

Projekt pri predmetu Praktikum I:

Moj Obrtnik

Avtorji: Tomaž Zajc, Marko Zmazek, Jure Turk, Luka Gričar, Dimitar Micevski

Mentor: Mitja Gradišnik

Kazalo

1. Uvod

# Besedilo naloge

**Moj obrtnik**

Načrtujte in izdelajte sistem, ki bo uporabnikov v pomoč pri iskanju obrtnikov ter pošiljanja poizvedb za želena obrtniška dela. V okviru sistema vzpostavite spletno stran, ki bo obiskovalcem omogočala pregled obrtnikov izbrane gospodarske kategorije ter storitev, ki jih ti obrtniki ponujajo. Vsakemu izmed obrtnikov, ki sodelujejo v spletni skupnosti omogočite, da na portalu strukturirano predstavijo sebe ter storitev, ki jih ponujajo (podprite samo obrtnike, ki so storitveno usmerjeni). Ker je vtis predhodnih odjemalcev storitev obrtnika ključen za proces odločanja o najemu obrtnika, odjemalcem portala omogočite komentiranje storitev obrtnikov. Da bi lahko uporabniki portala lažje našli ustreznega obrtnika, implementirajte tudi napredno iskanje po razpoložljivih obrtnikih. Uporabnik naj ima možnost povpraševanja pri izbranem obrtniku.

Mentor: Mitja Gradišnik

# Specifikacije in zahteve

* Iskanje obrtnikov
* Pošiljanje poizvedb (naročil)
* Pregled storitev glede na izbrane kriterije
* Predstavitev Obrtnika
* Komentiranje storitev
* Napredno iskanje
* Možnost povpraševanja pri izbranem obrtniku

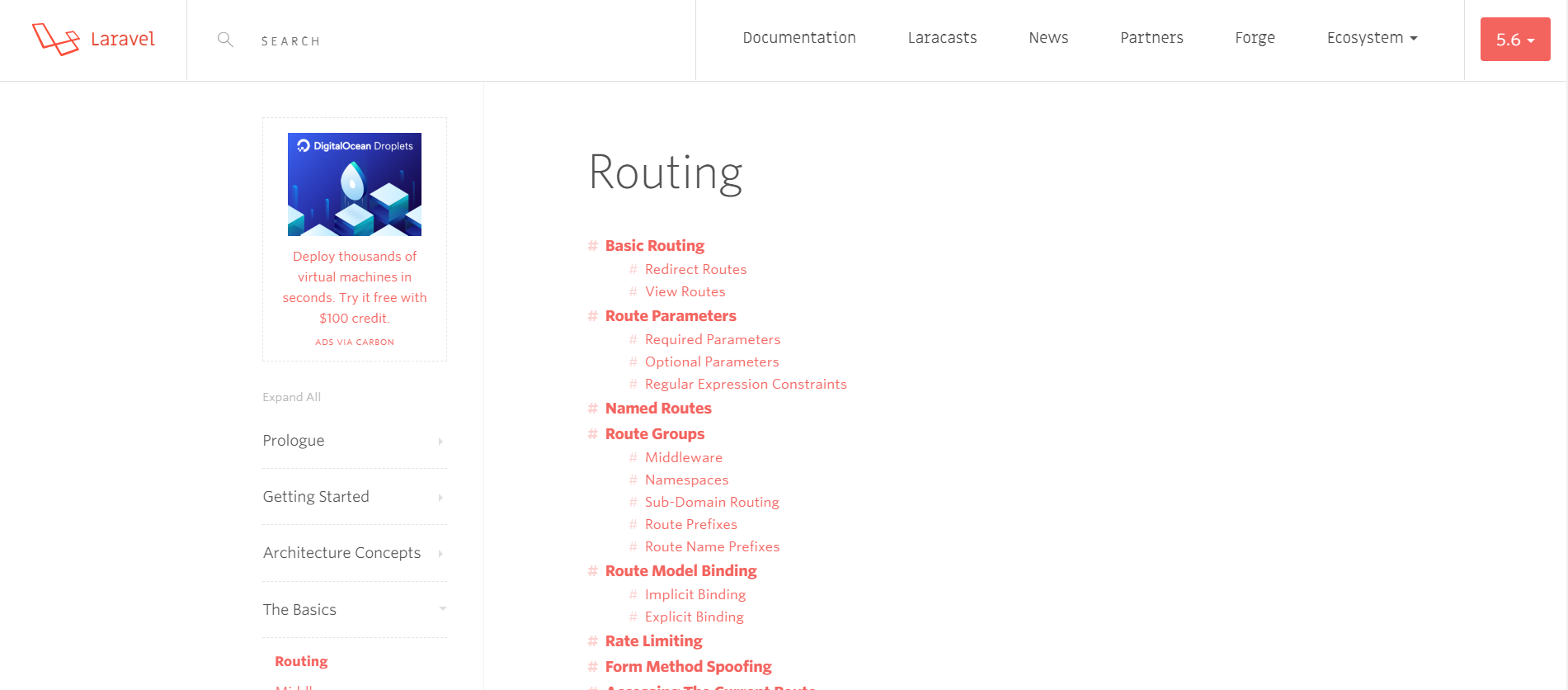
# Izbrano orodje in jeziki

Kot Izbrano orodje oziroma ti. »framework«, smo uporabili Laravel. To orodje se nam je zdelo primerno, saj smo kot programski jezik izbrali PHP. Nobeden izmed članov skupine ni še pred začetkom predmeta Praktikum uporabljal tega orodja, zato smo imeli na začetku nekaj težav z orodjem, saj ga še nismo znali uporabljat. Laravel nam je bil predlagan s strani asistenta pri predmetu Osnove spletnih tehnologij, saj smo trije člani naše skupine pri predmetu Praktikum za rešitev nalog pri predmetu Osnove spletnih tehnologij uporabljali programski jezik PHP. Kot Integrirano razvojno okolje pa smo poenoteno uporabiljali Microsoft visual studio code. Za ustvarjanje spletnih strani smo uporabli označevalni jezik HTML, za stilsko oblikovanje in razne efekte pa stilski jezik CSS.

1. Uporabljena orodja in rešitev problema

# Laravel

Laravel je eno izmed odprto kodnih orodij za izdelavo interaktivnih spletnih strani. Ustvarjeno je bilo specifično za programski jezih PHP. Tako kot springboot, je tudi laravel zasnovan na MVC arhitekturi. Ko smo se odločali za orodje, ki ga bomo uporabili nas je laravel prepričal s svojo preprostostjo in učinkovitostjo. Samo orodje ima zelo obsežno vendar prepsrosto dokumentacijo, ki nam je omogočala lažje razumevanje orodja in vseh funkcij ki jih orodje omogoča.



