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 sistem e-uprave koji omogućava registraciju i prijavljivanje korisnika, kao i pregled i pretragu različitih usluga i zahteva. Korisnik se može registrovati i prijaviti u sistem, gde prilikom registracije unosi osnovne podatke kao što su ime, prezime, JMBG, broj lične karte, adresu prebivališta i slično. Nakon registracije, korisnik može da se prijavi i pristupi svojim informacijama, kao i da pregleda različite usluge koje pruža e-uprava. Ove usluge uključuju različite administrativne i birokratske procese. Korisnik može podneti zahtev za određenu uslugu, gde se svaki zahtev beleži sa informacijama kao što su status zahteva, prioritet, rok obrade i dodatne napomene. Dodatno, postoji funkcionalnost za izvoz podataka o korisnicima u CSV format, što omogućava lakši pregled i analizu korisničkih podataka. Aplikacija nudi i opciju za pretragu korisnika na osnovu određenih kriterijuma kao što su mesto rođenja, opština prebivališta i datum rođenja, što olakšava lociranje specifičnih korisničkih profila. Na kraju, korisnici imaju opciju da se odjave iz sistema kada završe sa korišćenjem aplikacije.

# Implementacija

## Model

Postoje četiri različita modela: Usluga, User, Dokument i Zahtev.

Model "Usluga" predstavlja entitet usluga sa atributima kao što su naziv, opis, cena, kategorija, vreme obrade i drugi. Ovaj model ima i relaciju sa modelom "Zahtev", omogućavajući povezivanje usluga sa zahtevima korisnika.

Model "User" je odgovoran za upravljanje korisničkim podacima. On sadrži informacije kao što su ime, prezime, email adresa, uloga i drugi lični podaci korisnika. Ovaj model omogućava autentifikaciju i autorizaciju korisnika putem sanctum paketa.

Model "Dokument" se koristi za čuvanje informacija o različitim vrstama dokumenata, kao što su lične karte, pasoši ili izvodi. Ovaj model povezan je sa korisničkim modelom kako bi se omogućila veza između korisnika i njihovih dokumenata.

Model "Zahtev" predstavlja zahtev korisnika za određenom uslugom. Sadrži informacije o statusu zahteva, vremenu podnošenja zahteva, prioritetu i dodatnim napomenama. Ovaj model takođe ima veze sa korisničkim modelom kako bi se pratila povezanost sa konkretnim korisnikom, kao i sa modelom "Usluga" kako bi se odredila usluga na koju se zahtev odnosi.

## Migracije

Prva migracija se odnosi na tabelu "uslugas" i definiše strukturu za skladištenje informacija o uslugama. Tabela ima polja naziv, opis, cena, kategorija, vreme obrade, dokumentacija potrebna, digitalni potpis potreban i prioritet. Ova migracija takođe uključuje automatski generisana polja za praćenje vremena kreiranja i ažuriranja redova.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::create('uslugas', function (Blueprint $table) {

            $table->id();

            $table->string('naziv');

            $table->text('opis')->nullable();

            $table->decimal('cena', 8, 2)->nullable();

            $table->string('kategorija');

            $table->integer('vreme\_obrade')->nullable();

            $table->text('dokumentacija\_potrebna')->nullable();

            $table->boolean('digitalni\_potpis\_potreban')->default(false);

            $table->string('prioritet')->default('medium');

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('uslugas');

    }

};

Na slican nacin su implementirane i ostale migracije tipa „create“ – migracija za kreiranje tabele dokumenta, zahtevi…

Sledeca migracija dodaje nova polja u tabelu "users" kao što su prezime, ime oca, JMBG, broj lične karte, datum rođenja, mesto rođenja, adresa prebivališta i opština prebivališta. Ovo proširuje informacije o korisnicima kako bi se bolje upravljalo njihovim ličnim podacima.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::table('users', function (Blueprint $table) {

            $table->string('prezime');

            $table->string('imeOca')->nullable();

            $table->string('jmbg')->unique();

            $table->string('brLicneKarte')->unique();

