

**E-Menza**

Predmet: Administriranje baze podataka

**Profesor: Studenti:**

**dr Dušan Stefanović Aleksa Nejković PEp 2/17**

**Nemanja Božilović PEp 8/17**

**Miloš Stanojković PEp 19/17**

**Jun 2020.**

**Sadržaj**

[1. Uvod 3](#_Toc42843106)

[2. Instalacija i podešavanje projekta 3](#_Toc42843107)

[2.1 Instalacija Python-a 3](#_Toc42843108)

[2.2 Instalacija Flask biblioteke u okviru Python-a 5](#_Toc42843109)

[3. Arhitektura aplikacije 6](#_Toc42843110)

[3.1 Serverski deo 7](#_Toc42843111)

[3.2 Klijentski deo 7](#_Toc42843112)

[3.3 Baza podataka 7](#_Toc42843113)

[3.3.1 Relaciona baza podataka 7](#_Toc42843114)

[3.3.2 SQL jezik 7](#_Toc42843115)

[3.3.3 Funkcije 9](#_Toc42843116)

[3.3.4 Trigeri 12](#_Toc42843117)

[3.3.5 Pogledi 14](#_Toc42843118)

[3.3.6 Procedure 16](#_Toc42843119)

[3.4 Komunikacija 18](#_Toc42843120)

[4. Rad aplikacije 19](#_Toc42843121)

[4.1 Opis implementacije 19](#_Toc42843122)

[4.2 Opis funkcionalnosti - korisničko uputsvo 21](#_Toc42843123)

5. [Literatura 36](#_Toc42843124)

# 1. Uvod

U ovom projektu obrađena je izrada veb aplikacije koja je bazirana na realnom primeru studentskih menzi. Tehnologije koje su korišćene na klijentskoj strani su Python, CSS sa Bootstrap framework-om, ikonice iz FontAwesome grupacije, dok je na serverskoj strani korišćen Python sa Flask framework-om.

Pretraživač koji smo koristili za testoranje projekta je Google Chrome.

Aplikacija se sastoji iz dva dela. To su deo za administratora i deo za studenta.

Na samom početku aplikacije se bira način prijavljivanja (**administrator / student**). Nakon toga, pristupa se delu u kojem smo se prijavili sa dodatnim mogućnostima.

Prvi deo, odnosno deo **administrator** se sastojoji od sledećih delova kojim može manipulisati: ***Studenti, Restorani, Obroci, Banke, Porudžbine***. Nešto o svakom delu u predstojećem tekstu.

Deo sa studentima se sastoji iz delova: ***“Restorani” i “Račun” i “Vaše porudžbine”*** gde student ima mogućnost poručivanja hrane iz željenog restorana, kao i da izvrši uplatu novca na svom računu. Nešto više u nastavku dokumentacije.

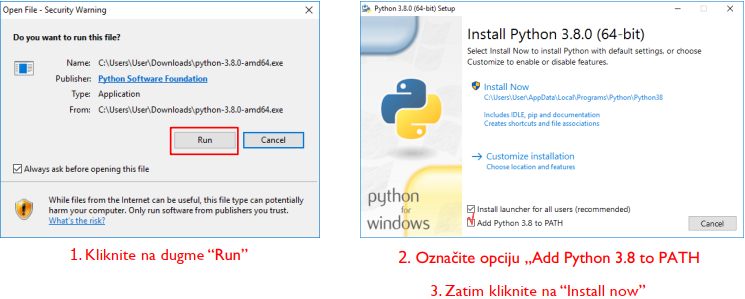
# 2. Instalacija i podešavanje projekta

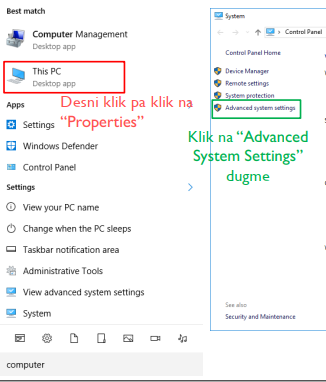
Da bismo pokrenuli samu aplikaciju neophodan nam je WAMP server koji koristimo kao deo koji nam omogućava korišćenje baze podataka odnosno MYSQL servis kao i grafički dizajn samog okruženja za kreiranje baze podataka - PHPMyAdmin, zatim instaliran Python i u okviru njega instaliran **Flask.**

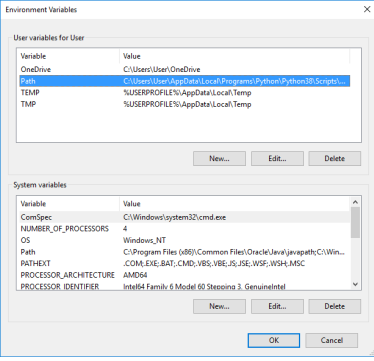
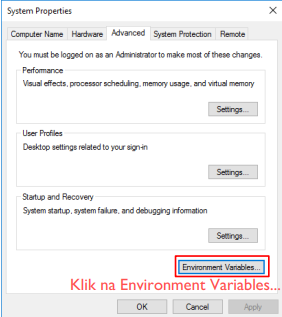
## 2.1 Instalacija Python-a

Za Windows operativne sisteme, koirsnik može preuzeti instalacione fajlove sa sledećeg linka:

<https://www.python.org/downloads/>

Zatim sledi instaliranje fajla kojeg smo preuzeli.****

****Nakon toga, podešavamo **environment path**:

****

Da bismo proverili da li smo uspešno instalirali Python na svom računaru, neophodno je da u Command prompt-u napišemo “**python**”:



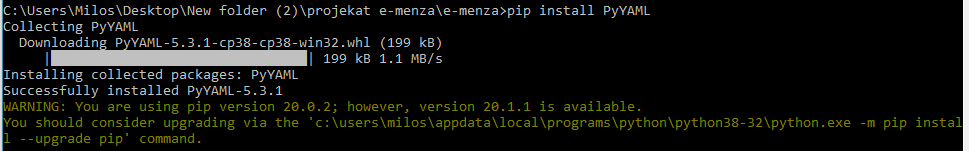
## 2.2 Instalacija Flask biblioteke u okviru Python-a

Instalacija Flask-a vrši se preko komandne linije: **py -3 -m pip install flask.**



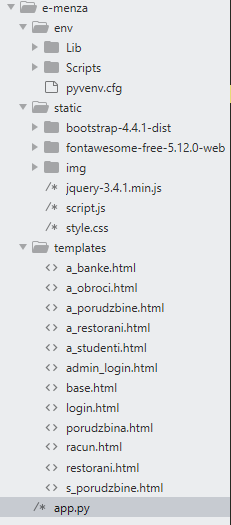
Sada imamo mogućnost korišćenja flask biblioteke, koju importujemo u python fajl.

Za proketanje projekta, moramo instalirati još jednu Python-ovu biblioteku koja se zove PyYalm. Služi za parsiranje u Python programskom jeziku. Instalacija se vrši na sledeći način:



# 3. Arhitektura aplikacije

Ovaaplikacijaima sledeću arhitekturu:



Sastoji se iz sledećih delova:

Deo pod nazivom **env** se kreira prilikom pokretanja python projekta i u njemu se smešta flask i druge neophodne biblioteke i moduli koje smo instalirali i importovali.

**Static** folder nam služi za smeštanje fajlova koje koristimo na klijentskoj strani. To su CSS, bootstrap, font-awesome set alata za font i ikone baziran na CSS-u i LESS-u i folder namenjen za smeštanje slika koje koristimo u projektu.

**Templates** folder služi za kreiranje html stranica, odnosno prikazivanje na veb strani. Svaka od ovih strana imaj svoju url adresu koje su definisane u fajlu **app.py.**  O ovom fajlu više u nastavku dokumentacije.

## 3.1 Serverski deo

Na serverskom delu imamo Flask, radni okvir za veb aplikacije koji daje skup opštih osnovnih tehnologija na kojima može da se gradi veb aplikacija. Komanda se povezuje na veb sajt PyPl, preuzima i instalira modul Flask i još 4 modula:

* Warkzeug
* MarkupSafe
* Jinja2
* itsdangerous

## 3.2 Klijentski deo

Na klijentskoj strani imamo skup tehnologija kao što su HTML kojim pravimo strukturu same stranice, zatim CSS za stilizovanje elemenata koji se prikazuju, imamo CSS-ovu biblioteku BOOTSTRAP koju smo takođe koristili.

## 3.3 Baza podataka

Baza podataka u ovom projektu ima izuzetno važnu ulogu. Potrebno je dobro osmisliti koncept baze podataka kao i vrstu iste pre same izrade samog projekta. U ovom projektu smo koristili MySQL relacionu bazu podataka. Kao podrška bazi korišćen je WAMP server. Zbog svoje moći i dobro konstruisanoj arhitekturi ovo je izuzetno korišćena tehnologija za izradu baza podataka.

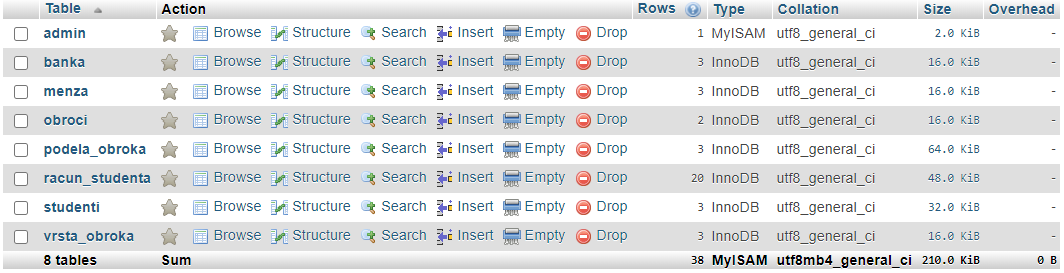
## 3.3.1 Relaciona baza podataka

Relaciona baza podataka je poseban tip baze podataka kod kojeg se organizacija zasniva na relacionom modelu. Podaci se u ovakvim bazama organizuju u skup relacija između kojih se definišu određene veze. Relacija se definiše kao skup n-torki sa istim atributima, definisanih nad istim domenima iz kojih mogu da uzimaju vrednosti. U relacionim bazama podataka, svaka relacija mora da ima definisan primarni ključ, koji predstavlja atribut pomoću kojeg se jedinstveno identifikuje svaka n-torka. Relacija opciono može da poseduje i spoljni ključ, preko kojeg ostvaruje vezu sa drugim relacijama.

