

*Jesenji semestar, 2016/17*

*PREDMET: IT350 Baze podataka*

*Projektni zadatak*

2 – Fabrika Bambi

Ime i prezime: **Nemanja Kuzmanović 2851**

**Sava Jeremić 2733**

**Bojan Petrović 2745**

Datum izrade: **27.01.2017**

*Beograd 2017.*

Sadržaj

[Abstrakt: 3](#_Toc473319304)

[Metodologija rada (način rada i korišćeni alati): 4](#_Toc473319305)

[Koncept: 5](#_Toc473319306)

[CRUD aplikacija i SQL upiti: 8](#_Toc473319307)

[Zaključak: 12](#_Toc473319308)

[Reference: 13](#_Toc473319309)

# Abstrakt:

**Opis problema:**

Fabrika Bambi je odlučila da angažuje studnete FIT-a da projektuju bazu podataka kojom bi poboljšala upravljanje svojim proizvodima i zalihama.

Fabrika se sastoji iz proizvodnih jedinica. Svaka proizvodna jedinica se identifikuje svojim brojem.

Proizvodna jedinica ima i svoje ime, mesto i ukupan kapacitet proizvodnje. Svaka jedinica može da proizvodi više proizvoda. Poznata je u svakom momentu, za svaku od proizvodnih jedinica i za svaki proizvod koji se u toj jedinici proizvodi, količina tog proizvoda, kao i minimumalni i maksimalni kapacitet proizvodnje.

Gotovi proizvodi se odlažu u skladišta. Skladišta se razlikuju po svom identifikacionom broju (koji mora biti sa spiska skladišta). Takođe, svako skladište ima svoj naziv, mesto u kome se nalazi i ukupni kapacitet zaliha koje čuva. U jednom skladištu može da se nalazi više vrsta proizvoda. U svakom trenutku je poznata trenutna količina svakog proizvoda u svakom skladištu.

Jedan proizvod može da se proizvodi u više proizvodnih jedinica fabrike Bambi, a takođe, može da se skladišti u bilo koje skladište ove fabrike. Svaki proizvod mora da ima svoj jedinstveni identifikacioni broj koji se nalazi na spisku proizvoda. Pored toga, proizvod mora da ima naziv, jedinicu mere i cenu.

Jedinična količina gotovog ili polugotovog proizvoda proizvodi se od određenih količina drugih poluproizvoda koje proizvodna jedinica naručuje iz svakog skladišta.

Narudzbenica jedne proizvodne jedinice mora da se uputi u tačno jedno skladište. Svaka narudzbenica mora da se identifikuje svojim brojem, da ima datum kada je poslata. Jednom porudzbenicom je moguće naručiti kako jedan, tako i više proizvoda ali količine tih proizvoda moraju biti navedene.

U okviru projekta treba:

1. Definisati konceptualni model podataka

2. Konceptualni model transformisati u fizički

3. Izvršiti transformaciju fizičkog modela u skripte za kreiranje elemenata baze podataka

4. Napraviti CRUD aplikaciju koja komunicira sa bazom podataka

5. Napisati SQL upite za sledeće probleme i omogućiti njihovo izvršavanje i prikaz rezultata kroz aplikaciju

Prikazati naziv i mesto proizvodne jedinice, kao i ukupnu cenu svih proizvoda koje je ona napravila.

Prikazati podatke o skladištima koji su čuvaju najviše različitih proizvoda. Voditi računa ako postoji dva ili više takva skladišta, prikazati podatke o svakom od njih.

Naći podatke o poslednjoj narudžbenici koju je bilo koja proizvodna jedinica uputila nekom skladištu, naziv te proizvodne jedinice, datum narudžbenice, naziv skladišta.

Sortirati u opadajućem redosledu prema ceni, sve proizvode koje Fabrika Bambi proizvodi.

Napraviti statistiku, na dnevnom nivou, za svaki od proizvoda koji je tog dana proizveden u količini od preko 100 kg. Prikazati datum, naziv proizvoda, količinu koja je tog dana proizvedena.

Za skladište u kome ima najviše robe, prikazati podatke o narudzbenicama iz tog skladišta za poslednjih mesec dana.

# Metodologija rada (način rada i korišćeni alati):

Prvenstveno, rad ima za cilj da se iskoriste znanja stečena na kursu IT350, konkretno, pravljenje konceptualnog modela baze podataka, sa svim entitetima i atributima koji odgovarajuća baza poseduje, određivanje primarnih kljuceva kao i kardinalnosti. Sve ovo se postiže alatom zvanim “PowerDesigner”. Kada se stečena znanja ubace u ovakav alat i osposobi se konceptualni model, potrebno je prebaciti konpectualni u fizički (kasnije će biti detaljnije objašnjeno). Na samom kraju u softveru potrebno je generisati iz ovako napravljene baze, SQL skriptu koja bi se importovanjem u odgovarajući alat prebacila, entitete u tabele, a attribute u same kolone te tabele. Ovaj projekat je za tu svrhu koristio “phpmyadmin” kao i “xampp” za podizanje virtuelnog servera.

U samom phpmyadmin-u su direktno ubačene vrednosti svake kolone i tabele kao što je npr: Proizvod, Proizvodna jedinica, Stanje itd.

