VELEUČILIŠTE U RIJECI

POSLOVNI ODJEL RIJEKA

Kolonić Laura

**Sustav za turističke izlete brodicama**

PROJEKTNA DOKUMENTACIJA

Rijeka, 2020.

VELEUČILIŠTE U RIJECI

POSLOVNI ODJEL RIJEKA

**Sustav za turističke izlete brodicama**

PROJEKTNA DOKUMENTACIJA

Kolegij: Izgradnja objektno orijentiranih aplikacija

Mentor: Davidović Vlatka, viši predavač

Student: Kolonić Laura

Rijeka, travanj 2020

**SADRŽAJ**

[1. Opis sustava 1](#_Toc42891490)

[2. Analiza sustava - specifikacija zahtjeva 2](#_Toc42891491)

[3. Scenariji 4](#_Toc42891492)

[3.1. Autentikacija administratora, klijenta 4](#_Toc42891493)

[3.2. Upravljanje brodicama (Unos, izmjena i brisanje) 7](#_Toc42891494)

[3.3. Upravljanje rutama 10](#_Toc42891495)

[3.4. Upravljanje kalendarom plovidbe 13](#_Toc42891496)

[3.5. Upravljanje zaposlenicima 16](#_Toc42891497)

[3.6. Izrada izvještaja 19](#_Toc42891498)

[3.7. Pregled gostiju na brodici 21](#_Toc42891499)

[3.8. On-line registracija klijenta (gosta) 23](#_Toc42891501)

[3.9. Rezervacija termina plovidbe 25](#_Toc42891502)

[3.10. Pregled ruta i brodica 28](#_Toc42891503)

[4. Dijagram klasa 30](#_Toc42891504)

[5. Dizajn sustava 31](#_Toc42891505)

[5.1. Početni prozor kod otvaranja stranice i prilikom rezervacije 31](#_Toc42891506)

[5.2. Administrator početna stranica 31](#_Toc42891507)

[5.3. Administrator opcija „Brodovi“ iz navigacije početne stranice 32](#_Toc42891508)

[5.4. Administrator – unos novog broda 32](#_Toc42891510)

[6. Implementacija sustava 33](#_Toc42891512)

[6.1. Radno okruženje aplikacije 33](#_Toc42891513)

[6.2.1. phpmyadmin 33](#_Toc42891514)

[6.2.2. XAMPP 33](#_Toc42891515)

[6.2.3. Visual Studio 34](#_Toc42891516)

[6.2.4. Laravel 34](#_Toc42891517)

[6.2. Dijelovi programskog koda 35](#_Toc42891518)

[6.2.1. Registracija gosta / administratora 35](#_Toc42891519)

[6.2.2. Kreiranje, izmjena, brisanje brodova 41](#_Toc42891520)

[6.2.3. Rezervacija putovanja 47](#_Toc42891521)

[7. Korištenje aplikacije 51](#_Toc42891522)

[8.Zaključak 52](#_Toc42891523)

[LITERATURA 53](#_Toc42891524)

# 1. Opis sustava

Sustav za turističke izlete brodicama ima jednostavnu ulogu, a ta je da olakša svakodnevno rukovanje s podatcima zaposlenima. Sustav će koristiti dva korisnika, to su administrator, koji je ujedno i vlasnik i klijent. Uloga administratora je da evidentira brodice, rute, te ažurira kalendar plovidbe. Također, po potrebi, može i otkazati plovidbe. Omogućena je i izrada izvještaja vlasniku, na taj način može dobiti uvid u najčešće posjećenije rute. Sustav sadrži i aktualnu vremensku prognozu, sukladno s njom, administrator može, kako je već spomenuto, otkazivati termine. Klijent ima mogućnost on-line registracije, pregleda brodica, ruta i kalendara, te rezervacije, kao i otkazivanja termina.

Akteri:

* Administrator (vlasnik)
* Klijent

# 2. Analiza sustava - specifikacija zahtjeva

Aplikacija će biti korištena od strane dva korisnika, to su administrator i klijent. U nastavku će detaljnije biti opisano koje mogućnosti ima svaki od korisnika.

**ADMINISTRATOR:**

* Evidencija brodica zajedno s njihovim karakteristikama.
* Evidencija ruta.
* Evidencija putovanja, na temelju brodica i ruta.
* Praćenje vremenske prognoze.
* Izrada izvještaja (najčešće korištene brodice, najposjećenije rute..)

**KLIJENT:**

* On-line registracija na stranici
* Pregled brodica, ruta, kalendara plovidbe
* Rezervacija termina plovidbe.

Slika 1. Use-case dijagram

Slika na kojoj se prikazuje tekst, karta

Opis je automatski generiran

Slika na kojoj se prikazuje tekst, karta

Opis je automatski generiran

*Izvor: autor*

# 3. Scenariji

## 3.1. Autentikacija administratora, klijenta

**Identifikacijski sažetak:** Korisnici sustava imaju mogućnost prijave u isti kako bi im bile omogućena određena svojstva koja sustav nudi za svaku kategoriju.