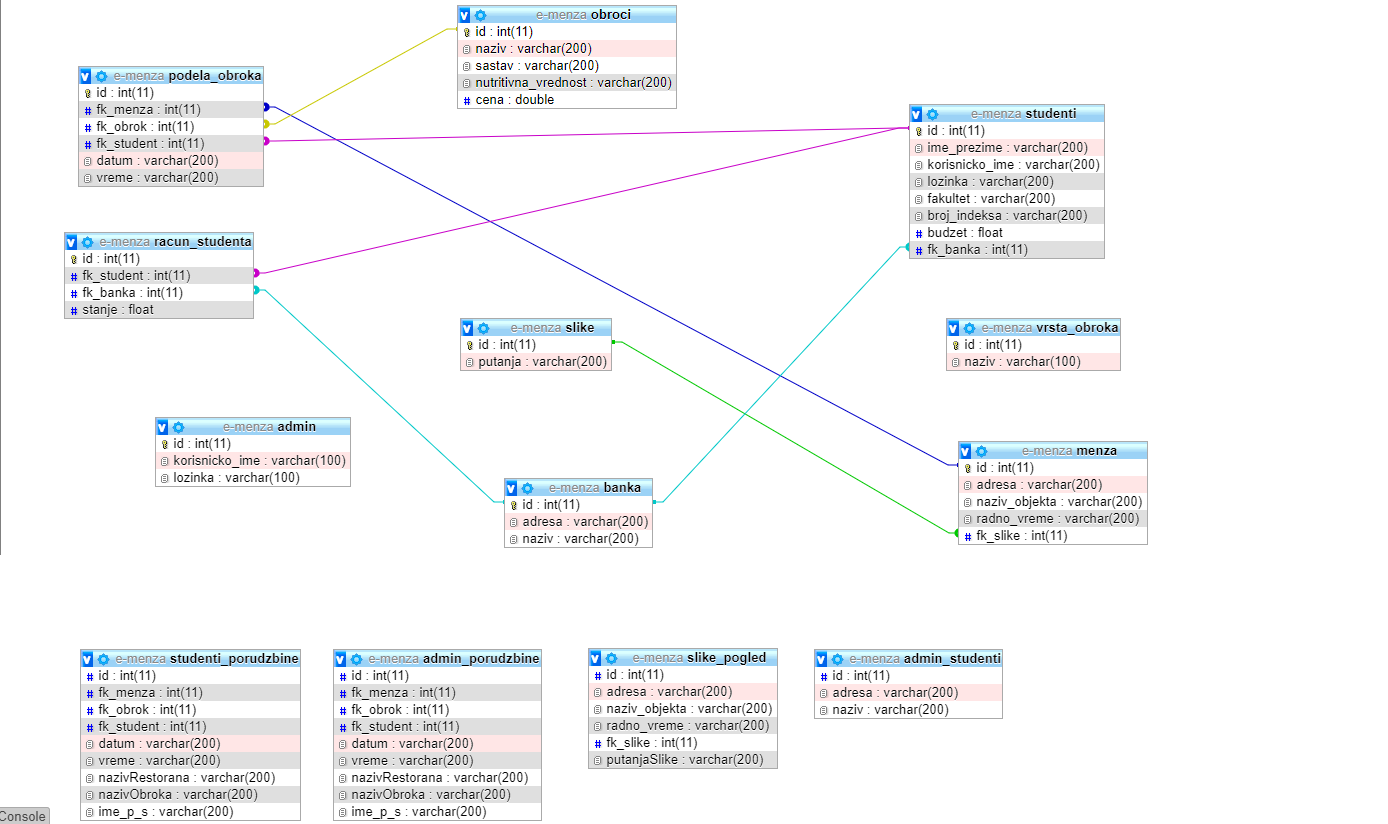
Upravljanje ovakvim bazama podataka se realizuje preko sistema za upravljanje relacionim bazama podataka. Među najpopularnijim takvim sistemima danas su: Microsoft SQL Server, Oracle Database, MySQL i drugi. Većina tih sistema koristi upitni jezik SQL za manipulaciju podacima.

## 3.3.2 SQL jezik

**SQL** (engl. Structured Query Language) je relacioni upitni jezik (ANSI i ISO standard). Relacije se kreiraju jednom naredbom i odmah su dostupne, što ga čini jednostavnim za korišćenje. Uniforman je, jer se svi podaci i rezultati operacija prikazuju u vidu tabele i omogućava interaktivno i klasično programiranje.

Prikaz tabela baze podataka u projektu:

ER dijagram baze podataka:



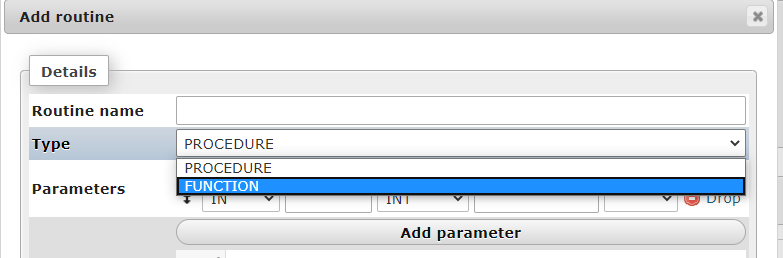
**U projektu su primenjene funkcije, trigeri, pogledi i procedure nad bazom podataka.**

## 3.3.3 Funkcije

Suština funkcija je da vraćaju vrednost koja može biti skalarna(jedna vrednost) ili tabelu.

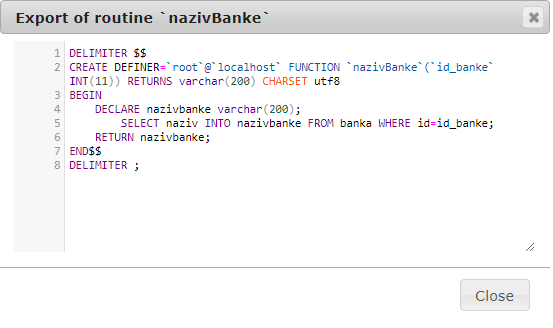
Funkcija je deo Select, Delete ili Update iskaza. Ukoliko vraća skalarnu vrednost poziva se u Select, Where ili Order segment. Ukoliko vraća tabelu poziva se u From segmentu SQL iskaza.

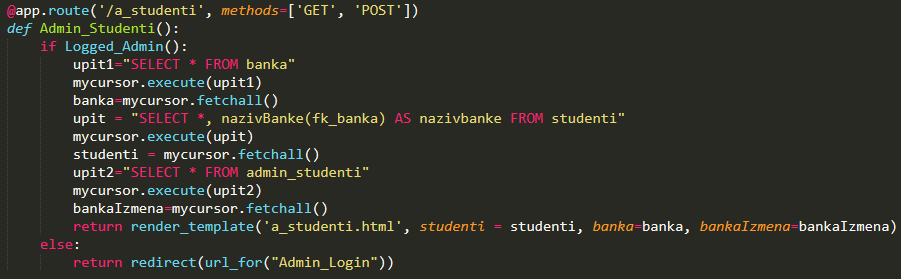
Funkcija se kreira na sledeći način:

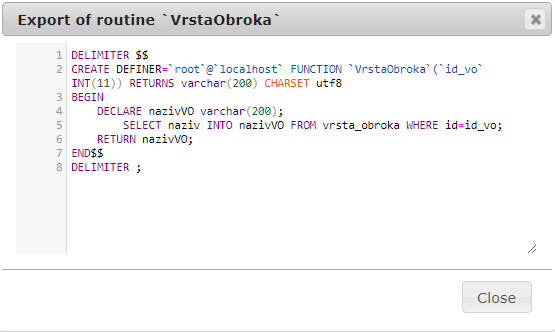
Iz menija u phpMyAdminu biramo “Routines”, zatim “Add routine” gde će nam se pojaviti sledeće:

U projektu smo napravili 6 funkcija.

Izgled kreiranih funkcija u phpMyAdmin-u:

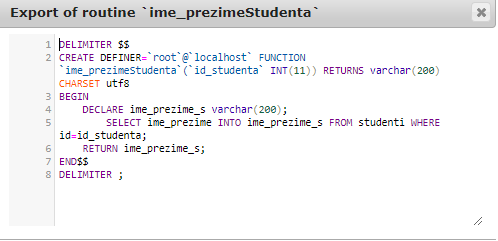
1. Funkcija - **nazivBanke**

Ova funkcija omogućava da se na osnovu uzetog ID-a vraća naziv banke u kojoj student ima račun. Sledi deo koda gde je funkcija pozvana:

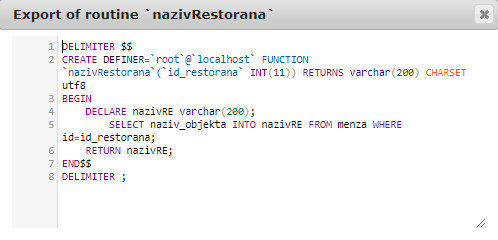
2. Funkcija - **VrstaObroka**

U ovoj funkciji, omogućeno je da se na osnovu uzetog ID-a, vraća obrok koji je student naručio.

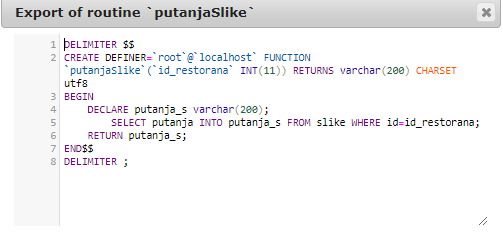
3. Funkcija - **ime\_prezimeStudenta**

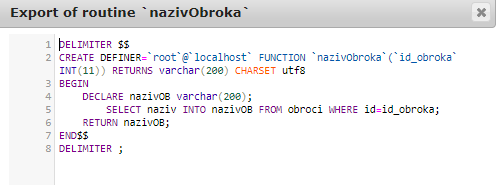


4. Funkcija - **nazivRestorana**



5. Funkcija- **putanjaSlike -** ova funkcija omogućava da se slika određenog restorana poveže sa njegovim ID.



6. Funkcija - nazivObroka 

## 3.3.4 Trigeri

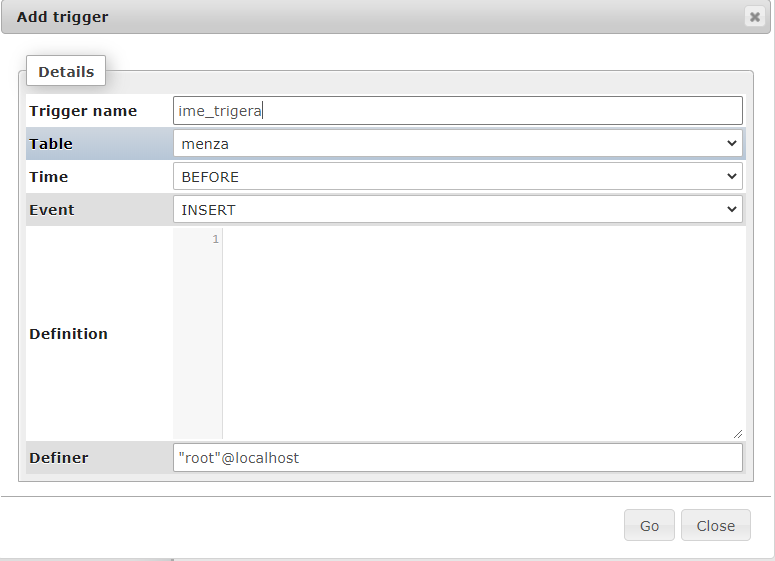
Triger je MySQL objekat koju se pokreće na određenu akciju ili kalkulaciju pre (before) ili posle (after) unosa, brisanja ili ažuriranja iskaza koji se izvršava. Triger može da se pokrene pre nego što se novi zapis unese u tabelu ili pošto se zapis ažurira.

Trigerima je dozvoljeno da menjaju podatke u bazi što obično i rade. Trigeri nikad ne vraćaju vrednost jer oni manipulišu sa podacima onog trenutka kada se desi događaj. Ne postoji komanda kojom se trigger pokreće, već se on pokreće kao reakcija na neki događaj (Update, Insert ili Delete).

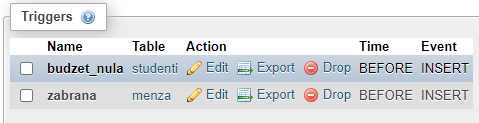
Trigerima možemo da beležimo svaku promenu u tabeli (istorija tabele).

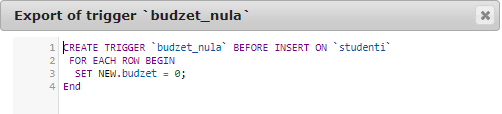
Mogu da se koriste da obezbede integritet podataka, kontrolišu dozvoljene podatke, sprečavaju greške.