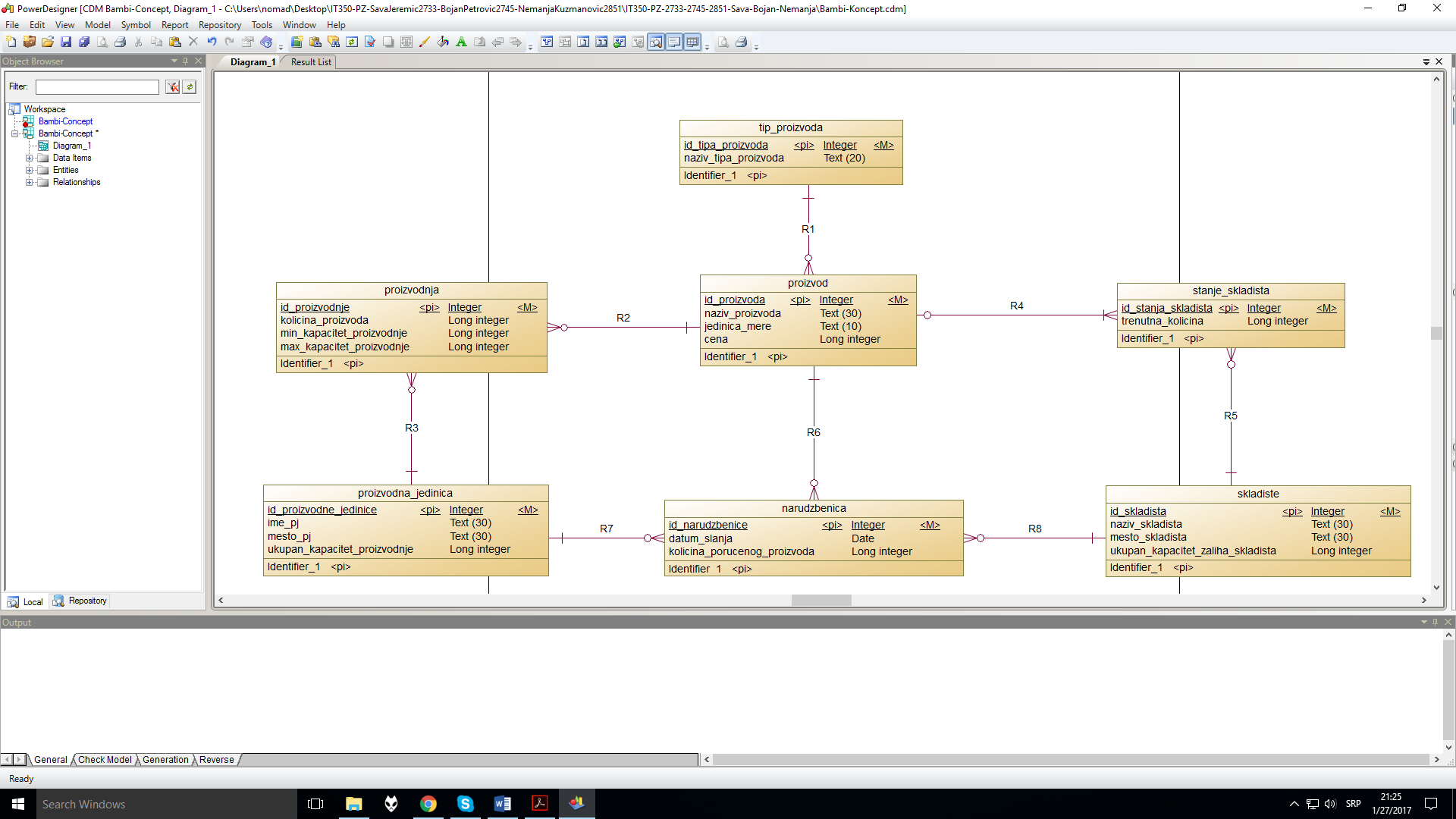
Pre pravljenja same CRUD (Create, Read, Update, Delete) aplikacije, generisani su zadati SQL upiti za određene operacije nad samom bazom podataka.

Na samom kraju se pristupilo pravljenu i CRUD aplikacije pomoću jezika “PHP”, “HTM”, “SQL”, “CSS”, i “JavaScript”.

Svaki deo projekta je testiran više puta i ustanovljeno je da je projekat spreman za dalju distribuciju.

# Koncept:

Dakle, kao što je navedeno, prvenstveno se krenulo u izradu konceptualnog modela baze podataka pomoću alata PowerDesigner. Napravljeni su entiteti i njihovi atributi sa primarnim ključevima a kasnije se pristupilo određivanju kardinalnosti između entiteta ovako napravljenog koncepta baze podataka.



*Slika 1: PowerDesigner - Koncept*

Konceptualni model je na osnovu priloženog zadatka izrađen na sledeći način.

Proizvodna jedinica i proizvod se nalaze u vezi više na više što rešavamo tako što ubacujemo među tabelu, ovde nazvanu Proizvodnja, u njoj čuvamo zahtevane atribute a pored njih, u fizičkom modelu će se spustiti primarni ključevi iz proizvodne jedinice, i proizvoda. Ovim dobijamo da u svakom trenutku možemo proveriti koja proizvodna jedinica proizvodi koji proizvod, u kojim količinama i koji su minimalni a koji maksimalni kapaciteti proizvodnje. Kardinalosti su određene na ovaj način iz razloga što ukoliko bi postojala proizvodnja mora da postoji barem jedna proizvodna jedinica i barem jedan proizvod. Dok u slučaju same proizvodne jedinice ona ne zavisi od proizvodnje, ona može postojati ali ne mora biti aktivna pa samim tim nema potrebe za ubacivanjem u proizvodnju. Istom logikom je rešena i kardinalnost kod proizvoda, on može da postoji ali ne znači da je primoran da se nalazi u samoj proizvodni, naime, može biti u skladištu kao gotov proivod.