**Akteri:** Administrator, klijent.

**Preduvjeti:** Podatci o registraciji u sustav postoje u bazi podataka.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

* 1. Prikaz glavnog izbornika
  2. Administrator odabir opciju prijava kao administrator
  3. Administrator unosi potrebne podatke
  4. Administrator potvrđuje unos
  5. Sustav uspoređuje unesene podatke s onima u bazi podataka
  6. Uneseni podatci i podatci u bazi podataka su identični
  7. Sustav javlja uspješnu prijavu i prikazuje glavni izbornik CMS administrator

**A: Alternativne sekvence:**

**A1: Prijava klijenta**

* 1. Prikaz glavnog izbornika
  2. Klijent odabir opciju prijava kao klijent
  3. Klijent unosi potrebne podatke
  4. Klijent potvrđuje unos
  5. Sustav uspoređuje unesene podatke s onima u bazi podataka
  6. Uneseni podatci i podatci u bazi podataka su identični
  7. Sustav javlja uspješnu prijavu i prikazuje glavni izbornik CMS klijent

**E: Sekvence s greškom:**

1. A5
2. Uneseni podatci i podatci u bazi podataka su različiti
3. Sustav javlja da prijava nije uspješna, te vraća korisnika na glavni izbornik

**Nefunkcionalni uvjeti (ograničenja):** /

**Dijagram aktivnosti:**

Slika 2. Dijagram aktivnosti – autentikacija administratora, klijenta  
Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.2. Upravljanje brodicama (Unos, izmjena i brisanje)

**Identifikacijski sažetak:** Administrator sustava ima mogućnost unosa brodica. Nakon što administrator unese podatke o brodici, stavka se sprema u bazu podataka. Ukoliko administrator želi izmijeniti ili izbrisati brodicu, to je omogućeno.

**Akteri:** Administrator.

**Preduvjeti:** Uspješna prijava administratora u sustav.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju brodice
  4. Administrator unosi potrebne podatke o brodici
  5. Administrator potvrđuje unos
  6. Sustav sprema unos i javlja da je novi unos uspješno spremljen

**A: Alternativne sekvence:**

**A1: Izmjena stavke**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju brodice
  4. Administrator pristupa pregledu brodica
  5. Administrator odabire opciju izmijeni
  6. Administrator mijenja podatke o brodici
  7. Administrator potvrđuje izmjenu
  8. Sustav javlja da je izmjena uspješno izvršena

**A2: Brisanje stavke**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju brodice
  4. Administrator pristupa pregledu brodica
  5. Administrator odabire opciju briši
  6. Administrator briše stavku
  7. Sustav javlja da je brisanje uspješno izvršeno

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja):** /

**Dijagram aktivnosti:**

Slika 3. Dijagram aktivnosti – upravljanje brodicama

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.3. Upravljanje rutama

**Identifikacijski sažetak:** Administrator sustava ima mogućnost unosa ruta. Nakon što administrator unese podatke o ruti, stavka se sprema u bazu podataka. Ukoliko administrator želi izmijeniti ili izbrisati rutu, to je omogućeno.

**Akteri:** Administrator.

**Preduvjeti:**

1. Uspješna prijava administratora u sustav.

2. Postojeće brodice u sustavu.

3. Postojeći zaposlenici u sustavu.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabir opciju rute
4. Administrator unosi potrebne podatke o ruti
5. Administrator potvrđuje unos
6. Sustav sprema unos i javlja da je novi unos uspješno spremljen

**A: Alternativne sekvence:**

**A1: Izmjena stavke**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabire opciju rute
4. Administrator pristupa pregledu ruta
5. Administrator odabire opciju izmijeni
6. Administrator mijenja podatke o ruti
7. Administrator potvrđuje izmjenu
8. Sustav javlja da je izmjena uspješno izvršena

**A2: Brisanje stavke**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju rute
  4. Administrator pristupa pregledu ruta
  5. Administrator odabire opciju briši
  6. Administrator briše stavku
  7. Sustav javlja da je brisanje uspješno izvršeno

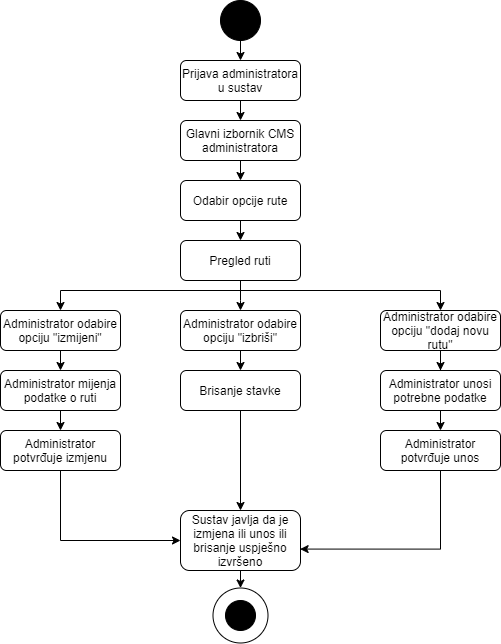
**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja):**

* U nazivu rute ne smiju biti brojevi
* Cijene su izražene u kunama (0,00 kn)

**Dijagram aktivnosti:**

Slika 4. Dijagram aktivnosti – upravljanje rutama



*Izvor: autor*

## 3.4. Upravljanje kalendarom plovidbe

**Identifikacijski sažetak:** Administrator sustava ima mogućnost ažuriranja kalendara plovidbe koji je vidljiv svim korisnicima. Nakon što administrator unese podatke o novoj ruti, pridružuje joj odgovarajući datum, zaposlenika i stavka se sprema u bazu podataka, te se kao takva prikazuje u kalendaru. Ukoliko administrator želi izmijeniti ili izbrisati kalendar, to je omogućeno.

