

VISOKA ŠKOLA STRUKOVNIH STUDIJA ZA INFORMACIONE I
KOMUNIKACIONE TEHNOLOGIJE

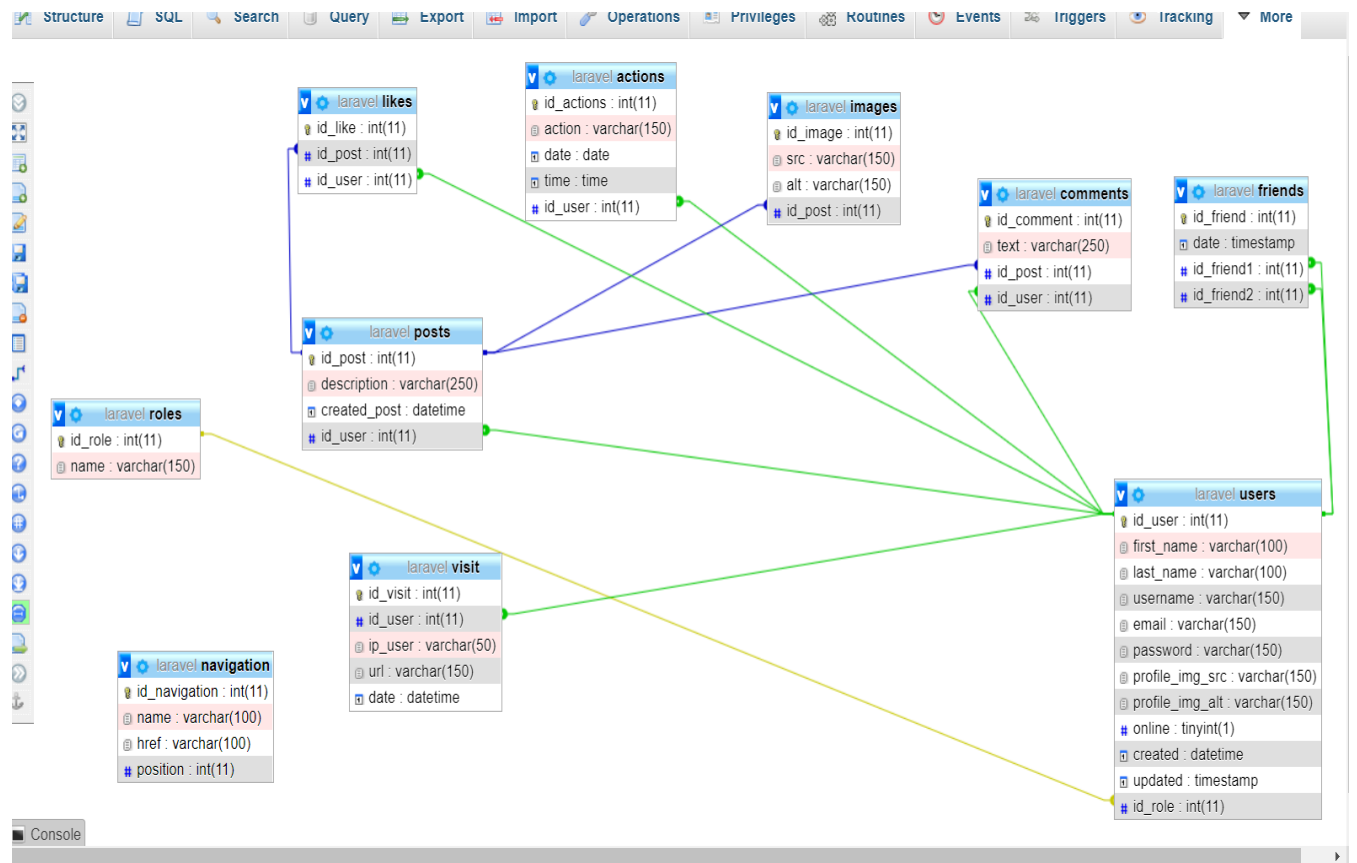
WEB PROGRAMIRANJE PHP2

Aleksandar Todorovic 173/17

<https://www.linkedin.com/in/aleksandar-todorovic-67bb061a2/>

Beograd 2020.

