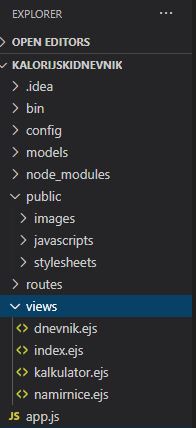
U ovom projektu nalazi se **package.json** fajl, to je fajl koji se inicijalizuje sa projektom, naravno na zahtev programera, i ukoliko popunimo sve informacije ispravno biće kreiran fajl. Zatim kad krenemo da instaliramo nove module i ukoliko upotrebimo neke ključne reči (-S , -save) prilikom instalacije u ovom fajlu biće nam upisani svi moduli koje koristimo za našu aplikaciju. U koliko je sve to ispravno kreirano, da bi smo na nekom drugom računaru pokrenuli i instalirali module, potreno je ukucati sledeću naredbu **npm install** ova naredba prvo pretražuje **package.json** fajl i u njemu traži i instalira sve dependence (tj. Module ili biblioteke) koji su potrebni za ovaj projekat. Moduli koji su potrebni za pokretanje ove aplikacije su: http-errors, path, body-parser, cookie-parser, cors i morgan.

## 2.2. Pokretanje aplikacije

## Da bismo pokrenuli projekat potrebno je da se nađemo u glavnom folderu gde se nalazi fajl app.js, iz tog foldera onda pokrećemo konzolu (cmd) i kucamo „npm start“.

# 3. Arhitektura aplikacije

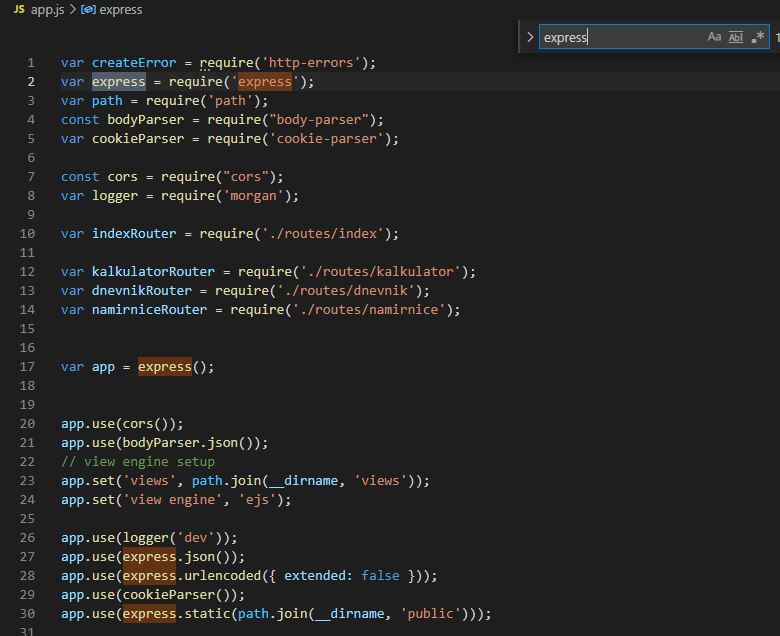
Aplikacija u sebi sadrži korenski (engl. Root) direktorijum '/' koji u sebi sadrži '/node\_modules' direktorijum koji sadrži module koji su potrebni NodeJS-u, '/public' direktorijum u kome je smešten klijentski deo aplikacije, kao i dve datoteke '/package.json' koji predstavlja JSON dokument koji opisuje samu aplikaciju i sadrži spisak modula od kojih je serverski deo aplikacije zavistan i app.js datoteka u kojoj je kod NodeJS servera. Svakoj datoteci iz '/public' direktorijuma je moguće pristupiti javno, odnosno direktno iz URL-a veb čitača. '/public' direktorijum se sastoji iz dnevnik.ejs' dokumenta koji predstavlja osnovni html dokument u kome će se dalje prikazivati aplikacija i 'app' direktorijuma u kome se nalaze kontroleri koji predstavljaju logiku klijentske aplikacije. U ’/views’ datoteci se nalazi sve html stranice u EJS formatu kojima pristupamo preko aplikacije. Arhitektura aplikacije i struktura foldera se mogu videti na slici 1.