*(slika 1: Primer dokumentacije)*

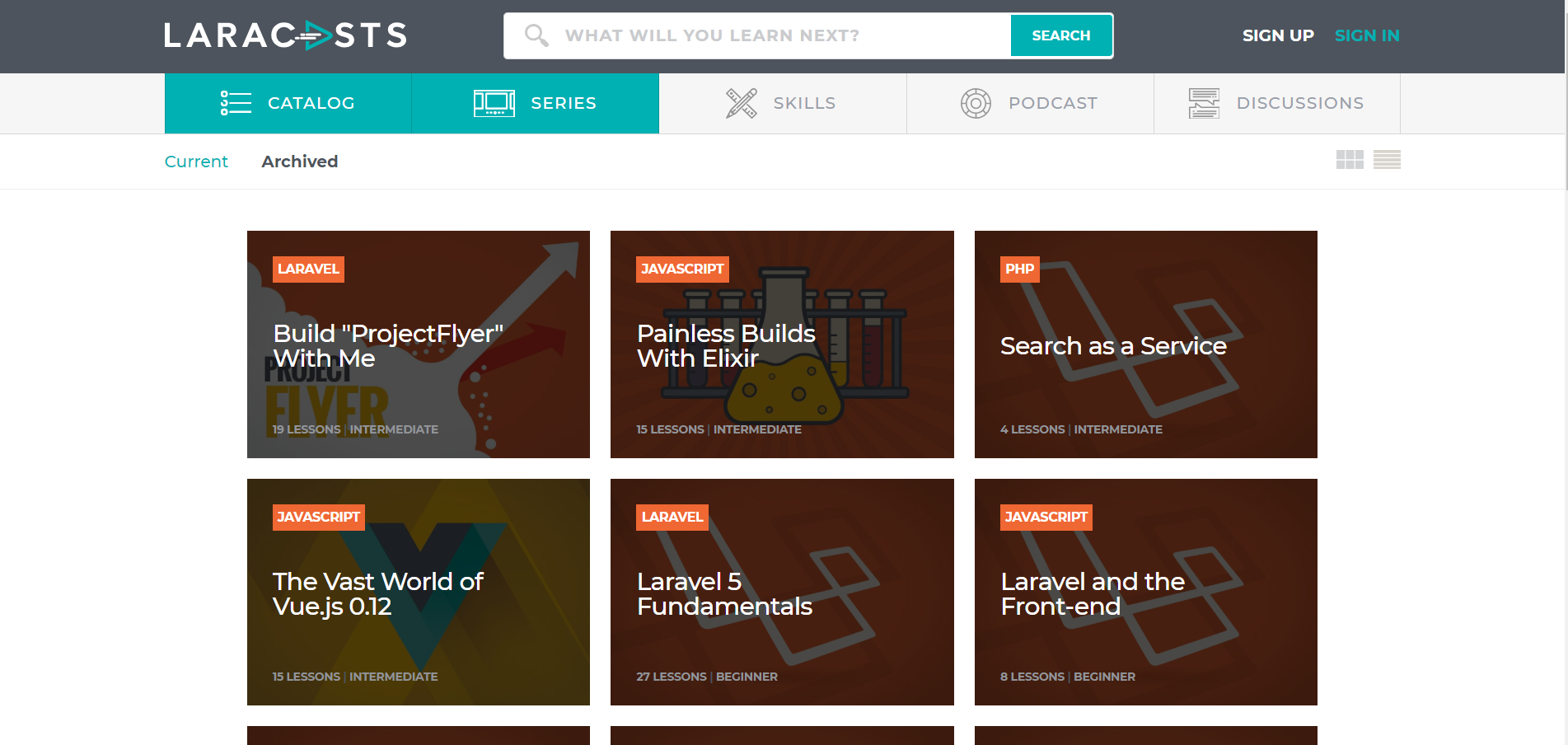
# Artisan

Artisan je implementiran vmesnik orodja laravel, zagotavlja nam veliko število ukazov, ki nam olajšajo pri razvoju aplikacije. Interakcija z artisanom poteka preko ukaznega poziva, omogoča pa tudi ustvarjanje novih ukazov, ki jih definiramo sami. Definitivno je artisan učinkovito in zelo uporabno orodje, ki nam je pri našem projektu bilo v veliko pomoč. Nekaj najpogostejših ukazov ki smo jih uporabljali pri razvoju naše aplikacije:

* Php artisan migrate (migracija podatkovne baze)
* Php artisan serve (Zažene aplikacijo na strežniku)
* Php artisan make(generira izbrane elemente npr: kontroler, blade…)

# Laracasts

K našemu spoznavanju orodja laravel je zraven dokumentacije veliko pripomogla tudi spletna stran laracasts. Laracasts je spletna stran namenjena vsem, ki se želijo seznaniti z orodjem laravel. Na njej najdemo veliko video vsebin, ki na kratek in jedrnat način predstavijo uporabo orodja laravel in njegovih funkcionalnosti. Razen video posnetkov, ki so namenjeni za orodje laravel lahko najdemo tudi ostale video vsebine, ki so povezane z orodjem laravel (PHP, javascript, testiranje, razvojna okolja…).