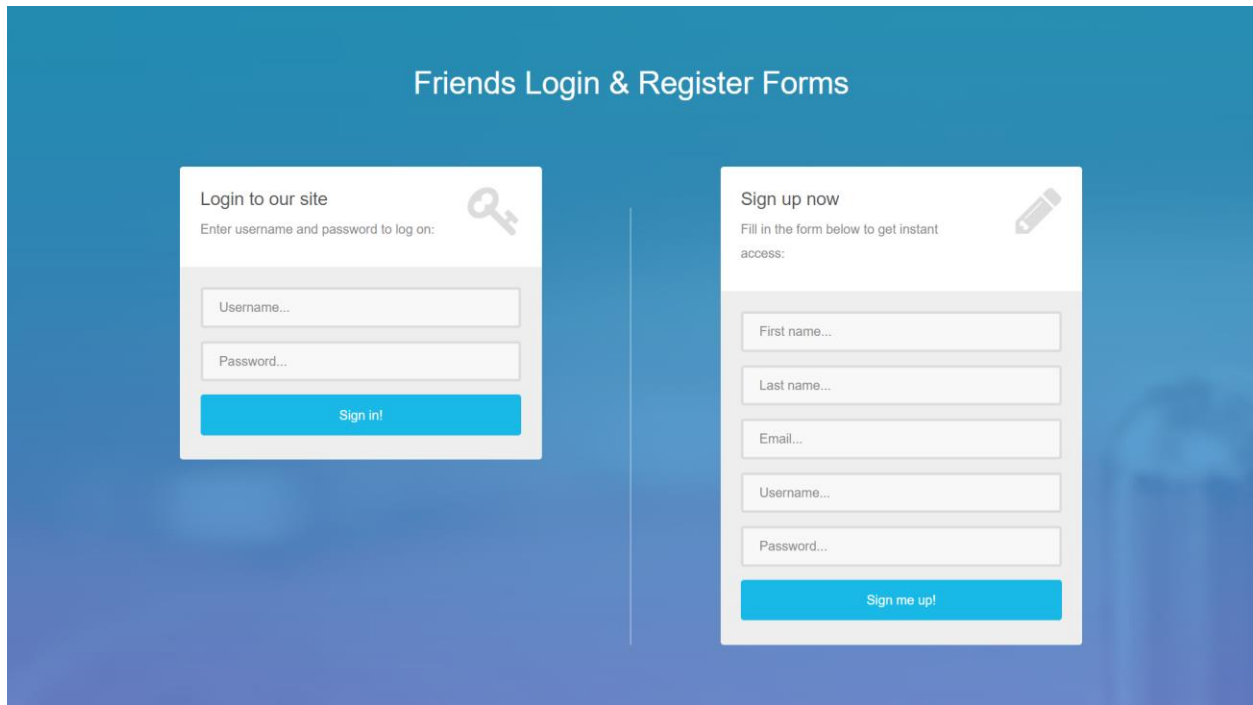
1. Baza podataka



2.Slike stranica I opis funkcionalnosti

Forma za registraciju I logovanje

Provera unose podataka putem regularnih izraza



The image displays two side-by-side login and registration forms on a blue background. The left form is titled 'Login to our site' and includes a key icon. It has input fields for 'Username...' and 'Password...' and a 'Sign in!' button. The right form is titled 'Sign up now' and includes a pencil icon. It has input fields for 'First name...', 'Last name...', 'Email...', 'Username...', and 'Password...', and a 'Sign me up!' button. Both forms have a light gray border and a white background.

Friends Login & Register Forms

Login to our site

Enter username and password to log on:

Username...

Password...

Sign in!

Sign up now

Fill in the form below to get instant access:

First name...

Last name...

Email...

Username...

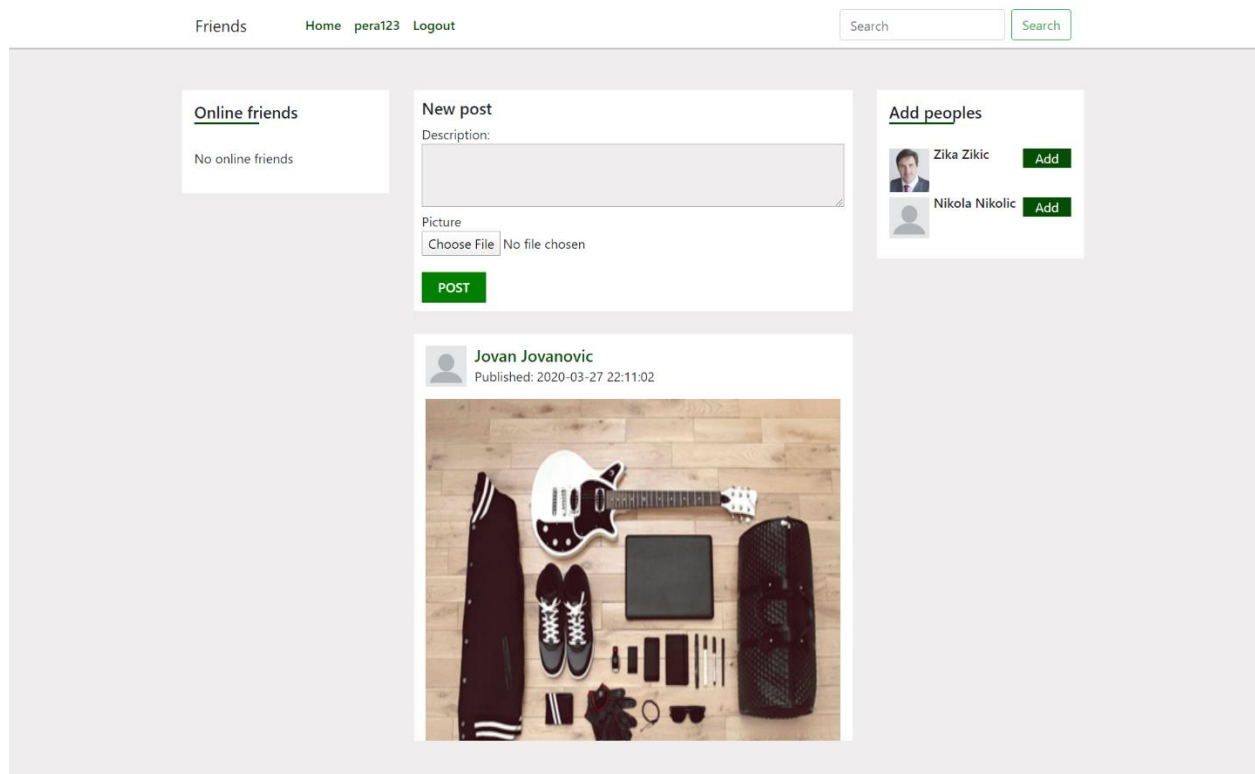
Password...

Sign me up!

Sa leve strane se prikazuju online prijatelji, dok sa desne se prikazuju sugestije za prijatelje koje su random generisane

U sredini se nalaze postovi korisnika i njegovih prijatelja


Moguće će je lakovanje posta i komentarisanje




Profilna strana korisnika


Nalaze se podaci o korisniku I njegovi postovi koje moze da edituje Ili obrise

[Friends](#) [Home](#) [pera123](#) [Logout](#)






[TIMELINE](#) [FRIENDS](#) [EDIT PROFILE](#)

