Web aplikacija za praćenje unosa nutrijenata - Backend

Fuček, Marko

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij

WEB APLIKACIJA ZA PRAĆENJE UNOSA NUTRIJENATA - BACKEND

Završni rad

Student: Marko Fuček

Mentor: izv. prof. dr. sc. ~~doc. dr. sc.~~ Krešimir Nenadić

Komentirano [KN1]: izv. prof. dr. sc.

Osijek, 2016.

SADRŽAJ

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

Oblikovano: Zadani font odlomka, Provjeri pravopis i gramatiku

1. UVOD

Tema ovog završnog rada je Web aplikacija za praćenje unosa nutrijenata - *Backend*. Cilj ovog rada je steći adekvatna praktična znanja i vještine iz područja ~~backend-a~~ na strani poslužitelja i baza podataka. Također jedan od ciljeva je razvijanje timskog rada i vrijednosti asociranih uz njega s obzirom na to da je praktični dio zadatka osmišljen u dvije cjeline (*backend* i *frontend*). U nastavku će ukratko biti opisana aplikacija, njeni dijelovi, funkcionalnost koju pruža, programski jezici, tehnologije korištene pri izradi i koji je njen cilj.

Pri izradi aplikacije korišten je Visual Studio 2015 i programski jezik C#, SQL Server 2014 Management Studio i jezik SQL, ASP.NET i Entity framework, MVC programski model.

Glavni cilj aplikacije je omogućiti registriranim korisnicima jasan i intuitivan unos dnevnih nutrijenata te prikaz statistike obroka u što jednostavnijem obliku. Ciljana publika su sve dobne skupine sposobne služiti se internet preglednikom.

Rad se sastoji od šest glavnih poglavlja. Prvo uvodno poglavlje opisuje problematiku rada. Drugo poglavlje opisuje tehnologije i alate korištene prilikom izrade. Treće poglavlje ističe strukturu baze podataka, te njezinu izradu. Četvrto poglavlje posvećeno je konkretnoj implementaciji tehnologija navedenih u drugom poglavlju i strukturi aplikacije. Peto poglavlje sadrži generalne testove funkcionalnosti aplikacije i baze podataka. Šesto poglavlje sadrži analizu i zaključak rada. ~~Ubaciti i kratki pregled sadržaja po poglavljima.~~

1.1. Zadatak rada

Objasniti važnost pravilne prehrane i unosa odgovarajućih nutrijenata. Modelirati bazu podataka koja omogućava spremanje podataka registriranim korisnicima o dnevnom unosu pojedinog nutrijenta u organizam. Omogućiti funkcionalnosti pregleda unosa prema različitim kriterijima. Izraditi poslužiteljski dio aplikacije i testirati rad cijelog sustava.

Oblikovano: Font: Kurziv

Komentirano [KN2]: na strani poslužitelja

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Boja fonta: Automatski

Oblikovano: Boja fonta: crvena

Komentirano [KN3]: 1.1

Oblikovano: Font: Podebljano, Boja fonta: Tekst 1

2. TEHNOLOGIJE KORIŠTENE U APLIKACIJI

2.1. Microsoft Visual Studio

Microsoft Visual Studio je alat ~~t za svakog developera i svaku aplikaciju~~pogodan za korištenje programerima početnicima ~~tako~~ i iskusnijim programerima, s ~~bogato~~bogatim, ~~integrirano~~ integriranim ~~razvojno~~ razvojnim okruženjem (IDE) kojeg je izradila ~~izrađen od strane~~ ~~f~~firma~~e~~ Microsoft. Koristi se za razvoj ~~zadivljujućih~~ jednostavnih i kompleksnih računalnih programa za Windows, Android i iOS operacijski sustav, web stranice, web aplikacije i ~~*cloud* servise~~*usluge na oblaku*.

Podržava razne programske jezike (C, C++, VB.NET, C#, F#), pruža podršku za neke druge jezike poput ~~(~~Python, Ruby, Node.js~~)~~ i pruža podršku XML/XSLT, HTML/XHTML, JavaScript i CSS.

Prethodnih godina Visual Studio bio je dostupan studentskoj populaciji preko Microsoftovog DreamSpark programa, no od nedavno postoji Community izdanje dostupno svima bez naknade.

2.1.1 ASP.NET

ASP.NET je otvoreno kodni, serverski orijentiran web aplikacijski framework dizajniran za razvoj dinamičnih web stranica i web aplikacija. Razvila ga je firma Microsoft ~~jen~~ početkom 2002. godine ~~od firme Microsoft~~ kao nasljednik Active Server Pages (ASP) tehnologije.

ASP.NET podržava tri različita razvojna modela. Web Pages, MVC (Model View Controller) i Web Forms.

2.1.2. C#

C# je ~~Jedan~~ jedan od mlađih programskih jezika najavljen 2000. godine, a firma Microsoft lansira~~n~~ 2002. godine, ~~od strane firme Microsoft~~ sastavni je dio .NET framework platforme. C# je jednostavan, moderan, objektno orijentiran programski jezik opće primjene, kao i većina današnjih programskih jezika više razine. C# 6.0 je zadnja ~~verzija~~ inačica jezika izdana sredinom 2015. godine.