**Akteri:** Administrator.

**Preduvjeti:**

1. Uspješna prijava administratora u sustav.

2. Postojeće rute u sustavu.

3. Postojeći zaposlenici u sustavu.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prijavljivanje administratora u sustav

1. Prikaz glavnog izbornika CMS – administratora
2. Administrator odabir opciju kalendar
3. Administrator pristupa kalendaru
4. Administrator odabire opciju „dodaj u kalendar“
5. Administrator unosi potrebne podatke za unos u kalendar
6. Administrator potvrđuje unos
7. Sustav sprema unos i javlja da je novi unos uspješno spremljen

**A: Alternativne sekvence:**

**A1: Izmjena stavke**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabire opciju kalendar
4. Administrator pristupa kalendaru
5. Administrator odabire opciju izmijeni
6. Administrator mijenja podatke u kalendaru
7. Administrator potvrđuje izmjenu
8. Sustav javlja da je izmjena uspješno izvršena

**A2: Brisanje stavke**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju kalendar
  4. Administrator pristupa kalendaru
  5. Administrator odabire opciju briši
  6. Administrator briše stavku
  7. Sustav javlja da je brisanje uspješno izvršeno

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja): /**

**Dijagram aktivnosti:**

Slika 5. Dijagram aktivnosti – upravljanje kalendarom plovidbe

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.5. Upravljanje zaposlenicima

**Identifikacijski sažetak:** Administrator sustava ima mogućnost unosa zaposlenika u sustav. Ukoliko administrator želi izmijeniti ili izbrisati zaposlenike, to je omogućeno.

**Akteri:** Administrator.

**Preduvjeti:**

1. Uspješna prijava administratora u sustav.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabir opciju zaposlenici
4. Administrator pristupa pregledu zaposlenika
5. Administrator odabire opciju „dodaj zaposlenika“
6. Administrator unosi potrebne podatke za unos zaposlenika
7. Administrator potvrđuje unos
8. Sustav sprema unos i javlja da je novi unos uspješno spremljen

**A: Alternativne sekvence:**

**A1: Izmjena stavke**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabire opciju zaposlenici
4. Administrator pristupa pregledu zaposlenika
5. Administrator odabire opciju izmijeni
6. Administrator mijenja podatke o zaposleniku
7. Administrator potvrđuje izmjenu
8. Sustav javlja da je izmjena uspješno izvršena

**A2: Brisanje stavke**

* 1. Prijavljivanje administratora u sustav
  2. Prikaz glavnog izbornika CMS – administratora
  3. Administrator odabir opciju zaposlenici
  4. Administrator pristupa pregledu zaposlenika
  5. Administrator odabire opciju briši
  6. Administrator briše stavku
  7. Sustav javlja da je brisanje uspješno izvršeno

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja):**

* Prilikom unosa datuma rođenja, datum ne smije biti iz budućnosti
* Ograničiti broj polja za unos OIB-a
* Prezime, ime ne smije sadržavati znamenke

**Dijagram aktivnosti:**

Slika 6. Dijagram aktivnosti – upravljanje zaposlenicima

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.6. Izrada izvještaja

**Identifikacijski sažetak:** Administrator ima mogućnost izrade izvještaja na temelju podataka koji se nalaze u bazi podataka. Administrator ima mogućnost kreiranja izvještaja za najposjećenije brodice, te za najposjećenije rute.

**Akteri:** Administrator.

**Preduvjeti:**

1. Uspješna prijava administratora u sustav.

2. Podatci o obavljenim rutama u bazi podataka.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabir opciju izvještaji
4. Administrator pristupa pregledu opcija za izradu izvještaja
5. Administrator odabire opciju „najposjećenije rute“
6. Administrator potvrđuje odabir
7. Sustav generira i prikazuje izvještaj

**A: Alternativne sekvence:**

**A1: Najkorištenije brodice**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administratora
3. Administrator odabir opciju izvještaji
4. Administrator pristupa pregledu opcija za izradu izvještaja
5. Administrator odabire opciju „najkorištenije rute“
6. Administrator potvrđuje odabir
7. Sustav generira i prikazuje izvještaj

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja): /**

**Dijagram aktivnosti:**

Slika 7. Dijagram aktivnosti – izrada izvještaja

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.7. Pregled gostiju na brodici

**Identifikacijski sažetak:** Vlasnik ima mogućnost pregleda svih gostiju koji su na brodici s obzirom na jedinstvenu šifru plovidbe.

**Akteri:** Administrator - vlasnik

**Preduvjeti:**

1. Uspješna prijava administratora u sustav.

2. On-line rezervacija rute od strane korisnika.

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prijavljivanje administratora u sustav
2. Prikaz glavnog izbornika CMS – administrator
3. Odabir opcije „pregled gostiju“
4. Administratoru se prikazuju klijenti na odabranoj ruti

**A: Alternativne sekvence: /**

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja): /**

**Dijagram aktivnosti:**

Slika 8. Dijagram aktivnosti – pregled gostiju na brodici

## 

## 3.8. On-line registracija klijenta (gosta)

**Identifikacijski sažetak:** Kako bi rezervirao termin plovidbe, gost se registrira u sustav pomoću obrasca. Nakon registracije, te kasnije uspješne autentikacije, klijent uz pregled termina i brodica, ima mogućnost rezervacije rute.

**Akteri:** Klijent

**Preduvjeti: /**

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Klijent odabire opciju registracija.
2. Klijent unosi potrebne podatke za registraciju.
3. Klijent potvrđuje prijavu.
4. Sustav javlja da je uspješno registriran.