Trigeri se kreiraju na sledeći način:

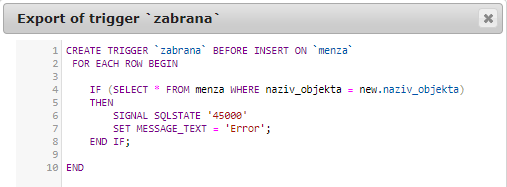
Iz menija u phpMyAdmin-u pristupamo delu pod nazivom “Triggers”, zatim “Add trigger” gde kreiramo novi triger.

Nakon toga nam se prikazuje prozor gde pišemo ime trigera, biramo tabelu iz baze, zatim kada se triger aktivira pre, posle ili umesto određenog događaja u tabeli.

U nastavku su prikazani trigeri koje smo kreirali i iskoristili u projektu:

**1. Triger** **- budžet\_nula** ima sledeću strukturu:

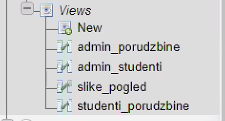
Ovaj triger smo u projektu koristili kako bi prilikom kreiranja novog studenta njegov budžet bio na nuli.

**2. Triger** - **zabrana** ima sledeću strukturu:

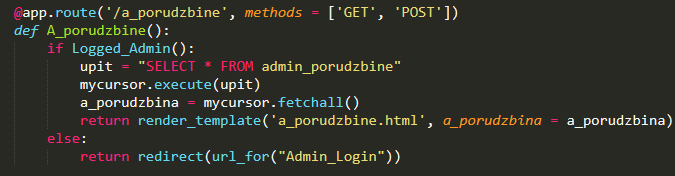
Ovaj triger smo koristili u projektu zbog zabrane kreiranja restorana sa istim nazivima.

## 3.3.5 Pogledi

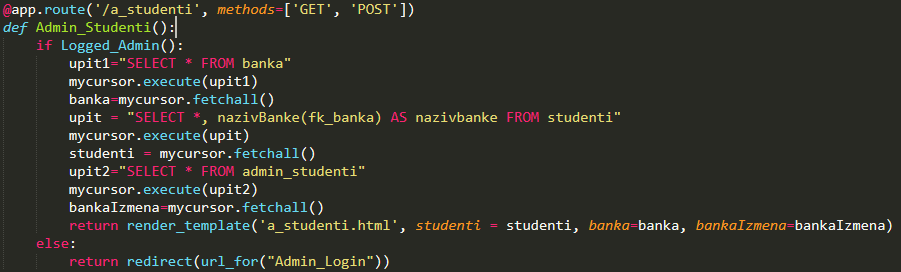
Pogled je logička tabela koja je definisana SELECT upitom sa JOIN klauzulom koja povezuje više fizičkih tabela. Promenom podataka u fizičkoj tabeli menja se i rezultat pogleda.

Kreirali smo 4 pogleda:

**1. Pogled: admin\_porudzbine**

Ovaj pogled se koristi u sledećem delu koda:

**2. Pogled: admin\_studenti**

Ovaj pogled se koristi u sledećem delu koda:

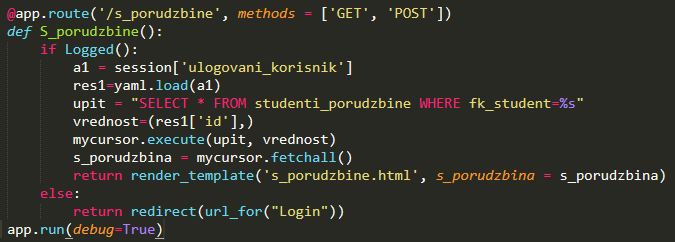
Služi za prikazivanje naziva i adresu banke unošenjem ID-a.

**3. Pogled - slike\_pogled**



Koristimo kako bismo prikazali odgovarajuću sliku.

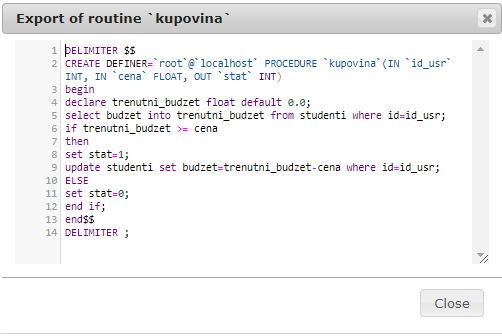
**4. Pogled - student\_porudžbine**



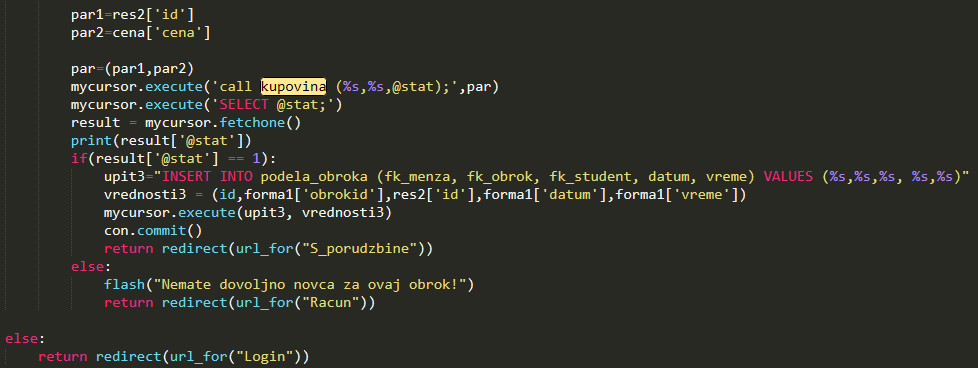
## 3.3.6 Procedure

Procedure obično imaju zadatak da promene vrednost u tabeli ali mogu i da vrate vrednost obično u formi 1 ili 0 i zavisnosti da li je zadatak urađen ili nije. Procedure se pozivaju komandom **exec**. Procedure se kreiraju na sličan način kao i funkcije s tim što se iz padajuće liste bira deo “Procedure”.

Mi smo u našem projektu kreirali procedure pod nazivom “**kupovina**” i ona ima sledeću strukturu:



Način pozivanja procedure u kodu:



Ova procedura je kreirana kako bi izvršili efikasno naručivanje hrane od strane studenta, tj. vrši se provera prilikom naručivanja, da li student ima dovoljno sredstava na računu. Ukoliko nema dovoljno novca, mora uplatiti u banci, dok je u suprotnom narudžbina omogućena.

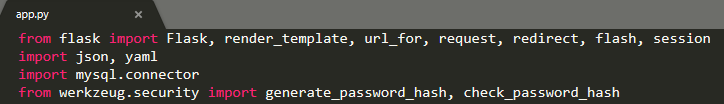
## 3.4 Komunikacija

U ovom delu su prikazane korišćene rute u projektu. Svaka ruta biće ukratko objašnjena.

|  |  |
| --- | --- |
| **Ruta** | **Opis** |
| **http://localhost:5000/login** | Početna strana, prijavljivanje studenta |
| **http://localhost:5000/admin\_login** | Početna strana, prijavljivanje administratora |
| **http://localhost:5000/restorani** | Restorani - Prikaz podataka o restoranima |
| **http://localhost:5000/racun** | Račun - prikaz stanja računa studenta |
| **http://localhost:5000/porudzbina/:id** | Porudzbina - Mogućnost poručivanja |
| **http://localhost:5000/a\_studenti** | Administrator - dodavanje/brisanje/izmena studenata |
| **http://localhost:5000/a\_restorani** | Administrator - dodavanje/brisanje/izmena restorana |
| **http://localhost:5000/a\_obroci** | Administrator - dodavanje/brisanje/izmena obroka |
| **http://localhost:5000/a\_banke** | Administrator - dodavanje/brisanje/izmena banaka |

# 4. Rad aplikacije

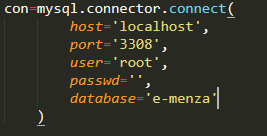
## 4.1 Opis implementacije



Projekat započinjemo u Python fajlu - app.py, gde na početku moramo da importujemo:

* flask,
* render\_template,
* url\_for, request,
* redirect,
* flash,
* session,
* json,
* yaml,
* mysql.connector
* generate\_password\_hash,
* check\_password\_hash

U nastavku imamo sledeći kod koji nam služi za konektovanje sa bazom podataka:



**host** - ime hosta,

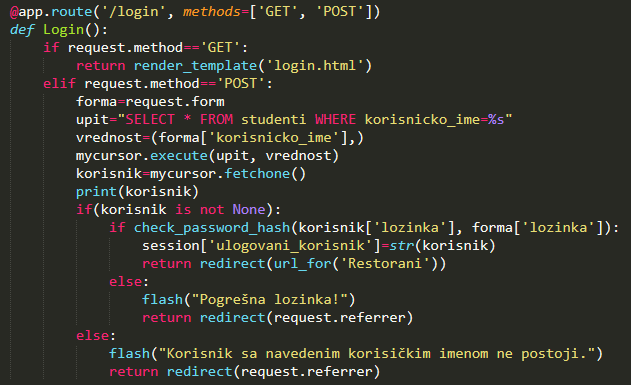
**port** - port u phpMyAdmin-u na kojem sluša MySQL

**user** - username prilikom logovanja na phpMyAdmin

**passwd** - šifra prilikom logovanja na phpMyAdmin

**database** - ime baze

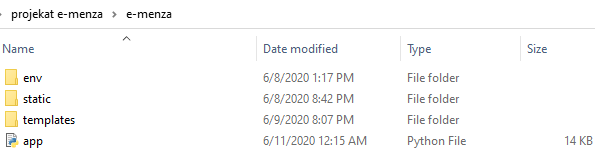
U nastavku **app.py** dela, kreiramo željene rute i definišemo određene fukcionalnosti nad njima.



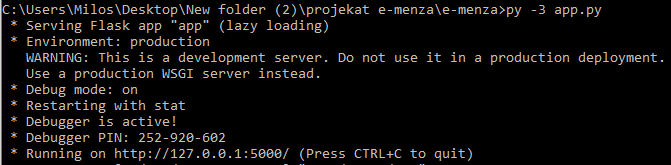
Ovo je primer kreiranja rute http://localhost:5000/**login**

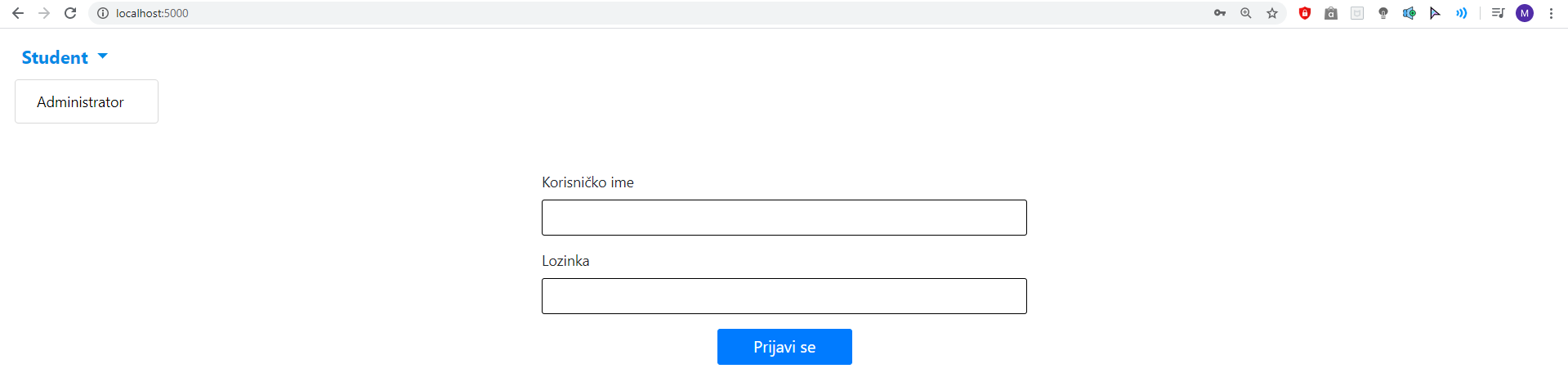
## 4.2 Opis funkcionalnosti - korisničko uputsvo

Projekat se pokreće tako što što najpre moramo upaliti WAMP server koji će nam omogućiti rad sa bazom povezanom sa datim projektom, ući u cmd i doći do lokacije gde se nalazi sledeće:



Zatim u cmd-u upisujemo:



Kada je sve to startovano pokrećemo Google Chrome i kucamo u url-u localhost:5000 jer flask sluša na ovom portu. Ova komanda će nas automatski odvesti na prvu stranicu projekta a to je biranje uloge (slika ispod).