Komentirano [KN4]: Ne koristiti ovakav stil pisanja

Komentirano [KN5]: Izbjegavati 'od strane'

Komentirano [KN6]: Nije dobra riječ za tehnički tekst

Komentirano [KN7]: usluge u oblaku.

Komentirano [KN8]: Bez zagrada

Komentirano [KN9]:

Komentirano [KN10]:

Komentirano [KN11]: inačica

2.1.3. MVC

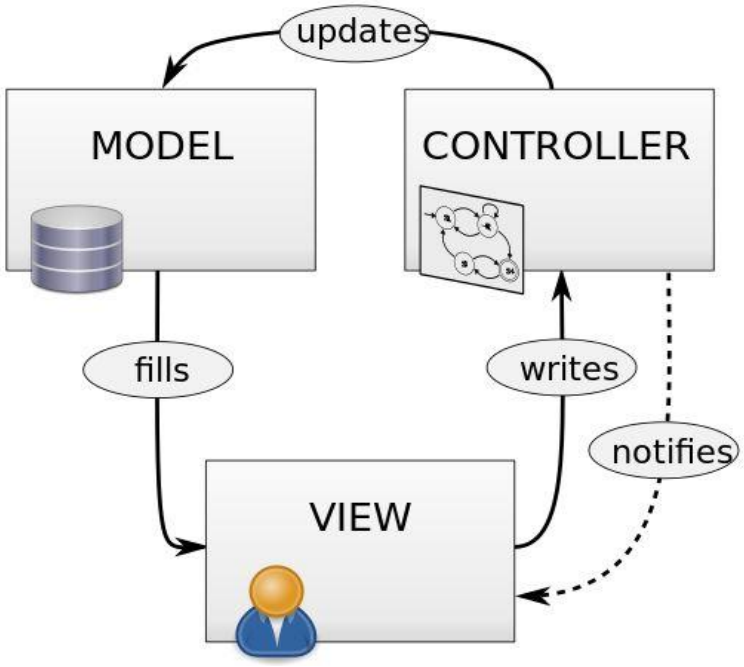
MVC je jedan od tri ASP.NET programska modela. *Model–View–Controller* tehnologija je arhitektonski obrazac koji se koristi u izradi programskih rješenja - programa. Koristi se kako bi se razdvojilo korisničko sučelje - *View* (HTML, CSS, JavaScript) od logike - *Controller* (upravlja ~~s~~korisničkim sučeljem) i modela podataka - *Model* (baza podataka).

Model je dio aplikacije odgovoran za rad s podacima, konekcije na bazu podataka, upite na bazu.

View je dio aplikacije koji upravlja ispisom podataka, vrlo često su pogledi ~~*views* (*views*pogledi~~) kreirani prema *Modelu*.

Controller je dio aplikacije zadužen za rukovanje korisničkom interakcijom.

Obično *controller* čita podatke s *view*-a, kontrolira korisnički unos i šalje unos modelu. Prednost *MVC modela* je njegova pogodnost za velike grupne projekte i kompleksne aplikacije, zato što omogućava fokusiranje na jedan aspekt aplikacije u danom vremenu - paralelan razvoj objekata *Model*,*View* i *Controller*. Time je olakšano testiranje, održavanje i nadogradnja aplikacije.



Sl. 2.1. Grafički prikaz odvijanja proces u MVC modelu

Komentirano [KN12]: stručne engleske nazive ukositi -Na svim mjestima u radu (to važi za sve naputke)

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Komentirano [KN13]: zamijeniti – hrvatski van zagrada a engleski u zagrade

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Komentirano [KN14]: Izvor slike ? Navesti ako je slika preuzeta od nekoga

2.2. SQL Server Management Studio

SQL Server Management Studio ili SSMS je integrirano okruženje za upravljanje SQL ~~Server~~ Poslužitelj infrastrukturom i Azure SQL Database. Prvo lansiran sa Microsoft SQL Server 2005 koji se koristi za konfiguraciju, upravljanje i administraciju svih komponenata unutar Microsoft SQL Servera. Alat uključuje skriptne uređivače kao i grafičke alate koji rade s objektima i funkcionalnostima poslužitelja. Središnja funkcionalnost SSMS-a je Object Explorer koji korisniku omogućava pretraživanje, označavanje i manipulaciju objektima na poslužitelju.

Komentirano [KN15]: Poslužitelj

2.2.1. SQL

Strukturirani jezik za pretraživanje (Structured Query Language) je ~~programski~~ strukturni jezik za pretraživanje, posebne namjene dizajniran za upravljanje podacima korištenim u relacijskim bazama podataka. Neproceduralni jezik koji upotrebljava ključne riječi kao *select*, *insert*, *delete* i slično kao dio naredbe za izvršavanje upita.

Komentirano [KN16]: Nije programski

Oblikovano: Font: Kurziv

Komentirano [KN17]: Ukositi i koristiti monospace font – Courier New, npr.

Komentirano [KN18]: Loš stil – malo bolje pojasniti i proširiti.

Originalno ~~baziran~~ temeljen na relacijskoj algebri, SQL se sastoji od podatkovno opisnog jezika (DDL), podatkovno manipulacijskog jezika (DML) i podatkovno kontrolnog jezika (DCL).

