SVEUČILIŠTE U MOSTARU

dr.sc. Tomislav Volarić, doc.

Upute za izradu projekta iz kolegija «Razvoj web aplikacija» (FSRE) i «Programiranje za Internet» (FPMOZ)

Za uspješno polaganje kolegija potrebno položiti pismeni dio ispita, preći online lekcije, izraditi i obraniti projekt.

Projekt se radi u timovima. Svaki tim se sastoji od najviše tri studenta. Ukoliko je ukupan broj studenata na kolegiju neparan, formira se tim koji se sastoji od četiri studenta.

Izrada projekta vrši se u više faza:

1. Definiranje tima

Na početku projekta potrebno je formirati tim od dva studenta. Studenti se međusobno dogovaraju oko teme i opisa projekta te naizmjenično predstavljaju projekt kroz različite faze.

2. Definiranje teme i opisa projekta

Za projekt je potrebno definirati temu i opis projekta (**vizija**). Tema projekta je **proizvoljna**, a u opisu projekta je potrebno navesti kome je projekt namijenjen i koji problem projekt rješava. Opis ne bi trebao biti ni prekratak, a ni preopširna. Najbolje se orijentirati prema zahtjevu pod točkom 4.

3. Definiranje vrsta korisnika i njihovih ovlasti

Nakon definiranja teme i opisa projekta potrebno je definirati vrste korisnika koje imaju pristup sustavu. Sustav treba imati **minimalno tri** vrste korisnika:

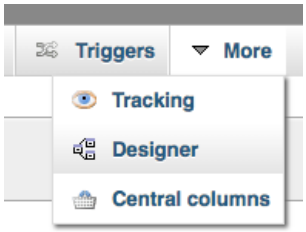
- Super Administrator – ima pristup svima elementima sustava i može kreirati administratore
- Administrator – ima pristup elementima sustava koje su definirane za administratora
- Običan korisnik – ima ograničen pristup sustavu
- Gost – korisnik koji se ne prijavljuje na sustav

Uz svaku vrstu korisnika potrebno je detaljno navesti koje točno ovlasti ta vrsta korisnika ima (napraviti dijagram slučaja uporabe engl. use case diagram). Napraviti tablice u bazi za spremanje «role and permission» informacija.

4. Definiranje baze podataka

Nakon definiranja vrste korisnika i njihovih ovlasti potrebno je izraditi bazu podataka za sustav. Baza bi se trebala sastojati od **najmanje 5** različitih tablica koje su povezane relacijama.

- Obavezno je da baza bude potpuno normalizirana, npr. greška je imati tablice Administratori, Korisnici i Gosti jer imaju iste atribute i samo se razlikuju po vrijednosti atributa *tip_korisnika*. To bi bio jedan primjer nenormalizirane baze podataka.
- Baza mora obvezno sadržavati tablicu korisnika s korisničkim imenom, lozinkom i vrstom korisnika (koristiti ENUM[1]).
- Baza također mora sadržavati bar jednu tablicu sa stranim ključem (ako postoji barem jedna relacija između tablica ovaj uvjet je ispunjen).
- Potrebno je izraditi i **ER dijagram** baze podataka. Dijagram baze je moguće napraviti u alatima poput MySQL Workbench ili phpMyAdmin. U slučaju da koristite phpMyAdmin, grafički prikaz ER dijagrama možete naći u phpMyAdmin **Desgineru** koji se nalazi pod ***More***:



5. Izrada korisničkog sučelja web aplikacije

Koristiti tehnologije koja je omogućena literaturom kroz predavanje i vježbe (materijali na Moodle sustavu) ... (dodatni bodovi korištenje ostalih tehnologija).

- Nabrojati glavne elemente web stranice.
- Definirati dizajn stranice kroz komponente te funkcionalnost weba.
- Stranica mora imati mogućnost prikaza na raznim uređajima različitih veličina ekrana - responzivni dizajn (eng. responsive design).
- Preporuka je da koristite Bootstrap s obzirom da «on» ispunjava sve ove zahtjeve. Dozvoljeno je koristiti neki Bootstrap predložak (Npr. https://startbootstrap.com) ili za dodatne bodove složiti vlastiti uz pomoć neke od Bootstrap tema (Npr. https://bootswatch.com)

6. Implementacija CRUD operacija

Nakon definiranja baze podataka potrebno je izvesti povezivanje PHP-a s bazom podataka i izraditi forme za unos (**C**reate), ispis (**R**ead), uređivanje (**U**pdate) i brisanje (**D**elete) svih elemenata u bazi podataka.