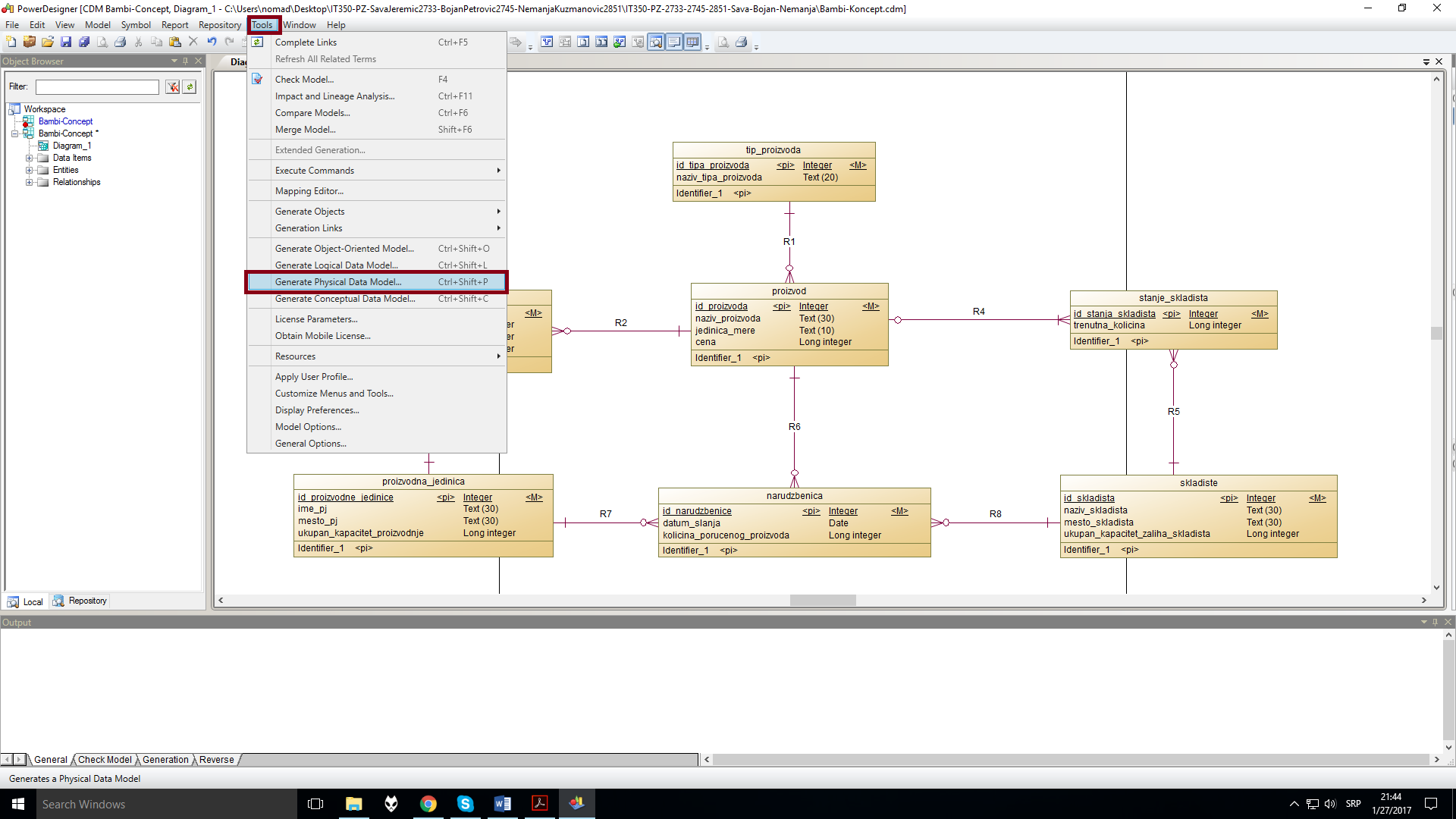
Tip proizvoda je rešen kao 1 – gotov i 2 – polugotov, iz razloga što se gotov proizvod pravi od više polugotovih, primera radi ukoliko imamo tip1 gotov proizvod nazvan krem banana, kako bi ovaj proizvod napravili potrebno nam je dva polu proizvoda (ako ne i više) , čokolada i ispuna, s tim što da bi napravili da bi napravili poluproizvod čokolada moramo imati takođe poluproizvode poput kakaoa i mleka, koji se mogu ili proizvoditi ili naručiti iz skladišta. S druge strane možemo istu tu čokoladu imati kao tip1 kao gotov proizvod, gde bi išla direktnu u distribuciju ka krajnjim korisnicima a ne u proizvodnju za spravljenje nekog gotovog proizvoda.

Takođe, skladište i proizvod imaju vezu više na više po tekstu, pa se i ova veza rešila među tabelom, ovoga puta nazvanom Stanje skladišta. Razlog toga je što u tekstu traži da u svakom trenutku znamo koji se proizvod nalazi u kom skladištu što ovim to i postižemo. U fizičkom modelu će se ključevi iz skladišta i proizvoda spustiti u ovu tabelu i imaćemo stanje, koji se proizvod nalazi u kom skladištu u kojoj količini.

Još jednom imamo vezu više na više a to je između tabela skladište i proizvodna jedinica, naime, po tekstu, proizodna jedinica može naručiti iz više skladišta potrebštine a takođe skladište može primiti više narudžbina.

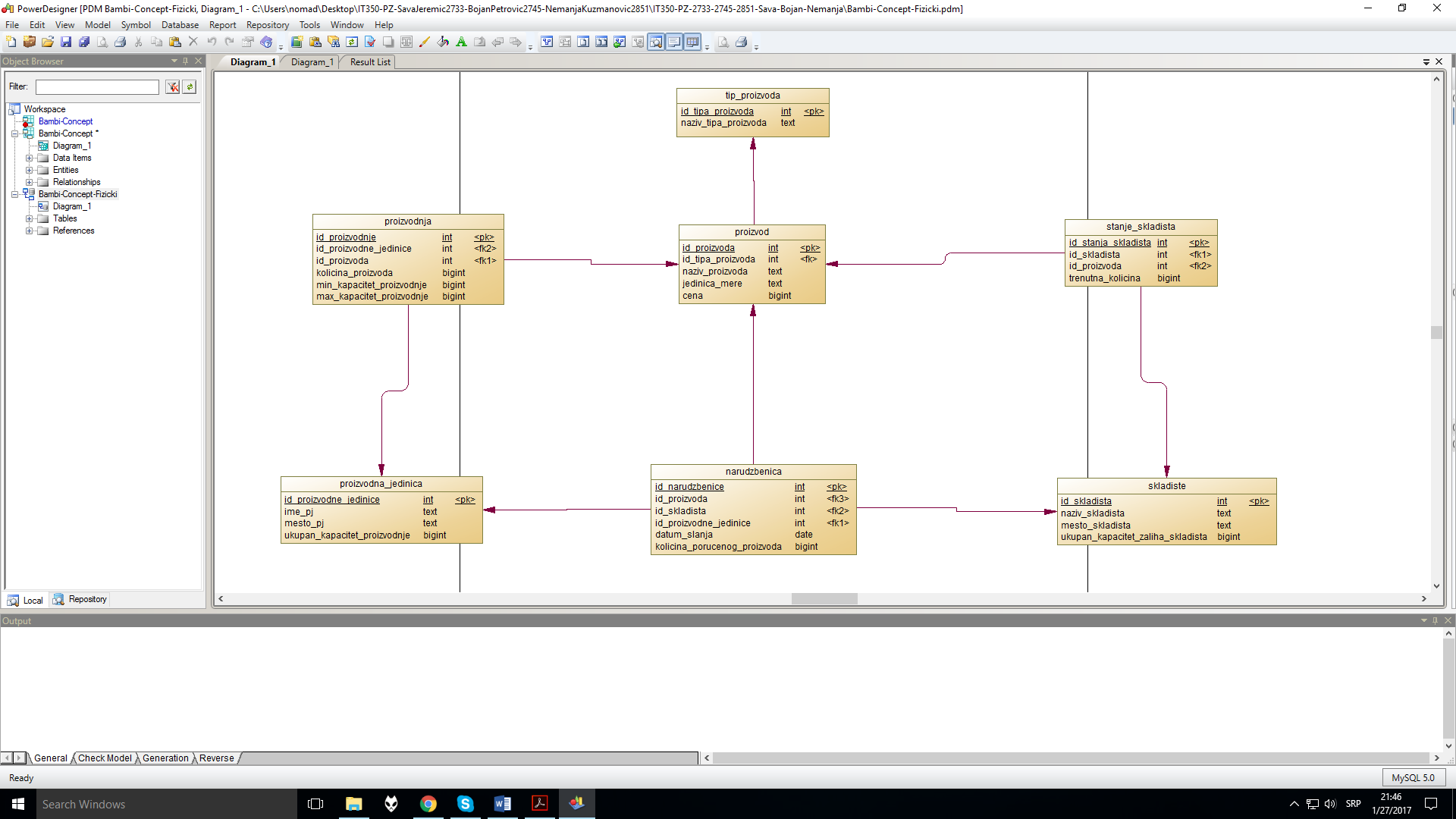
Pored toga traži se i implementacija narudžbenice koja bi imala ulogu da čuva podatke koja proizvodna jedinica u kom trenutku naručuje određeni proizvod iz samog skladišta. Tako u narudžbenici imamo potred podataka o jedinici koja naručuje, skladišta iz kog se naručuje i podatke o datumu kada se koji proizvod naručio u kojoj količini. Dodatno, iz tog razloga se i sam proizvod spustio u narudžbenicu kako bi imali uvid u to koji je proizvod naručen.  
  
