Univerzitet u Beogradu

Fakultet organizacionih nauka

Katedra za elektronsko poslovanje

Laravel

Domaći zadatak 2

|  |  |  |
| --- | --- | --- |
| Ime | Prezime | Broj indeksa |
|  |  |  |
| Link ka Github-u |  | |

Sadržaj

[1 Korisnički zahtev 3](#_Toc154095131)

[2 Implementacija 4](#_Toc154095132)

[2.1 Model 4](#_Toc154095133)

[2.2 Migracije 4](#_Toc154095134)

[2.3 Kontroleri 6](#_Toc154095135)

[2.4 Api rute 7](#_Toc154095136)

[3 REST API 8](#_Toc154095137)

[4 Korisničko uputstvo 17](#_Toc154095138)

# Korisnički zahtev

Potrebno je kreirati aplikaciju koja predstavlja bekend deo za chat aplikaciju. Korisnici se mogu registrovati i prijaviti, što im omogućava pristup personalizovanom iskustvu. Jednom prijavljeni, korisnici imaju mogućnost da pregledaju i ažuriraju svoje profile, uključujući promenu lozinke za dodatnu sigurnost. Takođe, aplikacija omogućava korisnicima da pretražuju druge korisnike, što može biti korisno za pronalaženje prijatelja ili kolega unutar platforme.

Središnji deo aplikacije su funkcije vezane za chat, što uključuje kreiranje, pregled, ažuriranje i brisanje chat soba. Ove sobe mogu biti pretraživane na osnovu imena i privatnosti, omogućavajući korisnicima da pronađu prostor koji odgovara njihovim potrebama. Pored toga, aplikacija se bavi i upravljanjem porukama unutar ovih chat soba, kao i upravljanjem korisnicima unutar tih soba.

# Implementacija

## Model

Evo kratkog opisa svake klase i njihovih međusobnih veza:

1. **ChatRoom**: Klasa **ChatRoom** predstavlja chat sobu. Sadrži atribute poput imena sobe, privatnosti, maksimalnog broja učesnika i opisa. U vezi je sa dve druge klase: **Message** i **User**. Metoda **messages()** ukazuje na vezu "jedan prema mnogo" sa klasom **Message**, što znači da jedna chat soba može sadržati mnogo poruka. Druga veza je preko metode **participants()**, koja ukazuje na "mnogo-prema-mnogo" vezu sa klasom **User**, što znači da jedna chat soba može imati više učesnika (korisnika).

Slika na kojoj se nalazi tekst, snimak ekrana

Opis je automatski generisan

1. **ChatRoomUser**: Iako sadržaj ove klase nije potpuno prikazan, iz komentara se može zaključiti da je to pivot klasa koja upravlja "mnogo-prema-mnogo" vezom između **ChatRoom** i **User** klasa. U Laravelu, pivot klase se koriste za manipulaciju dodatnim podacima u pivot tabeli, što sugeriše da ova klasa omogućava dodatnu logiku ili atribute koji se odnose na odnos između chat soba i korisnika.

Slika na kojoj se nalazi tekst, snimak ekrana, softver

Opis je automatski generisan

1. **Message**: Klasa **Message** predstavlja poruku unutar chat sobe. Sadrži atribute kao što su tekst poruke, ID korisnika koji ju je poslao, ID chat sobe u kojoj je poslata, vreme slanja, čitanja i brisanja. Ova klasa ima dve veze: **user()** i **chatRoom()**. Metoda **user()** ukazuje na "jedan prema jedan" vezu sa klasom **User**, što znači da svaka poruka ima tačno jednog autora. Metoda **chatRoom()** takođe ukazuje na "jedan prema jedan" vezu, ali sa klasom **ChatRoom**, označavajući chat sobu u kojoj je poruka poslata.

Slika na kojoj se nalazi tekst, snimak ekrana, softver

Opis je automatski generisan

1. **User**: Klasa **User** predstavlja korisnika u chat aplikaciji. Sadrži standardne atribute za korisnika kao što su ime, email i lozinka. Ova klasa ima veze sa **ChatRoom** i **Message** klasama. Veza sa **ChatRoom** klasom je realizovana preko pivot klase **ChatRoomUser**, koja upravlja "mnogo-prema-mnogo" vezom, dok je veza sa **Message** klasom direktna, gde svaki korisnik može biti autor više poruka.