Slika 1: struktura foldera

## 3.1 Serverski deo

Express.js je minimalni framework za NodeJS web aplikacije. Express sadrži metode za rutiranje, serviranje statičkih datoteka, korišćenje šablona rukovanje greškama, rukovanje sesijama i još dosta toga.



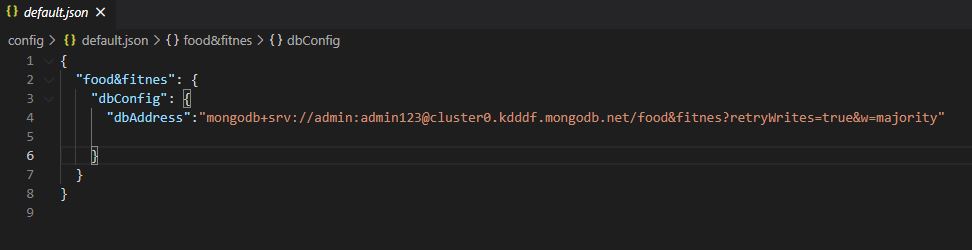
Ovde možemo videti kako smo uneli express modul, i stavili ga u promenljivu app i tako pozivamo funckiju express().

## 3.2 Klijentski deo

Što se tiče klijentskog dela, koristišćen je EJS templejtski jezik za renderovanje *HTML-a i CSS-a*. Za dodatnu stilizaciju stranice kao i mogućstva za lakšim kreiranjem responsive stranice koristimo Bootstrap framework. On sadrži dosta predefinisanih funkcija za stilizovanje tako da dosta umanjuje rad sa našim css dokumentom i takođe omogućava da aplikacija izgleda savršeno na svim veličinama ekrana. Neke funkcionalnosti sajta, kao za search, kalkulator, dnevnik koriscena je JavaScript i JQuery.

## 3.3 Baza podataka

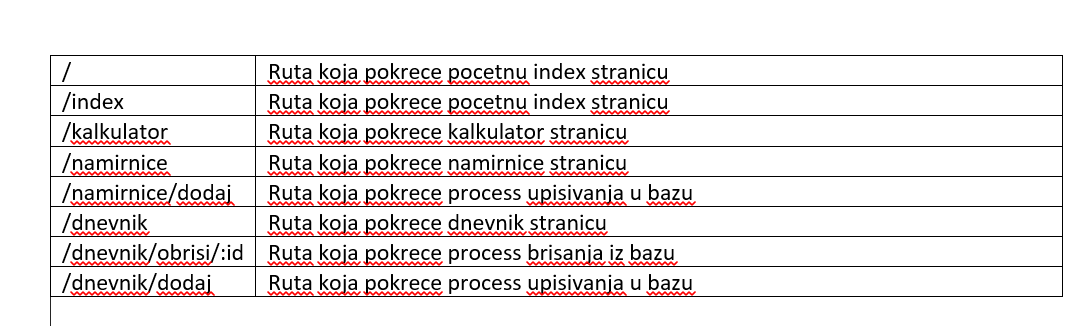
## MongoDB je NoSQL baza podataka, koja ne koristi SQL za povezivanje, nerelaciona je, distribuirana, otvorenog koda i horizontalno skalabilna. Napisana je u C++ jeziku i otvorenog je koda. MongoDB čuva podatke kao JSON dokumente sa dinamičkim šemama. JSON (JavaScript Object Notation) je otvoreni standard zasnovan na tekstu, osmišljen za razmenu podataka koji su pogodni za čitanje ljudima. Baza je uploadovana na cloud-u tako da ne treba instalacija za bazu.