Ovim je rešen sam koncept baze podataka zahtevan u tekstu projekta.

Sledeći korak je da se pristupi generisanju fizičkog modela, na sledeći način:



*Slika 2: Generisanje fizičkog modela*

Potom, dobijamo sledeće:



*Slika 3: fizički model*

U fizičkom modelu možemo detaljnije videti šta se desilo sa tabelama i koji primarni ključevi su se spustili u koju tabelu, ovako, imamo uvid u potpunu bazu podataka koja bi trebala biti importovana sledećom generizacijom sql skripte.

Kada generišemo skriptu to radimo na sledeći način – Database pa Generate Database, ovde imamo osnovna podešavanja za samu skriptu koja će biti generisana i možemo je importovati u phpmyadmin.

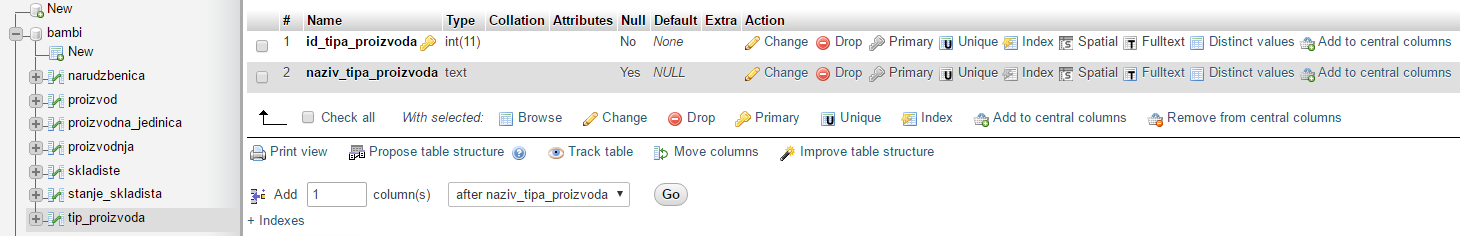
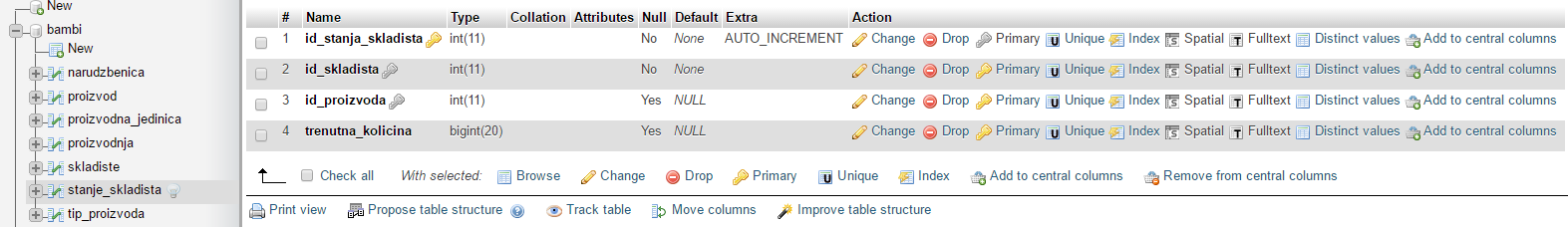
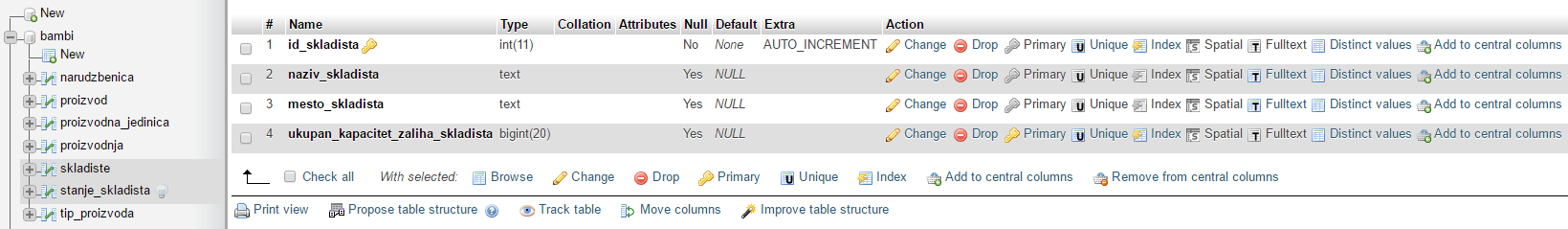