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Font

Opis je automatski generisan

## Migracije

Evo kratkog opisa svake od migracija na osnovu njihovih sadržaja:

* 2024\_01\_26\_221000\_create\_chat\_rooms\_table.php: Ova migracija kreira tabelu chat\_rooms u bazi podataka, dodajući kolone za ID, ime, privatnost (boolean), maksimalni broj učesnika i druge relevantne informacije vezane za chat sobe.

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

* 2024\_01\_26\_221819\_create\_messages\_table.php: Ova migracija stvara tabelu messages, sa kolonama za ID, tekst poruke, vreme slanja, vreme čitanja i druge atribute koji opisuju poruke unutar chat sobe.

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

* 2024\_01\_26\_223525\_add\_new\_columns\_to\_users\_table.php: Ova migracija dodaje nove kolone u tabelu users, uključujući kolone za profilnu sliku, datum rođenja i status korisnika, omogućavajući proširenje informacija o korisnicima.

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

* 2024\_01\_26\_223706\_create\_chat\_room\_user\_table.php: Ova migracija kreira pivot tabelu chat\_room\_user za upravljanje "mnogo-prema-mnogo" vezom između korisnika i chat soba, sa kolonama za ID korisnika i ID chat sobe.

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

* 2024\_01\_26\_224025\_rename\_column\_in\_chat\_rooms\_table.php: Ova migracija menja ime kolone private u is\_private u tabeli chat\_rooms, poboljšavajući jasnoću i konzistentnost naziva kolona.

Slika na kojoj se nalazi snimak ekrana, tekst, Font

Opis je automatski generisan

* 2024\_01\_26\_224430\_remove\_column\_in\_chat\_rooms\_table.php: Ova migracija uklanja kolonu user\_id iz tabele chat\_rooms, što sugeriše prilagođavanje strukture tabele za bolje upravljanje vezama između korisnika i chat soba.

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

* 2024\_01\_26\_224519\_add\_foreign\_keys\_messages\_table.php: Ova migracija dodaje strane ključeve u tabelu messages, povezujući je sa tabelama users i chat\_rooms kako bi se uspostavile relacije između poruka, korisnika koji su ih poslali, i chat soba u kojima su poslate. ​

Slika na kojoj se nalazi tekst, snimak ekrana, Font

Opis je automatski generisan

## Kontroleri

Kontroler ChatRoomUserController aplikaciji upravlja interakcijama sa pivot tabelom koja povezuje korisnike (users) i chat sobe (chat\_rooms). Evo detaljnog objašnjenja njegovih funkcionalnosti:

* Index Metoda: Metoda index služi za prikaz svih zapisa iz pivot tabele ChatRoomUser. Kada se pozove, ona dohvata sve zapisane veze između korisnika i chat soba i vraća ih kao JSON odgovor.
* Show Metoda: show metoda omogućava prikazivanje detalja specifičnog zapisa na osnovu njegovog ID-a. Ova metoda traži zapis u tabeli ChatRoomUser koristeći ID koji je prosleđen kao parametar i vraća detalje tog zapisa kao JSON odgovor.
* Store Metoda: Metoda store koristi se za kreiranje novog zapisa u pivot tabeli. Ova metoda prima HTTP zahtev sa potrebnim podacima (kao što su ID korisnika, ID chat sobe, vreme pridruživanja, uloga, vreme do kog je korisnik ućutkan), validira ove podatke, a zatim kreira novi zapis u tabeli ChatRoomUser. Ako podaci nisu validni, vraća greške kao JSON odgovor.
* Update Metoda: update metoda se koristi za ažuriranje postojećeg zapisa. Slično kao store metoda, prima HTTP zahtev sa podacima, vrši validaciju, i ažurira odgovarajući zapis u tabeli ChatRoomUser.
* Destroy Metoda: Ova metoda omogućava brisanje zapisa iz pivot tabele, ali koristi soft delete pristup. To znači da zapis nije trajno obrisan iz baze podataka, već je samo označen kao obrisan.
* Restore Metoda: restore metoda omogućava povratak soft deleted zapisa. Ako je zapis ranije "obrisan" (soft delete), ova metoda će ga ponovo aktivirati u tabeli.
* ForceDestroy Metoda: Za razliku od destroy metode, forceDestroy trajno briše zapis iz baze podataka. Ova metoda se koristi kada je potrebno potpuno ukloniti zapis, bez mogućnosti njegovog povratka.

