

**WEB PROGRAMIRANJE PHP2**

**DOKUMENTACIJA PROJEKTA**

Student:

Nevena Đaković 22/15

Beograd 2018.

SADRŽAJ

Table of Contents

[1.Opis funkcionalnosti 1](#_Toc510880473)

[2. Skica struktura stranica 1](#_Toc510880474)

[3. Dijagram baze podataka 14](#_Toc510880475)

[3.MVC Organizacija 15](#_Toc510880476)

[Kontroleri 15](#_Toc510880477)

[Modeli-eloqvent 17](#_Toc510880478)

[View 18](#_Toc510880479)

[4. javaScript kod 19](#_Toc510880480)

[5. PHP kod 24](#_Toc510880481)

# 1.Opis funkcionalnosti

|  |  |
| --- | --- |
| **Koriščeni jezici** | Html 5, CSS, PHP oop, laravel, JavaScript(Jquery) |
| **Korišćeni HTML šabloni (framework-ci)** | https://getbootstrap.com/ |
| **Koriščene biblioteke i njihovi url-ovi odakle ste ih preuzeli** | https://code.jquery.com/jquery-3.3.1.slim.min.js |
| **Korišćene biblioteke iz framework-a** | Laravel |
| **Korišćene php biblioteke van fremework-a** |  |
| **Koje funkcionalnosti su realizove putem AJAX-a** | Pregled pitanja i slika(stranicenje)  Anketa, Like, Unlike |
| **Navesti lokacije gde je napisan AJAX kod (fajlovi ili delovi html stranice)** | /js/js.js |
| **Navesti lokacije gde je napisan sopstveni javaScript kod (fajlovi ili delovi html stranice)** | /js/js.js |
| **URL sajta/aplikacije** | https://nevenadjakovic996.000webhostapp.com/ |
| **Pristupni parametri** | [admin@gmail.com](mailto:admin@gmail.com)  admin@gmail.com |

# 2. Skica struktura stranica

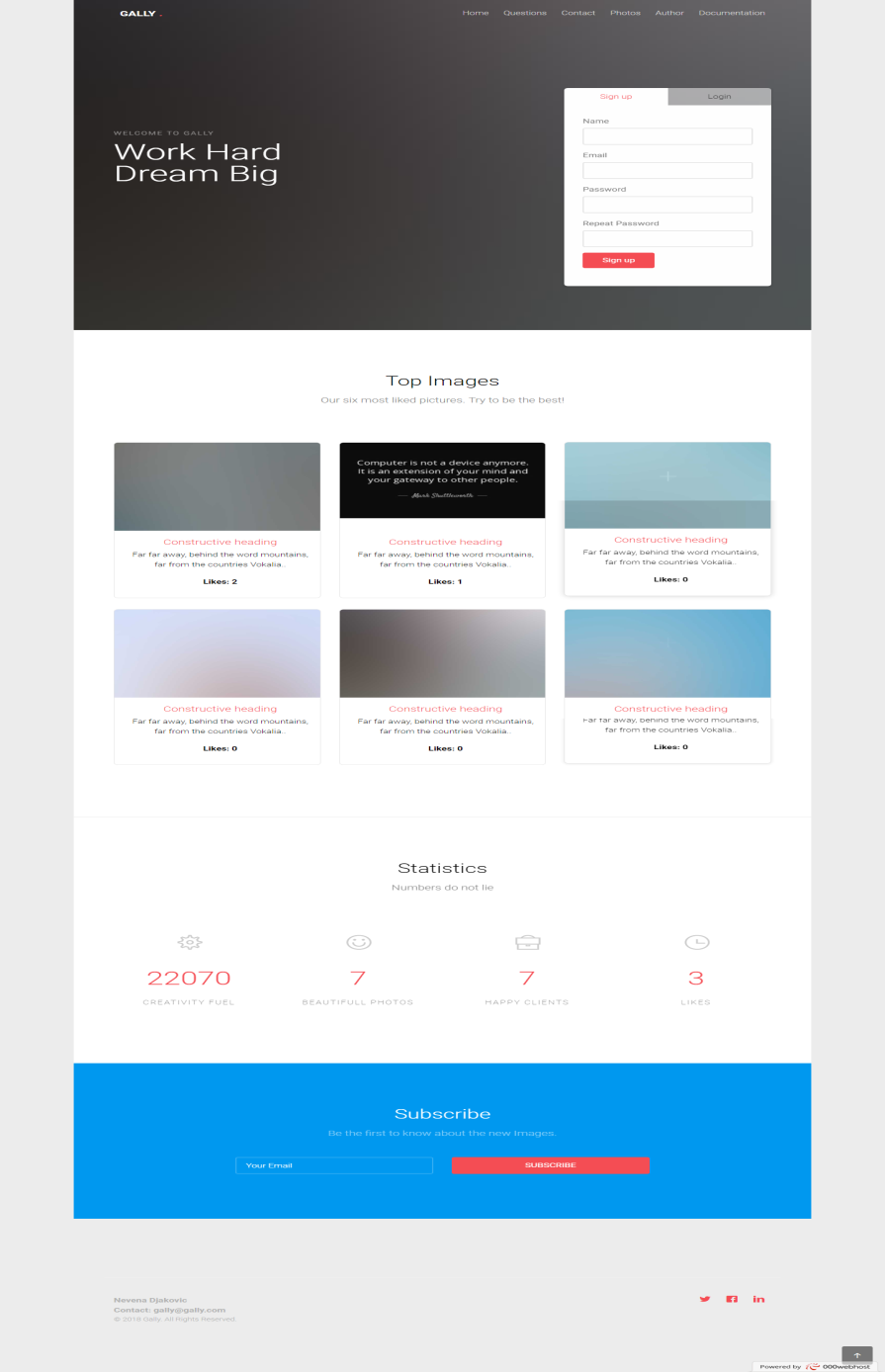
Početna stranica,

- korisnik ima opciju za **logovanje** i **registraciju** uz proveru regularnih izraza i prikaz obaveštenja.

-prikazano je prvih 6 slika koje imaju najviše lajkova. Klikom na njih otvara se **galerija.**

**-**prikazani su podaci iz baze o broju korisnika, broju slika i ukupnom broju lajkova.

-opcija za subscribe, uz proveru regularnih izraza.

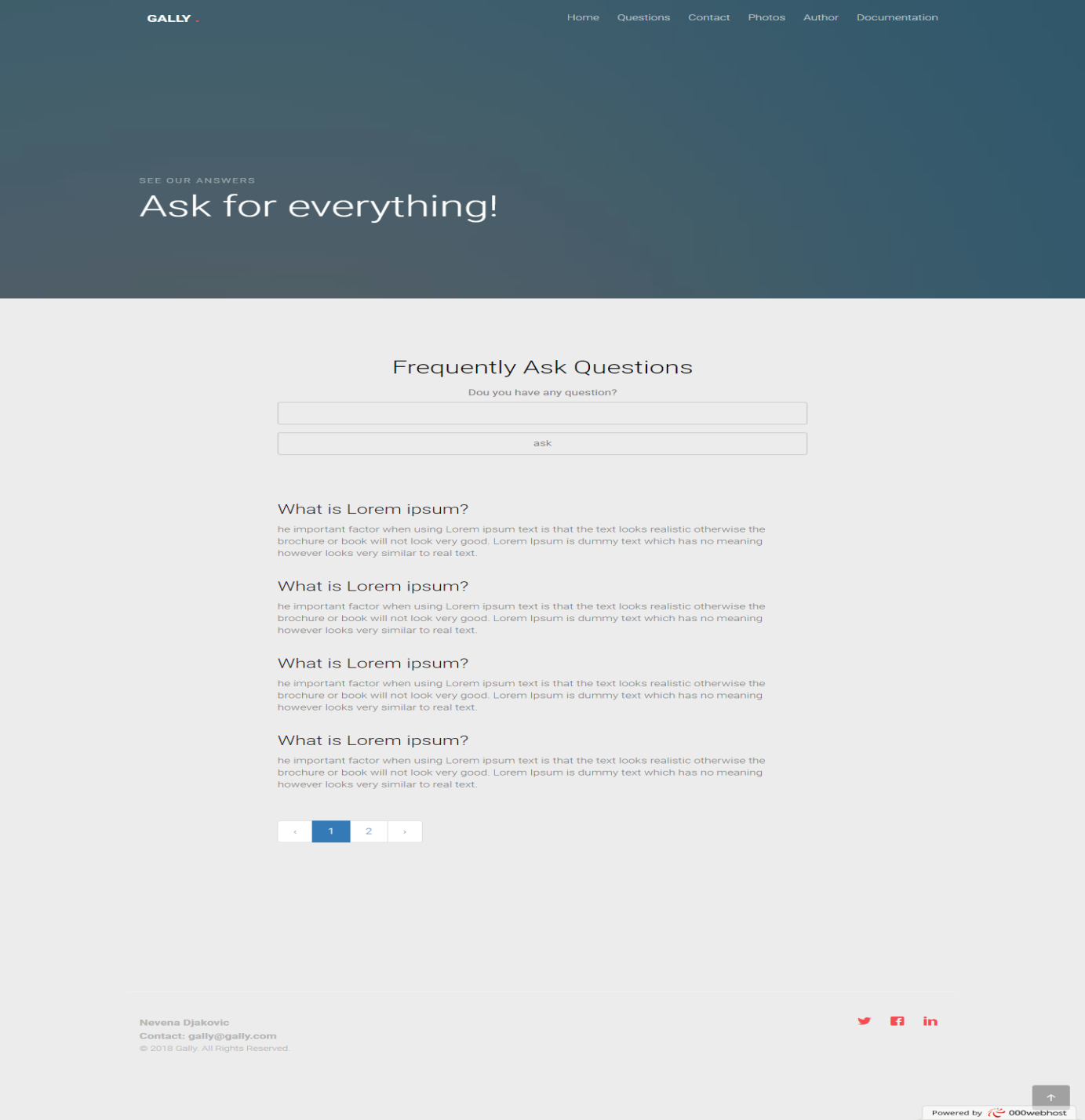


/questions

-neulogovan korisnik može da postavi pitanje.