Oblikovano: Font: Kurziv

Komentirano [KN19]: Temeljen

Osigurava naredbe za različite zadaće:

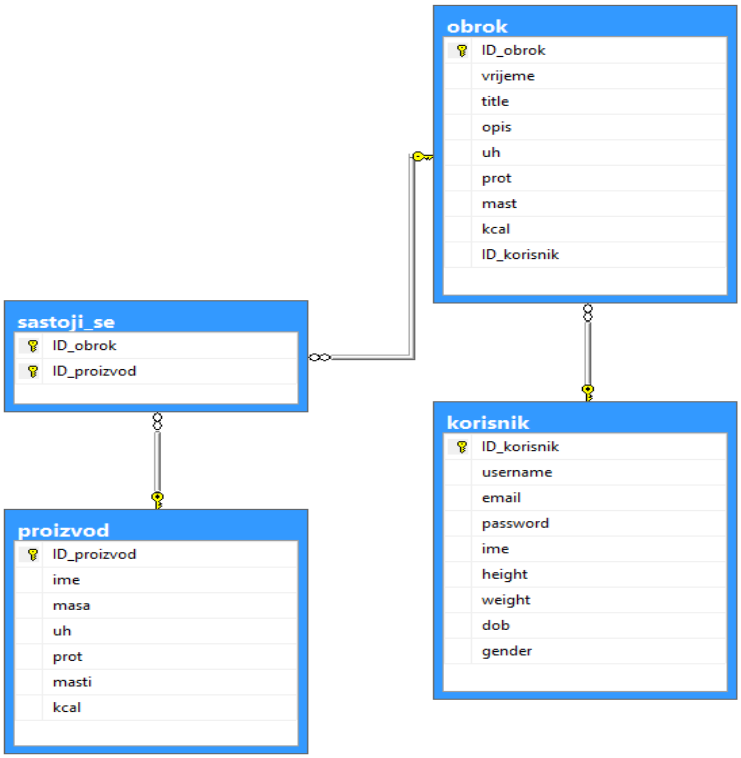
- Upite nad podacima
- Dodavanje, mijenjanje i brisanje redaka u tablicama
- Kreiranje, mijenjanje i brisanje objekata sheme
- Kontrolu pristupa objektima sheme i bazi podataka
- Osigurava konzistentnost baze podataka.

3. STRUKTURA BAZE PODATAKA

Baza je kreirana s ciljem da svaki korisnik može kreirati veći broj obroka u danu, a svaki obrok se sastoji od više proizvoda. Također vrijedi i obrnuto - više proizvoda može biti sadržano u više različitih obroka.

Koristeći programski alat SQL Server 2014 Management Studio napravljena je baza podataka koja se sastoji od 4 tablice (entiteta):

- *korisnik* - tablica registriranih korisnika
- *obrok* - tablica sastavljenih obroka
- *sastoji_se* - tablica veza između obrok i proizvod
- *proizvod* - tablica proizvoda dostupnih za sastavljanje obroka

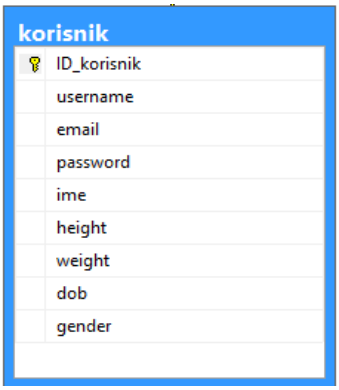


Sl. 3.1. ER dijagram modelirane baze podataka

Tablica *korisnik* sastoji se od 9 atributa kao što je prikazano na slici 3.2.:

Komentirano [KN20]: Pomaknuti na početak reda ulijevo

Oblikovano: Lijevo



Sl. 3.2. Tablica *korisnik*

1) *ID_korisnik* - jedinstveni identifikator za svakog korisnika
2) *username* - opis područja na kojem se nalazi korisničko ime
3) *email* - opis područja na kojem se nalazi adresa elektroničke pošte
4) *password* - opis područja na kojem se nalazi korisnikova lozinka
5) *ime* - opis područja na kojem se nalazi korisnikovo ime
6) *height* - opis područja na kojem se nalazi korisnikova visina u centimetrima
7) *weight* - opis područja na kojem se nalazi korisnikova ~~težina~~ masa u kilogramima
8) *dob* - opis područja na kojem se nalazi dob korisnika u godinama
9) *gender* - opis područja na kojem se nalazi spol korisnika

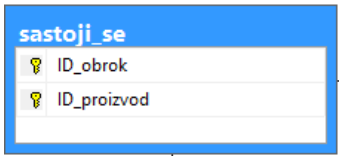
Komentirano [KN21]: MASA

Tablica *obrok* sastoji se od 9 atributa kao što je prikazano na slici 3.3.:

Sl. 3.3. Tablica *obrok*

1) *ID_obrok* - jedinstveni identifikator za svaki obrok
2) *vrijeme* - opis područja na kojem se nalazi vrijeme unosa obroka
3) *title* - opis područja na kojem se nalazi naziv obroka
4) *opis* - opis područja na kojem se nalazi opis obroka
5) *uh* - opis područja na kojem se nalazi ukupan vrijednost ugljikohidrata za taj obrok u gramima
6) *prot* - opis područja na kojem se nalazi vrijednost proteina za taj obrok u gramima
7) *mast* - opis područja na kojem se nalazi vrijednost masti za taj obrok u gramima
8) *kcal* - opis područja na kojem se nalazi vrijednost u kilokalorijama za taj obrok
9) *ID_korisnik* - opis područja na kojem se nalazi strani ključ