**Pera Peric**
Published: 2020-03-27 16:41:00



Perin prvi post

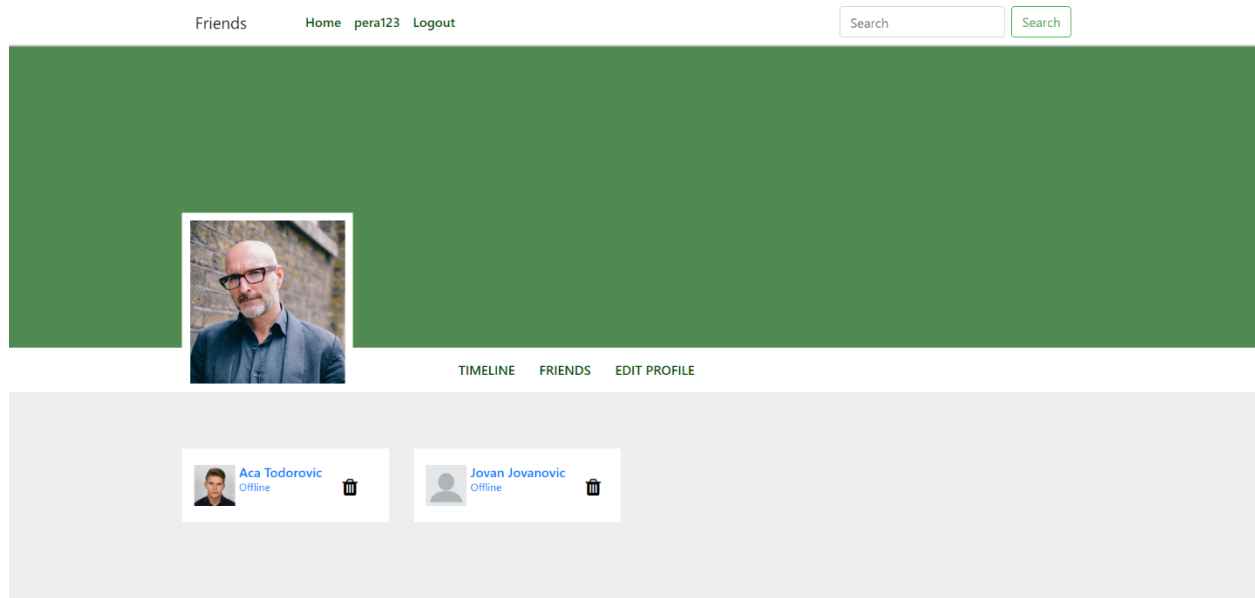
 10  12

 Pera Hehe

Strana na kojoj se prikazuju prijatelji korisnika

Moguće je brisanje prijatelja


Izmena profilne fotografije se izvodi klikom na istou




Stranica za editovanje korisnickih podataka

Kucanjem pojma u pretrazi izlistavaju se korisnici iz baze

[Friends](#) [Home](#) [pera123](#) [Logout](#)

 Aca Todorovic



[TIMELINE](#) [FRIENDS](#) [EDIT PROFILE](#)

Edit profile information

Profile image

No file chosen

Admin panel

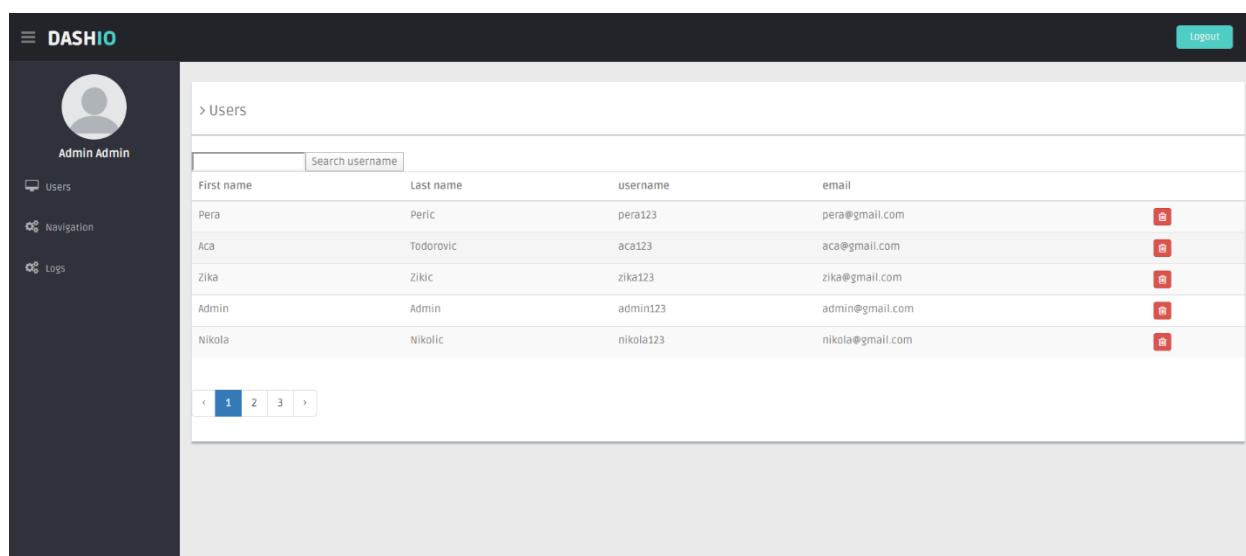
Omogucen je prikaz svih korisnika

Fitriranje po username I paginacija

Moguće je brisanje korisnika

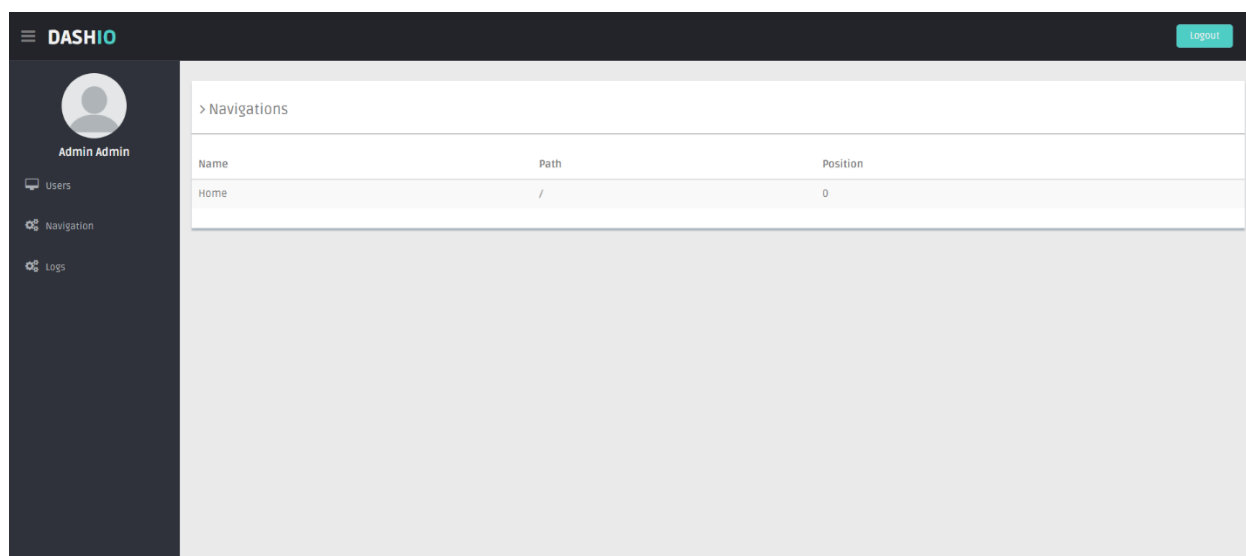
Dodavanje navigacije, prikaz

Prikaz aktivnosti korisnika, pristup stranicama, filtriranje aktivnosti po datumu



The screenshot shows the 'Users' management page in the DASHIO Admin Panel. The interface includes a dark sidebar with a user profile (Admin Admin) and navigation links for Users, Navigation, and Logs. The main content area displays a table of users with columns for First name, Last name, username, and email. A search bar is located above the table. The table lists six users: Pera Peric, Aca Todorovic, Zika Zikic, Admin Admin, and Nikola Nikolic. Each user entry has a red delete icon on the right. A pagination control at the bottom of the table shows page 1 is selected, with options for 2 and 3.