*(slika 2: Laracasts)*

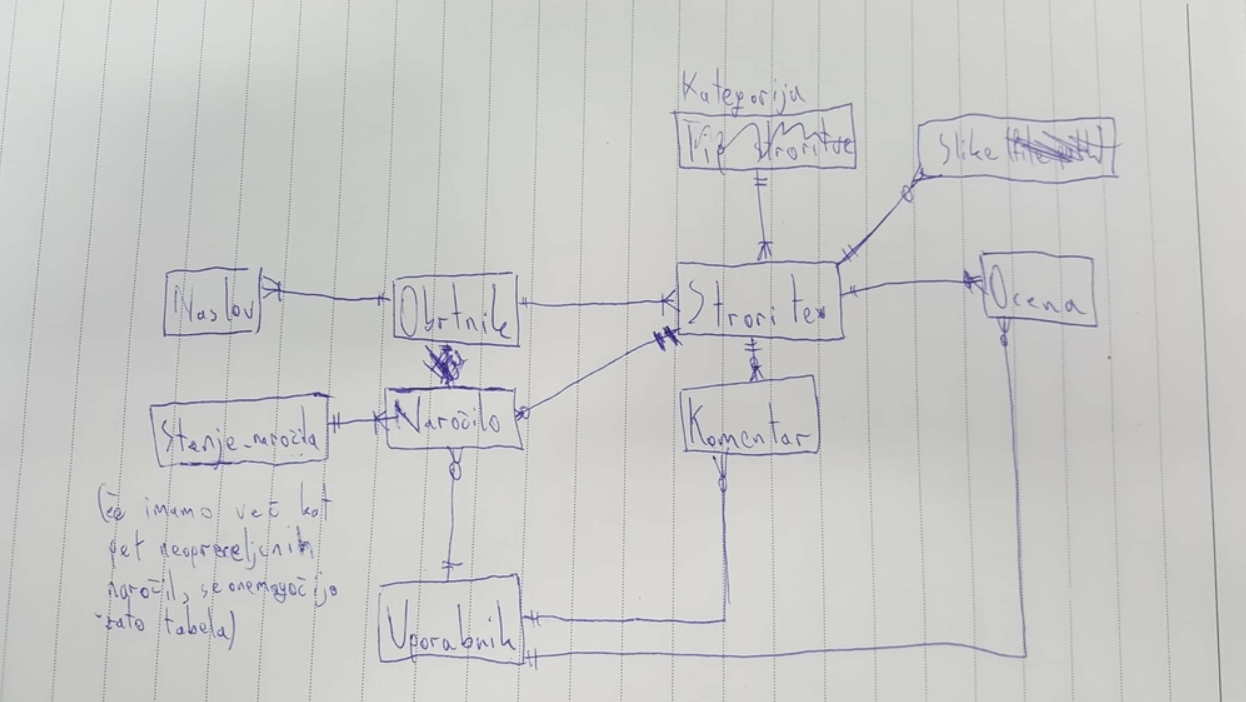
1. Podatkovna baza

Podatkovna baza se je postopoma spreminjala in dopolnjevala. Na začetku projekta smo po ustvarjenem ER modelu ustvarili skripto za ustvarjanje in povezovanje tabel, vendar smo kasneje naleteli na težave pri implementaciji podatkovne baze v naš projekt. Ker nismo našli rešitve, smo namesto vključitev podatkovne baze, generirali podatkovno bazo oziroma je to za nas naredil artisan. Z ukazom »php artisan migrate«, je orodje artisan za nas generiralo podatkovno bazo na podlagi podatkov, ki smo jih vnesli v migracije. Celoten proces [migracije](#migracije) je razložen in predstavljen v poglavju z programsko kodo.

# Načrtovanje podatkovne baze

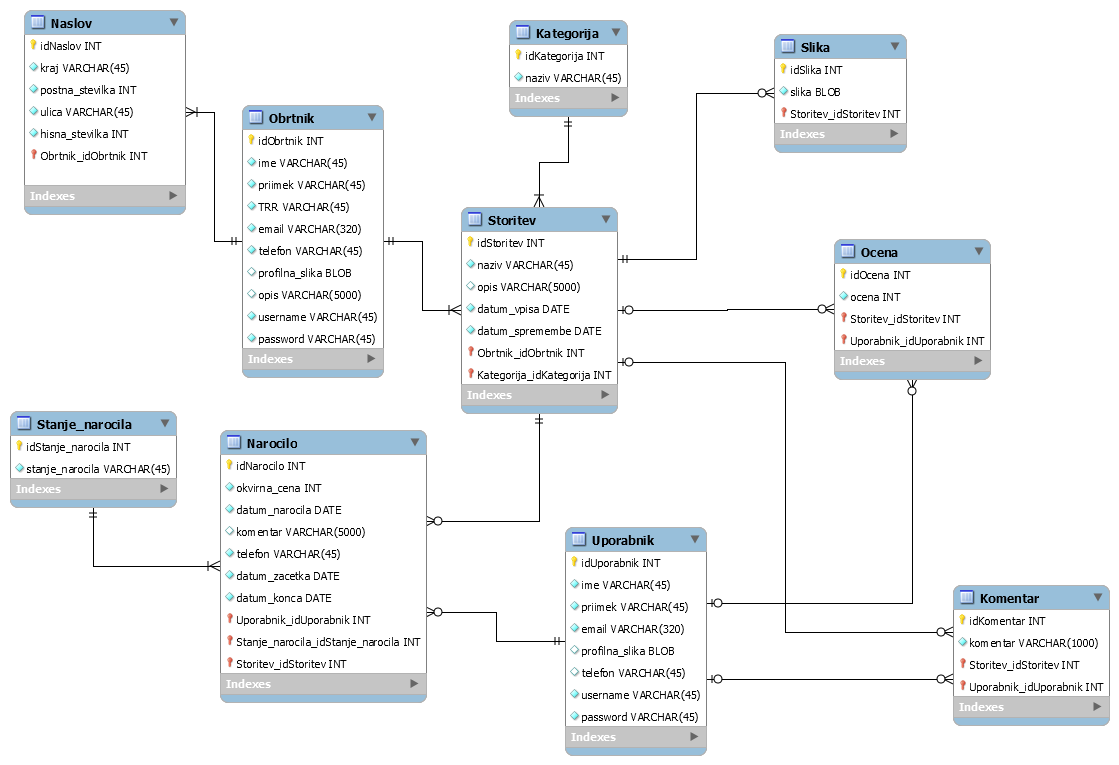
Razen majn pomembnih dodajanj in odstranjevanj atributov lahko razvoj naše podatkovne baze prikažemo v treh fazah. Pri vsaki izmed faz se je naša podatkovna baza spreminjala glede na potrebe in funkcionalnosti naše aplikacije.

**Faza 1:** Prvi korak pri načrtovanju podatkovne baze, je bila določitev entitet, njihovo poimenovanje in relacije med njimi. Tako smo na podlagi tega izdelali osnutek, ki nam je bil v pomoč pri nadaljnem modeliranju in razvoju podatkovne baze.