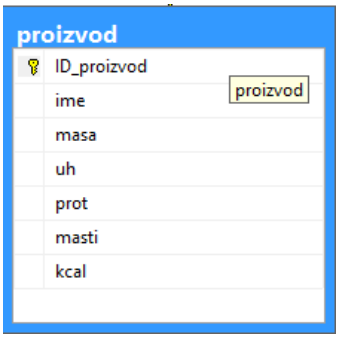
Tablica *sastoji_se* sastoji se od 2 atributa kao što je prikazano na slici 3.4.:



Sl. 3.4. Tablica *sastoji_se*

1) *ID_obrok* - jedinstveni identifikator obroka, prvi od dva primarna ključa
2) *ID_proizvod* - jedinstveni identifikator proizvoda, drugi od dva primarna ključa

Tablica *proizvod* sastoji se od 7 atributa kao što je prikazano na slici 3.5.:



Sl. 3.5. Tablica *proizvod*

1) *ID_proizvod* - jedinstveni identifikator proizvoda, primarni ključ
2) *ime* - opis područja za upis imena proizvoda
3) *masa* - opis područja za upis mase proizvoda u gramima
4) *uh* - opis područja za upis vrijednosti ugljikohidrata proizvoda u gramima
5) *prot* - opis područja za upis vrijednosti proteina proizvoda u gramima
6) *masti* - opis područja za upis masti proizvoda u gramima
7) *kcal* - opis područja na kojem se nalazi vrijednost u kilokalorijama za taj proizvod

Prikaz koda za kreiranje tablica u bazi podataka:

```
CREATE TABLE korisnik(

ID_korisnik INT IDENTITY(1,1),

username VARCHAR(20) NOT NULL,

email VARCHAR(40) NOT NULL,

password VARCHAR(30) NOT NULL,

ime CHAR(30) NOT NULL,

height SMALLINT NOT NULL,

weight SMALLINT NOT NULL,

dob SMALLDATETIME NOT NULL,

gender CHAR(1) NOT NULL DEFAULT 'M',

CONSTRAINT chk_sex CHECK (gender IN ('M','Ž')),

CONSTRAINT pk_korisnik PRIMARY KEY (ID_korisnik)

);
```

Sl. 3.6. Kod tablice *korisnik*

```
CREATE TABLE obrok(
ID_obrok INT IDENTITY(1,1),
vrijeme SMALLDATETIME NULL,
title CHAR(20) NOT NULL,
opis TEXT NULL,
uh SMALLINT NULL,
prot SMALLINT NULL,
mast SMALLINT NULL,
kcal SMALLINT NULL,
ID_korisnik INT,
CONSTRAINT pk_obrok PRIMARY KEY (ID_obrok),
CONSTRAINT fk_korisnik FOREIGN KEY (ID_korisnik) REFERENCES
korisnik(ID_korisnik)
);
```

Sl. 3.7. Kod tablice *obrok*

```
CREATE TABLE sastoji_se(
ID_obrok SMALLINT,
ID_proizvod SMALLINT,
CONSTRAINT pk_sastoji_se PRIMARY KEY (ID_obrok,ID_proizvod),
CONSTRAINT fk_obrok FOREIGN KEY(ID_obrok) REFERENCES obrok(ID_obrok),
CONSTRAINT fk_proizvod FOREIGN KEY (ID_proizvod) REFERENCES
proizvod(ID_proizvod)
);
```

Sl. 3.8. Kod tablice *sastoji_se*

```
CREATE TABLE proizvod(

ID_proizvod INT IDENTITY(1,1),

ime CHAR(30),

masa SMALLINT NOT NULL,

uh SMALLINT,

prot SMALLINT,

masti SMALLINT,

kcal SMALLINT,

CONSTRAINT pk_proizvod PRIMARY KEY (ID_proizvod),

);
```

Sl. 3.9. Kod tablice *proizvod*

Radi lakšeg upravljanja aplikacijom u tablicu proizvod uneseno je 29 najčešće korištenih proizvoda iz prehrambenih skupina (mliječni proizvodi, voće, povrće, meso, riba, orašasti plodovi, pića, masnoće, žitarice).

```
/*POPUNJAVANJE TABLICE proizvodi*/
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Posni sir','100','4','11','4','98');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Mlijeko','100','5','3','2','50');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Jaja','100','1','13','11','155');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Zob','100','66','17','7','389');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Piletina','100','0','27','14','238');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Orasi','100','14','15','65','654');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Riža','100','28','3','0','130');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Krumpir','100','17','2','0','77');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Banana','100','23','1','0','89');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Kakao','100','58','20','14','228');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Jogurt','100','5','4','3','48');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Parmezan','100','4','39','29','431');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Slanina','100','1','13','40','417');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Kulen','100','2','26','30','382');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Tuna','100','0','26','1','116');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Pastrva','100','0','18','5','119');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Lignje','100','3','16','1','92');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Jabuka','100','12','0','0','52');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Naranče','100','16','1','0','63');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Grašak','100','14','6','0','81');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Mrkva','100','9','1','0','41');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Luk','100','9','1','0','40');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Rajčica','100','4','1','0','17');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Špinat','100','4','3','0','23');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Maslac','100','1','1','80','744');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Maslinovo ulje','100','0','0','100','900');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Pivo','100','4','0','0','43');
INSERT INTO proizvod (ime, masa, uh, prot, masti, kcal) VALUES ('Crno vino','100','4','0','0','123');
```