**A: Alternativne sekvence: /**

**E: Sekvence s greškom:**

**E1: Korisničko ime već postoji**

1. A3

2. Sustav javlja da korisničko ime već postoji u sustavu

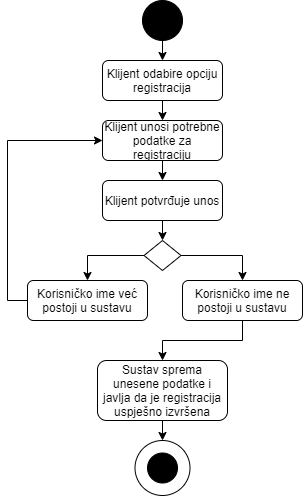
3. Sustav vraća gosta na A2

**Nefunkcionalni uvjeti (ograničenja):**

* Prilikom unosa datuma rođenja, datum ne smije biti iz budućnosti
* Ograničiti broj polja za unos OIB-a
* Prezime, ime ne smije sadržavati znamenke

**Dijagram aktivnosti:**

Slika 9. Dijagram aktivnosti – on line registracija gosta



*Izvor: autor*

## 3.9. Rezervacija termina plovidbe

**Identifikacijski sažetak:** Nakon uspješne registracije, klijent se prijavljuje u sustav i izvrašava rezervaciju termina plovidbe.

**Akteri:**  Klijent.

**Preduvjeti:** Uspješna prijava klijenta u sustav (potrebna je prethodna registracija).

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

* 1. Prijavljivanje klijenta u sustav
  2. Prikaz glavnog izbornika CMS – klijent
  3. Klijent odabire opciju rezervacija
  4. Klijent u kalendaru odabire željenu plovidbu
  5. Klijent pohranjuje odabir
  6. Sustav javlja da je ruta uspješno rezervirana

**A: Alternativne sekvence:**

**A1: Otkazivanje termina**

1. Prijavljivanje klijenta u sustav
2. Prikaz glavnog izbornika CMS – klijent
3. Klijent odabire opciju moje rezervacije
4. Klijent uklanja odabrani termin
5. Sustav javlja da je termin uspješno uklonjen

**E: Sekvence s greškom:**

**E1: Otkazivanje termina**

Klijent plovidbu ne može otkazati dan prije same plovidbe.

1. Prijavljivanje klijenta u sustav
2. Prikaz glavnog izbornika CMS – klijent
3. Klijent odabire opciju moje rezervacije
4. Klijent uklanja odabrani termin
5. Sustav javlja da termin nije uspješno uklonjen

**Nefunkcionalni uvjeti (ograničenja):**  /

**Dijagram aktivnosti:**

Slika 10. Dijagram aktivnosti – rezervacija termina plovidbe

Slika na kojoj se prikazuje snimka zaslona

Opis je automatski generiran

*Izvor: autor*

## 3.10. Pregled ruta i brodica

**Identifikacijski sažetak:** Klijent ima mogućnost pregleda ruta i brodica. Za potrebe pregleda ruta i brodica, nije se potrebno prijavljivati u sustav. Osim manualnog načina za pregled ruta i brodica koji je opisan u nastavku, klijent također preko kalendara može pristupiti opisu ruta.

**Akteri:** Klijent

**Preduvjeti: /**

**Tijek događaja:**

**G: Glavni uspješni scenarij:**

1. Prikaz glavnog izbornika CMS – klijent
2. Klijent odabir opciju pregled ruta
3. Klijentu se prikazuju postojeće rute s opisima

**A: Alternativne sekvence:**

**A1: Pregled brodica:**

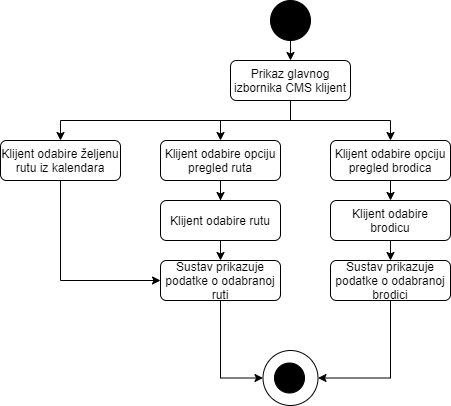
1. Prikaz glavnog izbornika CMS – klijent
2. Klijent odabire opciju pregled brodica
3. Klijentu se prikazuju postojeće brodice

**E: Sekvence s greškom:** /

**Nefunkcionalni uvjeti (ograničenja): /**

**Dijagram aktivnosti:**

Slika 11. Dijagram aktivnosti – pregled ruta i brodica



*Izvor: autor*

# 4. Dijagram klasa

*Slika 12. Dijagram klasa*

*Slika na kojoj se prikazuje snimka zaslona

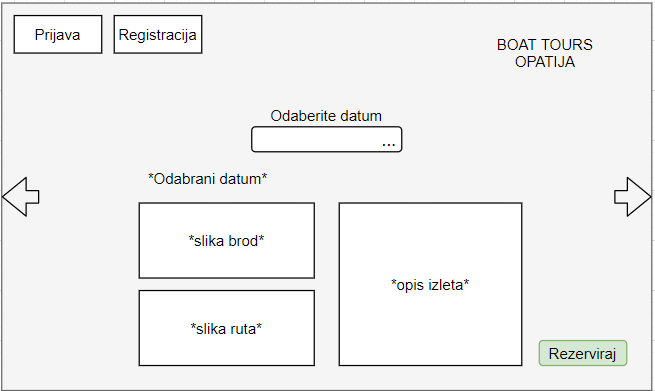
Opis je automatski generiran*

*Izvor: autor*

# 5. Dizajn sustava

U nastavku će biti prikazani neki prozori iz dijelova aplikacije.