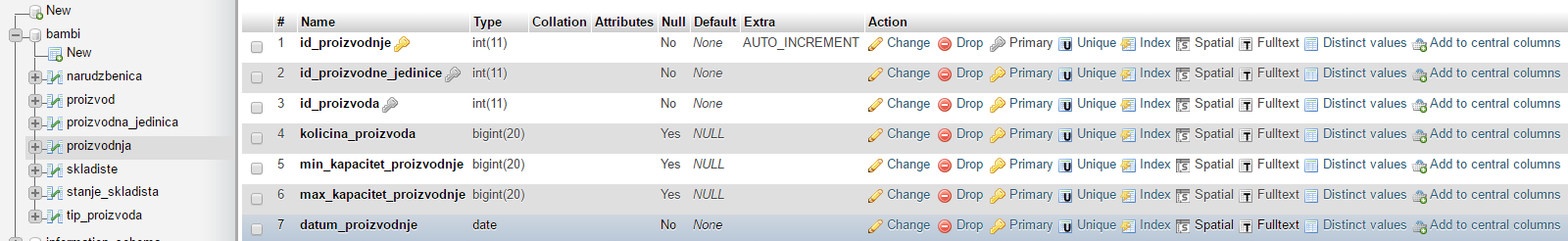
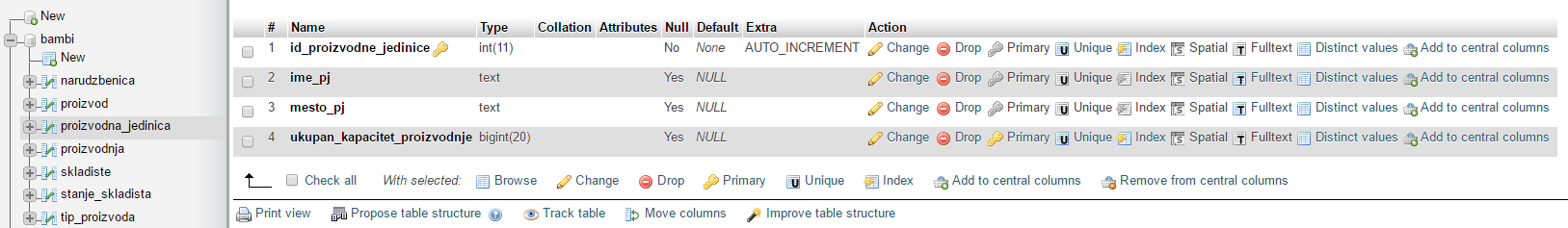
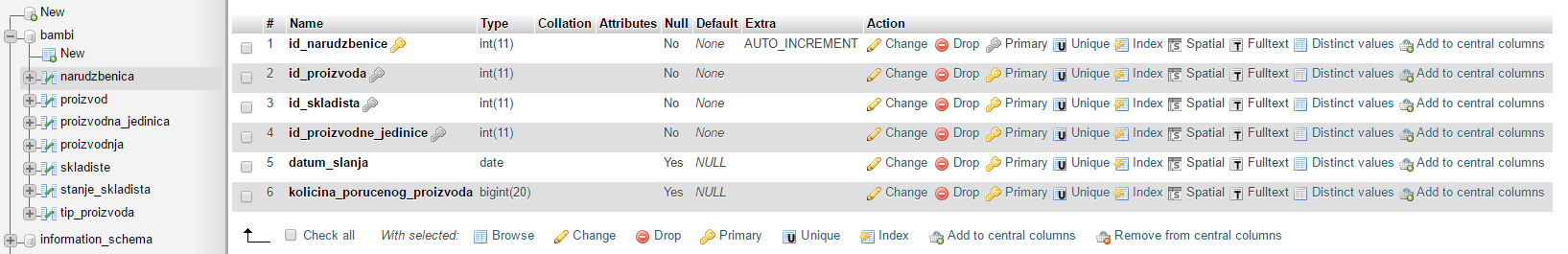
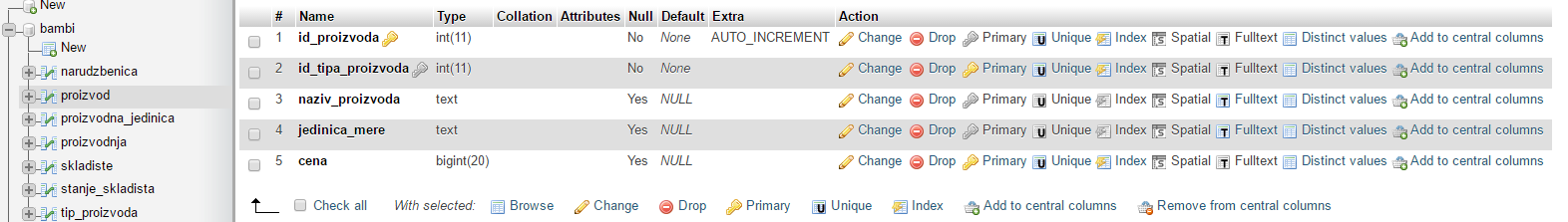
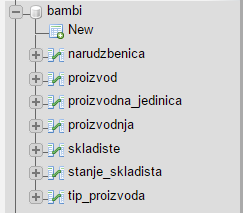
Napomena:

Skripta koju smo generisali je sa kardinalnostima, pa, po ubacivanju skripte za inserte tj. Detalje tabele i kolone može doći do greške u konstrejntovima. Ovo smo izbegli tako što smo prvo generisali skriptu bez kardinalnosti a u phpmyadminu-u u SQL skripti manuelno dodali konstreintove, ovim putem smo uspeli da izbegnemo određene greške samih alata kod ubacivanja i brisanja iz same baze, a isti ti problem su već poznati, i jedini način pored ovoga da se reše je uz trik da se foreign ključevi prvo uklone, pa se ubace određeni podatci a oni se potom dodaju, što dodano komplikuje stvari.