<?php

namespace App\Http\Controllers;

use App\Models\ChatRoomUser;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class ChatRoomUserController extends Controller

{

     // Prikazuje sve zapise iz pivot tabele

     public function index()

     {

         $chatRoomUsers = ChatRoomUser::all();

         return response()->json($chatRoomUsers);

     }

     // Prikazuje specifičan zapis iz pivot tabele

     public function show($id)

     {

         $chatRoomUser = ChatRoomUser::findOrFail($id);

         return response()->json($chatRoomUser);

     }

     // Kreira novi zapis u pivot tabeli

     public function store(Request $request)

     {

         $validator = Validator::make($request->all(), [

             'user\_id' => 'required|integer|exists:users,id',

             'chat\_room\_id' => 'required|integer|exists:chat\_rooms,id',

             'joined\_at' => 'required|date',

             'role' => 'required|string',

             'muted\_until' => 'nullable|date'

         ]);

         if ($validator->fails()) {

             return response()->json($validator->errors(), 400);

         }

         $chatRoomUser = ChatRoomUser::create($validator->validated());

         return response()->json($chatRoomUser, 201);

     }

     // Ažurira postojeći zapis u pivot tabeli

     public function update(Request $request, $id)

     {

         $chatRoomUser = ChatRoomUser::findOrFail($id);

         $validator = Validator::make($request->all(), [

             'user\_id' => 'integer|exists:users,id',

             'chat\_room\_id' => 'integer|exists:chat\_rooms,id',

             'joined\_at' => 'date',

             'role' => 'string',

             'muted\_until' => 'nullable|date'

         ]);

         if ($validator->fails()) {

             return response()->json($validator->errors(), 400);

         }

         $chatRoomUser->update($validator->validated());

         return response()->json($chatRoomUser);

     }

     // Briše zapis iz pivot tabele

     public function destroy($id)  //ova metoda radi soft delete

     {

         $chatRoomUser = ChatRoomUser::findOrFail($id);

         $chatRoomUser->delete();

         return response()->json(null, 204);

     }

     public function restore($id)  //ako je neki objekat obrisan, mozemo da ga povratimo iz baze jer on nije bas stvarno obirsan (jer smo korisitli soft delete)

    {

        $chatRoomUser = ChatRoomUser::withTrashed()->findOrFail($id);

        $chatRoomUser->restore();

        return response()->json($chatRoomUser, 200);

    }

    public function forceDestroy($id)  //ova metoda bi obrisala zaista objekat iz baze

    {

        $chatRoomUser = ChatRoomUser::withTrashed()->findOrFail($id);

        $chatRoomUser->forceDelete();

        return response()->json(null, 204);

    }

}

Kontroler **AuthController** upravlja autentifikacijom i registracijom korisnika, kao i ažuriranjem njihovih profila. Ima sledece metode:

* Metoda **register** omogućava registraciju novih korisnika. Ona validira podatke poslate preko HTTP zahteva, uključujući ime, email, lozinku, i druge opcionalne podatke. Ako je validacija uspešna, metoda kreira novog korisnika u bazi podataka i vraća podatke o korisniku kao JSON odgovor. Ukoliko postoji slika profila, ona se učitava i smešta u odgovarajući direktorijum.
* Metoda **login** omogućava korisnicima da se prijave. Ona proverava da li email i lozinka odgovaraju onima u bazi podataka. Ako su kredencijali ispravni, metoda generiše autentifikacioni token i vraća ga kao deo JSON odgovora.
* **logout** metoda omogućava korisnicima da se odjave iz sistema. Ova metoda briše trenutni pristupni token korisnika, efektivno završavajući sesiju.
* Metoda **userProfile** vraća podatke o profilu trenutno ulogovanog korisnika.
* **updateUser** metoda omogućava korisnicima da ažuriraju svoje podatke, uključujući ime, email, lozinku, sliku profila, i druge lične podatke. Svi podaci se validiraju pre ažuriranja.
* Metoda koja omogućava pretragu korisnika u bazi podataka na osnovu različitih kriterijuma kao što su ime, email, datum rođenja, status, i druge informacije. Rezultati pretrage se vraćaju kao JSON odgovor.
* Metoda **changePassword** omogućava korisnicima da promene svoju lozinku. Ona prvo proverava da li trenutna lozinka odgovara onoj u bazi podataka, a zatim ažurira lozinku u slučaju uspeha.