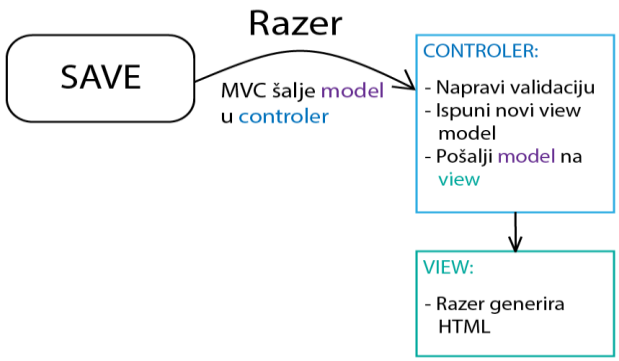
Sl. 3.10. Kod unosa vrijednosti u tablicu *proizvod*

4. STRUKTURA APLIKACIJE

4.1. Autorizacija korisnika i unos podataka

Kako bi korisnik mogao koristiti aplikaciju od njega se zahtjeva registracija preko danoge ~~forme~~ obrasca sadržane u jednom od view-ova prezentiranoge ~~od strane~~ kontrolerома. Nakon ispunjenih obveznih polja registracijskog~~ske~~ ~~forme~~ obrasca korisniku se prezentira Login ~~forma~~ obrazac kako bi mogao pristupiti aplikaciji.

Nakon što se korisnik uspješno ~~ulogirao~~ prijavio u aplikaciju prezentiran mu je *view* u kojem se traži detaljniji unos osobnih podataka (datum rođenja, spol, visina, težina) potrebnih za izračun *BMI* (indeks tjelesne mase) i *BMR* (bazalni metabolički utrošak), koji se izračunavaju klikom na *save* gumb u slučaju da su unesena sva potrebna polja. Istim klikom uneseni i izračunati podaci se spremaju u tablicu korisnik u bazi podataka.



Sl. 4.1. Dijagram toka validacije i generiranja *Profile information*

Komentirano [KN22]: obrazac

Komentirano [KN23]:

Komentirano [KN24]:

Komentirano [KN25]:

Komentirano [KN26]: ? prijpavio

Komentirano [KN27]: Ukositi engleske nazive

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Oblikovano: Font: Kurziv

Komentirano [KN28]: Ukositi imena objekata (funkcije, klase, kontrole, ...)

Oblikovano: Font: Kurziv

```
public ActionResult Register(LoginViewModel registerInfo)

    {
        // If exists in database, return "already exists"
        // If not, and success, redirect to profile screen
        // If not, and error, return "Error occured"
        List<korisnik> k = db.getDatabase().korisnik.Where(t => t.email ==
registerInfo.mail).ToList();
        if (k.Count == 0)
        {
            // insert into database
            korisnik newUser = new korisnik()
            {
                email = registerInfo.mail,
                password = registerInfo.password,
                username = "-",
                dob = 0,
                gender = 0,
                height = 0,
                ime = "-",
                weight = 0
            };

            db.getDatabase().korisnik.Add(newUser);
            db.SaveChanges();
            userHandler.setEmail(registerInfo.mail);
            return View("MyProfile", newUser);
        }
        else
        {
            // report "already registered"
            userHandler.setErrorMessage("Korisnik je već registriran.");
            return RedirectToAction("Login");
        }
    }
```

Sl. 4.2. Kod metode *Register*

Oblikovano: Font: (Zadano) Times New Roman, 12 točka

Oblikovano: Razmak Iza: 10 pt, Prored: 1,5 redak, Podesi razmak između azijskog i latiničnog teksta, Podesi razmak između azijskog teksta i brojeva, Obrub: Okvir: (Jedna puna crta, Automatski, 0,5 pt Širina retka)

Oblikovano: Font: (Zadano) Times New Roman, 12 točka, Boja fonta: Automatski

Oblikovano: Font: (Zadano) Times New Roman, 12 točka

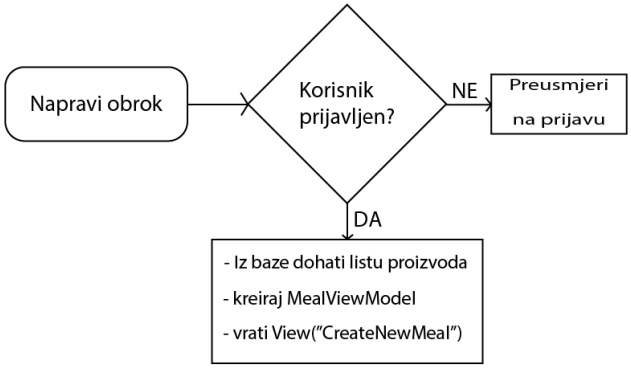
Oblikovano: Font: Boja fonta: crna

Oblikovano: Centrirano

Oblikovano: Font: Kurziv

4.2. Kreiranje novog obroka

Sljedeći prikaz korisniku je prikaz u kojem je omogućeno sastavljanje neograničenog broja dnevnih menija klikom na gumb "**+**". Klikom na gumb za dodavanje novog obroka korisniku je prezentiran *view* za odabir namirnica i njihove količine. Klikom na polje za unos namjernica, omogućen je unos proizvoda iz tablice *proizvod*, ispod tog polja je omogućen unos količine odabranog proizvoda. Nakon popunjenih polja "*Namjernica*" i "*Količina*", korisnik kao potvrdu unosa mora kliknuti na gumb "*ADD*" kako bi se odabrana namjernica dodala u novi obrok. Korisnik nakon kreiranog obroka klikom na gumb "*Done*" sprema dani obrok u tablicu *obrok* u bazi podataka.