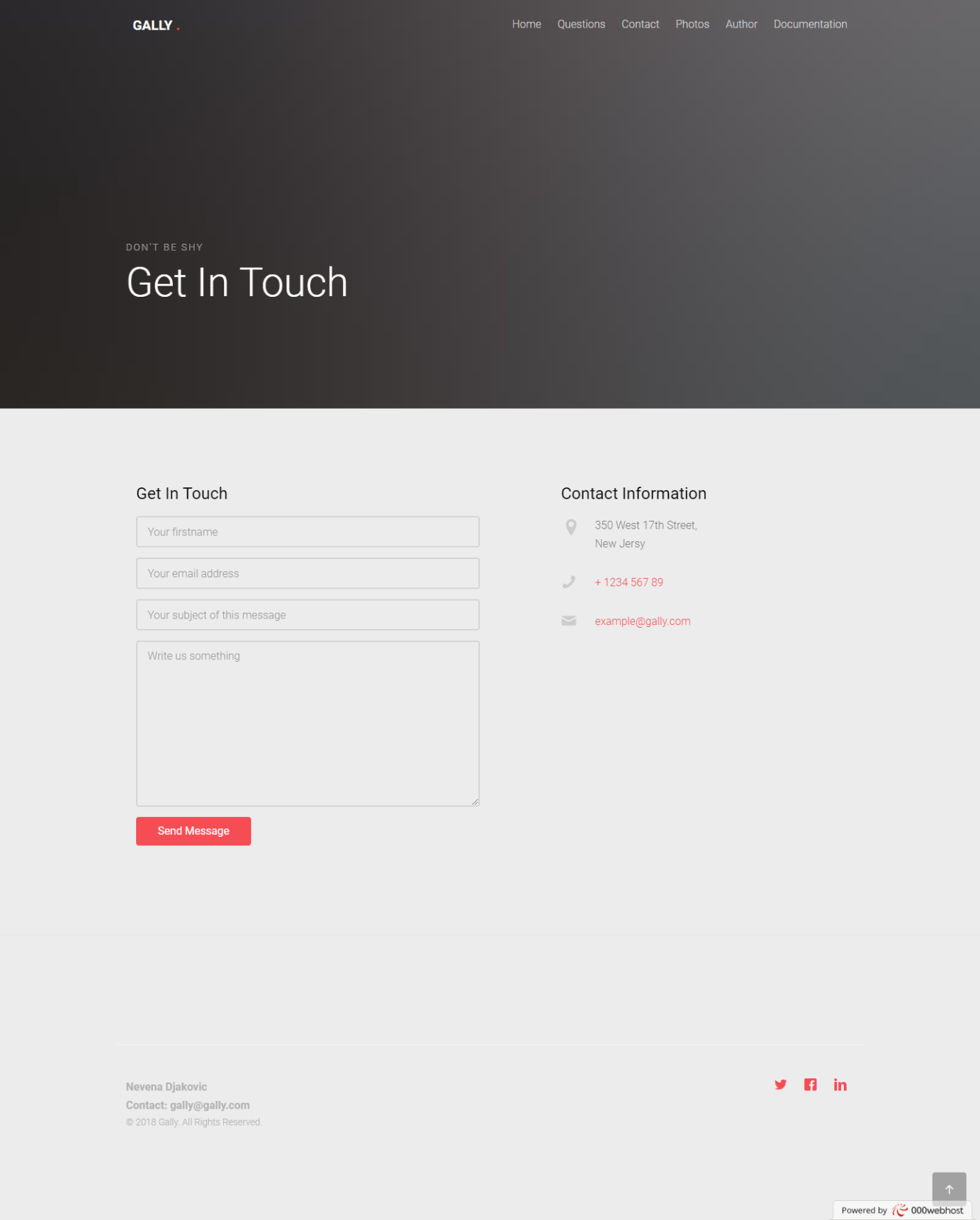
-prikaz pitanja na koje je admin dao odgovor

-**straničenje putem ajaxa**



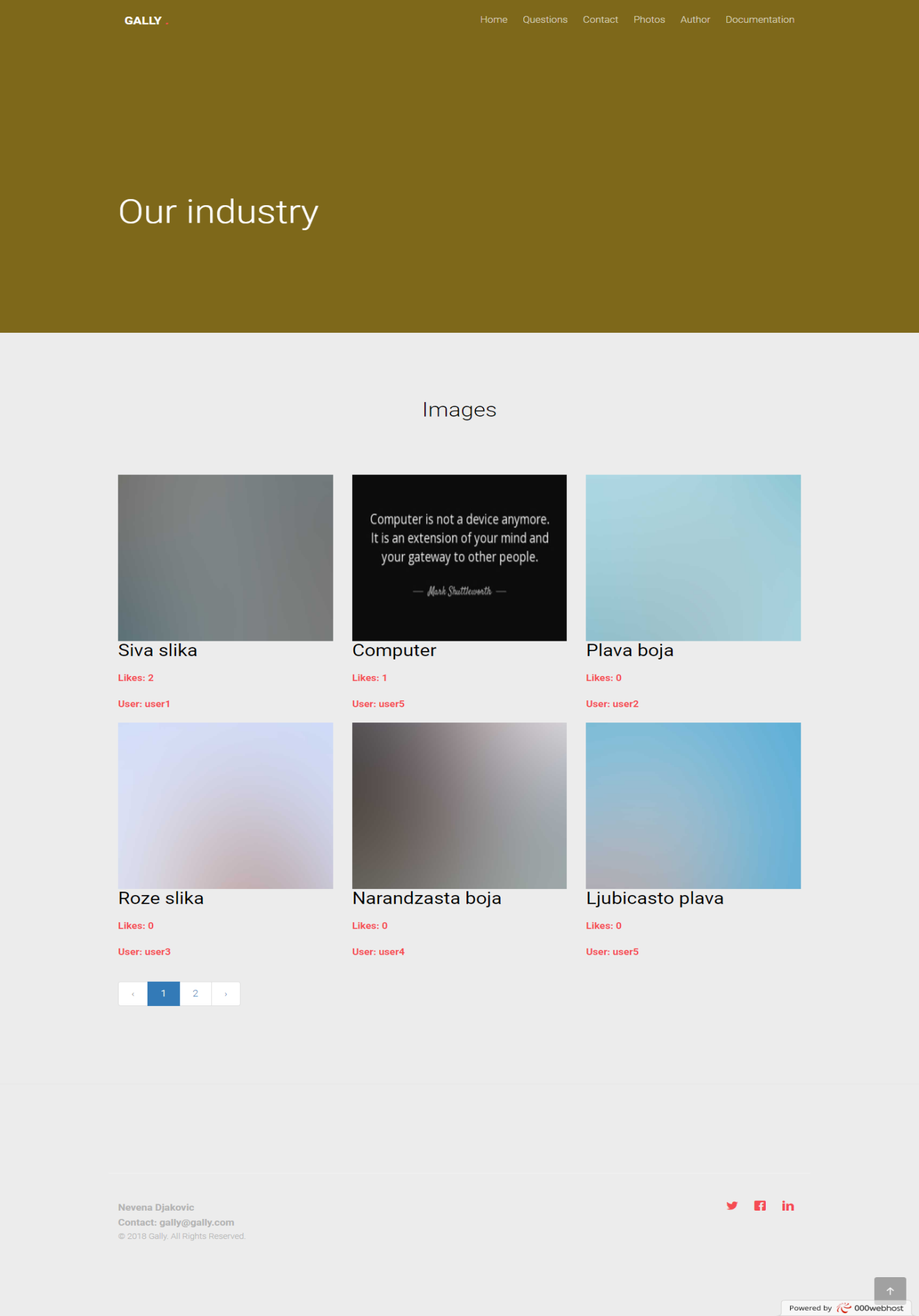
/contact

-neulogovan korisnik kao i ulogovan može poslati podatke za kontakt.

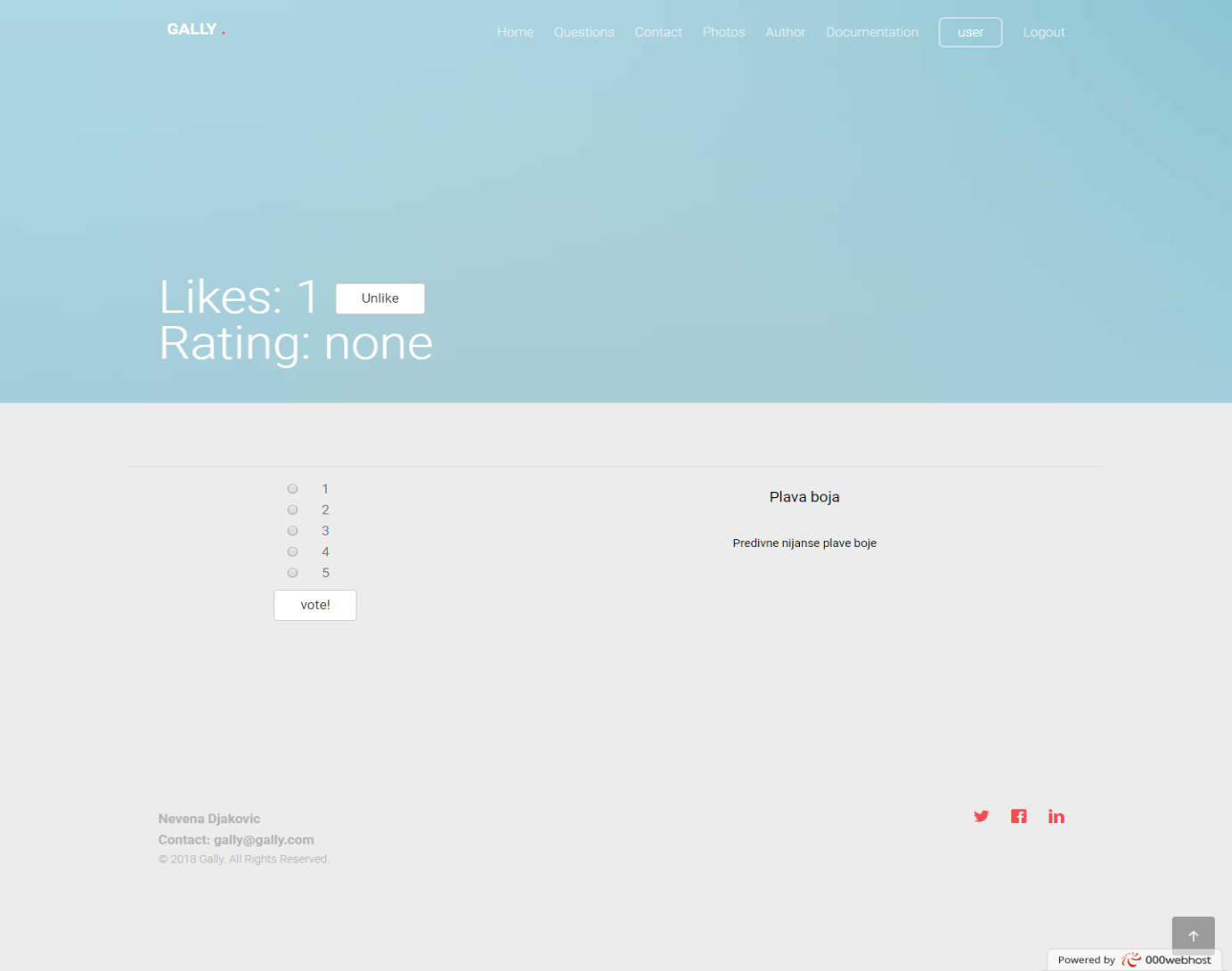
-provera regularnih izraza na **klijentskoj i serverskoj strani**.