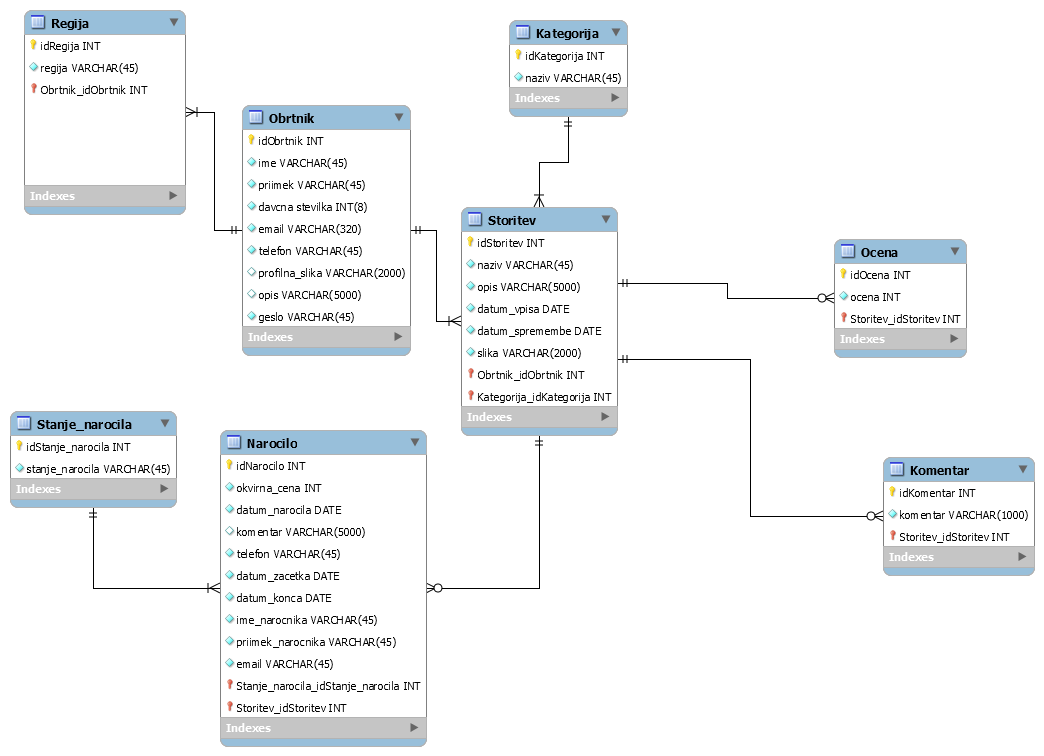
*(slika 3: Osnutek)*

**Faza 2:**  V drugi fazi načrtovanja smo na podlagi osnutka izdelali ER model, dodali atribute in dopolnili model.



*(slika 4: Prvoten ER model podatkovne baze)*

**Faza 3:**  V zadnji fazi načrtovanja smo iz naše podatkovne baze izločili entiteto uporabnika. Zaradi primanjkovanja časa in zahtevnosti projekta smo bili primorani odstraniti ta tip uporabnika. Vse entitete ki so bile povezane z uporabnikom smo zardi tega morali dopolniti z ustreznimi atributi, s katerimi smo lahko delno kompenzirali izločitev uporabnika.

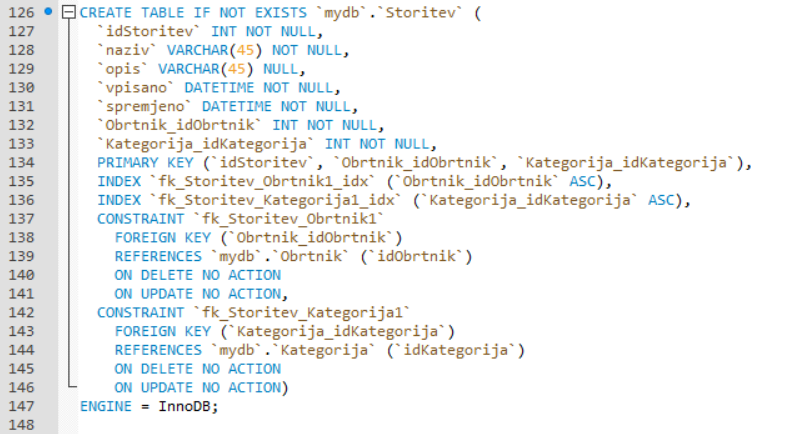
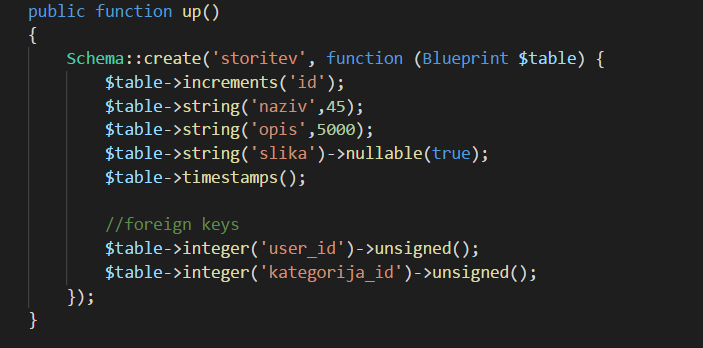


*(slika 5: Končen ER model naše podatkovne baze)*

# 3.2 Ustvarjanje podatkovne baze

Zaradi zapletov z implementacijo naše že obstoječe podatkovne baze, ki smo jo ustvarili z jezikom SQL , smo našo podatkovno bazo na nov način ustvarili z migracijami. Postopek migracij nam ni povzročal težav, saj smo s pomočjo pregledne dokumentacije in spletne strani laracasts hitro osvojili zahtevano znanje.

Primerjava obeh načinov:

*(slika 6: Ustvarjanje tabel v orodju MySQL) (slika 7: Ustvarjanje tabel z migracijami)*

Po postopku predefiniranja migracij smo z orodjem artisan ustvarili našo podatkovno bazo. Testne podatke in podatke potrebne za delovanje aplikacije smo nato v orodju phpMyAdmin z skripto vstavili v podatkovno bazo. V podatkovno bazo smo vključili tudi funkcijo ''on delete cascade'', ki ob izbrisu starševske tabele posredno izbriše tudi ostale tabele povezane z njo. Naša podatkovna baza je tako bila končana in je nemoteno delovala.



(slika 7: Skripta za vstavljanje podatkov v podatkovno bazo)

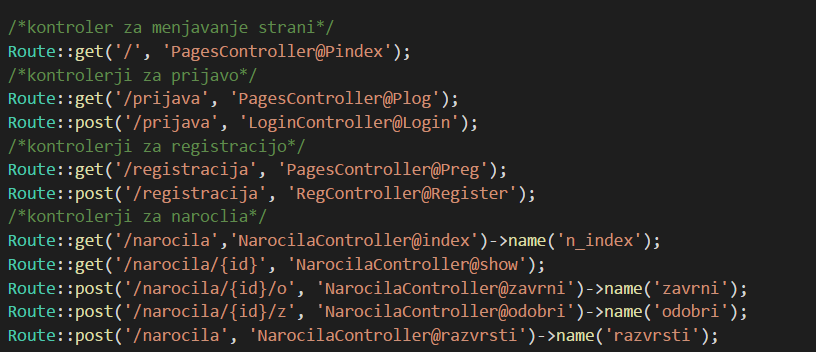
1. Uporabniški vmesnik
2. Opis komponent

V tem poglavju, so predstavljene glavne komponente naše aplikacije, le te so podrobneje predstavljene in obrazložene z izseki programske kode. Predstavljene komponente:

* Poti
* Kontorolerji
* Migracije
* Blade
* Mail

# Poti

Poti oziroma ''Routes'', so pomemben in potreben element kode saj se uporabljajo za koordinacijo med kontrolerji in usmerjajo aplikacijo po poteh ki so definirane.