## 3.4 Komunikacija

## U okviru fajla ’index.js’ nalaze se rute koje vode na odredjenu EJS stranicu. Serverska strana obrađuje taj zahtev i ispunjava klijetske.

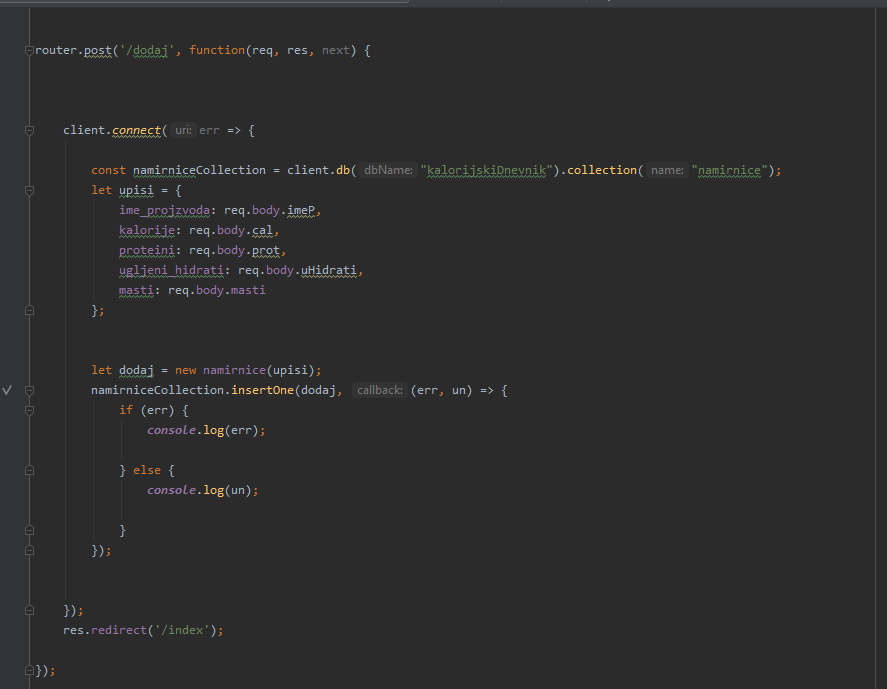
## Na ovoj slici se mogu videti rute kojima se pristupa preko localhost:3000, kao i opis ruta.



# 4. Rad aplikacije

## 4.1 Opis implementacije

U fajlu app.js se nalazi kod gde su implementirani moduli i baza.



Ovde imamo primer post metode koja vrši upis novih namirnica u bazu podataka na sledeći nacin.

Prvo se treba konektovati na bazu putem objekta cliket koji sadrži potrebne parametre za pristup bazi.

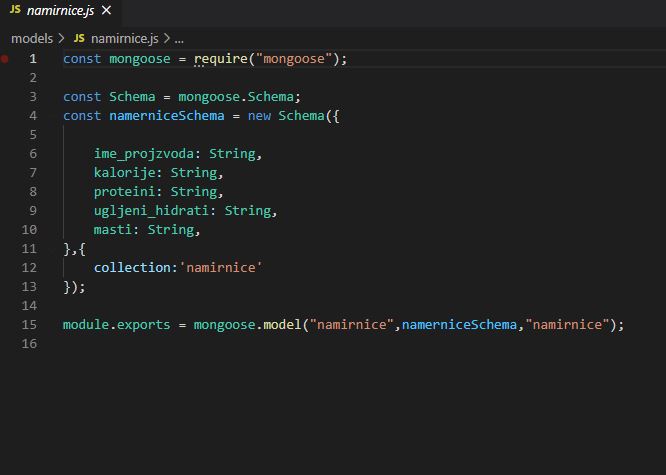
Zatim u promenljivu namirniceCollection smeštamo nad kojom će se kolekcijom vršiti upis u bazi.