/image

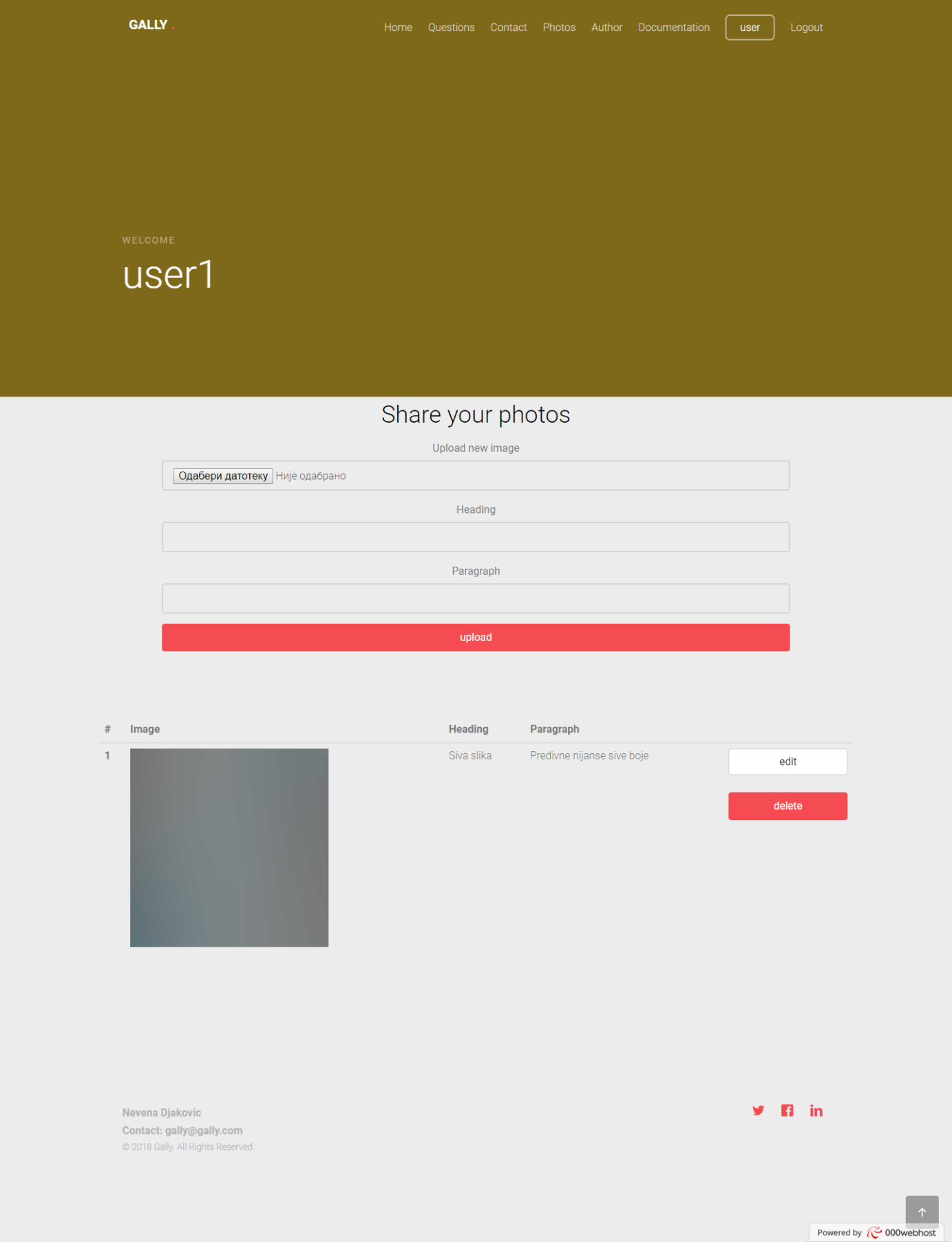
Prikaz svih slika uz paginaciju.



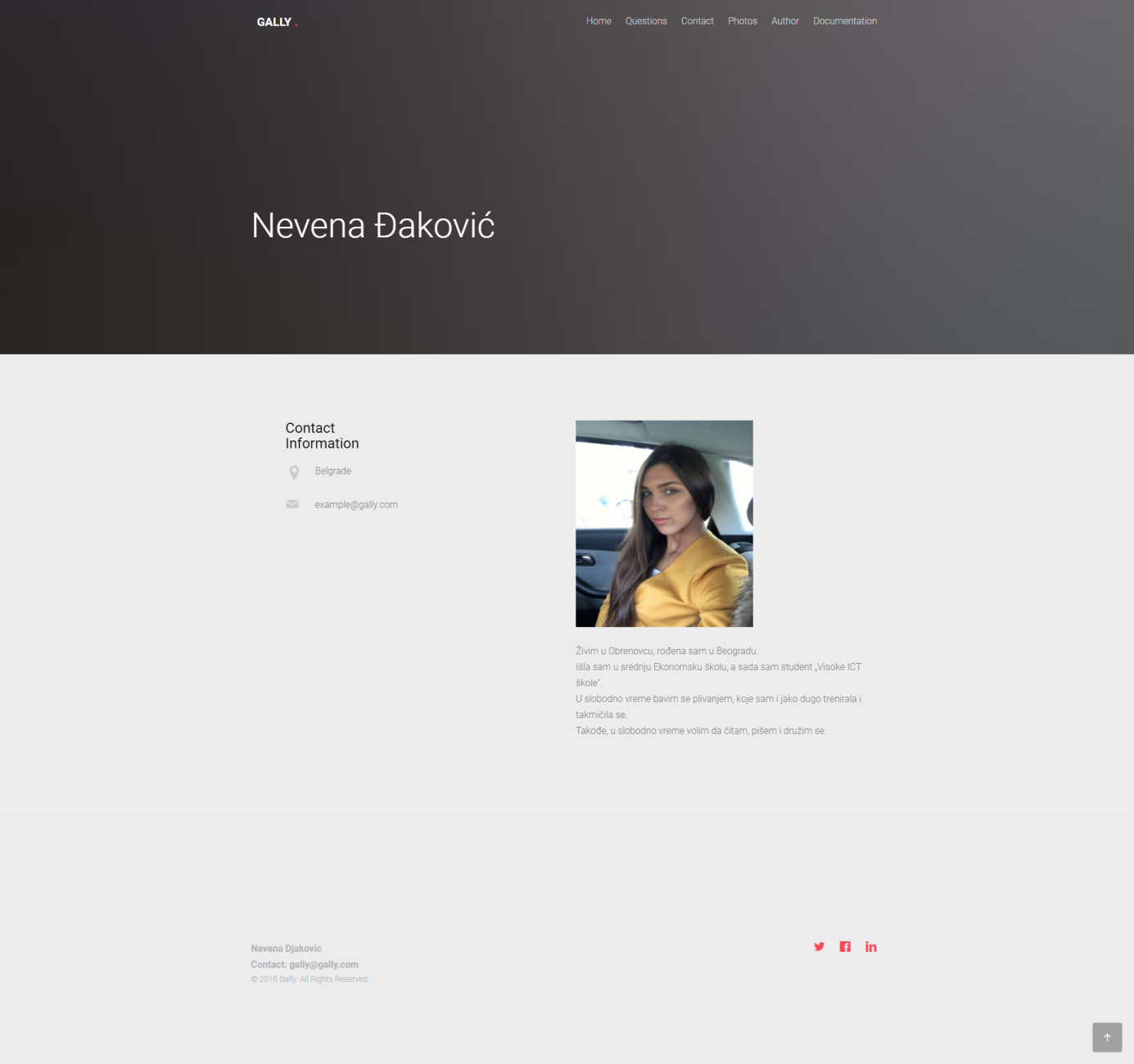
/image/{id}

-ulogovan korisnik ima opciju za lajkovanje i davanje ocene za anketu. Ukoliko je vec ocenio, nema opciju za ponovno ocenjivanje, odnosno dobija obaveštenje. Lajk može unlajkovati.  
Sve je odrađeno korišćenjem ajaxa.

/user

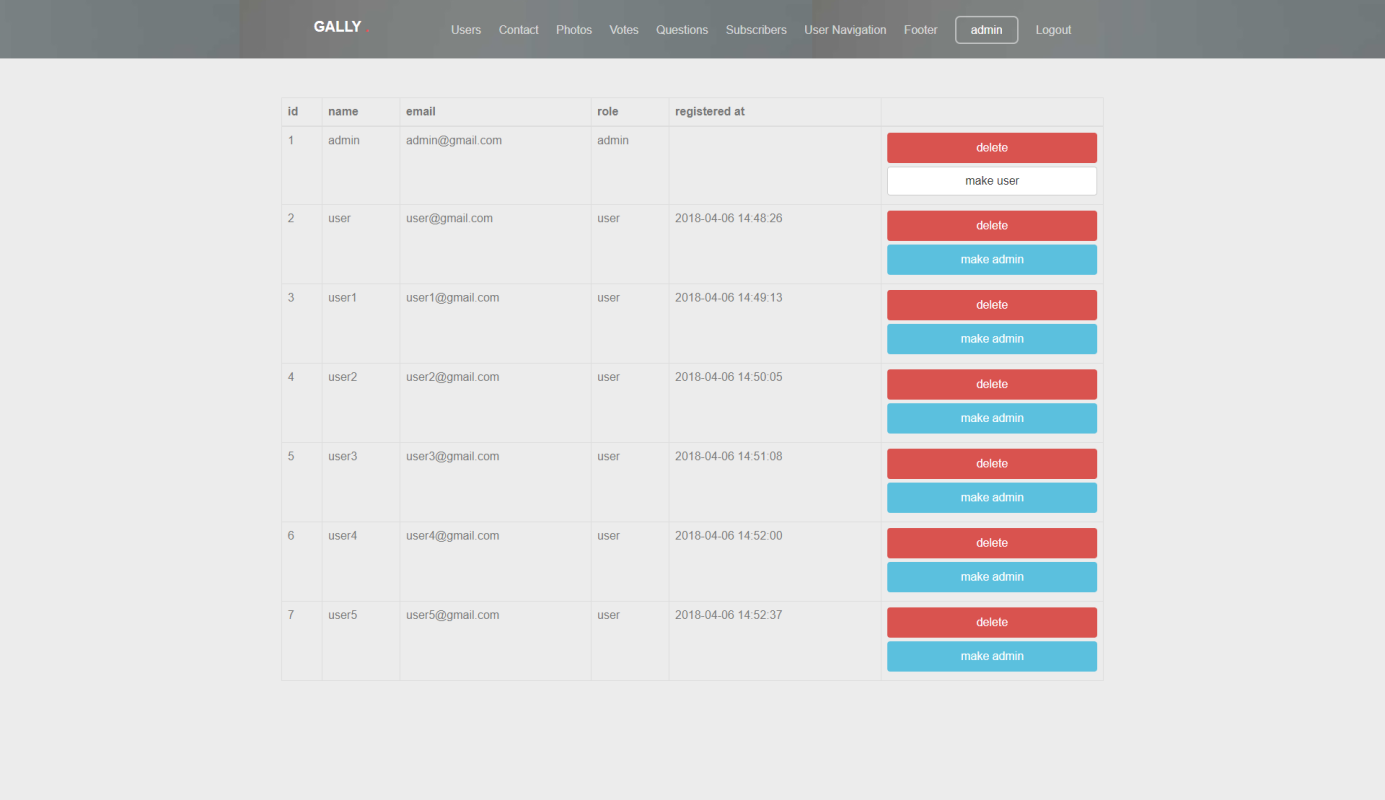
-korisnička strana  
-korisnik ima opciju da dodaje nove postove, da edituje postojeće I da ih briše takođe.