<?php

namespace App\Http\Controllers;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Validator;

use Illuminate\Support\Str;

class AuthController extends Controller

{

    public function register(Request $request)

    {

        $validator = Validator::make($request->all(), [

            'name' => 'required|string|max:255',

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

            'password' => 'required|string|min:6',

            'profile\_image' => 'nullable|image|max:2048',

            'date\_of\_birth' => 'required|date|before:today',

            'status' => 'sometimes|string|max:255',

            'last\_online\_at' => 'sometimes|date',

            'bio' => 'nullable|string',

            'location' => 'nullable|string|max:255',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 400);

        }

        // Učitavanje slike profila, ako je prisutna

        $profileImagePath = null;

        if ($request->hasFile('profile\_image')) {

            $profileImagePath = $request->file('profile\_image')->store('profile\_images', 'public');

        }

        // Kreiranje korisnika

        $user = User::create([

            'name' => $request->name,

            'email' => $request->email,

            'password' => Hash::make($request->password),

            'profile\_image' => $profileImagePath,

            'date\_of\_birth' => $request->date\_of\_birth,

            'status' => $request->status ?? 'offline',

            'last\_online\_at' => now(),

            'bio' => $request->bio,

            'location' => $request->location,

        ]);

        return response()->json($user, 201);

    }

    public function login(Request $request)

    {

        $request->validate([

            'email' => 'required|email',

            'password' => 'required',

        ]);

        $user = User::where('email', $request->email)->first();

        if (!$user || !Hash::check($request->password, $user->password)) {

            return response()->json(['message' => 'Neispravni podaci za prijavu'], 401);

        }

        $token = $user->createToken(Str::random(40))->plainTextToken;

        return response()->json(['token' => $token], 200);

    }

    public function logout(Request $request)

    {

        $request->user()->currentAccessToken()->delete();

        return response()->json(['message' => 'Uspešno ste se odjavili']);

    }

    public function userProfile(Request $request)  //profil ulogovanog korisnika

    {

        return response()->json($request->user());

    }

    public function updateUser(Request $request)

    {

        $user = $request->user();

        $validator = Validator::make($request->all(), [

            'name' => 'string|max:255',

            'email' => 'string|email|max:255|unique:users,email,' . $user->id,

            'password' => 'sometimes|min:6',

            'profile\_image' => 'nullable|image|max:2048',

            'date\_of\_birth' => 'date|before:today',

            'bio' => 'nullable|string',

            'location' => 'string|max:255',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 400);

        }

        // Ažuriranje podataka

        if ($request->has('name')) {

            $user->name = $request->name;

        }

        if ($request->has('email')) {

            $user->email = $request->email;

        }

        if ($request->has('password')) {

            $user->password = Hash::make($request->password);

        }

        if ($request->hasFile('profile\_image')) {

            $profileImagePath = $request->file('profile\_image')->store('profile\_images', 'public');

            $user->profile\_image = $profileImagePath;

        }

        if ($request->has('date\_of\_birth')) {

            $user->date\_of\_birth = $request->date\_of\_birth;

        }

        if ($request->has('bio')) {

            $user->bio = $request->bio;

        }

        if ($request->has('location')) {

            $user->location = $request->location;

        }

        $user->save();

        return response()->json($user);

    }

    public function searchUsers(Request $request)

{

    $query = User::query();

    // Filtriranje na osnovu imena

    if ($request->has('name')) {

        $query->where('name', 'like', '%' . $request->name . '%');

    }

    // Filtriranje na osnovu emaila

    if ($request->has('email')) {

        $query->where('email', 'like', '%' . $request->email . '%');

    }

    // Filtriranje na osnovu datuma rođenja

    if ($request->has('date\_of\_birth')) {

        $query->whereDate('date\_of\_birth', '=', $request->date\_of\_birth);

    }

    // Filtriranje na osnovu statusa

    if ($request->has('status')) {

        $query->where('status', 'like', '%' . $request->status . '%');

    }

    // Filtriranje na osnovu poslednjeg online statusa

    if ($request->has('last\_online\_at')) {

        $query->whereDate('last\_online\_at', '=', $request->last\_online\_at);

    }

    // Filtriranje na osnovu biografije

    if ($request->has('bio')) {

        $query->where('bio', 'like', '%' . $request->bio . '%');

    }

    // Filtriranje na osnovu lokacije

    if ($request->has('location')) {

        $query->where('location', 'like', '%' . $request->location . '%');

    }

    // Izvršavanje upita i vraćanje rezultata

    $users = $query->get();

    return response()->json($users);

    }

    public function changePassword(Request $request)

    {

        $user = $request->user();

        // Validacija podataka

        $request->validate([

            'current\_password' => 'required',

            'new\_password' => 'required|string|min:6|confirmed',

        ]);