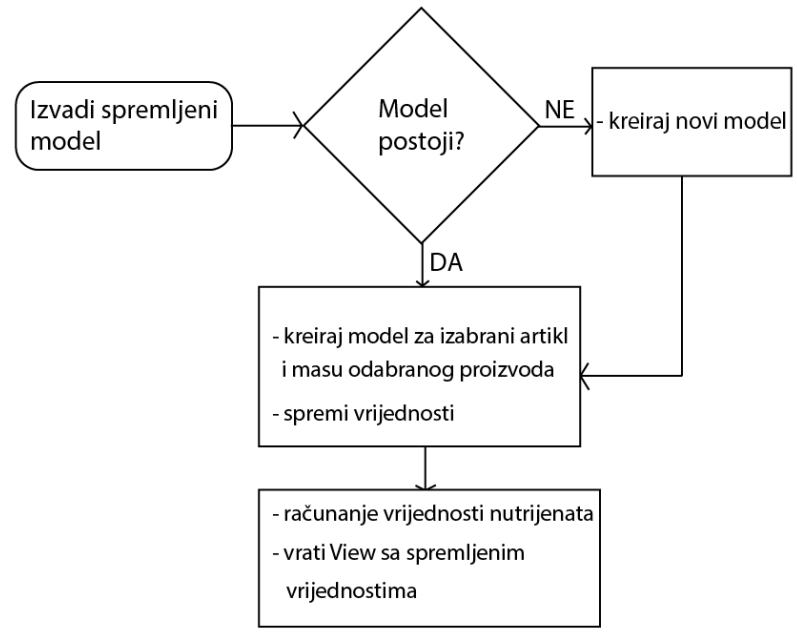
Sl. 4.2. Dijagram toka pristupu *Napravi obrok*

Oblikovano: Naslov 2, Lijevo
Oblikovano: Uvlaka: Prvi redak: 1,25 cm
Oblikovano: Font: Kurziv
Komentirano [KN29]:
Oblikovano: Font: Kurziv
Oblikovano: Font: Kurziv
Oblikovano: Font: Kurziv
Oblikovano: Font: Kurziv
Oblikovano: Font: Kurziv
Oblikovano: Centrirano
Oblikovano: Font: Kurziv

```
    public ActionResult CreateNewMeal()
    {
        if (!userHandler.isLoggedIn())
        {
            userHandler.setErrorMessage("Potrebna prijava za kreiranje novog obroka.");
            return RedirectToAction("Login");
        }
        userHandler.throwIfUserMissing(db);

        db.getDatabase().Configuration.ProxyCreationEnabled = false;
        List<proizvod> proizvodi = db.getDatabase().proizvod.ToList();
        CreateMealViewModel model = new CreateMealViewModel();
        model.proizvodi = proizvodi;
        return View("CreateNewMeal", model);
    }
```

Sl. 4.3. Kod metode *CreateNewMeal*



Sl. 4.4. Dijagram toka *UpdateNewMeal*

Oblikovano: Font: (Zadano) Times New Roman, 12 točka

Oblikovano: Obrub: Okvir: (Jedna puna crta, Automatski, 0,5 pt Širina retka)

Oblikovano: Font: Kurziv, Boja fonta: crna

Oblikovano: Font: Kurziv

```
    public ActionResult UpdateNewMeal(CreateMealViewModel passedInModel)
    {
        // Take the saved model out
        CreateMealViewModel saved =
System.Web.HttpContext.Current.Session["create_meal_saved_state"] as
CreateMealViewModel;
        if(saved == null)
            saved = new CreateMealViewModel();

        // Construct a model for newly picked food and save to cached model
        int productId = passedInModel.chosenProductId;
        double weight = passedInModel.chosenWeigth;
        db.getDatabase().Configuration.ProxyCreationEnabled = false;
        proizvod product = db.getDatabase().proizvod.FirstOrDefault(t => t.ID_proizvod ==
productId);
        if (product == null)
            throw new InvalidProgramException("Chosen product doesn't exist in the database");
        ChosenProduct newChosenProduct = new ChosenProduct(product.ime, weight);
        saved.chosenProducts.Add(newChosenProduct);

        // All products to choose from
        if(!saved.proizvodi.Any())
        {
            List<proizvod> allProducts = db.getDatabase().proizvod.Select(t => t).ToList();
            if (allProducts == null)
                throw new InvalidProgramException("No products found in the database.");
            saved.proizvodi = allProducts;
        }

        double factor = (weight / 100.0);
        saved.protWeight += product.prot * factor;
        saved.uhWeight += product.uh * factor;
        saved.fatWeight += product.masti * factor;
        saved.totalWeight += (saved.protWeight + saved.uhWeight + saved.fatWeight);
        saved.chosenWeigth = 0.0;
        saved.title = passedInModel.title;
        System.Web.HttpContext.Current.Session["create_meal_saved_state"] = saved;
        return View("CreateNewMeal", saved);
    }
```