First name	Last name	username	email	
Pera	Peric	pera123	pera@gmail.com	
Aca	Todorovic	aca123	aca@gmail.com	
Zika	Zikic	zika123	zika@gmail.com	
Admin	Admin	admin123	admin@gmail.com	
Nikola	Nikolic	nikola123	nikola@gmail.com	



The screenshot shows the 'Navigations' management page in the DASHIO Admin Panel. The interface is consistent with the previous screenshot, featuring the same sidebar and navigation links. The main content area displays a table with columns for Name, Path, and Position. The table contains one entry: 'Home' with a path of '/' and a position of '0'.

Name	Path	Position
Home	/	0

DASHIO

Logout

Admin Admin

Users

Navigation

Logs

> Actions

dd-----yyyy

Search

User	Action	Date	Time
pera123	login	2020-03-26	20:36:30
pera123	logout	2020-03-27	20:36:30
pera123	login	2020-03-27	20:36:30
pera123	logout	2020-03-27	20:36:30
pera123	login	2020-03-27	20:36:30

<

1

2

3

4

5

6

>

DASHIO

Logout

Admin Admin

Users

Navigation

Logs

> Visits

User	Ip	Url	Date
pera123	127.0.0.1	http://127.0.0.1:8000	2020-03-27 17:43:41
pera123	127.0.0.1	http://127.0.0.1:8000/user/1	2020-03-27 17:44:03
pera123	127.0.0.1	http://127.0.0.1:8000/logout	2020-03-27 17:45:38
pera123	127.0.0.1	http://127.0.0.1:8000	2020-03-27 21:23:10
pera123	127.0.0.1	http://127.0.0.1:8000/user/1	2020-03-27 21:24:47
pera123	127.0.0.1	http://127.0.0.1:8000/user/1	2020-03-27 21:25:16
pera123	127.0.0.1	http://127.0.0.1:8000/user/1	2020-03-27 21:26:00
pera123	127.0.0.1	http://127.0.0.1:8000/user/friends	2020-03-27 21:26:02
pera123	127.0.0.1	http://127.0.0.1:8000/user/1/edit	2020-03-27 21:26:09
pera123	127.0.0.1	http://127.0.0.1:8000/user/1/edit	2020-03-27 21:27:07

<

1

2

3

4

5

6

7

8

...

47

48

>

3.Kodovi

Web.php

```
<?php

/*
|-----
| 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('/', "HomeController@index")->middleware("login");

Route::get('/registration', function () {
    return view('registration');
});

Route::post("/doLogin", "LoginController@doLogin2")->name("doLogin");
Route::post("/doRegist", "RegistrationController@register")->name("doRegist");

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

Route::post('/posts', "PostController@store");
Route::get('/posts_ajax', "PostController@getPostAjax");
Route::get('post/{id}/destroy', 'PostController@destroy')->name('post_delete');
Route::get('post/{id}/edit', 'PostController@edit')->name('post_edit');
Route::post('post/{id}/update', 'PostController@update')->name('post_update');

Route::get("/user/friends", "UserController@friends");

Route::get("/user/{id}", "UserController@show")->name('user');
Route::get("/user/{id}/edit", "UserController@userForm")->name('profile_form');
Route::post("/user/profile_img", "UserController@editProfileImage");
Route::post('/user/{id}/edit', "UserController@edit")->name('edit_profile');

//ako su prijatelji
```

```

Route::get("/user_profile/{id}", "UserController@userProfile")-
>name('user_profile');
Route::get("/user_profile/{id}/friends", "FriendController@userProfileFriends")-
>name('user_profile_friends');

//friends
Route::get('/friend/{id}/delete', "FriendController@destroy")-
>name('friend_delete');
Route::get("/add_friend/{id}", "FriendController@addFriend")->name('add_friend');

//admin
Route::group(['middleware' => 'admin'], function () {
    Route::get("/admin/visits", "Admin\LogsController@visits")-
>name('logs_visit');
    Route::get("/admin/actions", "Admin\LogsController@actions")-
>name('logs_action');

    Route::get("/admin/user", "Admin\UserController@index")->name('user.index');
    Route::get("/admin/user/{id}/delete", "Admin\UserController@destroy");

    Route::get("/admin/navigation/create", "Admin\NavigationController@create")-
>name("navigation.create");
    Route::post("/admin/navigation", "Admin\NavigationController@store")-
>name("navigation.store");
    Route::get("/admin/navigation", "Admin\NavigationController@index")-
>name("navigation.index");
});

```

Api.php

```

<?php

use Illuminate\Http\Request;

/*
|-----

```

```

| API Routes
| -----
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| is assigned the "api" middleware group. Enjoy building your API!
|
| */

Route::middleware('auth:api')->get('/user', function (Request $request) {
    return $request->user();
});

Route::get("/user", "UserController@users");
Route::post("/comments/post", "CommentsController@postComment");
Route::post("/add-friend", "FriendController@addFriendAjax");
Route::post("/like_post", "LikeController@add");

```

CommentsController.php

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Comments;

class CommentsController extends Controller
{

    public function postComment(Request $request)
    {
        // return "Test";
        // dd($request->all());
        $comment = $request->input('comment');
        $idPost = $request->input('idPost');
        $idUser = $request->input('idUser');
    }
}

```

```

        $model = new Comments();
        try {
            $model->insert($comment, $idPost, $idUser);
            return response()->json(null, 200);
        } catch (\PDOException $ex) {
            dd($ex->getMessage());
            return response()->json(['message' => "Greska u bazi"], 500);
        }
    }
}

```

FriendController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Friend;
use Illuminate\Http\Request;
use Illuminate\Http\Response;

class FriendController extends Controller
{
    private $model;

    public function __construct()
    {
        $this->model = new Friend();
    }

    public function addFriendAjax(Request $request)
    {
        // return dd($request->all());
        $idSugestion = $request->input('idSugestion');
        $idUser = $request->input('idUser');

        try {
            $this->model->store($idSugestion, $idUser);
            return response(null, 200);
        } catch (\Exception $ex) {
            \Log::error($ex->getMessage());
        }
    }
}