        // Provera da li trenutna lozinka odgovara

        if (!Hash::check($request->current\_password, $user->password)) {

            return response()->json(['message' => 'Trenutna lozinka nije tačna'], 400);

        }

        // Ažuriranje lozinke

        $user->password = Hash::make($request->new\_password);

        $user->save();

        return response()->json(['message' => 'Lozinka je uspešno promenjena']);

    }

}

## Api rute

1. **/api/register** (POST):
   * Ova ruta omogućava registraciju korisnika u aplikaciju. Koristi **AuthController::register** metodu za obradu registracije korisnika.
2. **/api/login** (POST):
   * Ova ruta omogućava prijavljivanje korisnika u aplikaciju. Koristi **AuthController::login** metodu za obradu prijave korisnika.
3. **/api/users/search** (GET):
   * Ova ruta omogućava pretragu korisnika po imenu. Koristi **AuthController::searchUsers** metodu za pretragu korisnika.
4. **/api/chat-rooms** (API resurs):
   * Ovaj API resurs omogućava rad sa sobama za chat. Može se koristiti za kreiranje, pregled, ažuriranje i brisanje soba za chat.
5. **/api/chat-rooms/search** (GET):
   * Ova ruta omogućava pretragu soba za chat po imenu i privatnosti. Koristi **ChatRoomController::search** metodu za pretragu soba.
6. **/api/messages** (API resurs):
   * Ovaj API resurs omogućava rad sa porukama. Može se koristiti za kreiranje, pregled, ažuriranje i brisanje poruka.
7. **/api/chat-room-users** (API resurs):
   * Ovaj API resurs omogućava rad sa korisnicima koji su članovi soba za chat. Može se koristiti za pregled, kreiranje, ažuriranje i brisanje korisnika iz soba.
8. **/api/logout** (POST):
   * Ova ruta omogućava odjavljivanje prijavljenih korisnika iz aplikacije. Koristi **AuthController::logout** metodu za obradu odjave.
9. **/api/user/profile** (GET):
   * Ova ruta omogućava prijavljenim korisnicima da dobiju informacije o svom profilu. Koristi **AuthController::userProfile** metodu za dobijanje profila korisnika.
10. **/api/user/update** (PUT):
    * Ova ruta omogućava prijavljenim korisnicima da ažuriraju svoj profil. Koristi **AuthController::updateUser** metodu za ažuriranje profila korisnika.
11. **/api/change-password** (POST):
    * Ova ruta omogućava prijavljenim korisnicima da promene svoju lozinku. Koristi **AuthController::changePassword** metodu za promenu lozinke.

# REST API

U ovom poglavlju ce biti prikazani detalji o api rutama za kontrolere.