/author



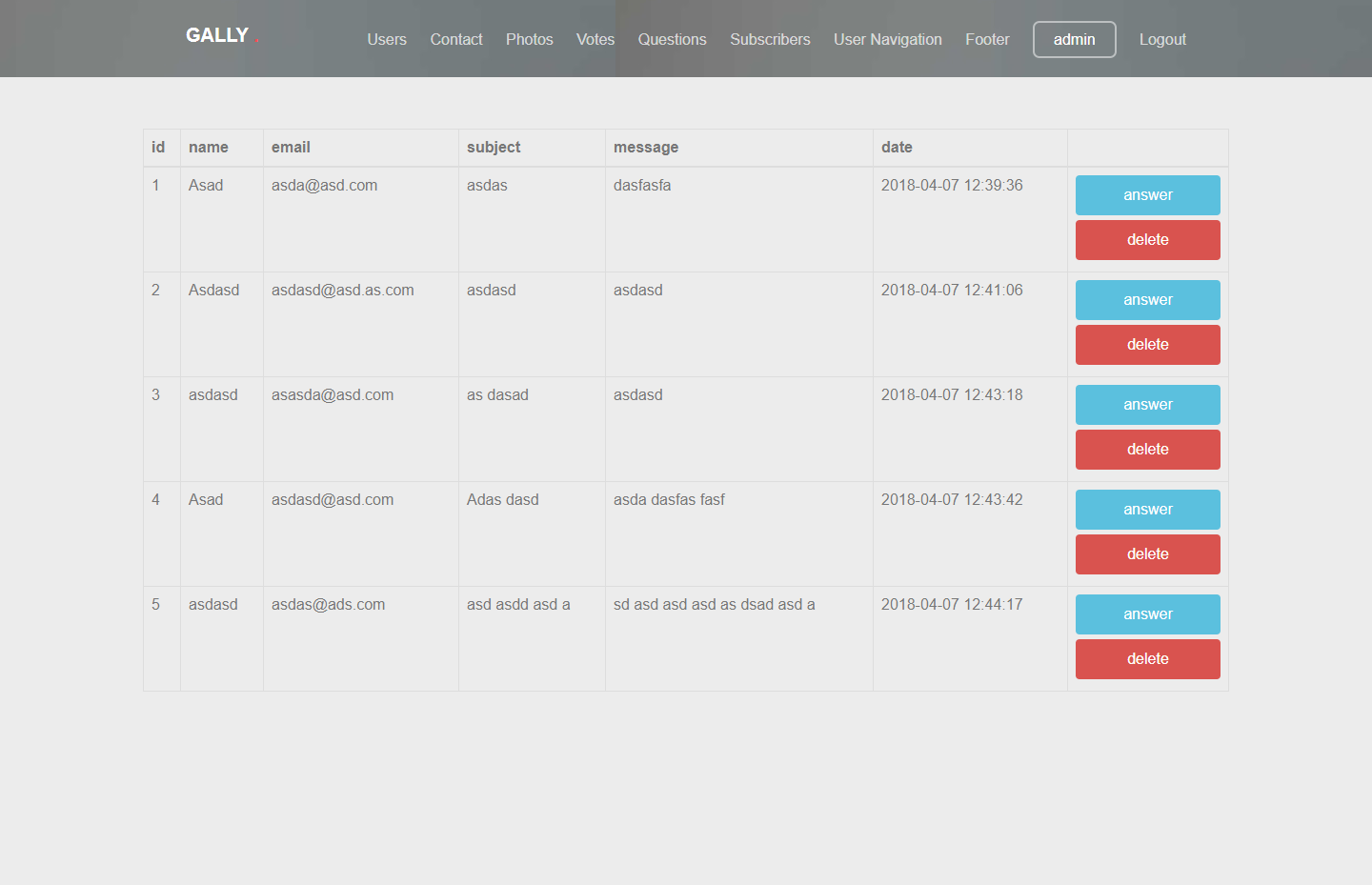
/admin/users

-prikaz korisnika, opcije za njihovo brisanje i dodavanje ima privilegije da budu admini, takođe i brisanje privilegije.