Deo u kojem se prijavljuju studenti.



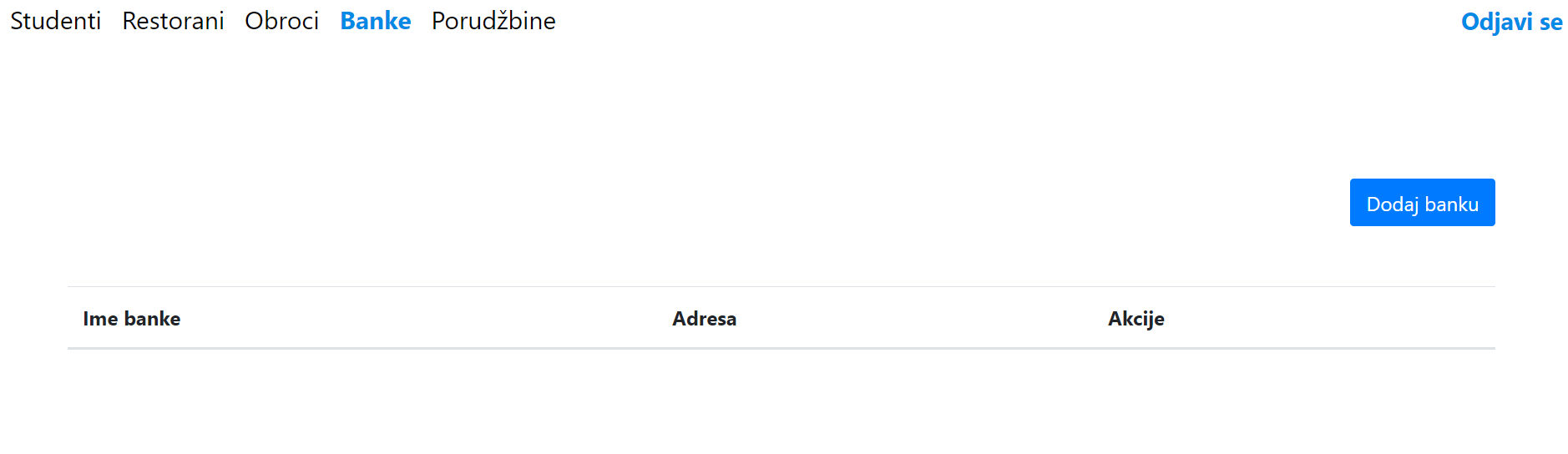
Deo u kojem se prijavljuje administrator.

Prijavom na sistem sa ulogom administratora pojavljuje se sledeća stranica:

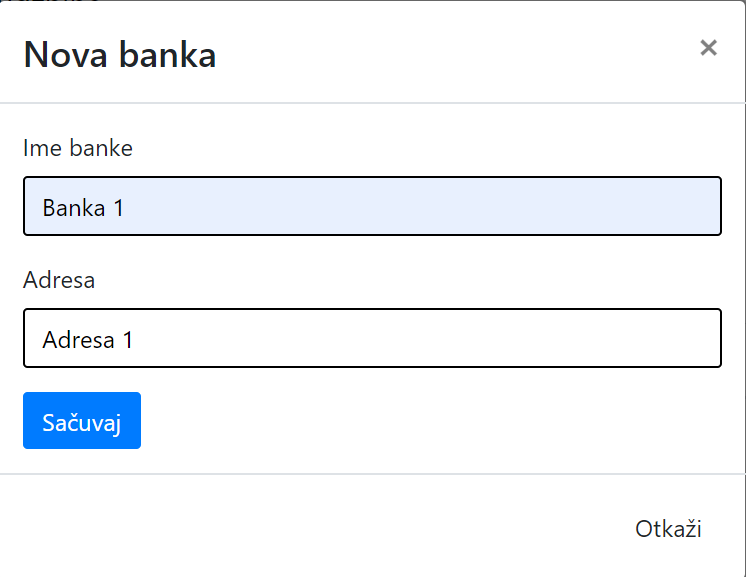


Kao što vidimo, na vrhu su prikazane stranice na kojim administrator može da pristupi, a svaka od njih će biti posebno objašnjena u nastavku.

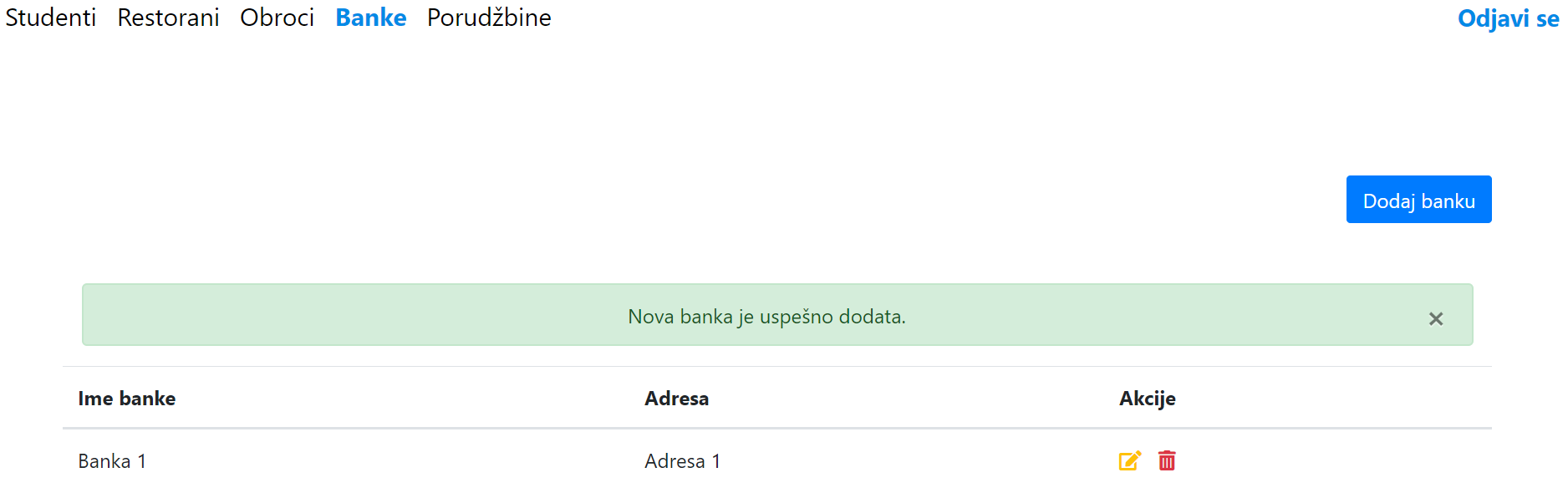
Da bismo kreirali novog studenta, moramo najpre kreirati banku u kojoj student mora imati račun u kojoj će uplaćivati novac.



Klikom na dodaj banku otvara nam se sledeći prozor u kojem možemo dodati novu banku, tako što unosimo Ime i Adresu banke.

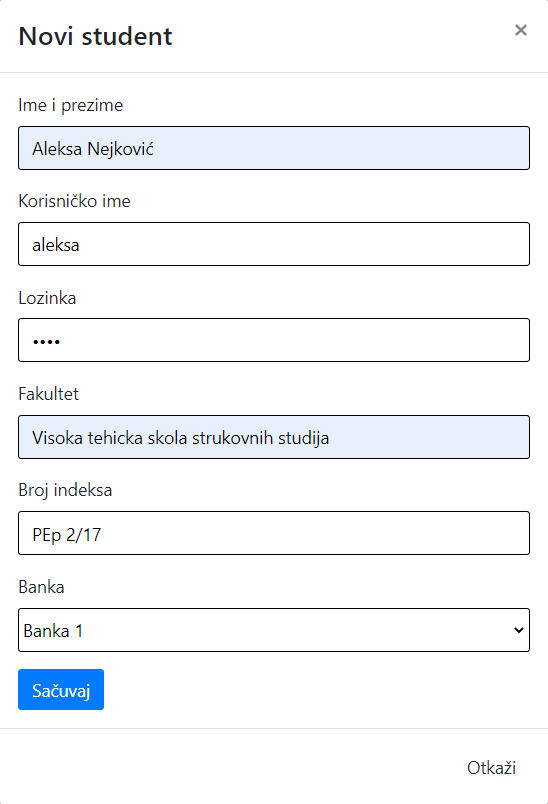


Nakon što smo uneli potrebne podatke, klikom na dugme “**Sačuvaj**” banka je dodata na listi:

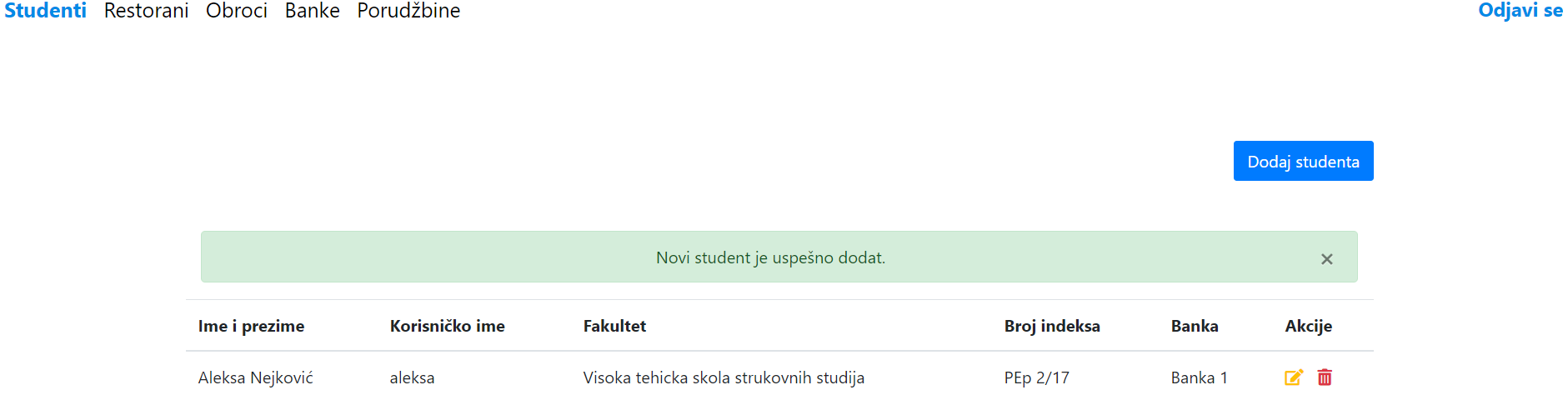


**Student**

Sada smo u mogućnosti da kreiramo novog studenta:

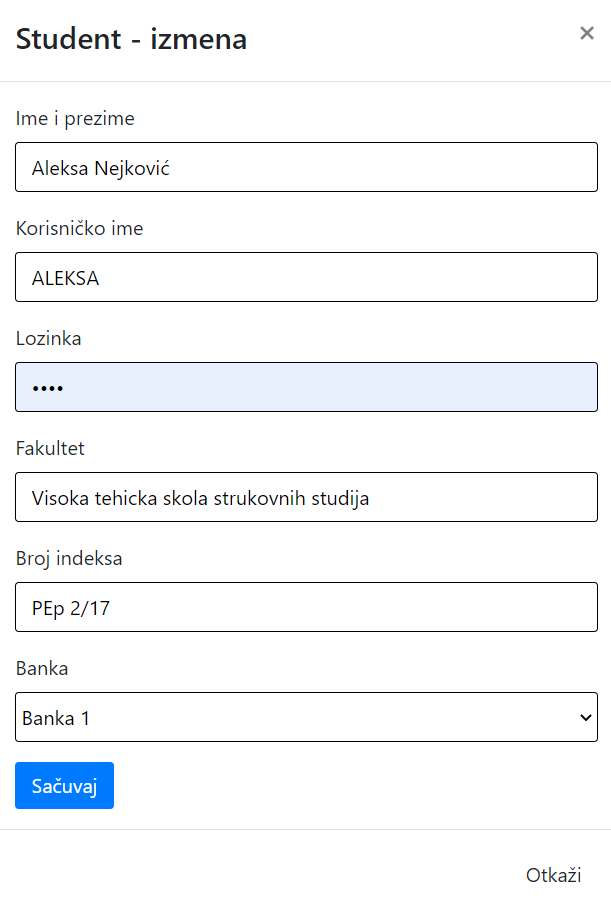


Klikom na dugme “**Sačuvaj**”, student je dodat u sledećem delu:

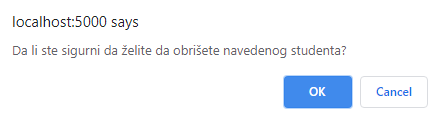


Kreiranjem novog studenta od strane administratora, ima mogućnost prijavljivanja na početnoj strani u delu prijavljivanja za studenta.

U delu “**Akcije**” možemo izmeniti informacije o studentu ili ga obrisati tako što kliknemo na odgovarajuću ikonicu.

**Izmena studenta**:

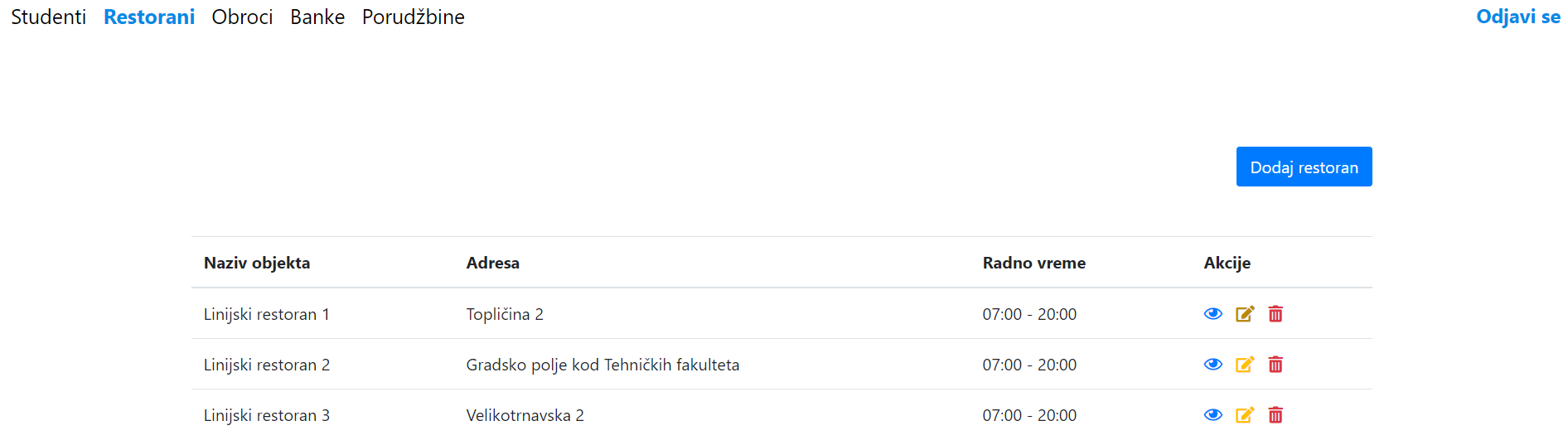
**Brisanje studenta:**



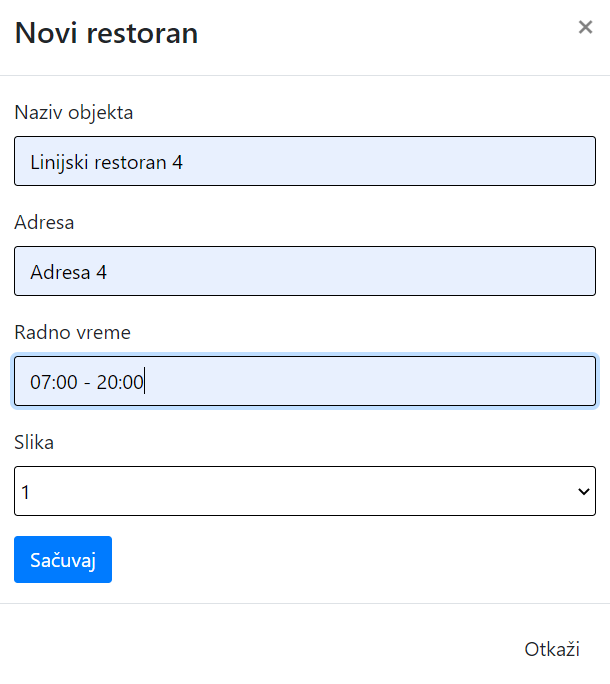
Klikom na dugme “**OK”** brišemo studenta iz baze.

**Restorani**

Ovde administrator može manipulisati restoranima (dodavati, menjati, brisati).



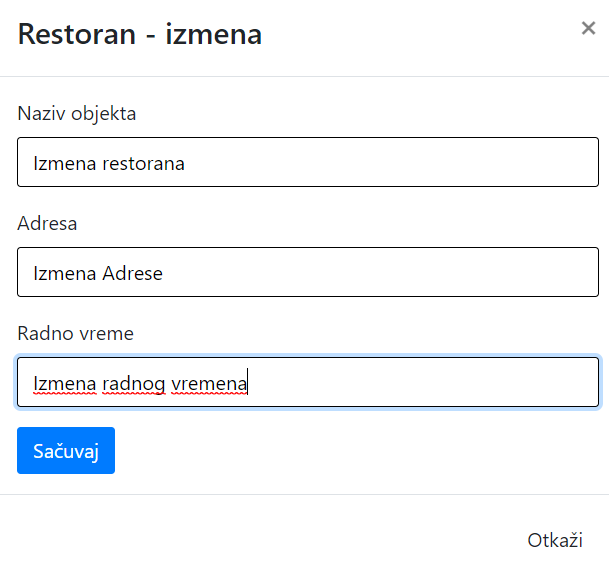
Prilikom klika na “**Dodaj restoran**” otvara nam se sledeći prozor:



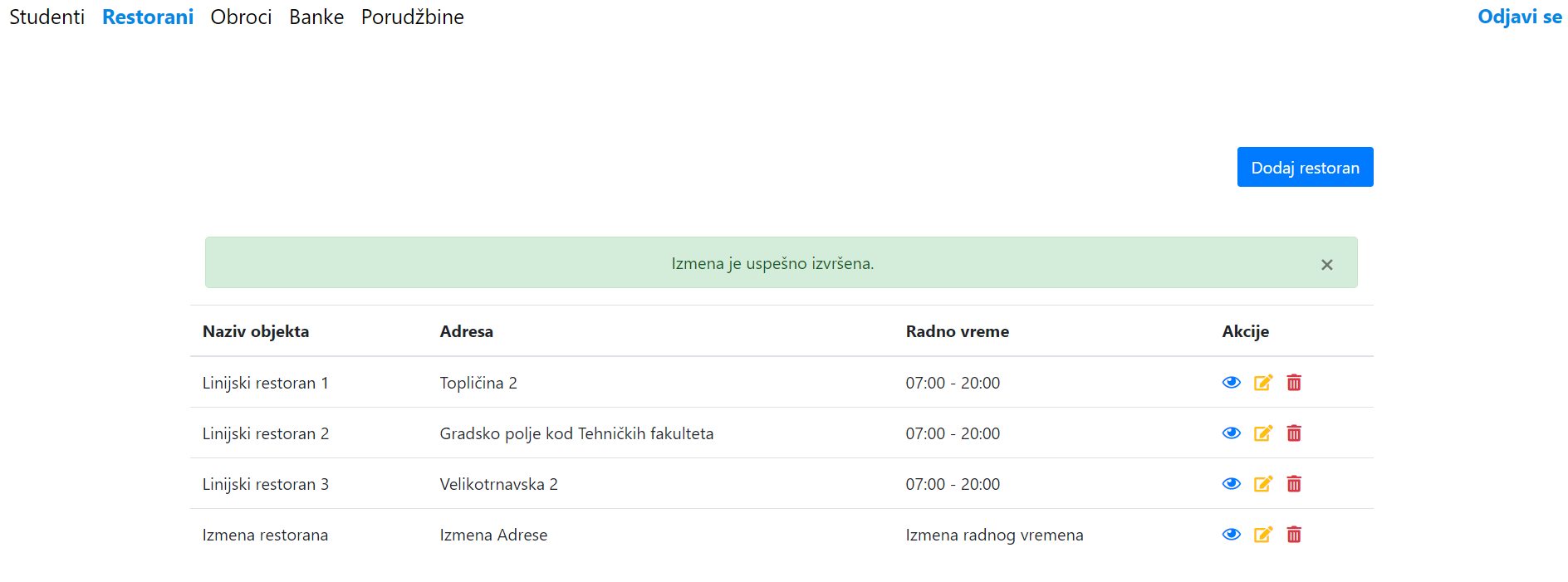
Administrator dodaje novi restoran tako što upisuje naziv objekta, adresu, radno vreme i bira odgovarajući sliku za dati restoran.