Spajanje na bazu podataka je standardna operacija i možete naći upute na https://www.w3schools.com/php/php_mysql_connect.asp. Dovoljno je spojiti se na bazu na proceduralnim načinom, a za dodatne bodove potrebno je koristiti objektno-orijentiran pristup (npr. PDO ili MySQLi OO).

- CRUD operacije se implementiraju tako što se rade SQL upiti nad svakom tablicom. Primjer upita za unos u bazu podataka bi bio: *INSERT INTO korisnici (kor_ime, lozinka) VALUES ('marko', 'marko12')*
- Za ispis svih korisnika (ili određenog korisnika), koristili bi *SELECT * FROM korisnici (WHERE id = 12)*
- Ukoliko želimo promijeniti lozinku korisnika sa ID-om 12, koristimo sljedeći upit: *UPDATE korisnici SET lozinka = 'mario21' WHERE id = 12*

[1] Enumerated (hrv. Pobrojenja) - Tip atributa koji je definiran s pobrojenom vrijednošću može poprimiti samo vrijednosti iz domene pobrojene vrijednosti. Primjer pobrojenja je sljedeći: **CREATE TYPE** *spol* **AS** *ENUM* ('M', 'Ž');

- I na koncu, ukoliko želimo istoga izbrisati *DELETE FROM korisnici WHERE id = 12, no preporuka je da se podaci ne brišu iz tablica već se koriste atributi statusa te taj status mijenja u aktivan, neaktivan. Pogledati prethodnu točku.*
- Korištenje model-view-controller (MVC) sheme unutar vašeg projekta je obavezno kako biste što lakše odvojili podatke od korisničkog sučelja.
- CRUD funkcionalnosti potrebno je obavezno integrirati u u MVC shemu, gdje su sve CRUD funkcije definirane unutar modela te potrebno je integrirati objektno relacijsku shemu (ORM).

Upozorenje: *Ako u DELETE upiti ne stavite neki uvjet izbrisati će se svi podaci iz tablice*

7. Implementacija sustava za prijavu i registraciju korisnika

Potrebno je u PHP-u implementirati prijavu i registraciju korisnika. Registracija je zapravo jedna od standardnih CRUD operacija, samo što je vidljiva svima koji dođu na web sustav.

- Da bi ste ispravno implementirali prijavu, potrebno je koristiti sesijske varijable (engl. Session, pogledati https://www.w3schools.com/php/php_sessions.asp).
- Ukoliko se korisnik uspješno prijavi, unutar tih varijabli zapisujemo osnovne podatke o njemu kako bi smo im jednostavno mogli pristupiti iz svakog .php dokumenta u našoj web aplikaciji.
- Bitno je voditi računa da li prijavljeni korisnik smije pristupiti nekim dijelovima sustavima, npr. gost ili obični korisnik ne bi trebao imati uvid u administraciju.
- Nije dovoljno u korisničkom sučelju sakriti linkove koji vode na administraciju, nego je obavezno da se unutar skripta koje su zadužene za administraciju odradi provjera prije svega prijave, a zatim i tipa korisnika, te u slučaju nedozvoljenog pristupa preusmjeriti korisnika na stranicu u kojoj piše da pristup nije dozvoljen. (*Error code: 403*).
- Korisnik u svakom dijelu sustava treba imati vidljiv link (minimalno) za odjavu, poželjno je implementirati ikonicu (opcionalno sa tekstom).

8. Implementacija preostalih funkcionalnosti sustava

Nakon implementacije prijave i registracije korisnika potrebno je implementirati preostale funkcionalnosti sustava. Ostale funkcionalnosti ovise o temi i namjeni sustava. Potrebno je implementirati min. 10/13 funkcionalnosti (dopušteno je koristiti više od 10 ☺)

- Web košaricu
- Pretraživanje stranice/korisnika/proizvoda
- Implementirati višestruke filtere (min. 3 uvjeta istovremeno)
- Upload slike, pdf
- Vremenska prognoza, svjetsko vrijeme (na vlastitom projektu prikazati podatke od vanjskog servisa)
- Implementirati PHP Framework (npr. Laravel, Codeigniter, itd.)
- Ispis – prilagoditi dio stranice za print
- Upravljanje korisnicima (urediti, obrisati, dodati, **promijeniti ulogu**)
- Korištenje AJAX[2]-a (primjer - https://www.w3schools.com/js/js_ajax_intro.asp)
- Korištenje JavaScript programskih MVC okvira kao što je AngularJS
- Korištenje pomagala za izgradnju korisničkog sučelja kao što je JQueryIU (Integracija odabira datuma engl. datepicker, dodavanje povuci i ubaci metoda engl, drag and drop) te brojnih drugih pogodnosti JQuery UI
- Korištenje WYSIWYG editora (primjer takvog editora je TinyMCE)
- Korištenje GIT-a za suradnju