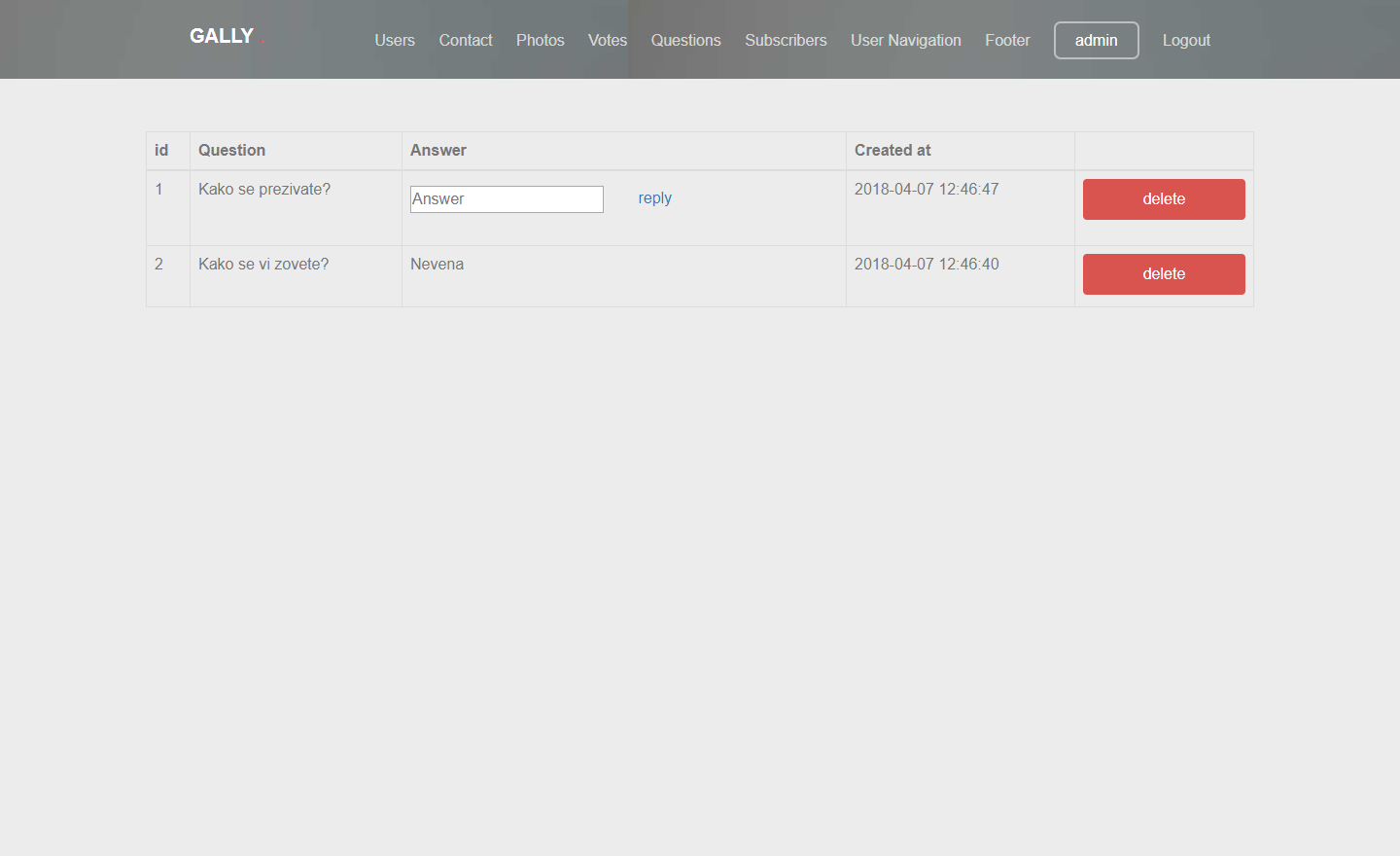


/admin/contact

-prikaz poruka, opcija za njihovo brisanje i odgovaranje

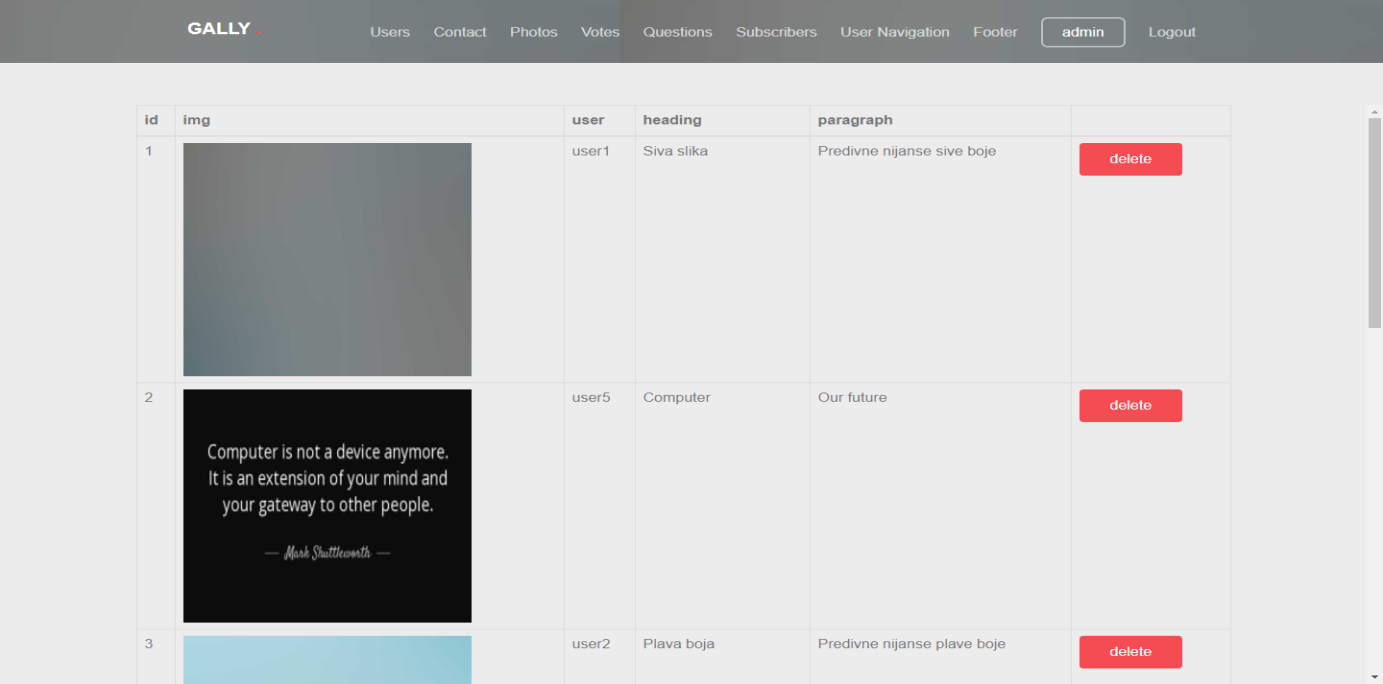


/admin/questions

-prikaz pitanja, opcija za odgovr i brisanje

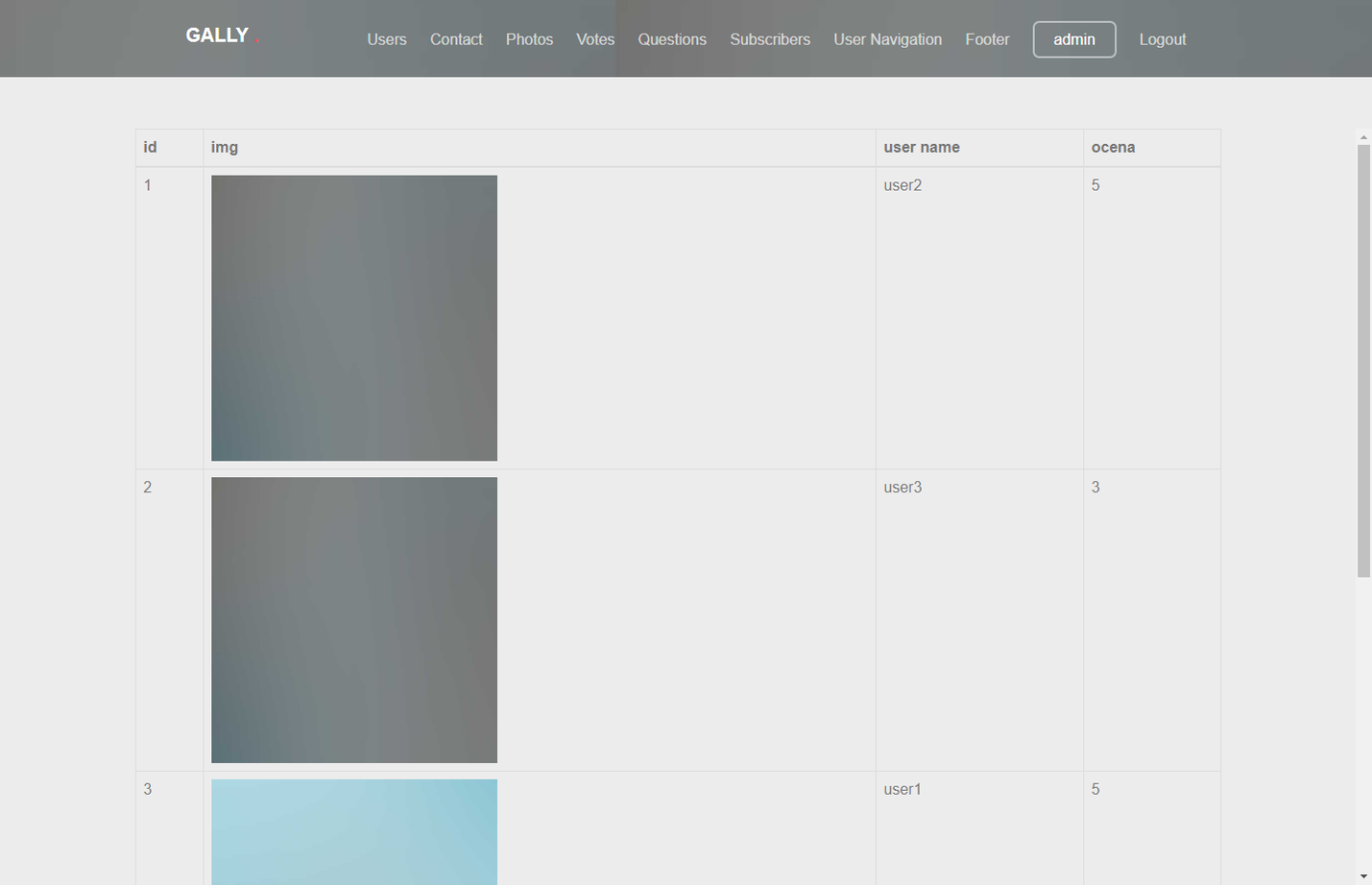
/admin/image

-prikaz svih postova i opcija za brisanje



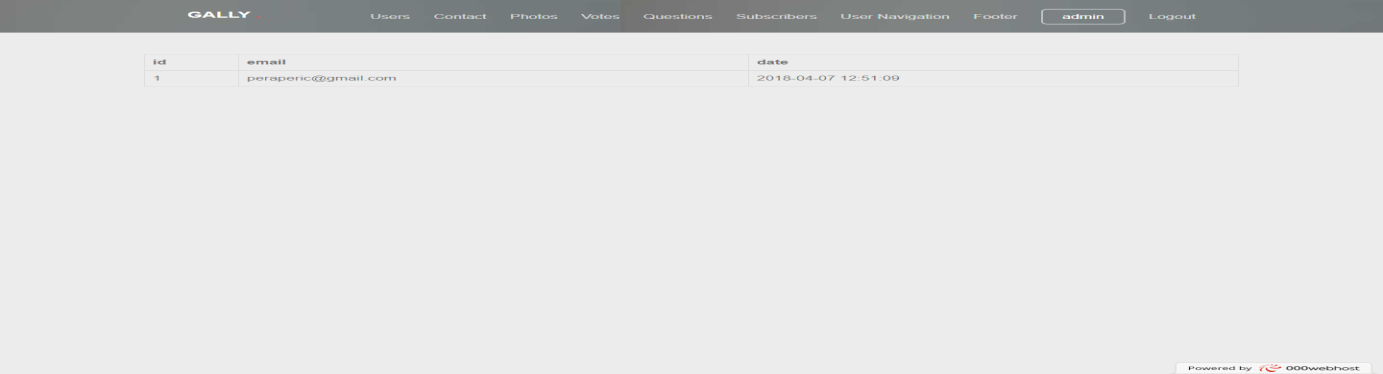
/admin/vote

-prikaz svih ocena

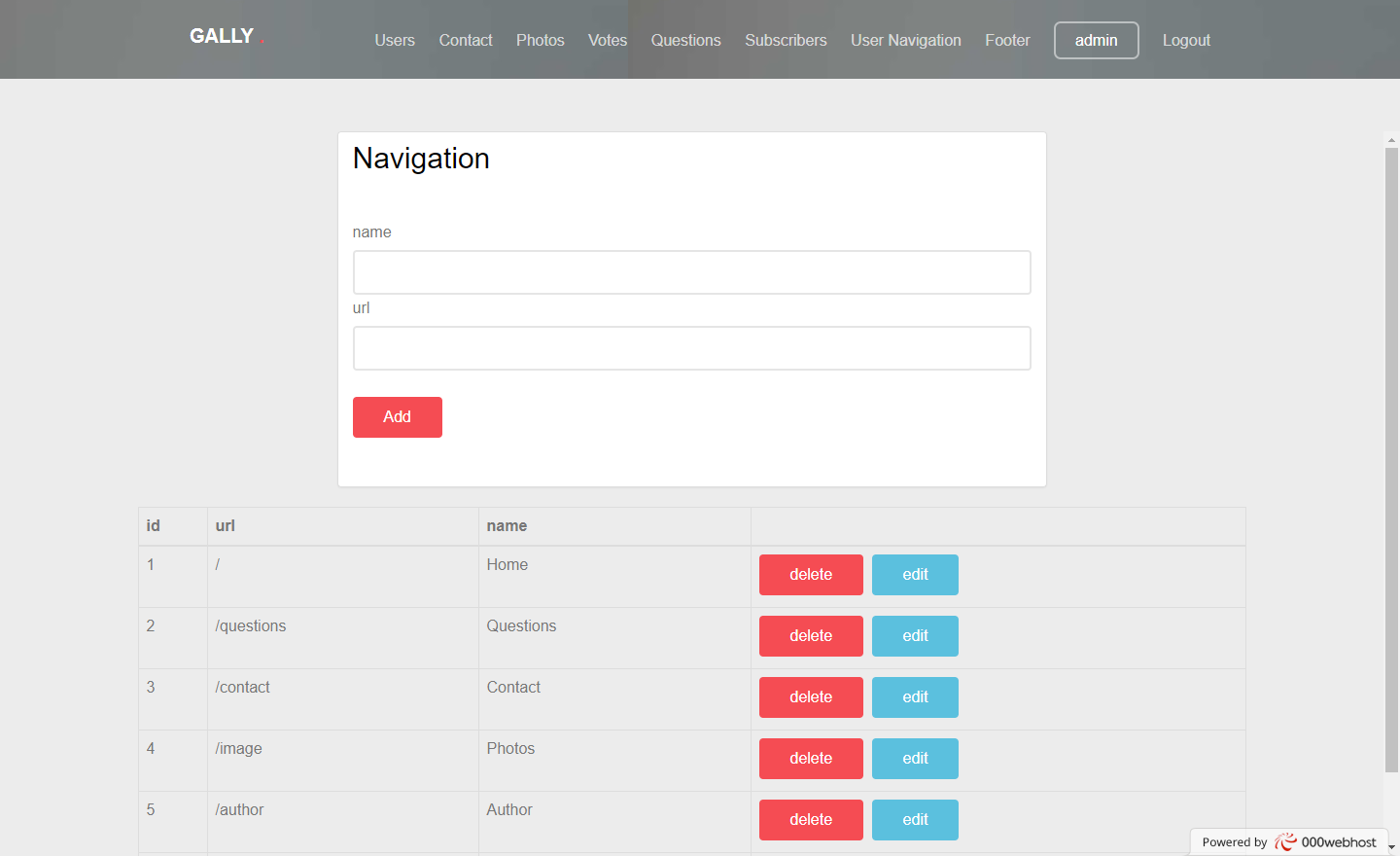


/admin/sub

-prikaz subscriba

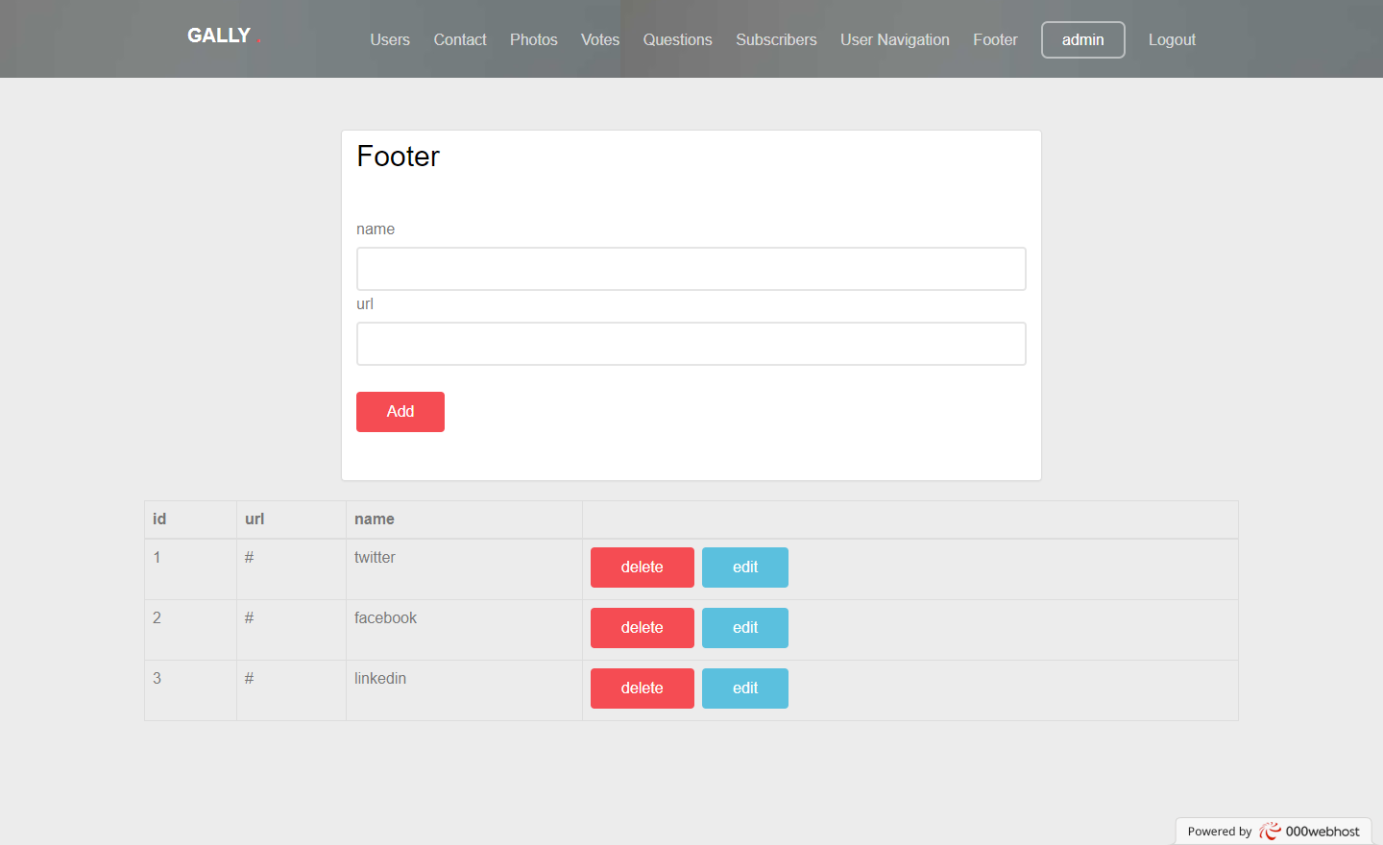


/admin/nav

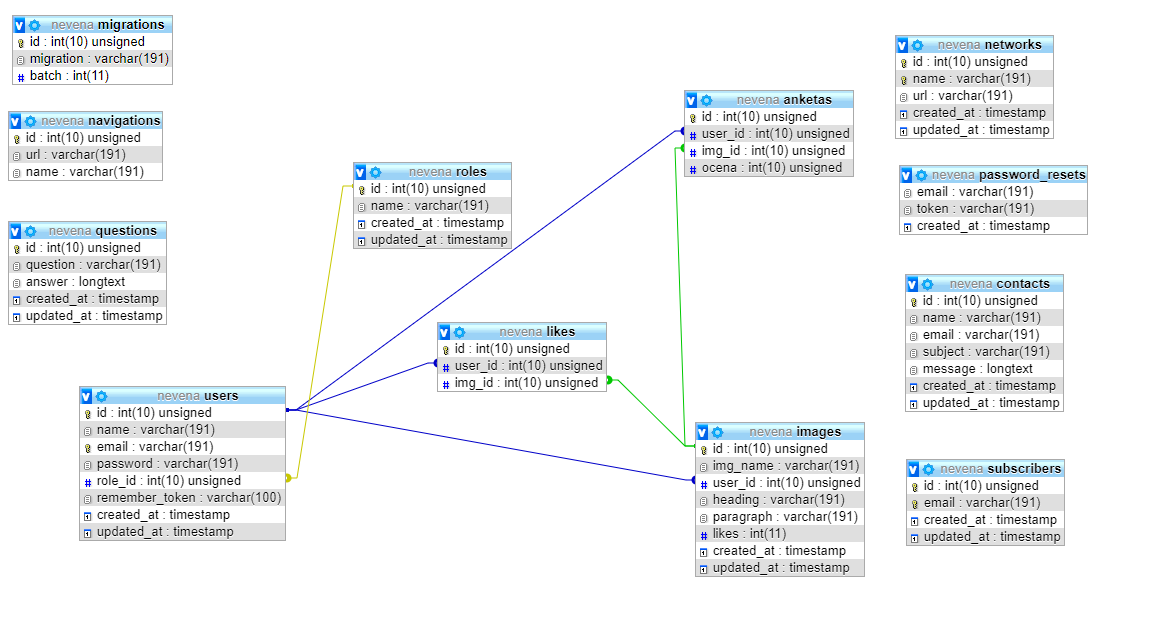
-prikaz korisnickog menija, opcija za brisanje i za editovanje

/admin/footer

-dodavanje, brisanje i editovanje mreza u futeru



# 3. Dijagram baze podataka



# 3.MVC Organizacija

# Kontroleri

|  |
| --- |
| BaseController |
| \_\_construct() |
| getMenu() |
| getNetworks() |
| getSixImages() |
| getLikes() |
| getNumberOfUsers() |
| getNumberOfImages() |
| getData() |
| getUserInstance() |
| getFooterInstance() |
| getNavInstance() |

|  |
| --- |
| ContactController |
| \_\_construct() |
| index() |
| store(Request $request) |
| show() |
| destroy($id) |

|  |
| --- |
| HomeController |
| \_\_construct() |
| autor() |
| index() |
| dokumentacija() |
| subscribe(Request $request) |
| show() |
| navbar() |
| navdelete($id) |
| navedit($id) |
| navupdate(Request $request) |
| footer() |
| footerdelete($id) |
| footeredit($id) |
| footerupdate(Request $request) |
| footerstore(Request $request) |
| navstore(Request $request |

|  |
| --- |
| ImageController |
| \_\_construct() |
| user(Request $request) |
| index(Request $request) |
| create(Request $request) |
| store(Request $request) |
| show(Request $request ,$id) |
| edit(Request $request,$id) |
| update(Request $request) |
| destroy(Request $request,$id) |
| destroyimgadmin(Request $request,$id) |
| like(Request $request) |
| unlike(Request $request) |
| showvote() |

|  |
| --- |
| LoginController |
| \_\_construct() |
| login(Request $request) |
| logout() |
| register(Request $request) |
| users() |
| destroy($id) |
| role($id) |
| roleuser($id) |

|  |
| --- |
| QuestionController |
| \_\_construct() |
| index(Request $request) |
| create(Request $request) |
| store(Request $request) |
| destroy($id) |

# Modeli-eloqvent

|  |
| --- |
| Anketa |
| Images() |
| Users() |

|  |
| --- |
| Contact |

|  |
| --- |
| Image |
| Users() |

|  |
| --- |
| Like |

|  |
| --- |
| Navigation |

|  |
| --- |
| Networks |

|  |
| --- |
| Question |

|  |
| --- |
| Role |

|  |
| --- |
| Subscriber |

|  |
| --- |
| User |
| Roles() |