*(slika x: Poti)*

*Primer:*

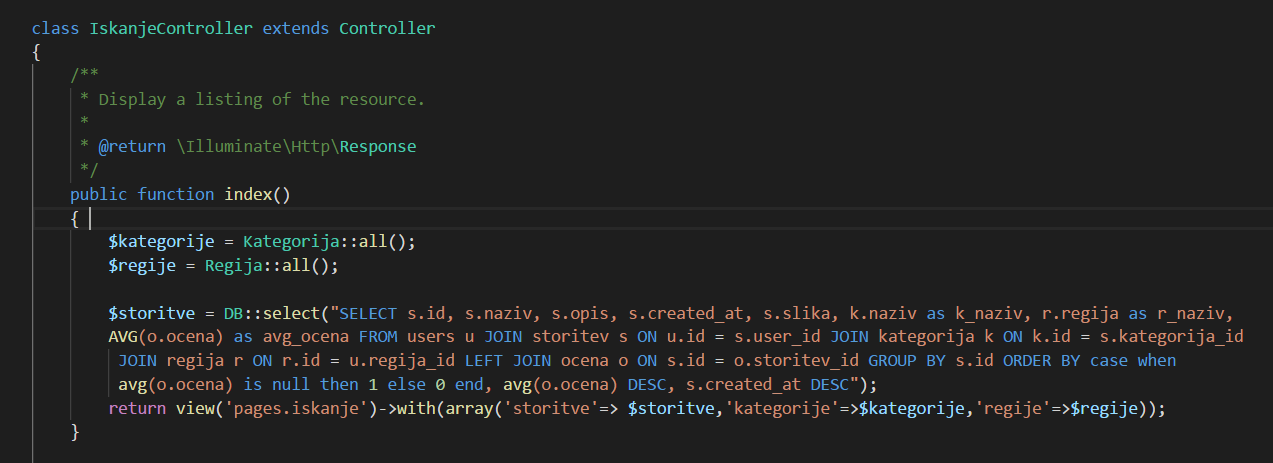
**Korak 1**: Uporabnik v uporabniškem v vmesniku izbere funkcijo iskanje.



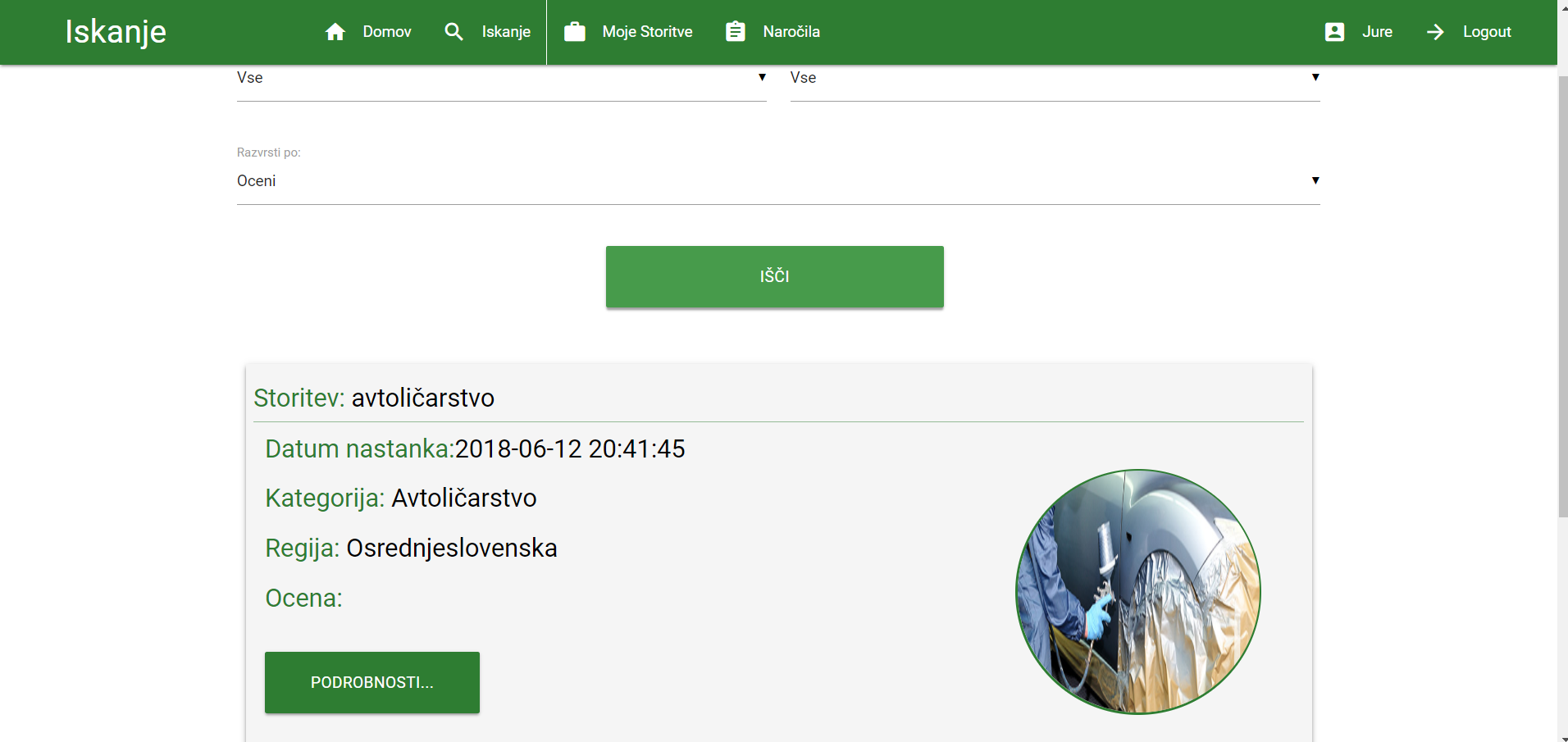
**Korak 2:** Ustrezna pot navedena v datoteki proži metodo, v kontrolerju ki je definiran v poti.