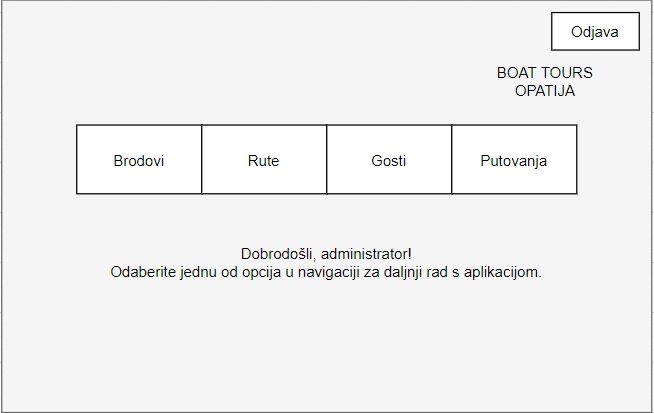
## 5.1. Početni prozor kod otvaranja stranice i prilikom rezervacije

*Slika 13. Početni prozor kod otvaranja stranice i prilikom rezervacije*

*žzvor: autor*

## 5.2. Administrator početna stranica

*Slika 14. Administrator početna stranica*



*Izvor: autor*

## 5.3. Administrator opcija „Brodovi“ iz navigacije početne stranice

Slika 15. Administrator opcija „Brodovi“ iz navigacije početne stranice

## 

Izvor: autor

## 5.4. Administrator – unos novog broda

Slika 16. Administrator – unos novog broda

## 

Izvor: autor

# 6. Implementacija sustava

U nastavku će biti prikazani dijelovi koda aplikacija, te radno okruženje u okviru kojega je aplikacija razvijana.

## 6.1. Radno okruženje aplikacije

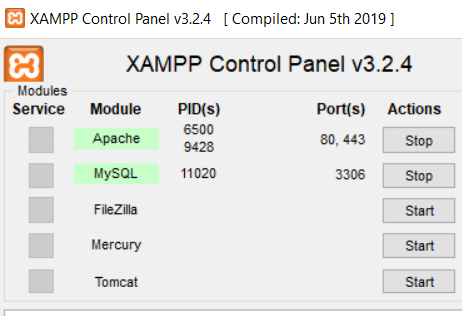
### 6.2.1. phpmyadmin

Phpmyadmin je besplatni softverski alat napisan u PHP-u, čija je namjena administriranje MySQL-a putem Web-a. Spomenuti alat korišten je na localhost-u kao baza podataka. Dakle, omogućio je upravljanje bazom, odnosno, svim funkcionalnostima koje korištenje iste podrazumijeva. Spomenuto se odnosi na upravljanje tablicama – unos, izmjena brisanje, pregled.

### 6.2.2. XAMPP

XAMPP predstavlja u potpunosti besplatan open-source serverski paket za jednostavnu instalaciju Apache servera za računala koja posjeduju operativni sustav Windows, Linux ili OS X. Namijenjen je za upotrebu na lokalnom poslužitelju, te služi, prvenstveno programerima, za testiranje aplikacije. Prije pokretanja same aplikacije i početka razvoja iste, bilo je potrebno pokrenuti Apache i MySQL na način koji prikazuje slika koja slijedi.

*Slika 17. Korištenje XAMPP*

**

*Izvor: autor*

### 6.2.3. Visual Studio

Visual Studio je korišten kao integrirano razvojno okruženje, te je razvijen od strane Microsoft-a. Služi za razvoj računalnih programa, web-stranica, raznih aplikacija i drugih usluga. Sadrži i uređivač programskog koda koji podržava IntelliSense (služi za predlaganje ostatka koda), te refaktoriranje koda. Također, sadrži i integrirani program za otklanjanje grešaka (engl. *debugger*). Osim spomenutog, sadrži i alate koji pomažu prilikom kreiranja aplikacija s grafičkim korisničkim sučeljem. Podržava različite programske jezike – ugrađeni programski jezici su sljedeći: C, C++, C++/CLI, VB.NET, C#, F#. Pruža i podršku za ostale jezike, kao što su Python, Ruby.

### 6.2.4. Laravel

Laravel je framework, koji kao takav pruža strukturu od koje gradimo aplikaciju. Dakle, dozvoljava da povežemo različite API-je, kako bi definirali konačnu strukturu buduće aplikacije. Ono za što je framework još zaslužan je sljedeće:

* Čini rad na kompleksnim projektima jednostavnijim.
* Pisanje čistog i ponovno upotrebljivog koda.
* Nameće pravila za implementaciju koda na konzistentan način.
* Testiranje i otkanjanje grešaka je jednostavnije.

Dakle, Laravel je otvorenog koda i posjeduje sve komponente za brz razvoj aplikacije. Dosljedno prati Model View Controller (MVC) arhitekturu. Iako se napominje da se koristi za velike projekte, a projekt izrađen za potrebe ovog kolegija nije veliki, nastojalo se bolje upoznati Laravel i način njegovog funkcioniranja.

## 6.2. Dijelovi programskog koda

U nastavku će biti prikazana izrada pojedinih dijelova programskog koda.