Sl. 4.5. Kod metode *UpdateNewMeal*

Oblikovano: Font: (Zadano) Times New Roman, 12 točka

Oblikovano: Obrub: Okvir: (Jedna puna crta, Automatski, 0,5 pt Širina retka)

Oblikovano: Font: Ne Kurziv

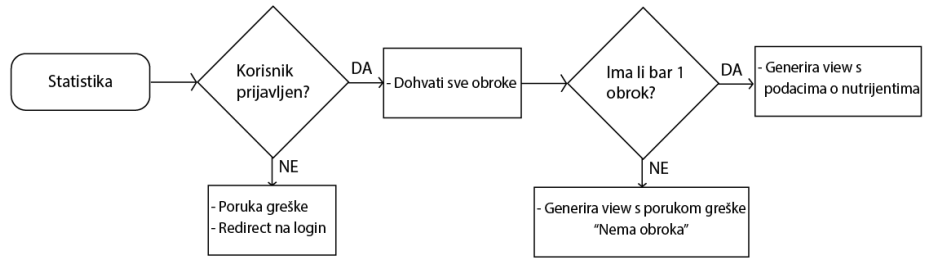
Oblikovano: Lijevo, Pomaci tabulatora: 2,14 cm, Lijevo

~~Na istoj stranici je i prikaz prethodno kreiranih obroka (ako postoje), te klikom na njih prikazuje se statistika za taj obrok.~~

4.~~3~~2. Ispis konačne statistike

Na istoj stranici je i prikaz prethodno kreiranih obroka (ako postoje), te klikom na njih prikazuje se statistika za taj obrok.

Na zahtjev korisnika, odnosno klikom na gumb za prikaz statistike, iz baze podataka se dohvaća svaki obrok, njegov sastav u proizvodima kreiran od tog korisnika u tom danu te se nakon što se izvrše svi izračuni on se u određenom prikazu prezentiraju.



Sl. 4.6. Dijagram toka *MyStats*

Oblikovano: Lijevo

Oblikovano: Obostrano

Oblikovano: Font: Kurziv

Oblikovano: Centrirano

```
        public ActionResult MyStats()
        {
            if (!userHandler.isLoggedIn())
            {
                userHandler.setErrorMessage("Potrebna prijava za pregled statistike.");
                return RedirectToAction("Login");
            }
            korisnik user = userHandler.throwIfUserMissing(db);

            int id = user.ID_korisnik;

            // Get all meals for this user
            ICollection<obrok> obroci = db.getDatabase().obrok.Select(x => x).Where(x =>
x.ID_korisnik == id).ToList();
            StatsViewModel model = new StatsViewModel();
            ICollection<MealViewModel> obrociModified = new
List<MealViewModel>(obroci.Count);
            if (obroci.Count == 0)
            {
                model.hasMeals = false;
                return View("MyStats", model);
            }
            double protTotal = 0.0;
            double uhTotal = 0.0;
            double fatTotal = 0.0;
            double allTotal = 0.0;
            // Transform data from "obrok" type to "obrokViewModel" type
            // Compute percentage of all prot, uh and fat values for graph
            foreach (obrok o in obroci)
            {
                // Transformation
                model.hasMeals = true;
                MealViewModel newObrok = new MealViewModel(o);
                obrociModified.Add(newObrok);
                // Computation
                protTotal += o.prot;
                uhTotal += o.uh;
                fatTotal += o.mast;
                allTotal += (((short)(o.prot + o.uh + o.mast)));
            }
            // Complete the view model with computed values
            model.meals = obrociModified;
            model.protPercent = protTotal / (allTotal * 1.0);
            model.uhPercent = uhTotal / (allTotal * 1.0F);
            model.fatPercent = fatTotal / (allTotal * 1.0F);
            return View(model);
            }
```

Sl. 4.5. Kod metode *MyStats*

5. TESTIRANJE

Ovdje su navedeni gGeneralni testovi funkcionalnosti koji su uspješno provedeni nad aplikacijom. Poduzete akcije, njihovi preduvjeti i očekivani rezultati za svaku od akcija. Složenija testiranja nije bilo potrebno provoditi zbog jednostavnosti funkcionalnosti projekta.

Komentirano [KN30]: Što je ovo? Rečenica treba imati određene dijelove – subjekt, objekt, predikat. Doraditi!!!

Test seq. ID	Poduzeta akcija	Preduvjet	Očekivani rezultati
1.	U web pregledniku upisati adresu aplikacije.	Nema preduvjeta	Korisnik je pristupio aplikaciji.
2.	Korisnik unosi svoje ime I zaporka.	Korisnik ima aktivan račun.	Korisnik se uspješno prijavio u app. Ukoliko je korisnik nije unio neko od obveznih polja ili je unos netočan, aplikacija će ispisati obavijest.
3.	Korisnik unosi svoj email.	Korisnik ima aktivan račun.	Korisnik se uspješno prijavio u aplikaciju. Ukoliko je korisnik unjeo email bez znaka @, aplikacija će ispisati obavijest.
4.	Korisnik unosi brojeve u polje.	Korisnik ima aktivan račun.	Korisnik uspješno nastavlja koristiti aplikaciju. Ukoliko je korisnik unjeo negativan broj ili znakove koji nisu brojevi aplikacija treba ispisati obavijest.