# CRUD aplikacija i SQL upiti:

U samom phpmyadmin-u se nije mnogo radilo, pored konstreintova i ubacivanja podataka jedino je deo SQL korišćen za isprobavanje određenih upita koji su kasnije implementirani u samoj aplikaciji i kroz ovaj deo će biti detaljno proučeni.

Baza phpmyadmin sa tabelama:



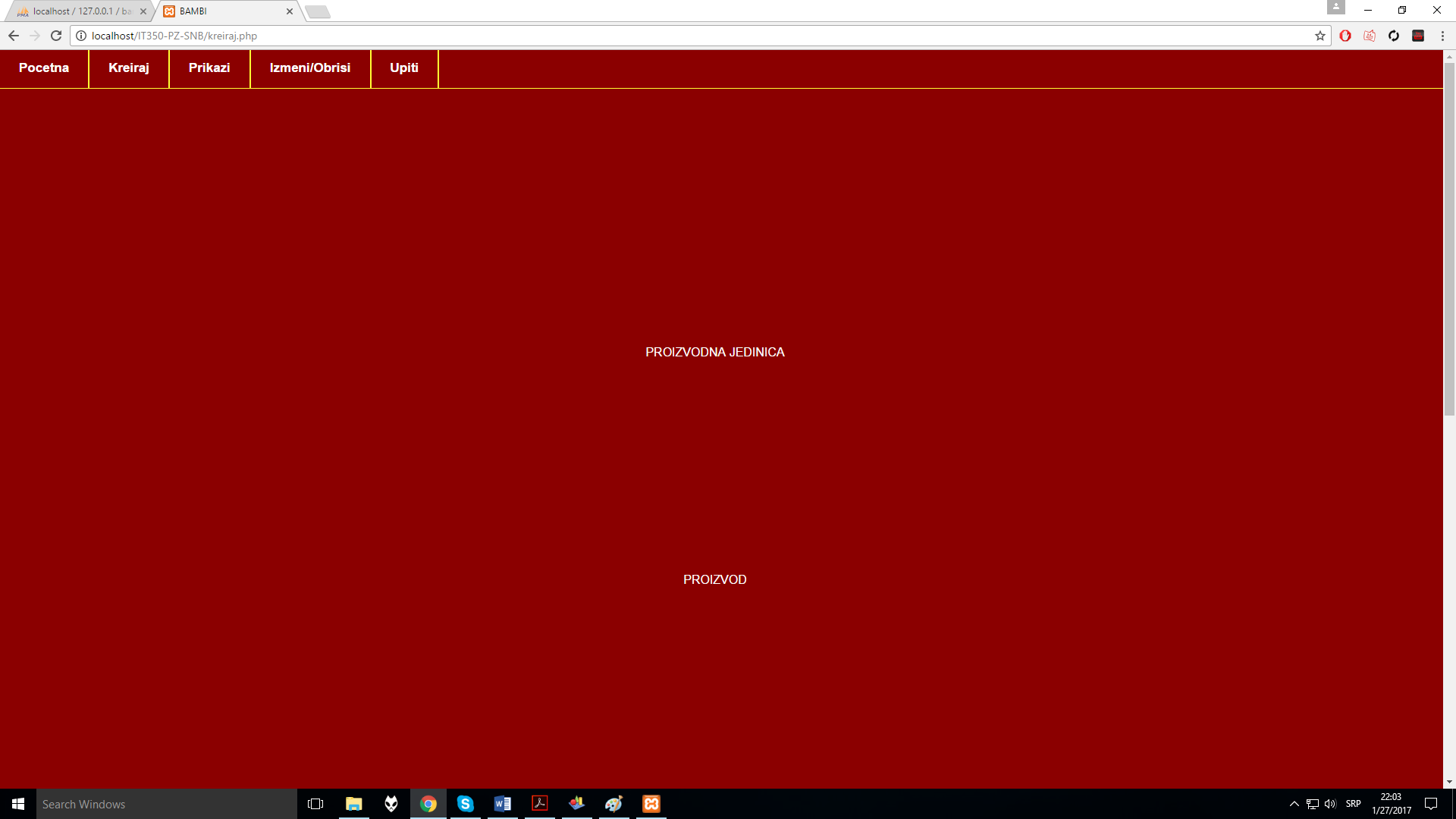
*Slika 4: phpmyadmin*

-CRUD aplikacija:



*Slika 5: CRUD aplikacija*

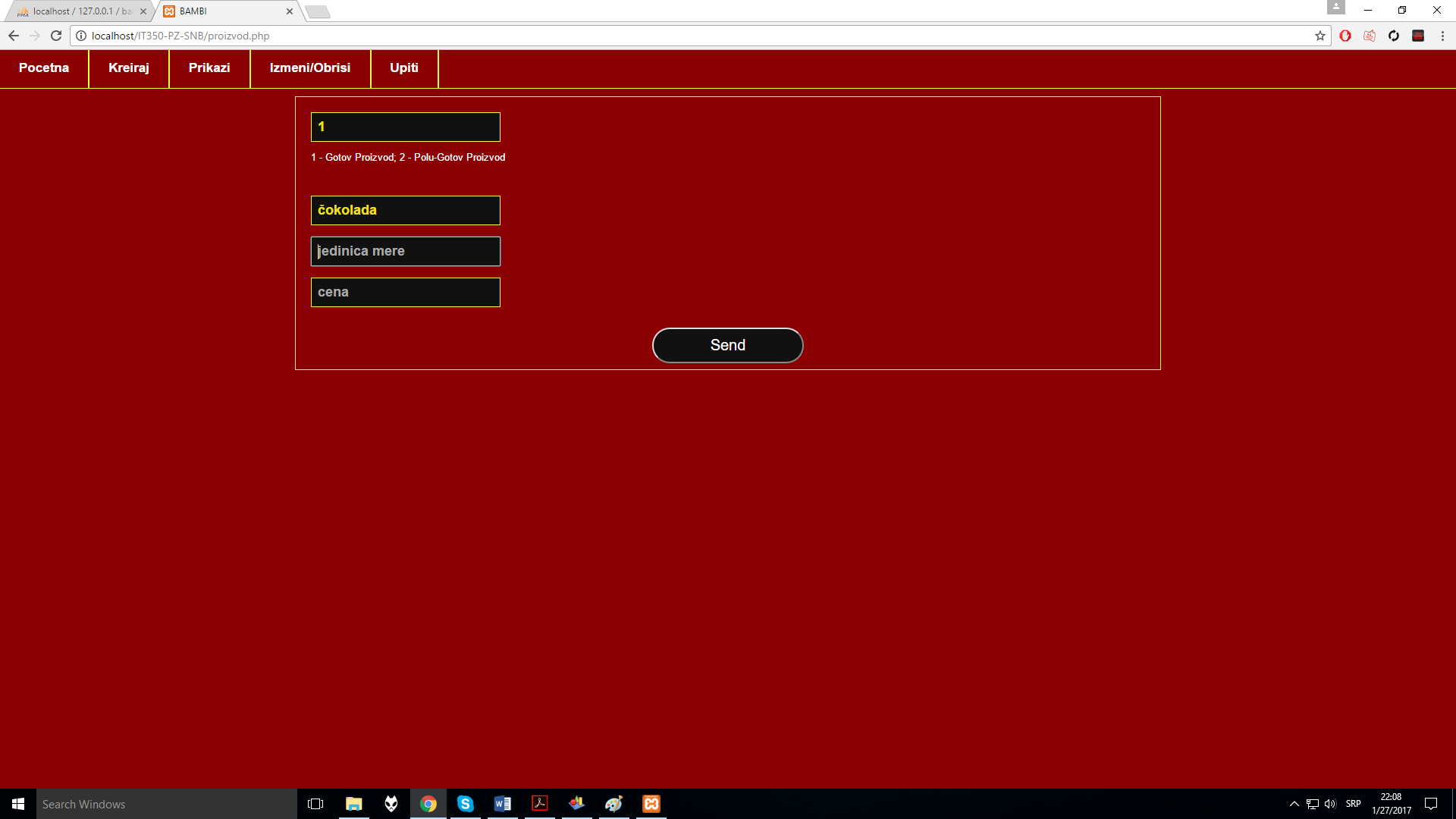
Početna strana CRUD aplikacije sadrži padajući navigacioni meni, veoma je jednostavna i laka za korišćenje, klikom na gornji padajući meni navodimo do određenog dela same aplikacije.



*Slika 6: Kreiraj*

Prva stvar koja upada u oči kada se klikne na stranu Kreiraj je skrolujući objekti, naime ovo je urađeno iz razloga što bi najverovatnije radnici koristili tablet ili mobilni telefon za pristup aplikaciji u svakom tenutku pa im se ovim to i omogućava (slike izostaju zbog ne odgovarajućih logoa, kako ne bi narušili izgled aplikacije a veoma je jednostavno da ih budući korisnici sami ubace po svom nahođenju – nismo dizajneri)

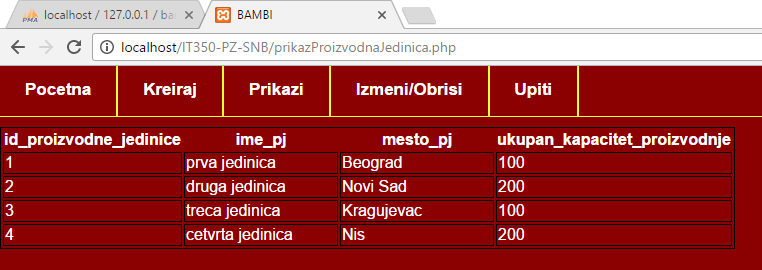
Ovakav meni nudi sledeće opcije: Proizvodna jedinica, Proizvod, Proizvodnja, Skladište, Stanje, Narudžbenica, Izabirom na odgovarajući taster dobijamo sledecu stanu na kojoj možemo uneti odgovarajuće podatke a ti podatci će se automatski ubaciti u bazu podataka.



*Slika 7: Kreiraj Meni*

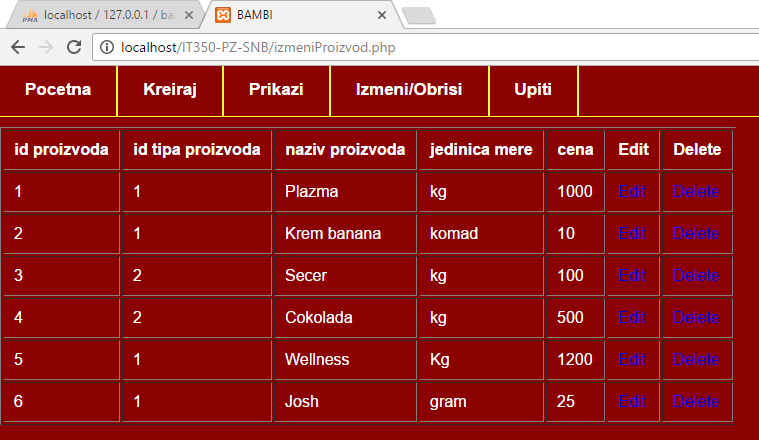
Prvenstveno biramo tip proizvoda kao što je već bilo reči, jedan je proizvod a dva polu proizvod, u ovom primeru je za ime uneta čokolada, a ispod se vide nazivi šta je neophodno uneti u sledeća polja, dakle, jedinica mere i cena.

Svaki deo kartice Kreiraj ima određenu formu za unos u samo bazu podataka koja se neće ovde detaljno prikazati.



*Slika 8: Prikazi*

U kartici prikaži takođe kao kod kartice kreiraj dobijamo meni sa izvorom stavki koje je moguće prikazati iz baze podataka, u ovom slučaju izabran je prikaz svih proizvodnih jedinica kao što se može videti na priloženoj slici.