Popunimo objekat podacima koji su pristigli sa frontenda i dobijamo promenljivu „upisi“.

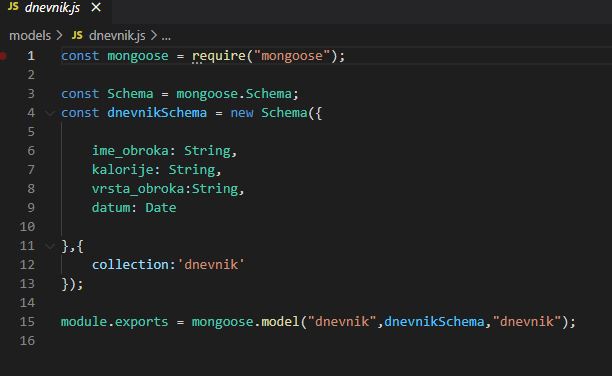
zatim kreiramo novi objekat 'dodaj' koji je inicijalizovan modelom namirnice i kome prosledjujemo podatke za upis.

Komandom insetOne porekcemo proces upisa u bazu.

U folderu models se nalazi model namirnice.js koji se može videti na sledećoj slici:

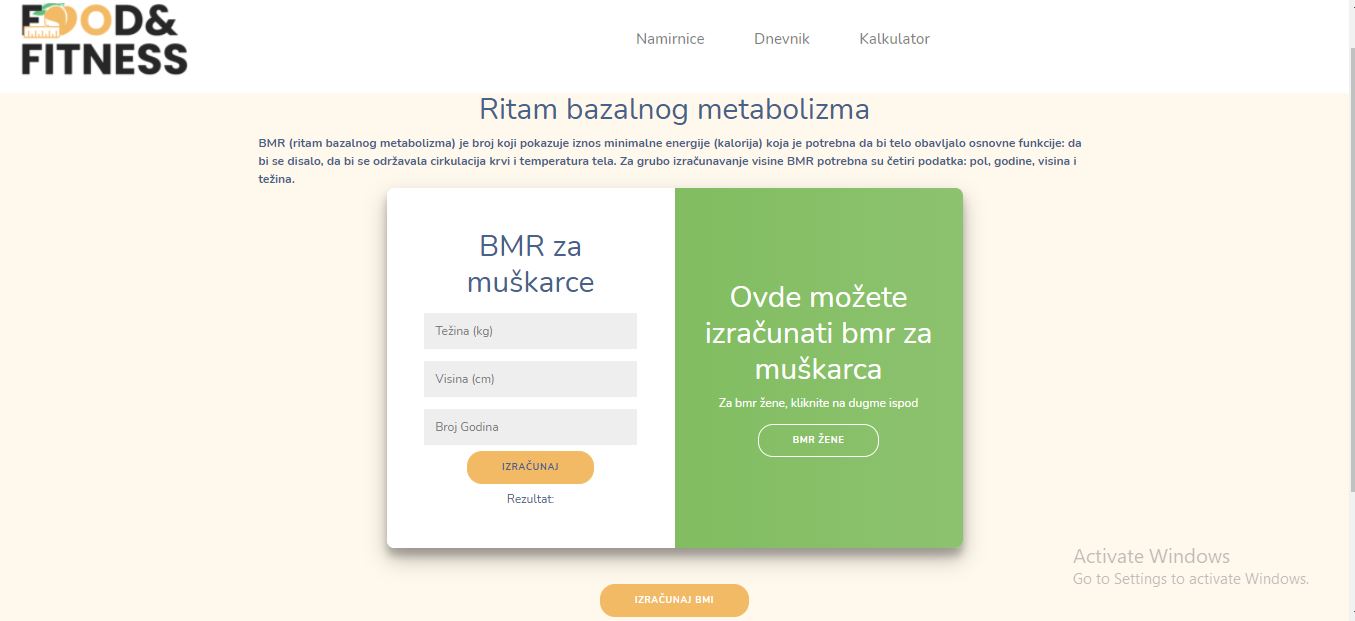


U folderu models se nalazi model dnevnik.js koji se može videti na sledećoj slici:

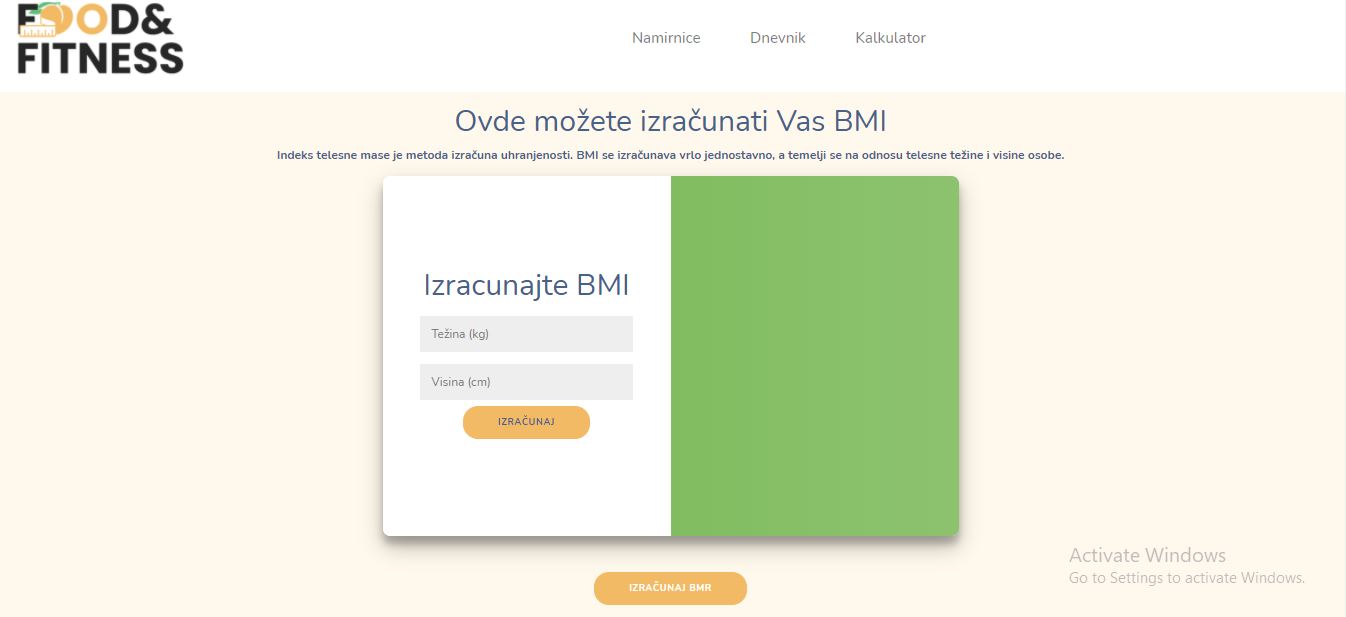


## 4.2 Opis funkcionalnosti – korisničko uputstvo

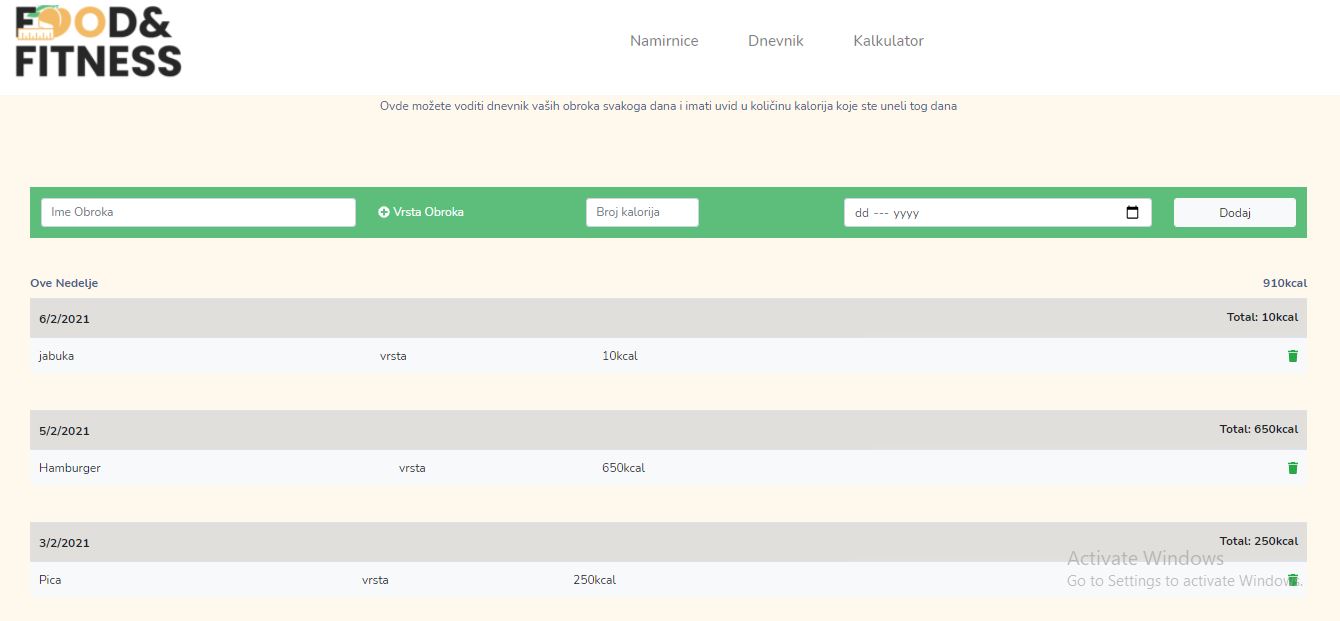
# Na kalkulator strani možemo da biramo između muškog i ženskog pola, gde unosimo naše podatke i dobija okvirno broj kalorija koja su nam potrebne.



BMI kalkulator, ponovnim unosom naših podataka možemo da vidimo kojoj grupi ljudi pripadamo u odnosu sa težinom.



Stranica dnevnik, je glavna strana i na njoj se se unosi ono što smo mi tokom dana jeli i dobijamo ukupan broj pojedenih kalorija.



Stranica namirnice, i forma za unos novih namirnica.