### 6.2.1. Registracija gosta / administratora

Kod 1. Rute

<?php

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Route;

/\*

|--------------------------------------------------------------------------

| Web Routes

|--------------------------------------------------------------------------

|

| Here is where you can register web routes for your application. These

| routes are loaded by the RouteServiceProvider within a group which

| contains the "web" middleware group. Now create something great!

|

\*/

Route::get('/', 'GostPutovanjeController@index')->name('welcome');

Auth::routes();

Route::get('/logout', 'Auth\LoginController@logout');

Route::get('/index', 'GostPutovanjeController@pregled')->name('pregledputovanja.pregled');

//ADMIN RUTE

Route::group(['middleware' => ['web','admin']], function()

{

    Route::get('/cms/index', 'HomeController@index')->name('admin');

    Route::resource('/cms/brod', 'BrodController');

    Route::resource('/cms/ruta', 'RutaController');

    Route::resource('/cms/zaposlenik', 'ZaposlenikController');

    Route::resource('/cms/kategorija', 'kategorijaGostController');

    Route::resource('/cms/putovanje', 'PutovanjeController');

    Route::get('/cms/gostPutovanje', 'pregledGostiController@index')->name('gostPutovanje.index');

    Route::get('/cms/gostPutovanje/pregled/', 'pregledGostiController@pregled')->name('gostPutovanje.pregled');

});

//AUTH RUTE

Route::group(['middleware' => ['web','auth']], function()

{

    Route::get('/index/{putovanje}', 'GostPutovanjeController@rezerviraj')->name('pregledputovanja');

    Route::get('/pregledrezervacija', 'GostPutovanjeController@pregledsvojihrezervacija')->name('pregledsvojihrezervacija.pregled');

    Route::get('/pregledrezervacija/{id}', 'GostPutovanjeController@izbrisirezervaciju')->name('izbrisirezervaciju');

});

*Izvor: autor*

Prije svega, u dokumentu web.php nailazimo na kod koji se nalazi na prethodnoj slici. Spomenuti dokument sadrži sve rute ove aplikacije. Kao što možemo i vidjeti iz prikazanog koda, rute su grupirane po grupama. Odnosno, imamo ne-grupirane rute, te imamo grupirane admin rute i auth rute. Ne-grupirane rute su dostupne svima, admin rute su dostupne adminu, a auth rute autentificiranom korisniku, što može biti admin ili gost koji je registiran, te potom prijavljen u aplikaciju. Dakle, kako bi se pristupilo određenom dijelu aplikacije, prvo se kroz rute proverava da li je korisnik prijavljen kao admin ili kao gost, ukoliko je, dozvoljava im rukovanje s rutama koje se nalaze u grupama. Kako je već spomenuto, ne-grupiranim rutama mogu svi pristupiti.

Kod 2. Registracija korisnika

<?php

namespace App\Http\Controllers\Auth;

use App\User;

use Carbon\Carbon;

use App\Http\Controllers\Controller;

use Illuminate\Support\Facades\Hash;

use App\Providers\RouteServiceProvider;

use Illuminate\Support\Facades\Validator;

use Illuminate\Foundation\Auth\RegistersUsers;

class RegisterController extends Controller

{

    /\*

    |--------------------------------------------------------------------------

    | Register Controller

    |--------------------------------------------------------------------------

    |

    | This controller handles the registration of new users as well as their

    | validation and creation. By default this controller uses a trait to

    | provide this functionality without requiring any additional code.

    |

    \*/

    use RegistersUsers;

    /\*\*

     \* Where to redirect users after registration.

     \*

     \* @var string

     \*/

    protected $redirectTo = RouteServiceProvider::HOME;

    /\*\*

     \* Create a new controller instance.

     \*

     \* @return void

     \*/

    public function \_\_construct()

    {

        $this->middleware('guest');

    }

    /\*\*

     \* Get a validator for an incoming registration request.

     \*

     \* @param  array  $data

     \* @return \Illuminate\Contracts\Validation\Validator

     \*/

    protected function validator(array $data)

    {

        return Validator::make($data, [

            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],

            'password' => ['required', 'string', 'min:8', 'confirmed'],

        ]);

    }

    /\*\*

     \* Create a new user instance after a valid registration.

     \*

     \* @param  array  $data

     \* @return \App\User

     \*/

    protected function create(array $data)

    {

        $user = User::create([

            'email' => $data['email'],

            'password' => Hash::make($data['password']),

            'oibUser' => $data['oibUser'],

            'imeUser' => $data['imeUser'],

            'prezimeUser' => $data['prezimeUser'],

            'datumRodenja' => $data['datumRodenja'],

        ]);

        $user->save();

        return $user;

    }

}

*Izvor: autor*

Prethodni kod prikazuje kontroler koji se koristi za registraciju. Za izradu ovog koda korišteni su razni tutorijali koji su pronađeni na Internetu. Možemo vidjeti kako funkcija create služi za kreiranje novog korisnika, pritom se spremaju uneseni podatci (od strane korisnika) u varijable, te se sve zajedno sprema pod tablicu Users u bazi podataka.

## 

### 6.2.2. Kreiranje, izmjena, brisanje brodova

Za kreiranje, izmjenu i brisanje, odabrani su brodovi, budući da je i za rute i za putovanja proces za navedeno isti.

Kod 3. Brod.php

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Brod extends Model

{

    protected $fillable=['id', 'nazivBrod', 'opisBrod', 'image'];

}

*Izvor: autor*

Datoteka brod.php sadrži podatke koji su navedeni u prethodnom kodu, ti podatci su isti onima iz baze podataka. Omogućuje komunikaciju s bazom podataka.