# 

# View

|  |
| --- |
| admin |
| contact |
| editfooter |
| editnav |
| footer |
| home |
| navbar |
| photos |
| questions |
| subscribe |
| votes |

|  |
| --- |
| ajax |
| anketa |
| photos |
| question |

|  |
| --- |
| components |
| adminnav |
| alerts |
| footer |
| navigation |

|  |
| --- |
| layout |
| Admin |
| home |

|  |
| --- |
| unautehnicated |
| autor |
| contact |
| image |
| index |
| photos |
| questions |

|  |
| --- |
| user |
| edit |
| profile |

# 4. javaScript kod

Js.js

$('#questions').on('click','.pagination a', function (e) {

e.preventDefault();

var url=$(this).attr('href').split('page=')[1];

$.ajax({

url: "questions?page="+url,

success: function (res) {

$('#questions').html(res);

},

error: function (err) {

console.log(err);

}

})

});

$('#photos').on('click','.pagination a', function (e) {

e.preventDefault();

var url=$(this).attr('href').split('page=')[1];

$.ajax({

url: "image?page="+url,

success: function (res) {

$('#photos').html(res);

},

error: function (err) {

console.log(err);

} })});

function proveracontact() {

var x=document.contactform;

var greske=new Array();

var name= x.name.value;

var email= x.email.value;

var subject= x.subject.value;

var message=x.message.value;

var rename=/[\w]{3,30}/;

var reemail=/^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/;

var resubject=/^\w+(\s+\w+)\*$/;

if(!name.match(rename)){

greske.push("Name myst be between 3 and 30 caracters");

}

if(!email.match(reemail)){

greske.push("Mail must be valid");

}

if(!subject.match(resubject)){

greske.push('Subject fild is invalid(only leters)');

}

if(message == ""){

greske.push('Message field is required');

}

if(greske.length == 0){

return true; }

else {

alert(greske);

return false; }}

function anketa() {

var ocena=$('input[name=radio1]:checked', '#anketa').val()

var imgid=$('#img\_id').val();

if(ocena !== undefined){

$.ajax({

headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

},

method: "post",

url: baseUrl + "/vote",

data:{ocena:ocena, imgid:imgid},

success: function (res) {

console.log(res);

if(res == 0){

alert("You have alredy voted");

}

else{

$('#rating').text(res);

}}

});

}

else alert("please select");

}

function like() {

var imgid=$('#img\_id').val();

brojlajkova++;

$.ajax({

headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

},

method: "post",

url: baseUrl + "/user/like",

data:{imgid:imgid},

success: function (res) {

if(res == 1){

$('#imglikes').html('<button class="btn btn-default" onclick="unlike()">Unlike</button>');