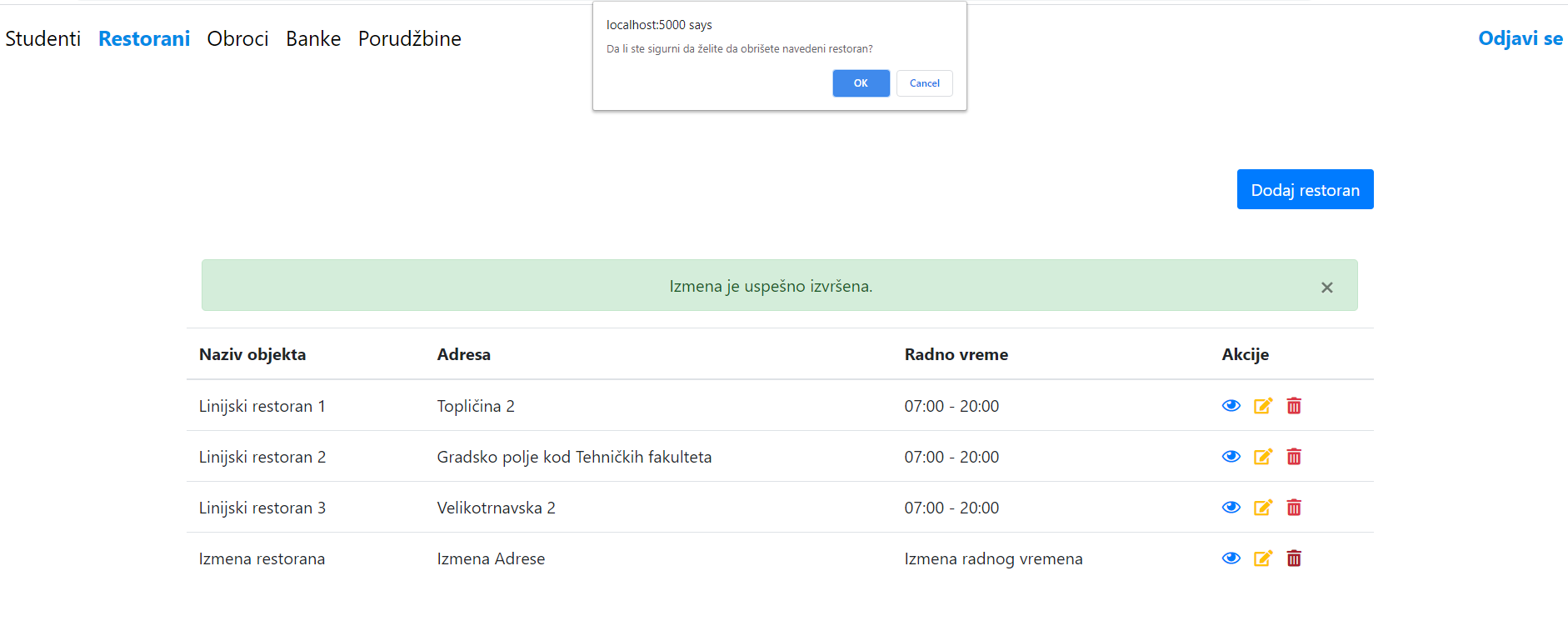
Novi restoran je sada dodat i u delu “**Akcije**” možemo pogledati izgled restorana, izvršiti izmene prilikom unošenja informacija kod dodavanja ili obrisati restoran.

**Izmena:**



Klikom na dugme “**Sačuvaj**” izmene će biti sačuvane. U nastavku je prikazana izmena:



Ukoliko želimo da izbrišemo određeni restoran, klikom na kantu u delu “**Akcije**”, restoran će biti obrisan.

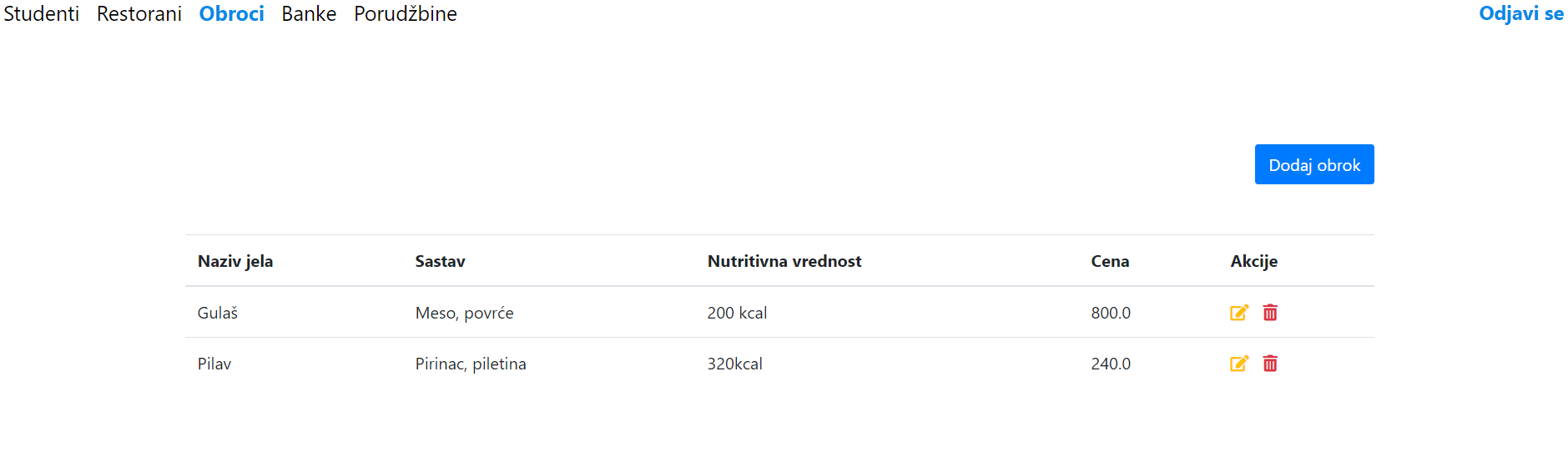
Vidimo iskačući prozor sa porukom da će navedeni restoran biti obrisan ako kliknemo **OK**.

Trenutno stanje restorana, nakon što smo 1 obrisali.

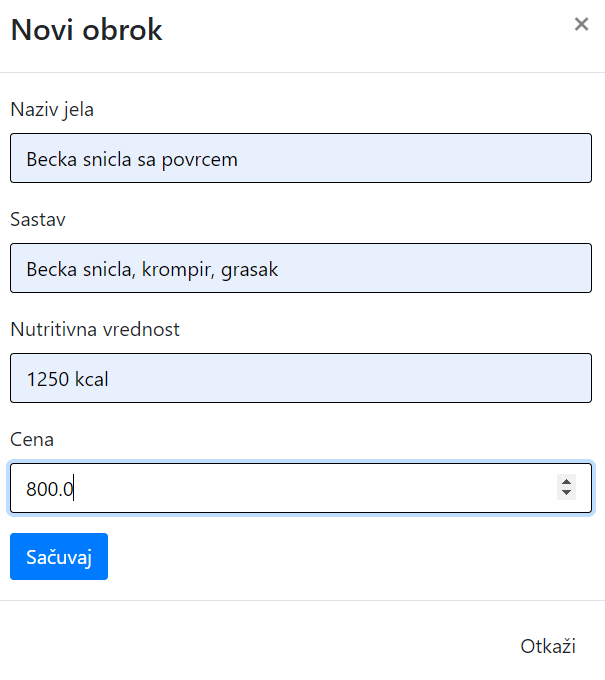


**Obroci**

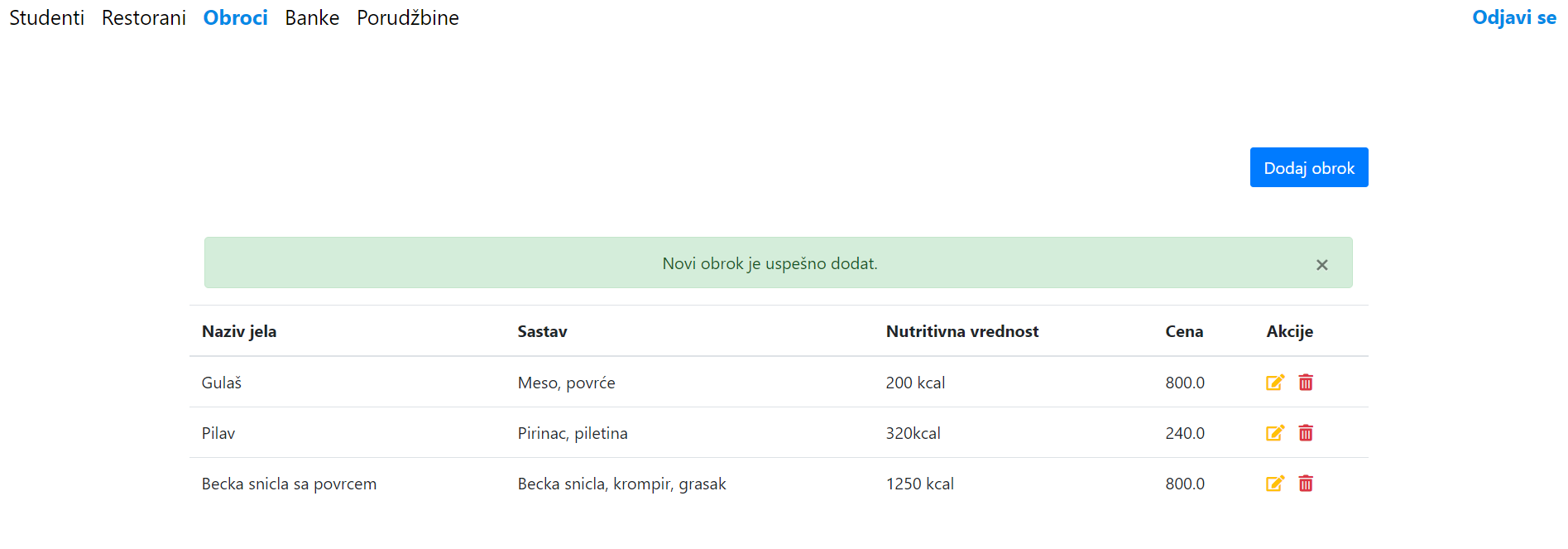
Sledeća stavka u administratorskom delu su “**Obroci**”.



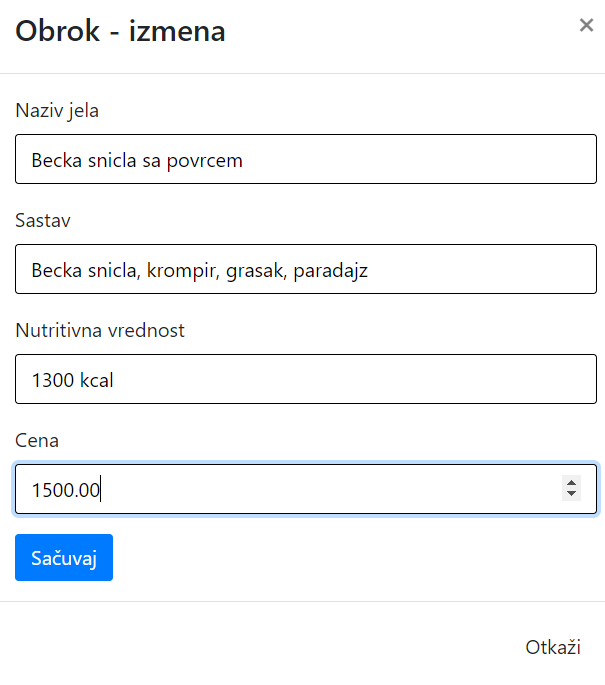
Ovaj deo služi za dodavanje jela koja su dostupna u restoranima, kao i dodatne informacije o njima. Administrator ima mogućnost da doda novi obrok klikom na dugme “Dodaj obrok”:



Nakon što smo kreirali novo jelo, ono se nalazi u listi dostupnih jela za naručivanje.

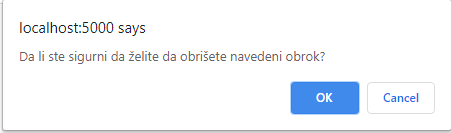


Kao i u prethodnim delovima, administrator ima mogućnost da obroke koje je dodao izmeni, ili obriše.