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih caskanja |
| HTTP metoda | GET |
| URL | /api/chat-rooms |
| URL parametri | nema |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | {[      {          "id": 1,          "name": "maiores",          "is\_private": 0,          "max\_participants": 9,          "description": "Ad eos et id esse ea suscipit et.",          "created\_at": "2024-01-26T22:50:35.000000Z",          "updated\_at": "2024-01-26T22:50:35.000000Z"      },      {          "id": 2,          "name": "in",          "is\_private": 1,          "max\_participants": 5,          "description": "Illum odio voluptas facilis fugiat et cum.",          "created\_at": "2024-01-26T22:50:35.000000Z",          "updated\_at": "2024-01-26T22:50:35.000000Z"      },      ..    } |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje chat rooma |
| HTTP metoda | POST |
| URL | /api/chat-rooms |
| URL parametri | ?name=soba&is\_private=0&max\_participants=5 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "name": "soba",      "is\_private": "0",      "max\_participants": "5",      "updated\_at": "2024-01-30T20:48:39.000000Z",      "created\_at": "2024-01-30T20:48:39.000000Z",      "id": 31  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje chat rooma |
| HTTP metoda | PUT |
| URL | /api/ chat-rooms |
| URL parametri | 31?name=soba&is\_private=0&max\_participants=5 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "id": 31,      "name": "soba",      "is\_private": "0",      "max\_participants": "5",      "description": **null**,      "created\_at": "2024-01-30T20:48:39.000000Z",      "updated\_at": "2024-01-30T20:48:39.000000Z"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Brisanje chat rooma |
| HTTP metoda | DELETE |
| URL | /api/ chat-rooms |
| URL parametri | 31 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | -No Content- |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih poruka |
| HTTP metoda | GET |
| URL | /api/ messages |
| URL parametri | nema |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | [      {          "id": 1,          "body": "Nostrum a sint debitis non corporis eum autem. Voluptas aperiam aut ut soluta voluptates id culpa. Iste ut eius enim necessitatibus. Dolores quidem dolor iure totam ut ut similique.",          "sent\_at": "2024-01-24 07:20:54",          "read\_at": "2024-01-19 18:56:47",          "deleted\_at": **null**,          "created\_at": "2024-01-26T22:50:35.000000Z",          "updated\_at": "2024-01-26T22:50:35.000000Z",          "user\_id": 11,          "chat\_room\_id": 11      },      {          "id": 5,          "body": "In laudantium possimus et. Repellendus non et excepturi et accusamus voluptatem. Sed molestiae quis velit aut saepe suscipit praesentium cupiditate. Libero consequuntur sit dolorem et accusantium et.",          "sent\_at": "2024-01-14 17:46:44",          "read\_at": "2024-01-09 21:32:33",          "deleted\_at": **null**,          "created\_at": "2024-01-26T22:50:35.000000Z",          "updated\_at": "2024-01-26T22:50:35.000000Z",          "user\_id": 15,          "chat\_room\_id": 15      }, |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje poruke |
| HTTP metoda | POST |
| URL | /api/messages |
| URL parametri | ?body=body poruke&user\_id=1&chat\_room\_id=1 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "body": "body poruke",      "user\_id": "1",      "chat\_room\_id": "1",      "updated\_at": "2024-01-30T20:51:36.000000Z",      "created\_at": "2024-01-30T20:51:36.000000Z",      "id": 11  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje poruke |
| HTTP metoda | PUT |
| URL | /api/ messages |
| URL parametri | 11?body=body poruke&user\_id=1&chat\_room\_id=1 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "id": 11,      "body": "body poruke",      "sent\_at": **null**,      "read\_at": **null**,      "deleted\_at": **null**,      "created\_at": "2024-01-30T20:51:36.000000Z",      "updated\_at": "2024-01-30T20:51:36.000000Z",      "user\_id": "1",      "chat\_room\_id": "1"  } |
| Format izlaznih parametara | application/json |
| Opis funkcije | Brisanje poruke |
| HTTP metoda | DELETE |
| URL | /api/ messages |
| URL parametri | 11 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | --No Content-- |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Registracija korisnika |
| HTTP metoda | POST |
| URL | /api/register |
| URL parametri | r?name=korisnik&email=korisnik@gmail.com&password=password&date\_of\_birth=2020-02-02 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "name": "korisnik",      "email": "korisnik@gmail.com",      "profile\_image": **null**,      "date\_of\_birth": "2020-02-02",      "status": "offline",      "last\_online\_at": "2024-01-30T20:57:55.471075Z",      "bio": **null**,      "location": **null**,      "updated\_at": "2024-01-30T20:57:55.000000Z",      "created\_at": "2024-01-30T20:57:55.000000Z",      "id": 32  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prijava korisnika |
| HTTP metoda | POST |
| URL | /api/login |
| URL parametri | email=korisnik@gmail.com&password=password |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "token": "4|7hPvAbxpbvswspwHiqA5VXbO0MBXhAlBGc15dhf8d3e09e76"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Odjava korisnika |
| HTTP metoda | POST |
| URL | /api/logout |
| URL parametri | nema |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Logged out"  } |
| Format izlaznih parametara | application/json |

# Korisničko uputstvo

Prijava korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, softver, broj

Opis je automatski generisan

Prikaz svih caskanja

Slika na kojoj se nalazi tekst, snimak ekrana, softver, broj

Opis je automatski generisan

Dodavanje chat room caskanja

Slika na kojoj se nalazi tekst, snimak ekrana, softver, broj

Opis je automatski generisan

Izmena chat rooma

Slika na kojoj se nalazi tekst, snimak ekrana, broj, Font

Opis je automatski generisan

Brisanje chat rooma

Slika na kojoj se nalazi tekst, snimak ekrana, broj, Font

Opis je automatski generisan

Prikaz svih poruka iz baze

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan

Slanje tj kreiranje poruke

Slika na kojoj se nalazi tekst, snimak ekrana, softver, broj

Opis je automatski generisan

Izmena poruke

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan

Brisanje poruke

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan

Pretraga chat rooms po raznim kriterijumima

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan

Pretraga korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Veb stranica

Opis je automatski generisan

Registracija korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan

Odjava korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, broj, softver

Opis je automatski generisan