```

```

        return response(['message' => 'Greska na serveru'], 500);
    }
}

public function addFriend($id)
{
    $idUser = session('user')->id_user;

    try {
        $this->model->store($id, $idUser);
        return redirect()->back();
    } catch (\Exception $ex) {
        \Log::error($ex->getMessage());
        // return response(['message' => 'Greska na serveru'], 500);
    }
}

public function destroy($id)
{
    try {
        $this->model->deleteFriend($id);
        return redirect()->back();
    } catch (\Exception $ex) {
        \Log::error($ex->getMessage());
        dd($ex->getMessage());
    }
}

public function userProfileFriends($id)
{
    $model = new Friend();
    $friends = $model->getFriendsOfUser($id);
    // dd($friends);
    return view("pages.user_profile_friends", ['friends' => $friends]);
}
}

```

HomeController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Friend;
use App\Models\Post;
use Illuminate\Http\Request;

class HomeController extends Controller
{
    public function index(Request $request)
    {
        //prikaz postova, prijatelja online i sugestije za prijatelja

        //sugestije
        // dd(session()->all());
        $modelFriend = new Friend();
        $idUser = session("user")->id_user;
        $friendsSugestion = $modelFriend->getSugestedFriendsForUser($idUser);
        // dd($friendsSugestion);
        $data['friendsSugestion'] = $friendsSugestion;

        //online prijatelji
        $onlineFriends = $modelFriend->getOnlineFriends($idUser);
        // dd($onlineFriends);
        $data['onlineFriends'] = $onlineFriends;

        $modelPost = new Post();
        $posts = $modelPost->postOfUserAndFriends($idUser);
        // dd($posts);
        $data['posts'] = $posts;

        return view('pages.main', $data);
    }
}
```

LikeController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Like;
use Illuminate\Http\Request;

class LikeController extends Controller
{
    private $model;
    public function __construct()
    {
        $this->model = new Like();
    }

    public function add(Request $request)
    {
        // dd($request->all());
        $idPost = $request->input('idPost');
        $idUser = $request->input('idUser');

        $exist = $this->model->existLike($idPost, $idUser);
        // dd($exist);

        try {
            if (!$exist) {
                $this->model->likePost($idPost, $idUser);
            } else {
                //obrise se like za taj post
                $this->model->deletePost($idPost, $idUser);
            }
            return response(['message' => 'uspesno'], 200);
        } catch (\Exception $ex) {
            \Log::error($ex->getMessage());
            return response(['message' => 'Greska na serveru'], 500);
        }
    }
}
```


LoginController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests\LoginRequest;
use App\Http\Requests\RegistrationRequest;
use App\Models\Actions;
use App\Models>Login;
use Illuminate\Http\Request;
use Session;
use Validator;

class LoginController extends Controller
{

    private $model;

    // public function __construct(Login $model)
    // {
    //     $this->model = $model;
    // }

    // public function doLogin(LoginRequest $request)
    // {
    //     //data
    //     $username = $request->input("username");
    //     $password = $request->input("password");

    //     // dd($request);

    //     // $user = $this->model->getUser($username, $password);

    //     // if ($user) {
    //     // } else {
    //     // }
    //     return redirect("/");
    // }

    public function doLogin2(Request $request)
    {

        // validate incoming request
        // dd($request->all());
    }
}
```

```

    $messages = [
        'username.required' => 'Username je obavezno polje',
        'password.required' => "Password je obavezno polje"
    ];

    $validator = Validator::make($request->all(), [
        "username" => "required|regex:/^[\\w!@#%$^&*]+$/",
        "password" => "required|regex:/\\d+/"|regex:/[a-
z]+/"|regex:/[!@#%$^&*?]+/" ,
    ], $messages);

    if ($validator->fails()) {
        // return redirect('/registration')
        // ->withErrors($validator, "login");
        return response()->json(['errors' => $validator->errors()-
>all()], 422);
    } else {
        //get user
        $model = new Login();
        // dd($request->all());
        $user = $model->getUser($request->input('username'), md5($request-
>input('password')));
        // dd($user);
        if ($user) {
            //ukoliko postoji korisnik
            $request->session()->put("user", $user);

            $modelAction = new Actions();
            $modelAction->actionsUser('login', $user->id_user);

            return response(['message' => "Uspesno logovanje"], 200);
        } else {
            return response(['message' => "Ne postoji korisnik"], 422);
        }
        // dd($user);
    }
}

public function logout(Request $request)
{
    $modelAction = new Actions();
    $modelAction->actionsUser('logout', session('user')->id_user);
}

```

```
        $request->session()->forget('user');
        $request->session()->flush();

        return redirect("/registration");
    }
}
```

PostController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests\PostRequest;
use App\Models\Image;
use App\Models\Post;
use App\Services\PostService;
use App\Services\UploadService;
use Illuminate\Http\Request;

class PostController extends Controller
{
    private $model;
    private $service;

    public function __construct()
    {
        $this->model = new Post();
        $this->service = new PostService();
    }

    public function store(PostRequest $request)
    {
        return $this->service->insert($request);
    }

    public function edit($id)
    {

```

```

        //forma za editovanje
        $post = $this->model->getPost($id);
        // dd($post);

        return view('pages.post_edit', ['post' => $post]);
    }

    public function update(PostRequest $request, $id)
    {
        // dd($request->all());
        // dd($request->file('post_img'))
        // $this->service->insert($request, $id);

        //fizicki ce upload na disk
        if ($request->hasFile('post_img')) {
            $uploadService = new UploadService();
            $image = $uploadService->upload($request->file('post_img'), 'posts');

            //update u bazi sliku za zadati id
            try {
                $modelImage = new Image();
                $modelImage->update($id, $image);
            } catch (\Exception $ex) {
                \Log::error($ex->getMessage());
            }
        }

        //update posta u bazu
        try {
            $description = $request->input('post_description');
            $this->model->updatePost($id, $description);
            return redirect()->back()->with('message', "Uspesno izmenjen post");
        } catch (\Exception $ex) {
            \Log::error($ex->getMessage());
            return redirect()->back()-
>with('message', "Greska prilikom izmene posta, pokusajte kasnije");
        }
    }

    public function getPostAjax()
    {

```

```

        $idUser = session('user')->id_user;

        try {
            $posts = $this->model->postOfUserAndFriends($idUser);
            return response($posts, 200);
        } catch (\Exception $ex) {
            \Log::error($ex->getMessage());
            return response(['message' => 'Greska na serveru'], 500);
        }
    }

    public function destroy($id)
    {
        try {
            $this->model->deletePost($id);
            return redirect()->back()->with('message', 'Uspesno izbrisan post');
        } catch (\Exception $ex) {
            \Log::error($ex->getMessage());
            return redirect()->back()->with('message', 'Greska na serveru');
        }
    }
}