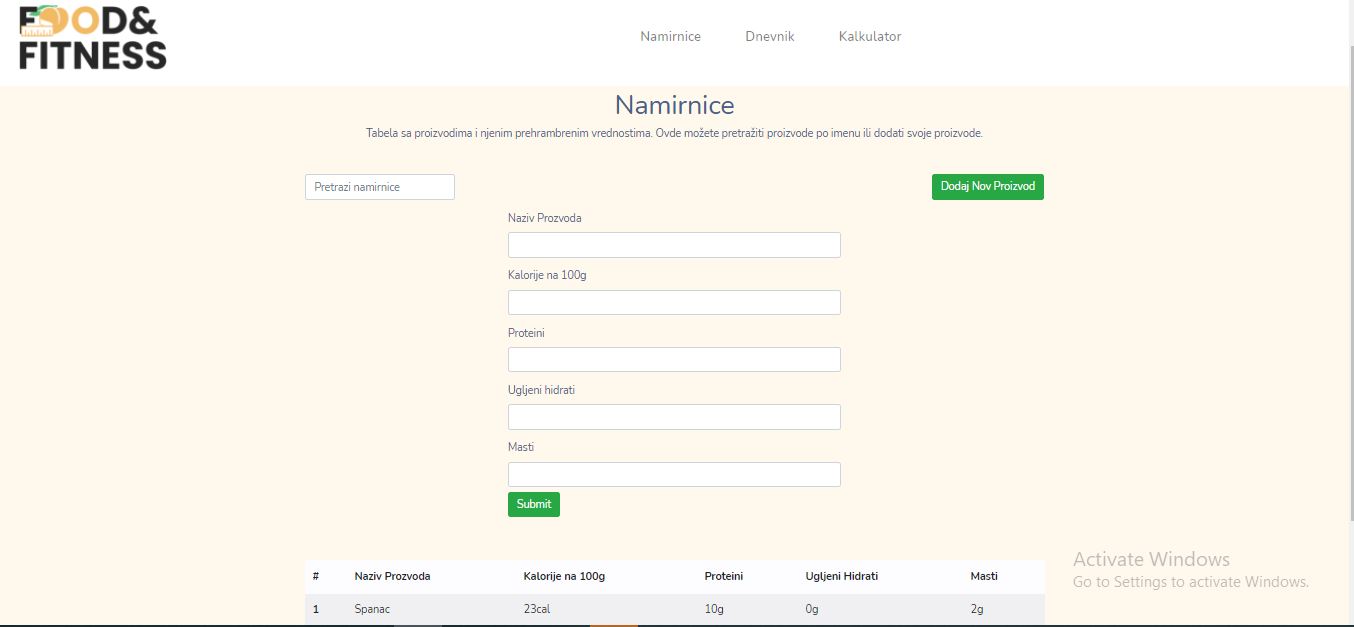
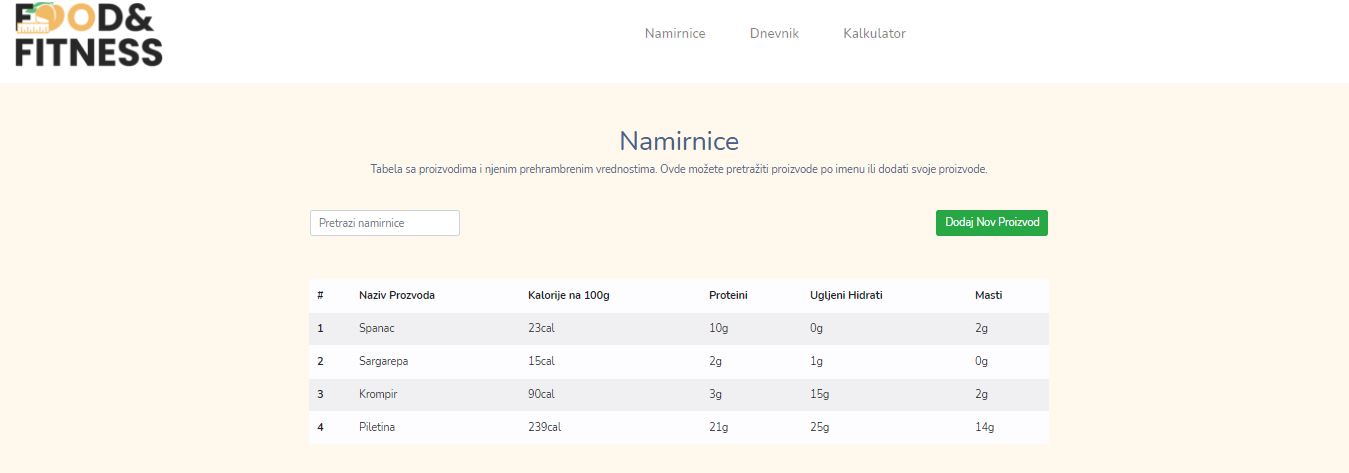


Tabela gde se ispisuju podaci iz baze koje smo uneli.



# 5. Literatura

Praktikum iz Klijent-server sistema

<https://www.npmjs.com/>

<https://www.w3schools.com/nodejs/>

<https://getbootstrap.com/>

<https://github.com/>

<https://youtube.com/>

<https://stackoverflow.com/>