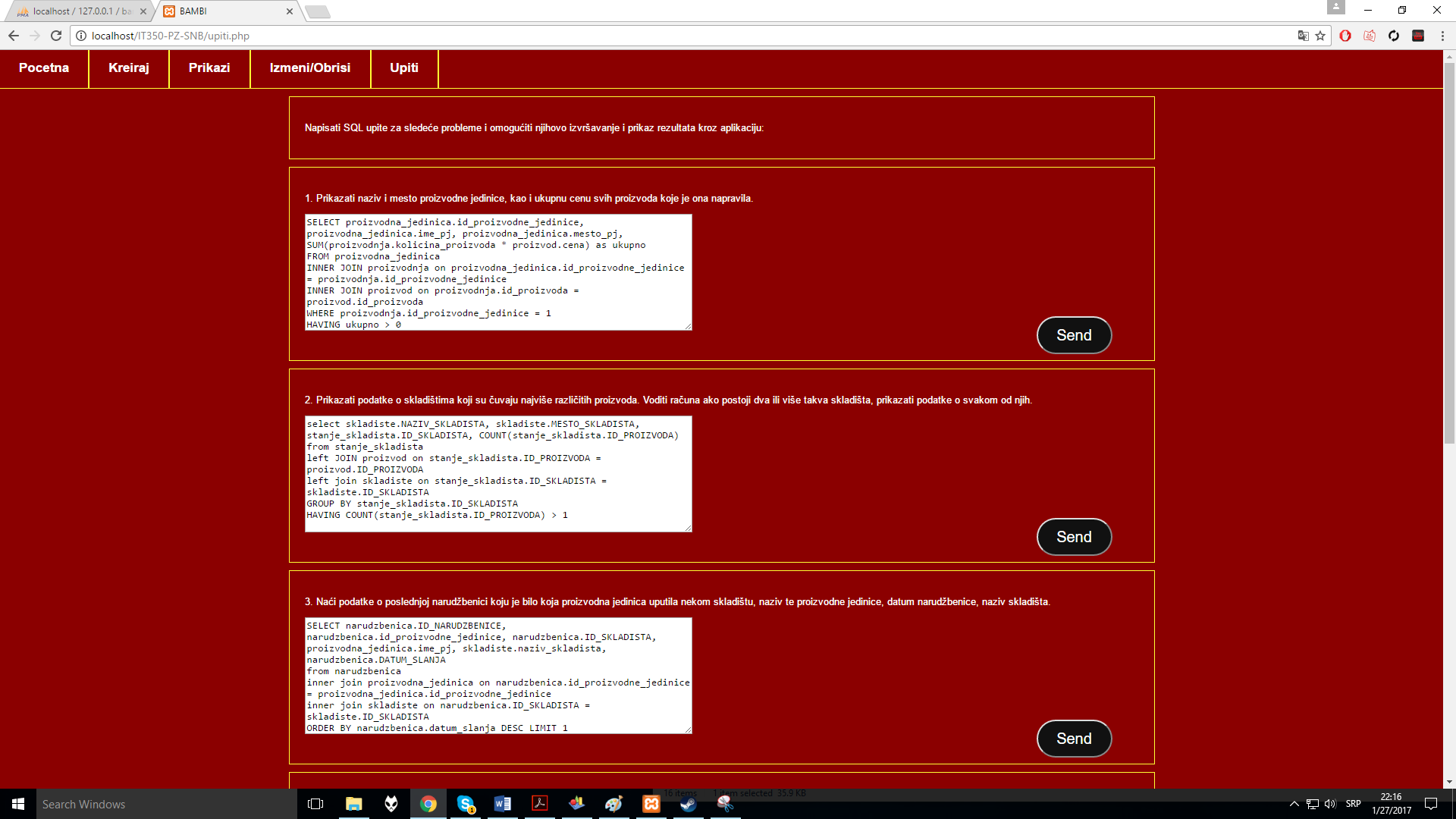


*Slika 9: Izmeni Obriši*

Pored osnovnih kartica za dodavanje i prikaz, naravno, implementirane su i ostale osnovne opcije same CRUD aplikacije a to su izmeni i obriši, ovoga puta kada izaberemo odgovarajuću karticu, ovoga puta proizvod, dobijamo detaljan prikaz svih proizvoda sa dodatnim opcijama za izmenu i brisanje, ovo je urađeno na ovaj način iz razloga što jako lako ovim putem možemo podeliti korisnike da vide prve dve strane, a administratoru dati dozvodu da pristupi samo ovoj strain.

Kada se izabere taster edit otvara se nova forma u kojoj se sadrže trenutni podatci i posle izmene i slanja tih podataka, ponovo će se prikazati ista tabela sa izmenjenih vrednostima.

Delete je veoma jasan, briše odgovarajući red iz tabele



*Slika 10: Upiti*

Poslednja kartica je implementirana samo za potrebe projekta i sadrži unešene SQL upite zadate projektnim zadatkom koji na zahtev asistenta možemo vrlo lako izmeniti, pritiskom na tater SEND, dobijamo tabelu u kojoj se određeni SQL upit prikazuje.

# Zaključak:

Projekat kao projekat nije bio komplikovan, prvenstveno sama baza i njen koncept su bili u većini slučaja veoma jasni, osim dela sa narudžbenicom, jer se pominje porudžbenica, ovo je moglo biti vrlo lako izbegnuto pominjanjem ponovo reči narudžbenica, ali smo prevazišli ovaj problem na način na koji smo razumeli. Primera radi sutra kada budemo nekome izrađivali bazu podataka uglavnom ćemo biti u kontaktu sa njima i moći da razrešimo sve nedoumice, ovako, kada imamo tekst ispred sebe neka pitanja su ostala i dalje otvorena i ne razjašnjena.

SQL upiti su vrlo jasni i nije ih bilo teško implementirati, kao i sve ostale delove vezane za phpmyadmin.

Veoma dobro nam je došao projekat kako bi obnovili povezivanje same baze i php aplikacije a i samog php-a.

Dosta nedoumica oko same normalizacije i projekta smo zajedničkim snagama prevazišli i sada mnogo bolje baratamo istim, stekli smo dosta novih znanja i veština i veoma smo zahvalni za ovakav projekat, mnogo nam je značio, i značiće u nastavku školovanja.

Nadamo se da smo zadovoljnili zahtevane kriterijume za ovaj projekat, a takođe se i izvinjavamo što nismo dizajnirali CRUD kao kolege jer smo shvatili da je focus na SQL upitima i samoj bazi a ne na šminkanju sajta, što se može lako i odraditi ukoliko je zahtevano.

Kao završnu reč, Bojan,Nemanja i Sava, se zahvaljuju na ukazanom projektu i sansi da sami iskuse projekat iz realnog života na kojim će sigurno u budućnosti raditi.

# Reference:

Predavanje, vezbe, IT350

W3schools.com

Php.net

Stackoverflow.com  
  
lams.metropolitan.ac.rs:8080/lams/