            $table->date('datumRodjenja');

            $table->string('mestoRodjenja');

            $table->string('adresaPrebivalista');

            $table->string('opstinaPrebivalista');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::table('users', function (Blueprint $table) {

            $table->dropColumn([

                'prezime',

                'imeOca',

                'jmbg',

                'brLicneKarte',

                'datumRodjenja',

                'mestoRodjenja',

                'adresaPrebivalista',

                'opstinaPrebivalista'

            ]);

        });

    }

};

Zatim imamo migraciju koja menja naziv polja "name" u tabeli "users" u "ime" radi bolje konzistentnosti i razumevanja podataka.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::table('users', function (Blueprint $table) {

            $table->renameColumn('name','ime');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::table('users', function (Blueprint $table) {

            $table->renameColumn('ime','name');

        });

    }

};

Imamo takodje i migraciju koja je zaduzena za kreiranje spoljnih kljuceva.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::table('zahtevs', function (Blueprint $table) {

            $table->foreign('korisnik\_id')->references('id')->on('users') ;

            $table->foreign('usluga\_id')->references('id')->on('uslugas') ;

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::table('zahtevs', function (Blueprint $table) {

            $table->dropForeign(['korisnik\_id']);

            $table->dropForeign(['usluga\_id']);

        });

    }

};

Poslednja migracija uklanja polje "velicina" iz tabele "dokuments" jer nije više potrebno za ovu tabelu.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

return new class extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::table('dokuments', function (Blueprint $table) {

            $table->dropColumn('velicina');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::table('dokuments', function (Blueprint $table) {

            $table->string('velicina');

        });

    }

};

## Kontroleri

Kontroler "UslugaController" upravlja akcijama povezanim sa uslugama, uključujući prikazivanje svih usluga, kreiranje nove usluge, ažuriranje postojeće usluge i brisanje usluge. Koristi se za manipulaciju podacima o uslugama, proveru i validaciju unesenih podataka te vraćanje odgovarajućih odgovora.

<?php

namespace App\Http\Controllers;

use App\Http\Resources\UslugaResource;

use App\Models\Usluga;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class UslugaController extends Controller

{

    public function index()

    {

        $usluge = Usluga::all();

        return response()->json(UslugaResource::collection($usluge));

    }

    public function store(Request $request)

    {

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

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

            'opis' => 'required|string|max:1000',

            'cena' => 'required|numeric|min:0',

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

            'vreme\_obrade' => 'required|integer|min:0',

            'dokumentacija\_potrebna' => 'sometimes|string|max:1000',

            'digitalni\_potpis\_potreban' => 'required|boolean',

            'prioritet' => 'required|string|in:low,medium,high',

        ]);

        if ($validator->fails()) {

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

        }

        $usluga = Usluga::create($validator->validated());

        return response()->json(new UslugaResource($usluga), 201);

    }

    public function update(Request $request, $id)

    {

        $usluga = Usluga::find($id);

        if (!$usluga) {

            return response()->json(['message' => 'Usluga not found'], 404);

        }

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

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

            'opis' => 'required|string|max:1000',

            'cena' => 'required|numeric|min:0',

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

            'vreme\_obrade' => 'required|integer|min:0',

            'dokumentacija\_potrebna' => 'sometimes|string|max:1000',

            'digitalni\_potpis\_potreban' => 'required|boolean',

            'prioritet' => 'required|string|in:low,medium,high',

        ]);

        if ($validator->fails()) {

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

        }

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

        return response()->json(new UslugaResource($usluga));

    }

    public function destroy($id)

    {

        $usluga = Usluga::find($id);

        if (!$usluga) {

            return response()->json(['message' => 'Usluga not found'], 404);

        }

        $usluga->delete();

        return response()->json(['message' => 'Usluga deleted successfully']);

    }

}

Kontroler "ZahtevController" se bavi akcijama povezanim sa zahtevima korisnika za određenim uslugama. Ovo uključuje prikazivanje svih zahteva, kreiranje novih zahteva, ažuriranje postojećih zahteva i brisanje zahteva. Kontroler takođe obavlja validaciju unesenih podataka, proveru postojanja povezanih resursa i vraćanje odgovarajućih odgovora.