9. Izrada tehničke i korisničke dokumentacije za sustav

Prije zadnje demonstracije i obrane projekta potrebno je izraditi tehničku i korisničku dokumentaciju projekta. Predlošci za tehničku i korisničku dokumentaciju dostupni su na Moodle sustavu.

[2] Ajax je tehnologija za asihroni pristup podacima, te slanje HTTP zahtjeva asinhrono u pozadini, koristi se u raznim implementacijama za učitavanje manje količine podataka (ne učitava se dijela stranica, već samo njen dio)

10. Demonstracija i obrana projekta

Kada student položi kolokvije ili pismeni ispit svi članovi tima moraju uživo demonstrirati projekt te predstaviti sve njegove funkcionalnosti. Nakon obrane slijedi ocjenjivanje projekta.

Tablica 1. Bodovna skala projekta (potrebno je imati minimalno 50 bodova da bi student mogao pristupiti obrani projekta)

Naziv funkcionalnosti	Obavezno	Bodovi
Web aplikaciju postaviti na besplatni (ili vlastiti) hosting	*	**0**
Vizija, predložak u wordu (popunjen) postaviti u web okruženje	*	**4**
Dijagram slučaja uporabe engl. use case diagram		**2**
Spojiti se na bazu na proceduralnim (standardnim) načinom	*	**4**
Spojiti se na bazu objektno-orijentiranim pristupom (npr. PDO ili MySQLi OO)		**4**
Implementacija prijave i registracije korisnika	*	**6**
Baza mora obvezno sadržavati tablicu korisnika s korisničkim imenom, lozinkom i vrstom korisnika i mora sadržavati bar jednu tablicu sa stranim ključem	*	**6**
Spremanje «role and permission» informacija[3]		**8**
Povezivanje PHP-a s bazom podataka i izrada forme za CRUD svih elemenata u bazi podataka	*	**10**
Implementacija Javascript, jQuery – po vlastitom izboru	*	**6**
Implementirati vlastiti predložak uz pomoć Bootstrap tema (Npr. https://bootswatch.com)		**6**
Implementirati PHP Framework (npr. Laravel, Codeigniter)		**16**
Korištenje model-view-controller (MVC) sheme unutar projekta je obavezno kako biste što lakše odvojili podatke od korisničkog sučelja		**6**
Integrirati objektno relacijsku shemu (ORM)		**6**
Korištenje JavaScript programskih MVC okvira kao što je AngularJS		**4**
Korištenje WYSIWYG editora		**2**
Korištenje GIT-a za suradnju		**10**
Korištenje vlastitih rješenja, izbor tehnologije koja nije navedena može zamijeniti bodove za navedene funkcionalnosti…		
Ukupno		**100**

Projekt se smatra završenim/predanim kada se pošaljete link i source code (zip/rar) datoteku na mail **tomislav.volaric@fpmoz.sum.ba**

Da bi student imao mogućnost usmeno izlagati završeni projekt potrebno je:

- položiti pismeni dio ispita (ili kolokvirati)
- dovršiti online lekcije na Moodle sustavu
- na stranici https://www.codecademy.com/ dovršiti [4]online tečajeve:
    - Introduction To JavaScript, Introduction to jQuery, Learn SQL

Bodovna skala predmeta

- Pismeni dio nosi 100 bodova (2 kolokvija po 50), potrebno je imati min 50%
- Projekt 100 bodova
- Moguće je dobiti dodatne bodove kroz predavanja i vježbe

100 bodova – dovoljan (2)
130 bodova – dobar (3)
160 bodova – vrlodobar (4)
180 bodova – izvrstan (5)

dr.sc. Tomislav Volarić, doc.

[3] da bi spremili u bazu podatke o ulogama i dozvolama (role and permission) trebate imati 4 tablice – *uloge* - pohranjuje ID uloge i naziv uloge, *dozvole* - pohranjuje ID dozvole i opis, *uloga_dozvola* – koja dozvola pripada kojoj ulozi, *korisnik_uloga* – koja uloga je dodijeljena kojem korisniku

[4] koliko dopušta besplatna verzija