Kod 4. BrodController dokument

1. <?php
2. namespace App\Http\Controllers;
3. use App\Brod;
4. use Illuminate\Http\Request;
5. use App\Http\Requests\StoreBrod;
6. use App\Http\Requests\UpdateBrod;
7. class BrodController extends Controller
8. {
9. /\*\*
10. \* Display a listing of the resource.
11. \*
12. \* @return \Illuminate\Http\Response
13. \*/
14. public function index()
15. {
16. return view('cms/brod/index')->with('brodice', Brod::all());
17. }
18. /\*\*
19. \* Show the form for creating a new resource.
20. \*
21. \* @return \Illuminate\Http\Response
22. \*/
23. public function create()
24. {
25. return view('cms.brod.create');
26. }
27. /\*\*
28. \* Store a newly created resource in storage.
29. \*
30. \* @param  \Illuminate\Http\Request  $request
31. \* @return \Illuminate\Http\Response
32. \*/
33. public function store(StoreBrod $request)
34. {
35. $path = $request-> file('image')->store ('image');
36. Brod::create([
37. 'nazivBrod'=>$request->nazivBrod,
38. 'opisBrod'=>$request->opisBrod,
39. 'image'=>$path,
40. ]);
41. return redirect(route('brod.index'));
42. }
43. /\*\*
44. \* Display the specified resource.
45. \*
46. \* @param  int  $id
47. \* @return \Illuminate\Http\Response
48. \*/
49. public function show($id)
50. {
51. //
52. }
53. /\*\*
54. \* Show the form for editing the specified resource.
55. \*
56. \* @param  int  $id
57. \* @return \Illuminate\Http\Response
58. \*/
59. public function edit(Brod $brod)
60. {
61. return view ('cms.brod.create')->with('brod', $brod);
62. }
63. /\*\*
64. \* Update the specified resource in storage.
65. \*
66. \* @param  \Illuminate\Http\Request  $request
67. \* @param  int  $id
68. \* @return \Illuminate\Http\Response
69. \*/
70. public function update(UpdateBrod $request, Brod $brod)
71. {
72. $brod->update([
73. 'nazivBrod'=>$request->nazivBrod,
74. 'opisBrod'=>$request->opisBrod,
75. 'image'=> $request->file('image')->store('image'),
76. ]);
77. return redirect(route('brod.index'));
78. }
79. /\*\*
80. \* Remove the specified resource from storage.
81. \*
82. \* @param  int  $id
83. \* @return \Illuminate\Http\Response
84. \*/
85. public function destroy(Brod $brod)
86. {
87. $brod->delete();
88. return redirect(route('brod.index'));
89. }}

Izvor: autor

U prethodnome kodu nalazi se kontroler. Izrada svih kontrolera izvršena je pomoću slijedećeg reda koda:

php artisan make: controller BrodController –recource

Zahvaljujući naredbi resource, automatski se kreiraju funkcije index, create i edit, koje možemo vidjeti u prethodnome kodu.

U kreiranu funkciju index, dodan je kod kao što prikazuje prethodna slika, dakle, ova funkcija vraća pogled na cms/brod/indeks. Funkciji create proslijeđen je pogled na cms.brod.create. Funkcija store pohranjuje uneseni brod u bazu podataka. Pomoću funkcije edit uređujemo / izmjenjujemo već postojeće podatke u bazi podataka, možemo vidjeti kako vraća pogled na create blade, ali s onim brodom koji je odabran (kako bi ga bilo moguće izmijeniti). Funkcija destroy pomaže nam kod brisanja podataka.

Kod 5. Create blade – brod

@extends('cms.master')

@section('content')

<div class="col-md-16" style="width: 730px;" >

<div class="card card-defatult">

    <div class="card-header">

        {{ isset($brod) ? 'Izmijeni brod' : 'Kreiraj brod'}}

    </div>

    <div class="card-body">

        <form action="{{ isset($brod) ? route('brod.update',  $brod->id) : route('brod.store') }}" enctype="multipart/form-data" method="POST">

            @csrf

            @if(isset($brod))

                @method('PUT')

            @endif

            <div class="form-group">

                <label for="nazivBrod">Naziv</label>

                <input type="text" id="nazivBrod" class="form-control" name="nazivBrod" value="{{ isset($brod) ? $brod->nazivBrod : ''}}">

            </div>

            <div class="form-group">

                <label for="opisBrod">Opis</label>

                <input type="text" id="opisBrod" class="form-control" name="opisBrod"

                value="{{ isset($brod) ? $brod->opisBrod : ''}}">

            </div>

            @if(isset($brod))

            <div class="form-group">

                <img src="/app/public/{{($brod->image)}}" alt="" style="width:150px">

            </div>

            @endif

            <div class="form-group">

                <label for="image">Image</label>

                <input type="file" class="form-control" name="image" id="image">

            </div>

            @include('layouts.errors')

            <div class="form-group">

                <button class="btn btn-success">

                    {{ isset($brod) ? 'Izmijeni brod' : 'Dodaj brod' }}                </button>

            </div>

        </form>

    </div>

</div>

</div>

@endsection

*Izvor: autor*

Prethodni kod prikazuje create blade za brod. Možemo vidjeti kako se prethodni kod poziva kod funkcije create, što je bilo prikazano u kodu prije ovoga. Možemo vidjeti dakle, u create blade kodu, proces koji se izvršava prilikom kreiranja, odnosno izmjene broda. Kako se podatci prosljeđuju, koristi se „PUT“ metoda. Možemo vdjeti kako je postavljena vrijednost polja za „naziv“, „opis“, te „sliku“.