```

RegistrationController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Actions;
use App\Models\User;
use Illuminate\Http\Request;
use Validator;

class RegistrationController extends Controller
{
    public function register(Request $request)
    {
        $validator = Validator::make($request->all(), [
            "first_name" => "required|regex:/^[A-Z]{1}[a-z]+$/",
            "last_name" => "required|regex:/^[A-Z]{1}[a-z]+$/",

```

```

        "email" => ["required", "email", "regex:/^([a-zA-Z0-9_+-.]+\@(([a-zA-Z0-9-]+\.)+([a-zA-Z0-9]{2,4})+$)/"],
        "username" => "required|regex:/^[\\w!@#%&*]+$/",
        "password" => "required|regex:/\\d+/"|regex:/[a-zA-Z0-9]{8,}/|regex:/[!@#%&*^&?]+/",
    ]);

    if ($validator->fails()) {
        // return redirect('/registration')
        // ->withErrors($validator, "registration");
        return response()->json(["errors" => $validator->errors()->all()], 422);
    }

    //insert into database
    // dd($request);
    $model = new User();
    try {
        $id = $model->insertUser($request->input("first_name"), $request->input("last_name"), $request->input("username"), $request->input("email"), md5($request->input("password")));

        $modelAction = new Actions();
        $modelAction->actionsUser('register', $id);
    } catch (\PDOException $ex) {
        \Log::error($ex->getMessage());
        return response(['greska' => "Greska prilikom registracija, pokušajte kasnije"], 500);
    }
}
}

```

UserController.php

```

<?php

namespace App\Http\Controllers;

use App\Models\Friend;

```

```

use App\Models\Post;
use App\Models\User;
use App\Rules>Password;
use App\Services\UploadService;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class UserController extends Controller
{

    private $model;

    public function __construct()
    {
        $this->model = new User();
    }

    public function profile(Request $request)
    {
        $user = $this->model->getUserProfileInformation($request);
        // dd($user);
        return view("pages.profile", ['user' => $user]);
    }

    public function users(Request $request)
    {
        $name = $request->input('name');
        try {
            $users = $this->model->getUsers($name);
            return response($users, 200);
        } catch (\PDOException $ex) {
            return response(['error' => $ex->getMessage()], 500);
        }
    }

    public function show($id)
    {
        $model = new Post();

        $data['posts'] = $model->getPostsOfUser($id);

        // dd($data);
        return view('pages.profile', $data);
    }
}

```

```

}

public function friends()
{
    $model = new Friend();
    $friends = $model->getFriendsOfUser(session()->get('user')->id_user);
    // dd($friends);
    return view("pages.friends", ['friends' => $friends]);
}

public function userForm($id)
{
    // dd($id);
    $user = $this->model->getOneUser($id);
    // dd($user);
    return view("pages.edit_profile", ['user' => $user]);
}

public function edit(Request $request, $id)
{
    // dd($request->all());

    $rules = [
        'first_name' => 'required|alpha|min:2|max:20',
        'last_name' => 'required|alpha|min:2|max:20',
        'email' => 'required|email',
        'username' => 'required',
    ];

    //ukoliko se vrzi izmena passworda
    if ($request->input('password') != null) {
        $rules['password'] = "required|regex:/\d+/" . "regex:/[a-z]+/" . "regex:/[!@#$%^&*?]+/";
    }

    // dd($rules);

    $validator = \Validator::make($request->all(), $rules);
    $validator->validate();

    $firstName = $request->input('first_name');
    $lastName = $request->input('last_name');

```



```

$username = $request->input('username');
$email = $request->input('email');
// $password = $request->input('password');
$profileImg = $request->file('edit_profile_img');

try {
    //ubaci id user
    $this->model-
>updateUser($id, $firstName, $lastName, $username, $email);

    //resetovati sesiju
    $user = $this->model->updateSession($id);
    session()->put("user", $user);

    return redirect()->back()-
>with("message", "Uspesno ste izmenili informacije");
} catch (\Exception $ex) {
    \Log::error($ex->getMessage());
    return redirect()->back()-
>with("message", "Doslo je do greske pokusajte kasnije");
    // dd($ex->getMessage());
}

// return view("pages.edit_profile");
}

public function editProfileImage(Request $request)
{
    // dd($request->all());
    $resource = new UploadService();
    $image = $resource->upload($request->file('profile_img_file'), "users");
    $idUser = $request->input('user_id');
    // dd($image);

    try {
        $model = new User();
        $model->updateProfileImage($image, $idUser);

        //resetovati sesiju
        $user = $model->updateSession($idUser);
        session()->put("user", $user);

        return redirect()->back();
    } catch (\Exception $ex) {
        \Log::error($ex->getMessage());
    }
}

```

```
        // dd($ex->getMessage());
    }
}

public function userProfile($id)
{
    //da li su prijatelji
    $modelFriends = new Friend();
    $friends = $modelFriends->checkIfIsFriend($id, session('user')->id_user);
    // dd($id);
    // dd(session('user')->id_user);
    // dd($friends);
    $modelUser = new User();
    $user = $modelUser->getOneUser($id);
    session()->put("user_profile", $user);

    if ($friends) {
        $model = new Post();
        $data['posts'] = $model->getPostsOfUser($id);
        return view("pages.user_profile", $data);
    }

    return view("pages.user_profile_not_friends");
    //ukoliko su prijatelji

    // dd($data);
}
}
```

Middleware

IsLoginMiddleware.php

```
<?php

namespace App\Http\Middleware;

use Closure;

class IsLoginMiddleware
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param \Closure $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        if (session()->has('user')) {
            return $next($request);
        }
        return redirect("/registration");
    }
}
```

VisitMiddleware.php

```
<?php

namespace App\Http\Middleware;

use Closure;
use Illuminate\Support\Facades\DB;
```

```

class VisitMiddleware
{
    /**
     * Handle an incoming request.
     *
     * @param \Illuminate\Http\Request $request
     * @param \Closure $next
     * @return mixed
     */
    public function handle($request, Closure $next)
    {
        if (session()->has('user')) {
            // dd(session()->all());
            DB::table('visit')
                ->insert(
                    [
                        'id_user' => session('user')->id_user,
                        'ip_user' => $request->ip(),
                        'url' => $request->url()
                    ]
                );
        }
        return $next($request);

        // DB::table('visits')
        // ->inser([
        //     ''
        // ])
    }
}

```

AdminMiddleware.php