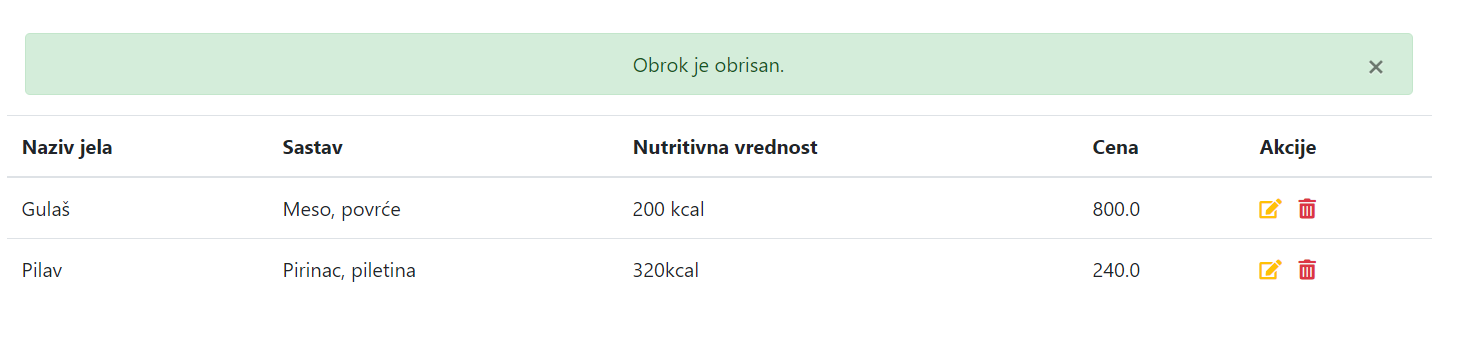
**Izmena:**

****

**Brisanje:** Klikom na drugu ikonicu u delu akcije, brišemo jelo iz baze:

****

Izgled dela “**Obroci**” nakon brisanja željenog jela.

****

**Porudžbine**

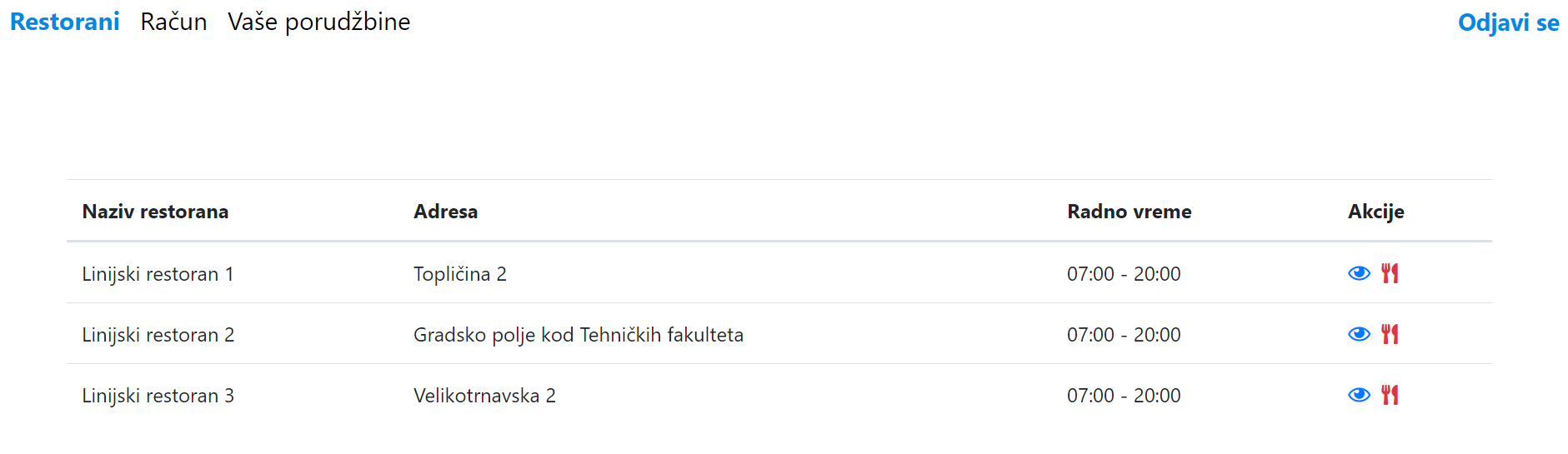
U ovom delu se podaci prikazuju nakon što student poruči željeno jelo, tako da ćemo se sada odjaviti na dugme “**Odjavi se”.**

**Student - početna strana, prijavljivanje**

Upisujemo korisničko ime i lozinku koje smo upisali u administratorskom delu prilikom dodavanja novog studenta.

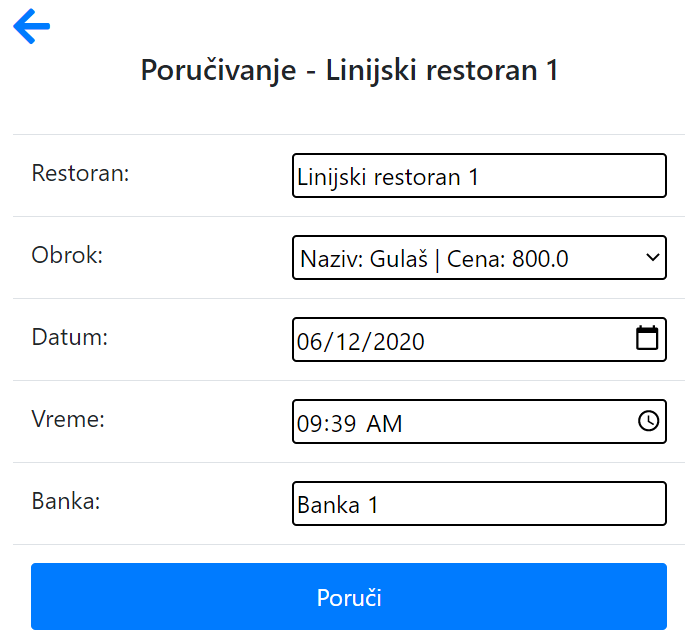
****

Na početnoj strani imamo sledeće:



Deo “**Restorani”** gde imamo prikaz restorana sa informacijama koje smo dodali u administratorskom delu gde student ima mogućnost izvrši poručivanje hrane na sledeći način:

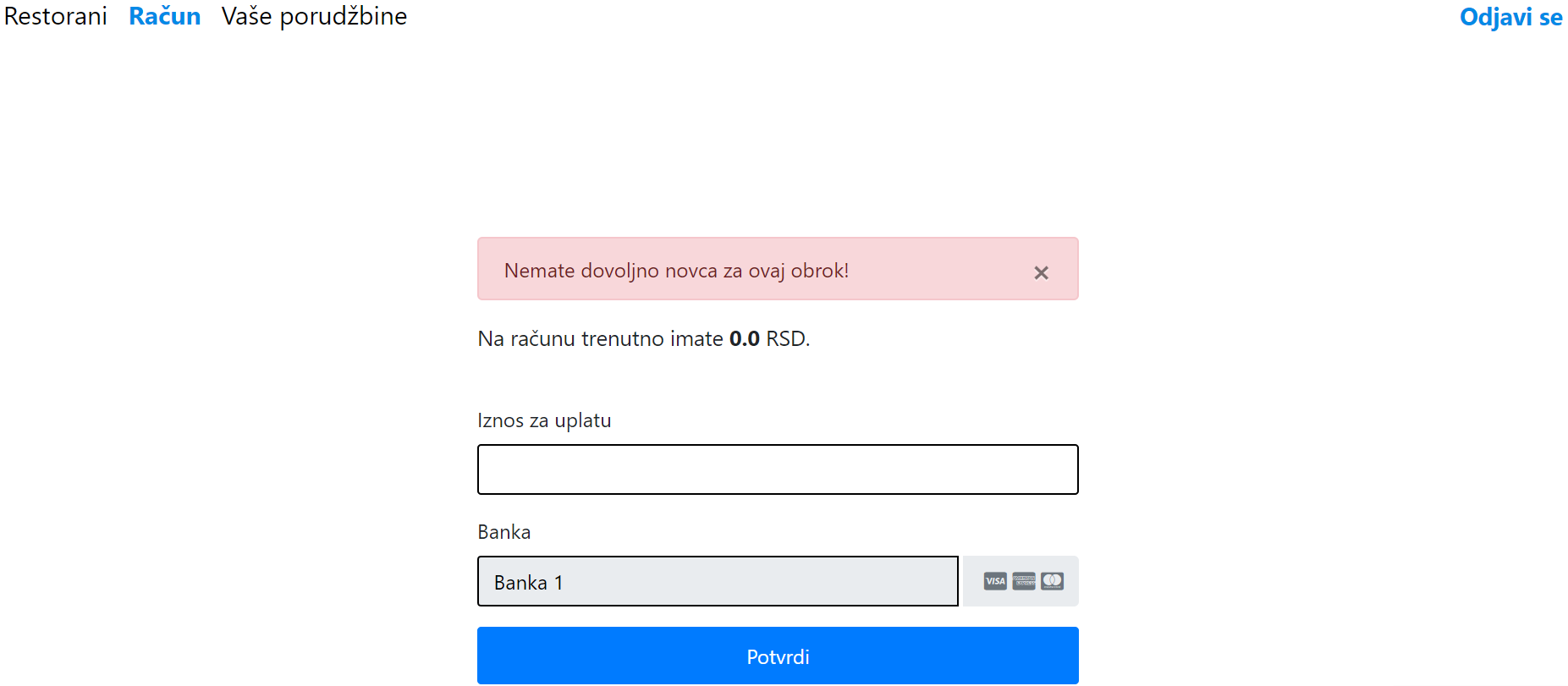
Klikom na drugu ikonicu u delu “**Akcije**” student najpre bira u kom će restoranu (menzi) poručiti željenu hranu.



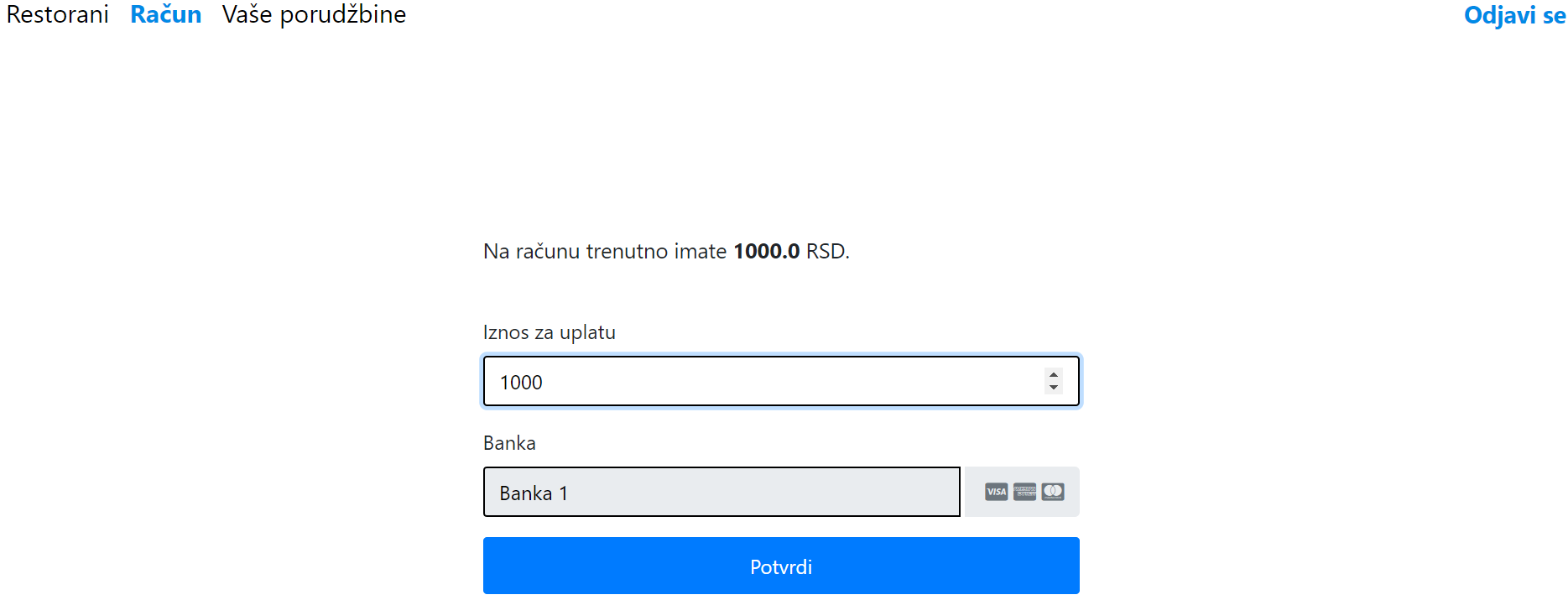
U ponudi su nekoliko jela koja su dodata u administratorskom delu i njihove cene, zatim datum i vreme kada student želi naručiti hranu i banka koja pripada studentu.