$('#numberoflikes').text(brojlajkova);

}

else{

alert('try again later');

} } });}

function unlike() {

var imgid=$('#img\_id').val();

brojlajkova--;

$.ajax({

headers: {

'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')

},

method: "post",

url: baseUrl + "/user/unlike",

data:{imgid:imgid},

success: function (res) {

if(res == 1){

$('#imglikes').html('<button class="btn btn-default" onclick="like()">Like </button>');

$('#numberoflikes').text(brojlajkova); }

else{

alert('try again later');

}

}

});

}

# 5. PHP kod

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Navigation;

use App\Networks;

use App\Image;

use App\User;

class BaseController extends Controller

{

private $data;

private $menu;

private $networks;

private $imageInstance;

private $userInstance;

public function \_\_construct()

{

$this->menu=new Navigation();

$this->networks=new Networks();

$this->imageInstance=new Image();

$this->userInstance=new User();

$this->data['menu']=$this->getMenu();

$this->data['network']=$this->getNetworks();

$this->data['images']=$this->getSixImages();

$this->data['likes']=$this->getLikes();

$this->data['numberofusers']=$this->getNumberOfUsers();

$this->data['numberofimages']=$this->getNumberOfImages(); }

public function getMenu(){

return $data['menu']=$this->menu->get();

}

public function getNetworks(){

return $data['network']=$this->networks->get();

}

public function getSixImages(){

return $this->imageInstance->with('users')->orderBy('likes', 'DESC')->take(6)->get();

}

public function getLikes(){

return $this->imageInstance->sum('likes');

}

public function getNumberOfUsers(){

return $this->userInstance->get()->count();

}

public function getNumberOfImages(){

return $this->imageInstance->get()->count(); }

public function getData(){

return $this->data; }

public function getUserInstance(){

return $this->userInstance; }

public function getFooterInstance(){

return $this->networks; }

public function getNavInstance(){

return $this->menu;

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Contact;

class ContactController extends BaseController

{

private $contactInstance;

public function \_\_construct()

{

parent::\_\_construct();

$this->contactInstance=new Contact();

}

public function index() {

return view('unautehnicated.contact',['data'=>$this->getData()]);

}

public function store(Request $request) {

$request->validate([

'name' => 'required|min:3',

'email' => 'email|required',

'subject' =>'required|min:3',

'message'=>'required|min:5',

]);

try{

$this->contactInstance->name= $request->name;

$this->contactInstance->email= $request->email;

$this->contactInstance->subject=$request->subject;

$this->contactInstance->message=$request->message;

$this->contactInstance->save();

return back();

}catch (\Exception $ex){

\Log::error('greska prilikom slanja contact poruke' .$ex->getMessage() ." Name:". $request->name . " Message:" .$request->message);

return back(); }

}

public function show() {

$contact=$this->contactInstance->get();

return view('admin.contact', ['contact'=>$contact]);

}

public function destroy($id)

{

try{

$this->contactInstance->where('id', $id)->delete();

return back();

}catch (\Exception $exception){

\Log::error('greska prilikom brisanja konakta iz admin pane;la');

return back();

}

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Subscriber;

class HomeController extends BaseController

{

private $subscribeInstance;

public function \_\_construct()

{

parent::\_\_construct();

$this->subscribeInstance=new Subscriber();

}

public function autor(){

return view('unautehnicated.autor',['data' => $this->getData()]);

}

public function index(){

return view('unautehnicated.index',['data' => $this->getData()]);

}

public function dokumentacija(){

$headers = array(

'Content-Type: application/pdf',

);

return response()->download(public\_path('dokumentacija.pdf'), 'dokumentacija.pdf', $headers); }

public function subscribe(Request $request){

$request->validate([

'subemail' => 'required|email|unique:subscribers,email',

]);

try{

$this->subscribeInstance->email=$request->subemail;

$this->subscribeInstance->save();

return back()->with('success', 'Thank you for subscribe!');

}catch (\Exception $ex){

\Log::error('Greska prilikom subscriba'. $ex->getMessage());

return back()->with('fail', 'We are sorry, try again latter to subscribe');

}

}

public function show(){

$sub=$this->subscribeInstance->get();

return view('admin.subscribe', ['sub' => $sub]);

}

public function navbar(){

return view('admin.navbar', ['nav' => $this->getMenu()]);

}

public function navdelete($id){

try{

$this->getNavInstance()->where('id', $id)->delete();

return back();

}catch (\Exception $exception){

\Log::error('greska prilikom brisanja navigacije!'.$exception->getMessage());

// dd($exception->getMessage());

return back();

}

}

public function navedit($id){

try{

$nav=$this->getNavInstance()->where('id', $id)->get();

return view('admin.editnav',['nav'=>$nav, 'data'=>$this->getData()]);

}catch (\Exception $exception){

\Log::error('greska prilikom pozivanja edita navigacije!'.$exception->getMessage());

// dd($exception->getMessage());

return back();

}

}

public function navupdate(Request $request){

try{

$this->getNavInstance()->where('id', $request->id)->update(['url' => $request->url, 'name'=> $request->name]);

return redirect('/admin/nav');

}catch (\Exception $exception){

\Log::error('greska prilikom pozivanja update navigacije!'.$exception->getMessage());

return back();

}

}

public function footer(){

return view('admin.footer', ['footer' => $this->getNetworks()]);

}

public function footerdelete($id){

try{

$this->getFooterInstance()->where('id', $id)->delete();

return back();

}catch (\Exception $exception){

\Log::error('greska prilikom brisanja futera!'.$exception->getMessage());

// dd($exception->getMessage());

return back();

}

}

public function footeredit($id){

try{

$nav=$this->getFooterInstance()->where('id', $id)->get();

return view('admin.editfooter',['nav'=>$nav, 'data'=>$this->getData()]);

}catch (\Exception $exception){

\Log::error('greska prilikom pozivanja edita futera!'.$exception->getMessage());

// dd($exception->getMessage());

return back();

}

}

public function footerupdate(Request $request){

try{

$this->getFooterInstance()->where('id', $request->id)->update(['url' => $request->url, 'name'=> $request->name]);

return redirect('/admin/footer');

}catch (\Exception $exception){

\Log::error('greska prilikom pozivanja update futera!'.$exception->getMessage());

return back(); } }

public function footerstore(Request $request){

try{

$this->getFooterInstance()->insert(['url' => $request->url, 'name'=> $request->name]);

return back();

}catch (\Exception $exception){

\Log::error('greska prilikom dodavanja u futer!'.$exception->getMessage());

return back();

}

}

public function navstore(Request $request){

try{

$this->getNavInstance()->url = $request->url;

$this->getNavInstance()->name = $request->name;

$this->getNavInstance()->save();

return back();

}catch (\Exception $exception){

\Log::error('greska prilikom dodavanja u navigacioni meni!'.$exception->getMessage());

return back();

}

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Image;

use App\Anketa;

Use App\Like;

class ImageController extends BaseController

{

private $imageInstance;

private $anketaInstance;

private $likeInstance;

public function \_\_construct()

{

parent::\_\_construct();

$this->imageInstance=new Image();

$this->anketaInstance=new Anketa();

$this->likeInstance=new Like();

}

public function user(Request $request){

$sesija=$request->session()->get('user');

$images=$this->imageInstance->where('user\_id', $sesija[0]['id'])->get();

return view('user.profile',['data' => $this->getData(), 'images' => $images]);

}

public function index(Request $request) {

$images=$this->imageInstance->with('users')->orderBy('likes', 'desc')->paginate(6);

if($request->ajax()){

return view('ajax.photos', ['photos' => $images])->render();

}

if($request->session()->has('admin')){

return view('admin.photos', compact('images'));

}

return view('unautehnicated.photos',['data' => $this->getData(), 'photos' => $images]);

}

public function create(Request $request) {

$sesija=$request->session()->get('user');

try{

$provera=$this->anketaInstance->where(['user\_id'=>$sesija[0]['id'],'img\_id'=>$request->imgid])->get();

if($provera->count() !== 0){

return 0;

}

else {

$this->anketaInstance->user\_id = $sesija[0]['id'];

$this->anketaInstance->img\_id = $request->imgid;

$this->anketaInstance->ocena = $request->ocena;

$this->anketaInstance->save();

$rez=$this->anketaInstance->where('img\_id','like',$request->imgid)->avg('ocena');

return response($rez,200); }

}catch (\Exception $ex){

\Log::error('Greska prilikom upisa u anketu'. $ex->getMessage());

return 0; } }

public function store(Request $request) {

$request->validate([

'image' => 'required|image|mimes:jpeg,png,jpg,gif',

'heading'=> 'required|min:5|regex:/^[A-Z]{1,}/',

'paragraph'=>'required|min:5',

]);

$sesija=$request->session()->get('user');

$imageName = time() . '\_' . request()->image->getClientOriginalName();

try{

request()->image->move(public\_path('images'), $imageName);

$this->imageInstance->img\_name = $imageName;

$this->imageInstance->heading=$request->heading;

$this->imageInstance->paragraph=$request->paragraph;

$this->imageInstance->user\_id=$sesija[0]['id'];

$this->imageInstance->likes=0;

$this->imageInstance->save();

return back()

->with('success', 'You have successfully upload image.');

}catch (\Exception $ex){

\Log::error('Greska prilikom dodavanja posta'. $ex->getMessage());

dd($ex->getMessage());

return back() ->with('fail', 'We are sorry, try again later.'); } }

public function show(Request $request ,$id)

{

$podaci=$this->imageInstance->where('id', $id)->get();

$rez=$this->anketaInstance->where('img\_id',$id)->avg('ocena', 3);

$sesija=$request->session()->has('user');

if($sesija === false){

return view('unautehnicated.image', ['data' => $this->getData(), 'podaci' => $podaci, 'rez' => $rez]); }

else{

$sesija=$request->session()->get('user');

$isliked=$this->likeInstance->where(['user\_id'=>$sesija[0]['id'], 'img\_id'=>$id])->first();

if($isliked===null){

return view('unautehnicated.image', ['data' => $this->getData(), 'podaci' => $podaci, 'rez' => $rez,'like'=> 0]);

}

else{

return view('unautehnicated.image', ['data' => $this->getData(), 'podaci' => $podaci, 'rez' => $rez,'like'=>1]);

}

}

}

public function edit(Request $request,$id)

{

$podaci=$this->imageInstance->where('id', $id)->get();

return view('user.edit', ['data' => $this->getData(), 'podaci' => $podaci]);

}

public function update(Request $request)

{

$request->validate([

'heading'=> 'required|min:5|regex:/^[A-Z]{1,}/',

'paragraph'=>'required|min:5',

]);

$sesija=$request->session()->get('user');

if(isset($request->image)){

$request->validate([

'image' => 'required|image|mimes:jpeg,png,jpg,gif',

]);

try{

$oldimage=$this->imageInstance->select('img\_name')->where(['id' => $request->id, 'user\_id' => $sesija[0]['id']])->get();