```

<?php

namespace App\Http\Middleware;

use Closure;

class AdminMiddleware
{

```

```

/**
 * Handle an incoming request.
 *
 * @param \Illuminate\Http\Request $request
 * @param \Closure $next
 * @return mixed
 */
public function handle($request, Closure $next)
{
    if (session()->has('user')) {
        if (session()->get('user')->id_role == "2") {
            return $next($request);
        }
    }
    return redirect("/");
}
}

```

Modeli

Actions.php

```

<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Actions
{
    public function actionsUser($action, $id)
    {
        return DB::table('actions')
            ->insert([

```

```
        'action' => $action,  
        'id_user' => $id  
    ]]);  
    }  
}
```

Comments.php

```
<?php  
  
namespace App\Models;  
  
use Illuminate\Support\Facades\DB;  
  
class Comments  
{  
  
    public function insert($text, $idPost, $idUser)  
    {  
        DB::table("comments")  
            ->insert([  
                'text' => $text,  
                "id_post" => $idPost,  
                "id_user" => $idUser  
            ]]);  
    }  
}
```

Friend.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Friend
{

    public function getSugestedFriendsForUser($idUser)
    {
        // dd($idUser);
        // return DB::table('user')
        //     ->select("id", 'first_name', "last_name")
        //     ->where('user.id', "!=", $idUser)
        //     ->whereNotExists(function ($query) {
        //         $query->select(DB::raw(1))
        //         ->from('friends')
        //         ->whereRaw("id_friend1 = 1") // promeni
        //         ->whereRaw("id_friend2 = user.id");
        //     })
        //     ->inRandomOrder()
        //     ->limit('2')
        //     ->get();

        return DB::select("SELECT * from users where users.id_user != $idUser AND
not EXISTS(SELECT * from friends where id_friend1 = $idUser AND id_friend2 = use
rs.id_user ) order by rand() limit 2");

        // DB::table('users')
        //     ->whereExists(function($query)
        //     {
        //         $query->select(DB::raw(1))
        //         ->from('orders')
        //         ->whereRaw('orders.user_id = users.id');
        //     })
        //     ->get();
    }

    public function getOnlineFriends($idUser)
    {
        return DB::table('users')
```

```

        ->join('friends', 'users.id_user', 'friends.id_friend2')
        -
>select('users.first_name', "users.last_name", "users.profile_img_src", "users.pr
ofile_img_alt")
        ->where("friends.id_friend1", $idUser)
        ->where('users.online', '1') //promeni na 1
        ->get();
    }

    public function getFriendsOfUser($idUser)
    {
        return DB::table("users as u")
            ->join("friends as f", "u.id_user", "f.id_friend2")
            ->where("f.id_friend1", $idUser)
            ->get();
    }

    public function store($idSugestion, $idUser)
    {
        //insert friend
        return DB::table('friends')
            ->insert([
                ['id_friend1' => $idSugestion, 'id_friend2' => $idUser],
                ['id_friend1' => $idUser, 'id_friend2' => $idSugestion]
            ]);
    }

    public function deleteFriend($id)
    {
        // dd($id);
        DB::table('friends')
            ->where('id_friend1', $id)
            ->delete();

        DB::table('friends')
            ->where('id_friend2', $id)
            ->delete();
    }

    public function checkIfIsFriend($id, $idFriend)
    {
        return DB::table('friends')

```



```
        ->where('id_friend1', $id)
        ->where('id_friend2', $idFriend)
        ->first();
    }
}
```

Image.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Image
{
    public function insert($idPost, $image)
    {
        // dd($image);
        DB::table("images")
            ->insert([
                "src" => $image['file_name'],
                'alt' => $image['alt'],
                'id_post' => $idPost
            ]);
    }

    public function update($idPost, $image)
    {
        return DB::table("images")
            ->where('id_post', $idPost)
            ->update([
                "src" => $image['file_name'],
                'alt' => $image['alt']
            ]);
    }
}
```

```
}
```

Like.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Like
{
    public function likePost($idPost, $idUser)
    {
        return DB::table('likes')
            ->insert([
                'id_post' => $idPost,
                'id_user' => $idUser
            ]);
    }

    public function existLike($idPost, $idUser)
    {
        return DB::table('likes')
            ->where([
                ['id_post', $idPost],
                ['id_user', $idUser]
            ])->first();
    }

    public function deletePost($idPost, $idUser)
    {
        return DB::table('likes')
            ->where([
```

```
        ['id_post', $idPost],  
        ['id_user', $idUser]  
    ]->delete();  
    }  
}
```

Login.php

```
<?php  
  
namespace app\Models;  
  
use DB;  
  
class Login  
{  
  
    public function getUser($username, $password)  
    {  
        return DB::table('users')  
            // ->select('id', 'username')  
            ->where([  
                ['username', $username],  
                ['password', $password]  
            ])->first();  
    }  
}
```

Logs.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Logs
{
    public function getVisits()
    {
        return DB::table('visit as v')
            ->select('v.*', 'u.username')
            ->join('users as u', 'v.id_user', 'u.id_user')
            ->paginate(10);
    }

    public function getActions($request = null)
    {
        $query = DB::table('actions as a')
            ->select('a.*', 'u.username')
            ->join('users as u', 'a.id_user', 'u.id_user');

        if ($request) {
            $query->where('date', $request->get('date_search'));
        }

        return $query->paginate(5);

        // return DB::table('actions as a')
        //     ->select('a.*', 'u.username')
        //     ->join('users as u', 'a.id_user', 'u.id_user')
        //     ->paginate(10);
    }
}
```

Navigation.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;

class Navigation
{
    public function insert($path, $name, $position)
    {
        return DB::table('navigation')
            ->insert([
                'href' => $path,
                'name' => $name,
                'position' => $position
            ]);
    }

    public function getNav()
    {
        return DB::table('navigation')->get();
    }
}
```

Post.php

```
<?php

namespace App\Models;

use Illuminate\Support\Facades\DB;
```

```

class Post
{
    public function insert($request)
    {
        // dd(session('user')->id);
        return DB::table('posts')
            ->insertGetId([
                'description' => $request->input('post_description'),
                "id_user" => session('user')->id_user
            ]);
    }

    public function updatePost($idPost, $description)
    {
        return DB::table('posts')
            ->where('id_post', $idPost)
            ->update([
                'description' => $description
            ]);
    }

    public function postOfUserAndFriends($idUser)
    {
        $posts = DB::select("SELECT p.*, i.*, u.* FROM users u INNER JOIN posts p
        ON u.id_user = p.id_user INNER JOIN images i ON p.id_post = i.id_post WHERE p.i
        d_user = $idUser or p.id_user IN (select u.id_user from users u inner join friend
        s f on u.id_user = f.id_friend2 WHERE f.id_friend1 = $idUser) ORDER BY p.created_
        post DESC");

        foreach ($posts as $p) {
            $comments = $this->getPostComment($p->id_post);
            $numberOfComments = $this->getNumberOfCommentsForPost($p->id_post);
            $p->comments = $comments;
            $p->numberOfComments = $numberOfComments;

            $likes = $this->getNumberOfLikes($p->id_post);
            $p->numberOfLikes = $likes;
        }
    }
}