Klikom na dugme poruči, postoje dve mogućnosti:

* Prva je, ukoliko student nema dovoljno novca na računu, najpre mora uplatiti kredit.

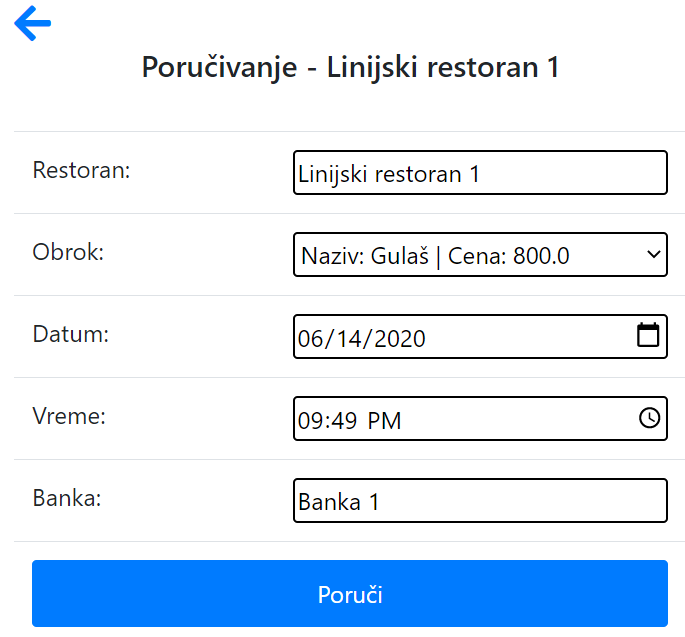


Student unosi željenu svotu novca kako bi mogao obaviti poručivanje.

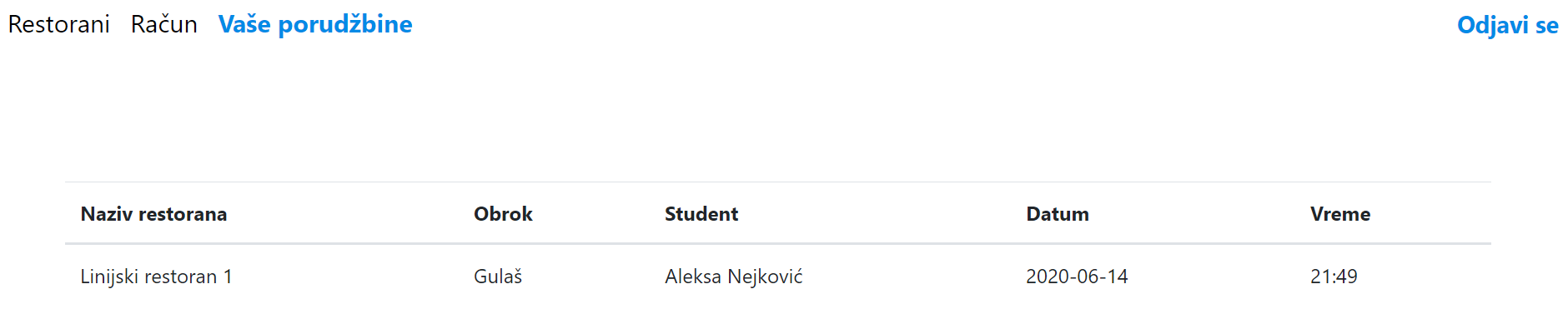


Student je uplatio 1000 dinara, gde se jasno može videti nakon svake uplate stanje na računu.

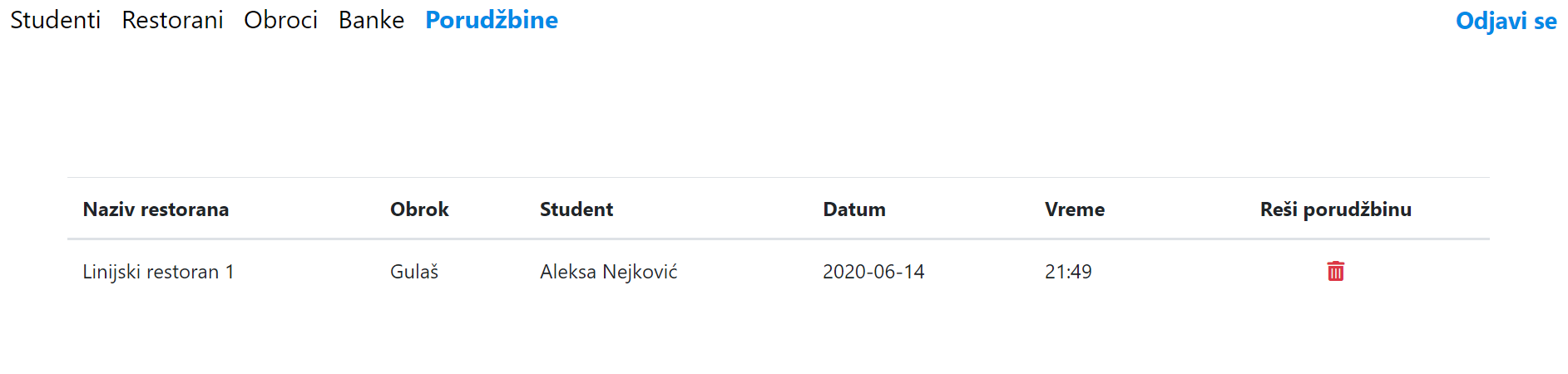
S obzirom da je student sada u mogućnosti da poruči željenu hranu, dolazimo do drugog dela:



Vraćamo se u delu “Poručivanje” gde sada student ima dovoljno sredstva na računu za poručivanje hrane.

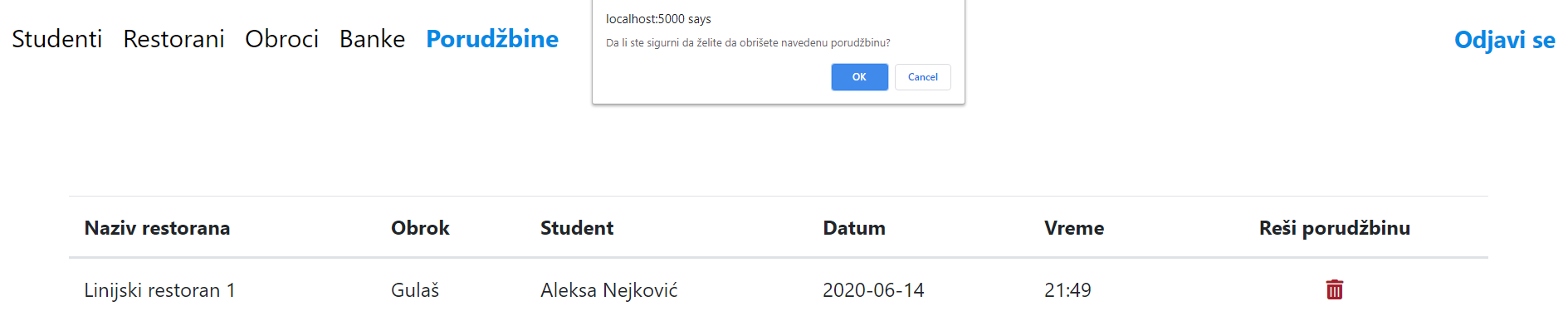


Klikom na “**Poruči”** dolazimo do dela pod nazivom “**Vaše porudžbine”,** gde se čuvaju informacije o poručivanju koje smo uneli u prethodnim delovima.

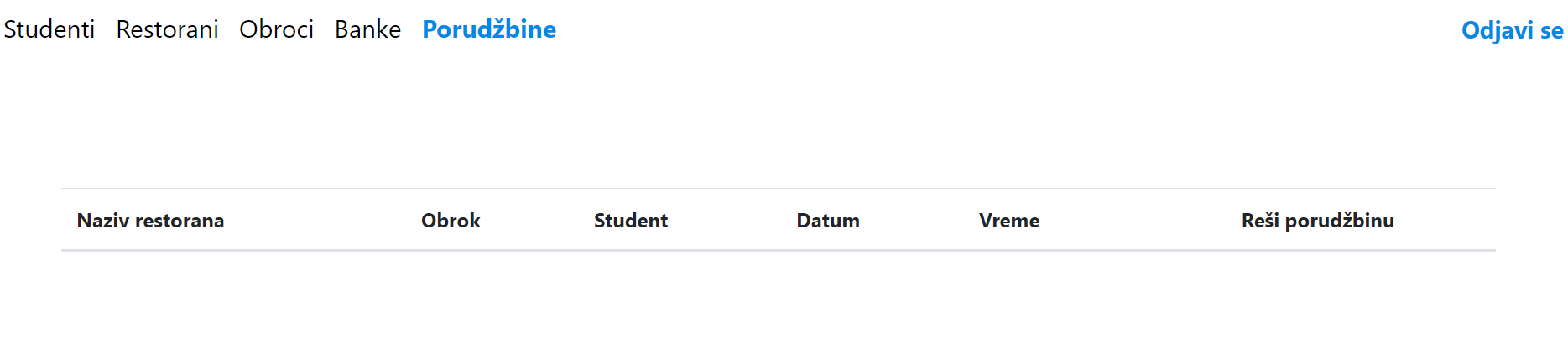
Sada ćemo se vratiti u “**administratorskom”** delu da prikažemo poslednju stavku u ovom delu:

U delu porudžbine se takođe pamte podaci o naručivanjima ail i mogućnošću brisanja porudžbine od strane administratora.

Klikom na ikonicu ispod dela “**Reši porudžbinu”** se omogućuje ova funkcija:



Klikom na dugme “**OK”** brišemo ovu narudžbinu iz baze, gde nakon promene ona izlgeda ovako:



# 5. Literatura

1. Dušan Stefanović, Administriranje baze podataka - predavanja i vežbe
2. <https://dev.mysql.com/doc/>
3. <https://dev.mysql.com/doc/refman/8.0/en/trigger-syntax.html>
4. <https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html>
5. <https://dev.mysql.com/doc/refman/8.0/en/functions.html>
6. <https://www.mysqltutorial.org/mysql-views-tutorial.aspx/>
7. <https://www.mysqltutorial.org/>