5.	Korisnik unosi korisničko ime u polje za unos.	Nema preduvjeta.	Korisnik se uspješno uspio registrirati. Ukoliko unese korisničko ime koje već postoji u bazi, aplikacija treba ispisati obavijest.
6.	Korisnik upisuje podatke.	Korisnik ima aktivan račun.	Korisnik uspješno nastavlja koristiti aplikaciju. Te svi unosi koje je korisnik unjeo ulaze u okvire unosa istog.
7.	Korisnik odabire spol na stranici "profile".	Korisnik ima aktivan račun.	Korisnik je uspio označiti samo jedan radio button, te nastavlja popunjavati potrebne podatke o sebi.

Interakcija aplikacije s bazom podataka je testirana u sljedečim koracima:

Test seq. ID	Poduzeta akcija	Preduvjet	Očekivani rezultati
1.	Korisnik unosi podatke u obvezna (NOT NULL) polja.	Korisnik ima aktivan račun.	Uneseni podaci su uspješno umetnuti u adekvatnu bazu podataka
2.	Korisnik traži pregled statistike	Korisnik ima aktivan račun.	Aplikacije iz baze dohvaća i prezentira tražene podatke po zadanim kriterijima ispisa

6. ZAKLJUČAK

ASP.NET MVC je projekt koji definitivno pruža mnogo u pogledu fleksibilnost, skalabilnosti, održavanja aplikacija i podjele posla na njegove segmente što mu je velika prednost, a i činjenica da je Visual Studio Community moćan i svestran razvojni alat koji će zbog Microsoftove mudre odluke da ga učini besplatnim daleko doprinijeti cjelokupnoj programerskoj zajednici otvorenog koda.

U sklopu rada napravljena je web aplikacija pomoću koje korisnik ima mogućnost praćenja unosa dnevnih nutrijenata te prikaz statistike.

Aplikacija je uspješno testirana za brojne korisničke slučajeve navedene u prethodnom poglavlju i radi u skladu s postavljenim očekivanjima.

~~Dodati da je provedeno testiranje i da aplikacija radi ono što treba.~~

LITERATURA

[1] Microsoft Visual Studio, stranica https://msdn.microsoft.com/en-us/library/dd831853.aspx (pristup: 1.6.2015.)

[2] C#, stranica https://msdn.microsoft.com/en-us/library/kx37x362.aspx (pristup: 2.6.2016.)

[3] SQL Server Management Studio, stranica https://msdn.microsoft.com/en-us/library/hh213248.aspx (pristup: 3.6.2016.)

[4] SQL, stranica http://www.w3schools.com/sql/default.asp (pristup: 4.6.2016.)

[5] MVC, stranica http://www.w3schools.com/aspnet/mvc_intro.asp (pristup: 5.6.2016.)

[6] ASP.NET, stranica http://www.w3schools.com/aspnet/default.asp (pristup 2.6.2016.)

[7] Testiranje aplikacije, http://www.softwaretestinghelp.com/sample-test-cases-testing-web-desktop-applications/ (pristup 9.6.2016.)

[8] MVC model, https://www.pinterest.com/dotnettutorial/aspnet-mvc/ (pristup 11.2.2017.)

SAŽETAK

Ovaj rad predstavlja projekt izrade web aplikacije koja omogućava jednostavan unos dnevnih nutrijenata i ispis statistike. Cilj izrade ove aplikacije je steći praktična znanja iz informacijskih sustava i timskog rada. Projekt je izrađen uz pomoć programskih alata SQL Server 2014 Management Studio i Visual Studio 2015. U razvojnom alatu SQL Server 2014 Management Studio napravljena je baza podataka, a u Visual Studio napravljena je logika, funkcionalnost i prikaz podataka.

Ključne riječi: Web aplikacija, MVC, ASP.NET, statistika prehrane

ABSTRACT

~~WEB APPLICATION FOR NUTRITION MANAGEMENT – BACKEND~~

Oblikovano: Precrtano

Web application for monitoring consummation of nutrients - Backend

This paper presents a project of development a web application that provides an easy nutrient intake monitoring and statistics. The aim of this application is to acquire practical knowledge of information systems and team work. The project was created with the help of software tools SQL Server 2014 Management Studio and Visual Studio 2015. The database was created in development tool SQL Server 2014 Management Studio, and the Visual Studio created the logic, functionality and display data.

Keywords: Web application, MVC, ASP.NET, nutrition statistics

ŽIVOTOPIS

Marko Fuček rođen je 10.7.1990. godine u Koprivnici, Republika Hrvatska. Osnovnu školu pohađao je u Đurđevcu od 1997. – 2004. godine. Srednju strukovnu školu završio je u Đurđevcu 2010. godine te stekao zvanje Tehničar za računarstvo. Godine 2012. upisao se na stručni studij Elektrotehničkog fakulteta u Osijeku, smjer Informatika. Od 11.7.2016. polaznik SPAN Development Academy.