```

```

        $userSession = session('user');
        // dd($userSession);
        $p->userFromSession = $userSession;
    }
    // dd($posts);

    return $posts;
}

public function getPostComment($idPost)
{
    return DB::table("comments as c")
        ->join("users as u", "c.id_user", "u.id_user")
        ->where("id_post", $idPost)
        ->get();
}

public function getNumberOfCommentsForPost($idPost)
{
    return DB::table("comments")
        ->where("id_post", $idPost)
        ->count();
}

public function getNumberOfLikes($idPost)
{
    return DB::table("likes")
        ->where("id_post", $idPost)
        ->count();
}

function getPostsOfUser($idUser)
{
    // dd(session('user'));
    $posts = DB::table('posts as p')
        ->join("images as i", "p.id_post", "i.id_post")
        ->where('id_user', $idUser)
        ->orderBy('created_post', 'desc')
        ->get();

    foreach ($posts as $p) {

```

```

        $comments = $this->getPostComment($p->id_post);
        $p->comments = $comments;
    }
    // dd($posts);
    return $posts;
}

public function deletePost($id)
{
    return DB::table('posts')
        ->where('id_post', $id)
        ->delete();
}

public function getPost($id)
{
    return DB::table('posts as p')
        ->join('images as i', 'p.id_post', 'i.id_post')
        ->where('p.id_post', $id)
        ->first();
}
}

```

User.php

```
<?php
```



```

namespace app\Models;

use DB;
use Illuminate\Http\Request;

class User
{
    public function insertUser($firstName, $lastName, $username, $email, $password)
    {
        // dd($lastName);
        return DB::table('users')->insertGetId(
            [
                'first_name' => $firstName,
                'last_name' => $lastName,
                'username' => $username,
                'email' => $email,
                'password' => $password
            ]
        );
    }

    public function getUserProfileInformation($request)
    {
        return DB::table('users')
            // ->select('username', 'img_src', 'img_alt')
            ->where('id_user', $request->session()->get('user')->id_user)
            ->first();
    }

    // SELECT * from user where not EXISTS(SELECT * from friends where id_friend1
    = 1 AND id_friend2 = user.id)
    // SELECT * from user where user.id != 1 AND not EXISTS(SELECT * from friends
    where id_friend1 = 1 AND id_friend2 = user.id)

    public function getUsers($name)
    {
        return DB::table("users as u")
            -
            >select("u.id_user", "u.first_name", "u.last_name", "u.profile_img_src")
            ->where('u.first_name', 'like', "%" . $name . "%")
            ->orWhere("u.last_name", 'like', "%" . $name . "%")
            ->get();
    }
}

```

```

}

public function getUserWithPosts($id)
{
    $user = $this->getOneUser($id);

    $posts = $this->getUserPosts($id);

    // dd($posts);
    $user->posts = $posts;
    return $user;
    // dd($user);
}

public function getOneUser($id)
{
    return DB::table('users')
        ->where('id_user', $id)->first();
}

public function getUserPosts($id)
{
    return DB::table('posts as p')
        ->join('images as i', 'p.id_post', "i.id_post")
        ->where('id_user', $id)
        ->get();
}

public function getAllUsers($request = null)
{
    $query = DB::table('users');
    if ($request) {
        // dd($request->get('user_search'));
        $query->where("username", "like", "%" . $request-
>get("user_search") . "%");
    }

    return $query->paginate(5);
}

```

```

public function delete($id)
{
    return DB::table('users')->where('id_user', $id)->delete();
}

public function updateProfileImage($file, $idUser)
{
    return DB::table('users')
        ->where('id_user', $idUser)
        ->update([
            'profile_img_src' => $file['file_name'],
            'profile_img_alt' => $file['alt']
        ]);
}

public function updateSession($idUser)
{
    return DB::table('users')
        ->where([
            ['id_user', $idUser]
        ])->first();
}

public function updateUser($id, $firstName, $lastName, $username, $email)
{
    // dd($id);
    return DB::table('users')
        ->where('id_user', $id)
        ->update([
            'first_name' => $firstName,
            'last_name' => $lastName,
            'username' => $username,
            'email' => $email,
            'updated' => date('Y-m-d H:i:s', time())
        ]);
}
}

```

Services

PostService.php

```
<?php

namespace App\Services;

use App\Http\Requests\PostRequest;
use App\Models\Image;
use App\Models\Post;
use App\Services\UploadService;
use Illuminate\Support\Facades\DB;

class PostService
{
    private $model;

    public function __construct()
    {
        $this->model = new Post();
    }

    public function insert($request)
    {
        $uploadedImage = $this->uploadImage($request);
        // dd($uploadedImage);

        $this->model = new Post();
        // dd($request->all());
        DB::beginTransaction();
        try {
            $idPost = $this->model->insert($request);
            // dd($idPost);
            $this->insertImage($idPost, $uploadedImage);
            DB::commit();
            return redirect('/');
        } catch (\PDOException $ex) {
            dd($ex->getMessage());
            DB::rollBack();
        }
    }
}
```

```

    }
}

public function insertImage($idPost, $image)
{
    $model = new Image();
    $model->insert($idPost, $image);
}

public function uploadImage($request)
{
    $image = $request->file('post_img');
    // if ($image->isValid()) {
    //     dd($image);
    // }
    // dd($image);
    $service = new UploadService();
    return $service->upload($image, "posts");
}
}

```

UploadService

```

<?php

namespace App\Services;

class UploadService
{
    public function upload($file, $path = null)
    {
        $alt = $file->getClientOriginalName();
        $fileName = time() . $alt;

        // dd($fileName);
        if ($path) {

```

```
        $file->move(\public_path() . "/img/$path", $fileName); //promeni putanju
    } else {
        $file->move(\public_path() . "/img", $fileName); //promeni putanju
    }

    return [
        'file_name' => $fileName,
        'alt' => $alt
    ];
}
}
```