**Korak 3:** prožijo se metode v navedenem kontrolerju.



**Korak 4:** Metoda iz podatkovne baze izbere vse storitve in jih vrne prikazane na strani ''pages.iskanje''.

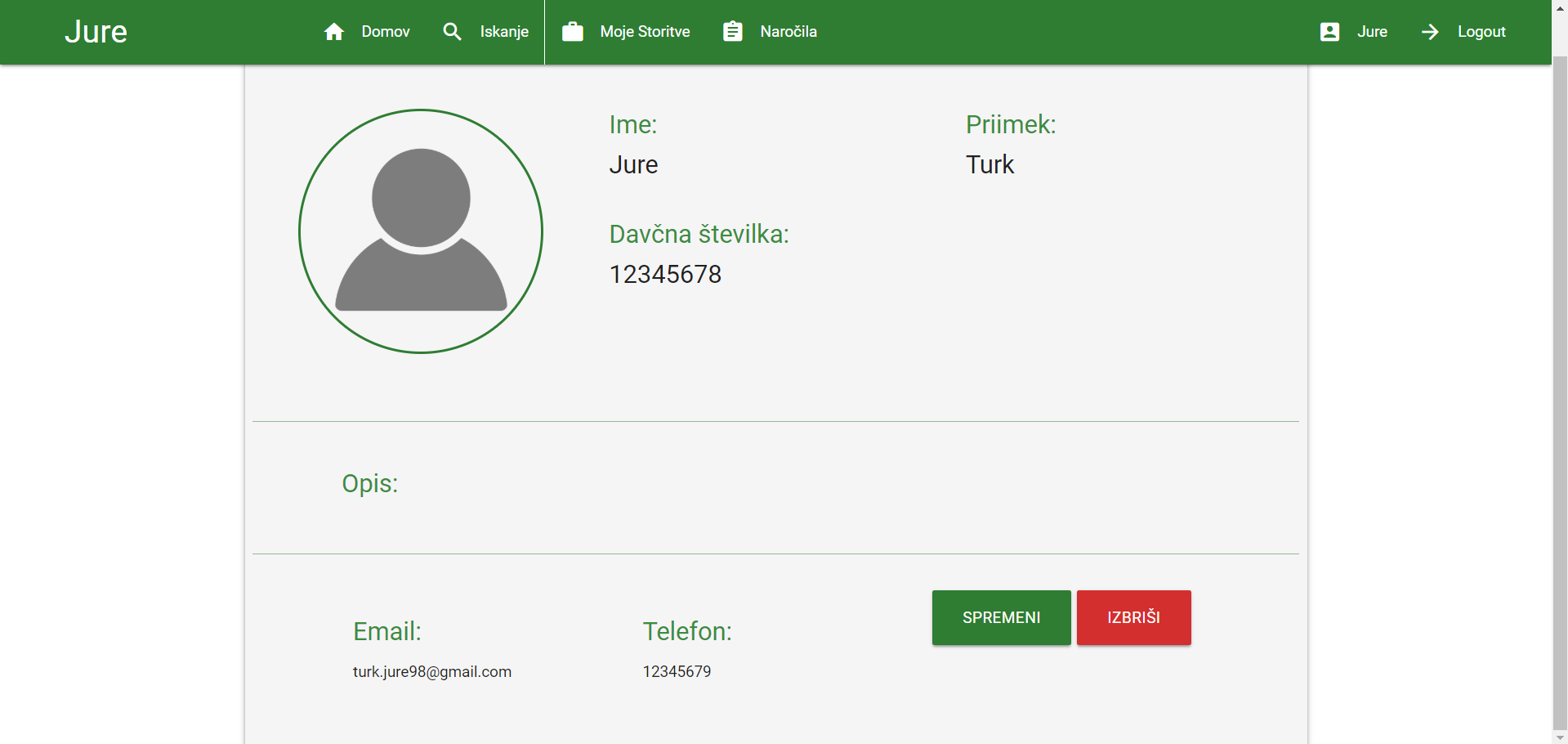


# Kontrolerji

Kot je že prikazano v zgornjem primeru, lahko kontrolerji vračajo različnen ''view'' oziroma pogled v obliki spletne strani. V njih pa vključujemo vse preostale funkcionalnosti.

*Primer:*

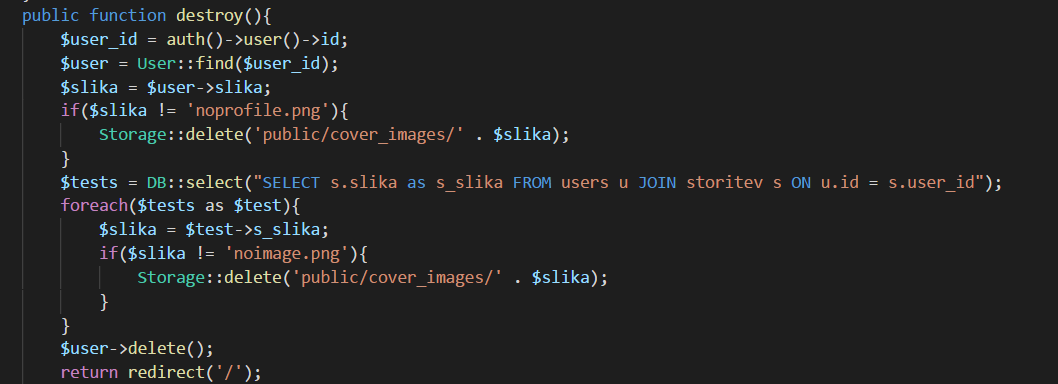
**Korak 1:** Uporabnik želi izbrisati svoj profil, zato izbere gumb za izbris svojega profila.



**Korak 2:** Ustrezna pot, kliče metodo v kontrolerju.



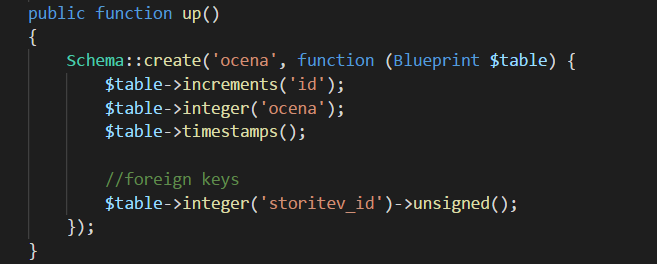
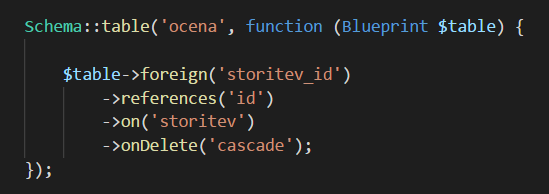
**Korak3:** Metoda izbriše uporabnika iz podatkovne baze.



# Migracije

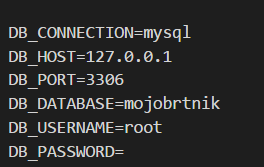
Ker smo imeli težave pri implementaciji podatkovne baze, smo podatkovno bazo ustvarili na novo s pomočjo migracij. Migracije delujejo tako, da v terminalu s pomočjo artisan-a ukažemo aplikaciji da generira podatkovno bazo definirano v migracijah. Definirati je potrebno vsako tabelo, njene atribute, ter tuje ključe.

**Korak 1:** V datoteko z migracijami definiramo vse tabele, ki jih potrebujemo in definiramo tuje ključe, ter druge tabele na katere tuji ključi referencirajo.

(slika x: Predefiniranje tabele za oceno) (slika x: Tuji ključi za tabelo ocena)