$imageName = time() . '\_' . request()->image->getClientOriginalName();

unlink(public\_path() ."/images/".$oldimage[0]['img\_name']);

request()->image->move(public\_path('images'), $imageName);

$this->imageInstance->where(['id' => $request->id, 'user\_id' => $sesija[0]['id']])->update([

'img\_name' => $imageName,

'heading' => $request->heading,

'paragraph' =>$request->paragraph,

]);

return redirect()->action('ImageController@user')->with('success', 'You have successfully update post.');

}catch (\Exception $ex){

\Log::error('Greska prilikom update posta i menjanja fotke'. $ex->getMessage());

return back()->with('fail', 'We are sorry, try again later.'); } }

try{

$this->imageInstance->where(['id' => $request->id, 'user\_id' => $sesija[0]['id'] ])->update([

'heading' => $request->heading,

'paragraph' =>$request->paragraph,

]);

return redirect()->action('ImageController@user')->with('success', 'You have successfully update post.');

}catch (\Exception $ex){

return back() ->with('fail', 'We are sorry, try again later.');

}

}

public function destroy(Request $request,$id)

{

$sesija=$request->session()->get('user');

try{

$oldimage=$this->imageInstance->select('img\_name')->where(['id' => $request->id, 'user\_id' => $sesija[0]['id']])->get();

//dd(public\_path() ."/images/".$oldimage[0]['img\_name']);

unlink(public\_path() ."/images/".$oldimage[0]['img\_name']);

$this->imageInstance->where(['id' => $request->id, 'user\_id' => $sesija[0]['id']])->delete();

return back()->with('success', 'You have successfully deleted post');

}catch (\Exception $ex){

\Log::error('Greska prilikom brisanja posta'. $ex->getMessage());

return back()->with('fail', 'try again later');

}

}

public function destroyimgadmin(Request $request,$id)

{

try{

$oldimage=$this->imageInstance->select('img\_name')->where('id', $request->id)->get();

unlink(public\_path() ."/images/".$oldimage[0]['img\_name']);

$this->imageInstance->where('id', $request->id)->delete();

return back()->with('success', 'You have successfully deleted post');

}catch (\Exception $ex){

\Log::error('Greska prilikom brisanja posta'. $ex->getMessage());

return back()->with('fail', 'try again later');

}

}

public function like(Request $request){

try{

$sesija=$request->session()->get('user');

$this->imageInstance->where('id',$request->imgid)->increment('likes');

$this->likeInstance->insert(['user\_id'=>$sesija[0]['id'], 'img\_id'=>$request->imgid]);

return 1;

}catch (\Exception $exception){

return 2;

}

}

public function unlike(Request $request){

try{

$sesija=$request->session()->get('user');

$this->imageInstance->where('id',$request->imgid)->decrement('likes');

$this->likeInstance->where(['user\_id'=>$sesija[0]['id'], 'img\_id'=>$request->imgid])->delete();

return 1;

}catch (\Exception $exception){

return 2;

}

}

public function showvote(){

$votes=$this->anketaInstance->with('users','images')->orderby('img\_id')->get();

return view('admin.votes', ['votes'=>$votes]);

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\User;

class LoginController extends Controller

{

private $userInstance;

public function \_\_construct()

{

$this->userInstance=new User();

}

public function login(Request $request){

$request->validate([

'email' => 'required|email',

'pass'=>'required|min:4',

]);

$email=$request->email;

$password=md5($request->pass);

try {

$result=$this->userInstance->with('roles')->where([

'email'=> $email,

'password' => $password,

])->get();

$session=$this->userInstance->where([

'email'=> $email,

'password' => $password,

])->get();

if($result->count() == 1){

$name=$result[0]['roles']['name'];

if($name=='user'){

session()->put('user', $session);

return redirect('/user');

}

if($name == 'admin'){

session()->put('admin', $session);

return redirect('/admin/users');

}

}

else return back()->withInput()->with('fail', 'Sorry, user with that informations does not exists');

}catch (\Exception $ex){

\Log::error('Greska prilikom logovanja'. $ex->getMessage());

return back()->with('fail', 'Sorry, user with that informations does not exists');

}

return back();

}

public function logout(){

session()->invalidate();

return redirect('/');

}

public function register(Request $request){

$request->validate([

'username' => 'required|email|unique:users,email|max:40',

'password'=>'required|min:4|max:20',

'name'=>'required|max:40|unique:users,email',

]);

if($request->password !== $request->password2){

return back()->withInput()->with('fail','passwords must be equals');

}

try {

$this->userInstance->name=$request->name;

$this->userInstance->email=$request->username;

$this->userInstance->password=md5($request->password);

$this->userInstance->role\_id=2;

$this->userInstance->save();

}catch (\Exception $ex){

\Log::error('Greska prilikom registracije'. $ex->getMessage());

return back()->with('fail', 'We are sorry, try again latter');

}

try

{

$result=$this->userInstance->where([

'email'=> $request->username,

'password' => md5($request->password),

])->get();

if($result->count() == 1){

session()->put('user',$result);

return redirect('/user');

}

}catch (\Exception $ex){

\Log::error('Greska prilikom logovanja, nakon registracije'. $ex->getMessage());

return back()->with('fail', 'Now you can login');

}

return back();

}

public function users(){

$users=$this->userInstance->with('roles')->get();

return view('admin.home',compact('users'));

}

public function destroy($id){

try{

$this->userInstance->where('id', $id)->delete();

return back();

}catch (\Exception $exception){

\Log::error('Greska prilikom brisanja korisnika!!!!'. $exception->getMessage());

dd($exception->getMessage());

return back();

}

}

public function role($id){

try{

$this->userInstance->where('id', $id)->update(['role\_id'=>'1']);

return back();

}catch (\Exception $exception){

\Log::error('Greska prilikom menjanja role korisnika!!!!'. $exception->getMessage());

return back();

}

}

public function roleuser($id){

try{

$this->userInstance->where('id', $id)->update(['role\_id'=>'2']);

return back();

}catch (\Exception $exception){

\Log::error('Greska prilikom menjanja role korisnika!!!!'. $exception->getMessage());

return back();

}

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Question;

class QuestionController extends BaseController

{

private $questionInstance;

public function \_\_construct()

{

parent::\_\_construct();

$this->questionInstance=new Question();

}

public function index(Request $request)

{

if($request->session()->has('admin')){

$questions=$this->questionInstance->orderBy('answer', 'asc')->get();

return view('admin.questions', compact('questions'));

}

else {

$questions=$this->questionInstance->where('answer','NOT LIKE',null)->paginate(4);

}

if($request->ajax()){

return view('ajax.question', ['questions' => $questions])->render();

}

return view('unautehnicated.questions',['data' => $this->getData(), 'questions' => $questions]);

}

public function create(Request $request)

{

try{

$this->questionInstance->where('id', $request->id)->update(

['answer'=>$request->answer]);

return back();

}catch (\Exception $exception){

\Log::error('problem prilikom odgovaranja na pitanje');

return back();

}

}

public function store(Request $request)

{

$request->validate([

'quesiton' => 'required|min:4'

]);

try{

$this->questionInstance->question=$request->quesiton;

$this->questionInstance->save();

return back();

}catch (\Exception $ex){

\Log::error('problem prilikom postavljanja pitanja'. $ex->getMessage() . $request->quesiton);

return back();

}

}

public function destroy($id) {

try{

$this->questionInstance->find($id)->delete();

return back();

}catch (\Exception $exception){

dd($exception);

return back();

}

}

}

-------------------------------------------------Modeli------------------------------------------------------

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Anketa extends Model

{

public $timestamps=false;

public function Images(){

return $this->belongsTo('App\Image','img\_id', 'id');

}

public function Users(){

return $this->belongsTo('App\User','user\_id', 'id');

}

}

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Image extends Model

{

protected $fillable=['user\_id',];

public function Users(){

return $this->hasOne('App\User','id','user\_id');

}

}

<?php

namespace App;

use Illuminate\Notifications\Notifiable;

use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable

{

use Notifiable;

protected $fillable = [

'name', 'email', 'password','id',

];

public function Roles(){

return $this->hasOne("App\Role", "id", "role\_id");

}

}

Web.php

Route::post('/vote','ImageController@create');

Route::get('/dokumentacija', "HomeController@dokumentacija");

Route::get('/', 'HomeController@index');

Route::get('/questions', 'QuestionController@index');

Route::post('/questions/store','QuestionController@store')->name('question');

Route::post('/subscribe', 'HomeController@subscribe')->name('subscribe');

//Login Controller

Route::post('/login',"LoginController@login")->name("login");

Route::post('/register',"LoginController@register")->name("register");

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

//ImageController

Route::get('/image', 'ImageController@index');

Route::get('/image/show/{id}','ImageController@show');

//contactController

Route::get('/contact','ContactController@index');

Route::get('/author','HomeController@autor');

Route::post('/contacta','ContactController@store')->name('contact');

//user

Route::group(["prefix"=>"/user",'middleware'=>'user'], function (){

Route::get('/','ImageController@user');

Route::post('/store','ImageController@store')->name('user.upload');

Route::get('/image/edit/{id}','ImageController@edit');

Route::get('/image/delete/{id}','ImageController@destroy');

Route::post('/image/update','ImageController@update')->name('user.update');

Route::post('/like','ImageController@like');

Route::post('/unlike','ImageController@unlike');

});

//admin

Route::group(["prefix"=>"/admin",'middleware'=>'admin'], function (){

Route::get("/users","LoginController@users");

Route::get("/home/user/{id}/delete","LoginController@destroy");

Route::get("/home/user/{id}/role","LoginController@role");

Route::get("/home/user/{id}/role/user","LoginController@roleuser");

Route::get("/questions","QuestionController@index");

Route::post('/answer/{id}/reply','QuestionController@create');

Route::get('/questions/{id}/delete','QuestionController@destroy');

Route::get('/image','ImageController@index');

Route::get('/image/delete/{id}','ImageController@destroyimgadmin');

Route::get('/vote','ImageController@showvote');

Route::get('/contact','ContactController@show');

Route::get('/contact/{id}/delete','ContactController@destroy');

Route::get('/sub','HomeController@show');

Route::get('/nav','HomeController@navbar');

Route::get('/footer','HomeController@footer');

Route::get('/nav/delete/{id}','HomeController@navdelete');

Route::get('/nav/edit/{id}','HomeController@navedit');

Route::post('/nav/update','HomeController@navupdate')->name('nav.update');

Route::post('/nav/store','HomeController@navstore')->name('nav.store');

Route::get('/footer/delete/{id}','HomeController@footerdelete');

Route::get('/footer/edit/{id}','HomeController@footeredit');

Route::post('/footer/update','HomeController@footerupdate')->name('footer.update');

Route::post('/footer/store','HomeController@footerstore')->name('footer.store');

});