### 6.2.3. Rezervacija putovanja

Kod 6. Rezervacija putovanja

<?php

namespace App\Http\Controllers;

use App\Putovanje;

use Carbon\Carbon;

use App\gostPutovanje;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class GostPutovanjeController extends Controller

{

    public function index()

    {

        return view('welcome')->with('putovanja', Putovanje::all());

    }

    public function pregled(Request $request)

    {

        $datum = $request->datum;

        return view('welcome')->with('putovanja', Putovanje::where('datum', '=', $datum)->get());

    }

  public function rezerviraj($putovanje)

    {

        $userBorn = new Carbon(Auth::user()->datumRodenja);

        $dateToday = new Carbon(now());

        $godine\_gosta = $dateToday->diffInYears($userBorn);

        if($godine\_gosta<12){

            $cijena=Putovanje::find($putovanje)->cijena/2;

        }else{

            $cijena=Putovanje::find($putovanje)->cijena;

        }

        $gost\_id= Auth::user()->id;

        $gost\_putovanje = Putovanje::find($putovanje)->with('gost\_putovanjes')

        ->leftJoin('gost\_putovanjes', 'gost\_putovanjes.idPutovanje', '=', 'putovanjes.id')

        ->where('idGost', '=', $gost\_id)

        ->where('idPutovanje', '=', $putovanje)->get();

        foreach($gost\_putovanje as $gost\_p){

            if($gost\_putovanje){

                $status = "Rezervacija već postoji";

                return view('rezervacija', compact('status'));

            }

        }

            $rezervacija = gostPutovanje::create([

                'idGost' => Auth::user()->id,

                'idPutovanje' => $putovanje,

                'cijenaGosta' => $cijena,

            ]);

            return view('rezervacija', compact('rezervacija'));

        /\*if(gostPutovanje::where('idGost', '=', Auth::user()->id)->count() >0

            && gostPutovanje::where('idPutovanje', '=', Putovanje::find($putovanje)->id)->count() >0){

                $status = "Rezervacija već postoji";

                return view('rezervacija', compact('status'));

            }

        $rezervacija = gostPutovanje::create([

            'idGost' => Auth::user()->id,

            'idPutovanje' => Putovanje::find($putovanje)->id,

            'cijenaGosta' => $cijena,

        ]);

        return view('rezervacija', compact('rezervacija'));\*/

    }

    public function destroy(Putovanje $rezervacija)

    {

        $rezervacija->delete();

        return redirect(route('rezervacija.index'));

    }

    public function pregledsvojihrezervacija()

    {

        $gostputovanja = gostPutovanje::with('gost', 'putovanje')

        ->leftJoin('users', 'users.id', '=', 'gost\_putovanjes.idGost')

        ->where('idGost', '=', Auth::user()->id)->get();

        return view('pregledsvojihputovanja',compact('gostputovanja'));

    }

    public function izbrisirezervaciju($id)

    {

        $putovanje = gostPutovanje::where('idPutovanje','=',$id)

        ->where('idGost','=', Auth::user()->id)->delete();

        return redirect(route('pregledsvojihrezervacija.pregled'));

    }

}

*Izvor: autor*

Prethodni kod prikazuje GostPutovanjeController, u okviru kojega je i definiran proces rezervacije putovanja od strane gosta (ili vlasnika). Možemo, iz prikazanog koda vidjeti kako imamo funkciju rezerviraj u okviru koje se nalazi kod zaslužan za mogućnost registracije. Naišla sam na poteškoće prilikom izrade koda koji bi omogućio da gost s pripadajućim idGosta može rezervirati više putovanja koja imaju različito idPutovanje. Problem koji mi se pojavljivao nalazi se i u prethodnome kodu (dio u komentaru, u funkciji rezerviraj). Spomenuti problem manifestirao se na način da bi gost rezervirao jedno putovanje, te bi mu se prikazala obavijest da je rezervacija uspješno izvršena. No, gost ne bi mogao rezervirati više, osim tog putovanja, niti bilo koje drugo. Spomenuti problem je riješen i prikazan je u okviru funkcije rezervacija. Ono što se dešava je sljedeće. Dohvaća se id od autentificiranog korisnika, te se sprema u varijablu $gost\_id, također se podatci spremaju u $gost\_putovanje – prvo pronađe putovanje gdje je idGost=login gost id, te idPutovanje=idPutovanje – kada se klikne na gumb „Odaberi“. Zatim u okviru petlje foreach, true je, ukoliko postoji neki podatak od onoga što smo u prijašnjem dijelu koda tražili. Dakle, ako ima podataka u bazi, javiti će da „rezervacija već postoji“, u suprotnome izvršiti će se dio koda ispod s varijablom $rezervacija, gdje će se u istu spremiti novo kreirana rezervacija.

# 7. Korištenje aplikacije

Upute za preuzimanje, otvaranje i korištenje aplikacije, kao i sve verzije aplikacije nalaze se na sljedećoj poveznici: <https://github.com/laurakolonic/iooa2020_turisticki_izleti_brodicama>.

# 8.Zaključak

Budući da mi programiranje koristeći Laravel framework nije bilo previše poznato prilikom samog početka izrade projekta, mogu reći kako su mi razni tutorijali na Internetu, kao i dokumentacija Laravela zaista bili od koristi.

Prilikom razvoja projekta, naišla sam na nekoliko poteškoća i nejasnoća, no ništa što se nije uspjelo riješiti. Također je, proces jednog dijela u okviru kojega mi se i javio problem, opisan u prethodnim dijelovima ove dokumentacije.

# LITERATURA

Laravel – dokumentacija: <https://laravel.com/docs/7.x/installation>