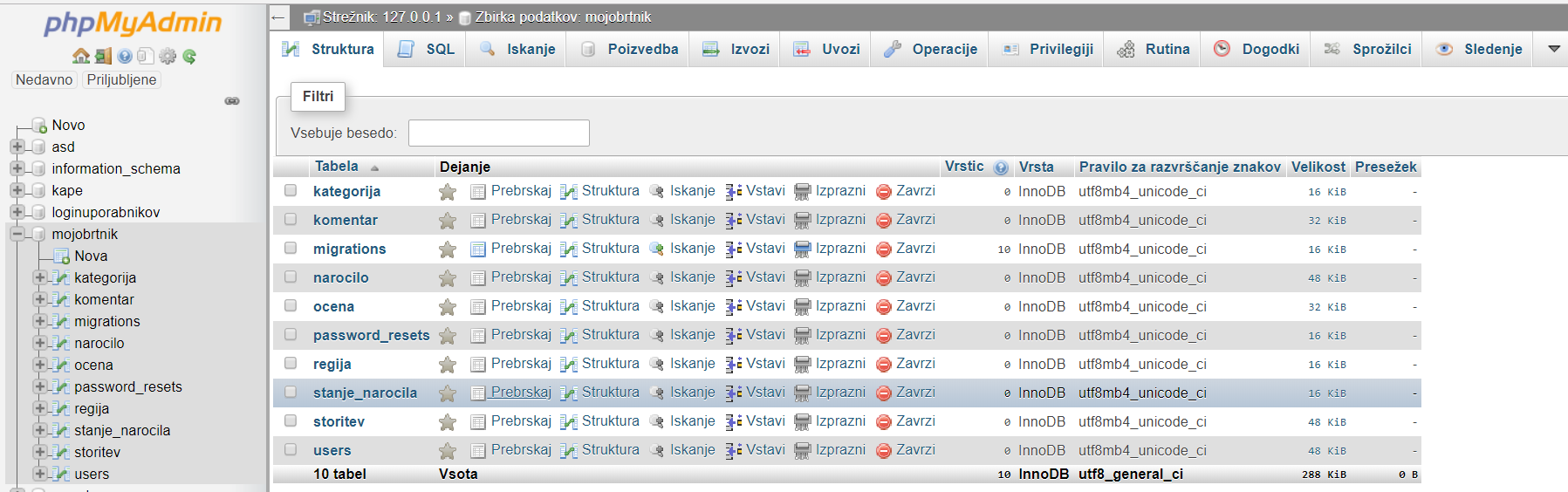
**Korak 2:** V datoteki ENV, navedemo vse atribute potrebne za povezavo z podatkovno bazo. Ko to storimo moramo tudi ustvariti podatkovno bazo z imenom navedenim v ENV datoteki. To smo storili

(slika x: Env datoteka)

**Korak 3:** Z ustreznim ukazom v ukaznem pozivu migriramo tabele .

 (slika x: Artisan ukaz)

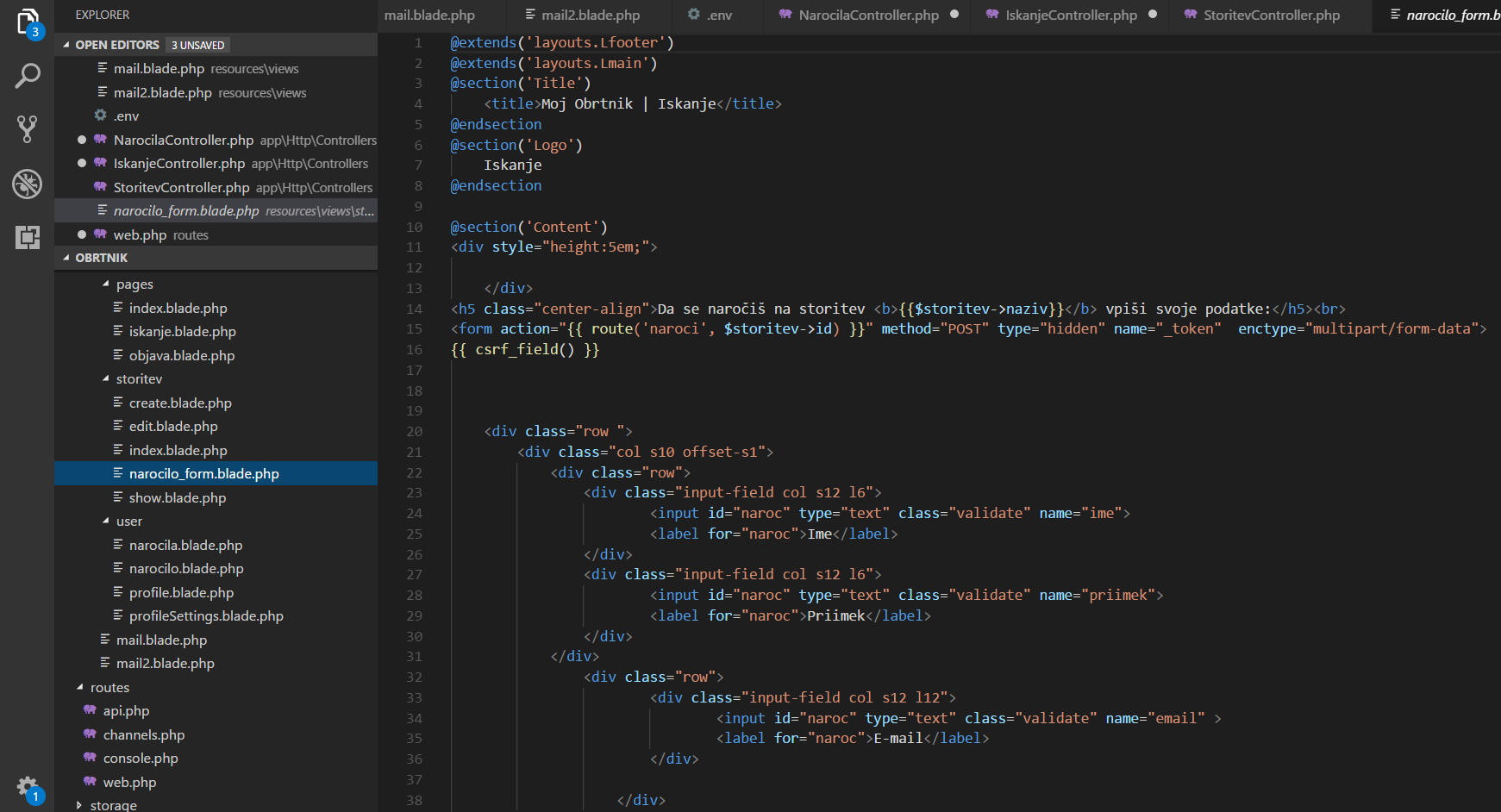
**Korak 4** : S podatkovno bazo lahko upravljamo v orodju phpMyAdmin . Tukaj smo vnašali testne podatke, in nasplošno izvajali ostale interakcije z podatkovno bazo.



(slika x: phpMyAdmin)

# Blade

Če se ravnamo po MVC modelu, komponenta blade predstavlja View oz. pogled. Blade je statična komponenta. Ustvarjena z HTML .



# Prijava in registracija

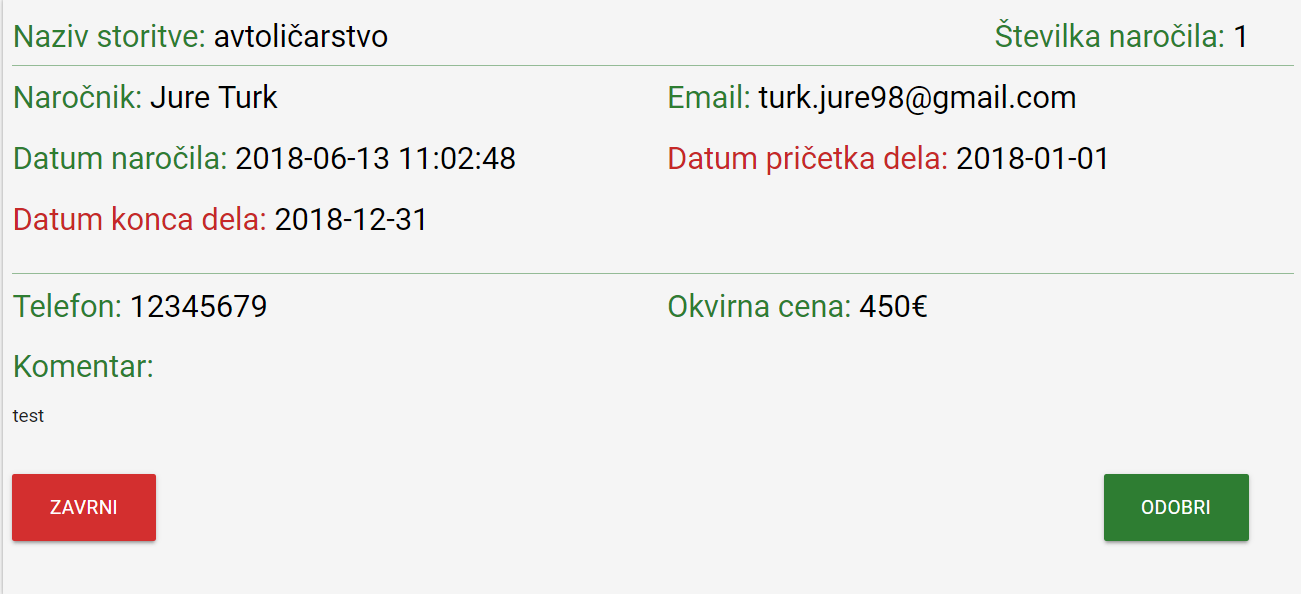
Prija

# E-pošta

V naš projekt smo uvedli tudi pošiljanje E-pošte, saj je to način komunikacije z uporabnikom. Ko obrtnik odobri oziroma zavrne naročilo, uporabnik ki je naročilo oddal dobi po e-pošti obvestilo o tem, ali je njegovo naročilo bilo sprejeto oziroma zavrnjeno.

*Primer delovanja e-pošte:*

**Korak 1:** Obrtnik odobri naročilo, ki mu je bilo podano s strani uporabnika.

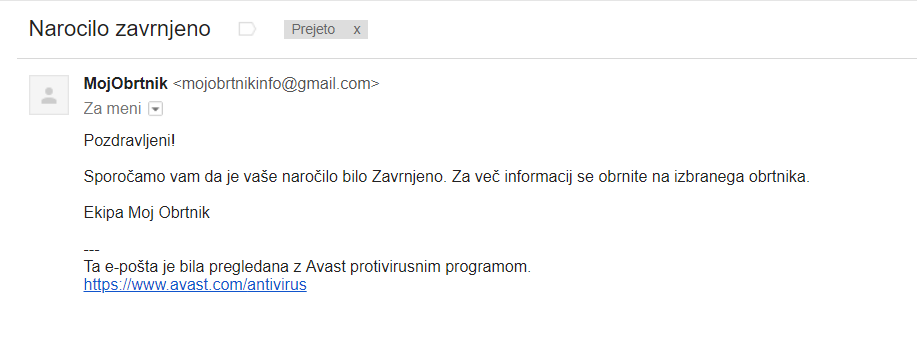


(slika x: Naročilo)

**Korak 2:** Metoda proži funkcijo ki pošlje e-pošto, ista metoda je bila implementirana za obe funkcionalnosti aplikacije(odobri in zavrni), le podloga v kateri je določeno ali je uporabnikovo naročilo bilo sprejeto ali zavrnjeno se v metodi spremeni da uporabnik dobi ustrezno e-pošto.

(slika x: Metoda za pošiljanje e-pošte)

**Korak 3:** Uporabnik prejme ustrezno e-pošto.

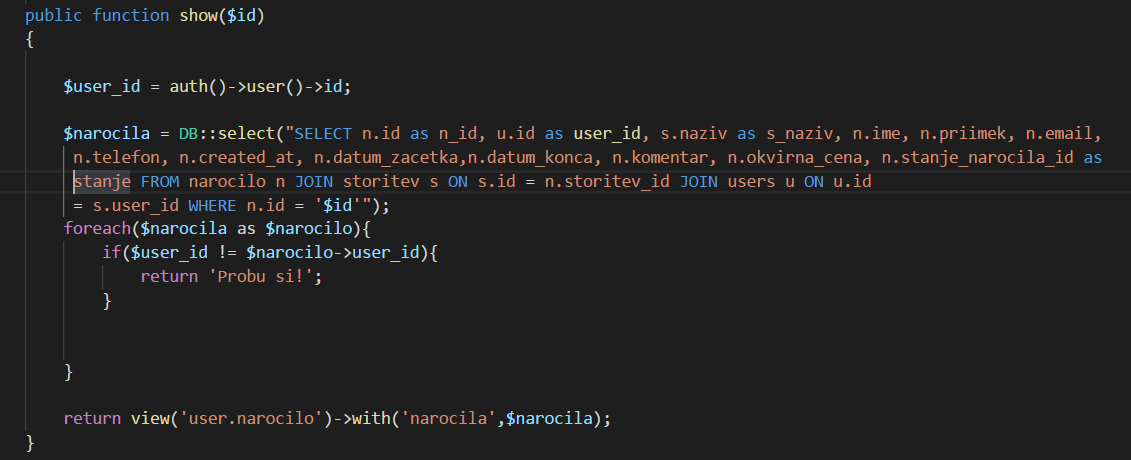


(slika x: Primer e-pošte)

# Naročila, storitve in autorizacija

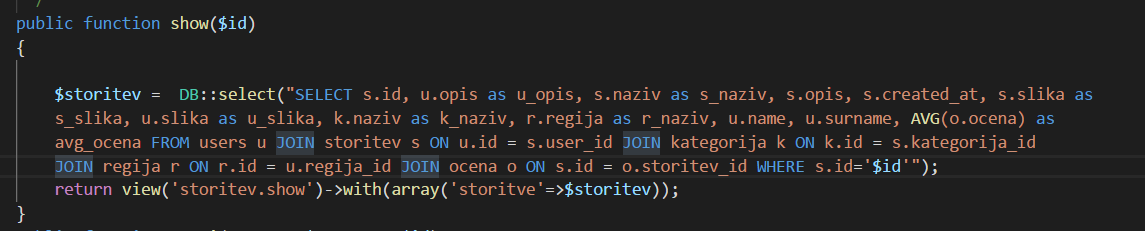
Ena izmed glavnih funkcionalnosti naše aplikacije je prikaz storitev in naročil. Storitve se prikažejo navadnemu uporabniku aplikacije. Naročila pa le obrtniku kateremu pripada določeno naročilo, tukaj je pomemben element naše aplikacije tudi verifikacijo.

Spodaj navedena metoda prikaže naročila, ki pripadajo določenemu uporabniku.



(slika x: Metoda za prikaz naročil)

Metoda ki iz podatkovne baze prikaže vse storitve uporabniku. Da



1. Varnost
2. Testiranje
3. Zaključek