<?php

namespace App\Http\Controllers;

use App\Http\Resources\ZahtevResource;

use App\Models\Zahtev;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class ZahtevController extends Controller

{

    public function index()

    {

        $zahtevi = Zahtev::all();

        return response()->json(ZahtevResource::collection($zahtevi));

    }

    public function store(Request $request)

    {

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

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

            'usluga\_id' => 'required|exists:usluge,id',

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

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

            'processed\_at' => 'nullable|date',

            'request\_priority' => 'required|string|max:255',

            'additional\_notes' => 'nullable|string|max:1000',

            'processing\_deadline' => 'nullable|date',

        ]);

        if ($validator->fails()) {

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

        }

        $zahtev = Zahtev::create($validator->validated());

        return response()->json(new ZahtevResource($zahtev), 201);

    }

    public function update(Request $request, $id)

    {

        $zahtev = Zahtev::find($id);

        if (!$zahtev) {

            return response()->json(['message' => 'Zahtev not found'], 404);

        }

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

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

            'usluga\_id' => 'required|exists:usluge,id',

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

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

            'processed\_at' => 'nullable|date',

            'request\_priority' => 'required|string|max:255',

            'additional\_notes' => 'nullable|string|max:1000',

            'processing\_deadline' => 'nullable|date',

        ]);

        if ($validator->fails()) {

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

        }

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

        return response()->json(new ZahtevResource($zahtev));

    }

    public function destroy($id)

    {

        $zahtev = Zahtev::find($id);

        if (!$zahtev) {

            return response()->json(['message' => 'Zahtev not found'], 404);

        }

        $zahtev->delete();

        return response()->json(['message' => 'Zahtev deleted successfully']);

    }

}

U oba kontrolera su implementirane sledece metode

* "index": Služi za prikazivanje liste resursa.
* "store": Služi za kreiranje novog resursa.
* "update": Služi za ažuriranje postojećeg resursa.
* "destroy": Služi za brisanje postojećeg resursa.
* "show": Služi za prikazivanje detalja određenog resursa.

Kontroler "AuthController" odgovoran je za autentifikaciju i registraciju korisnika u aplikaciji. Metoda "register" omogućava registraciju novih korisnika sa validacijom unesenih podataka, a nakon uspešne registracije generiše se i šalje JWT token za autentifikaciju. Metoda "login" omogućava prijavu postojećih korisnika, takođe uz upotrebu JWT tokena. Kontroler takođe ima metodu "logout" za odjavu korisnika, "searchUsers" za pretragu korisnika prema različitim kriterijumima i "exportUsersToCsv" za izvoz podataka o korisnicima u CSV formatu. Takođe, pruža i mogućnost dobijanja informacija o trenutno prijavljenom korisniku putem metode "getCurrentUser".

<?php

namespace App\Http\Controllers;

use App\Http\Resources\UserResource;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Validator;

class AuthController extends Controller

{

    public function register(Request $request)

{

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

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

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

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

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

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

        'jmbg' => 'required|string|max:13|unique:users',

        'brLicneKarte' => 'required|string|max:20|unique:users',

        'datumRodjenja' => 'required|date',

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

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

        'opstinaPrebivalista' => 'required|string|max:255'

    ]);

    if ($validator->fails()) {

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

    }

    $user = User::create([

        'ime' => $request->ime,

        'prezime' => $request->prezime,

        'imeOca' => $request->imeOca,

        'email' => $request->email,

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

        'jmbg' => $request->jmbg,

        'brLicneKarte' => $request->brLicneKarte,

        'datumRodjenja' => $request->datumRodjenja,

        'mestoRodjenja' => $request->mestoRodjenja,

        'adresaPrebivalista' => $request->adresaPrebivalista,

        'opstinaPrebivalista' => $request->opstinaPrebivalista

    ]);

    $token = $user->createToken('auth\_token')->plainTextToken;

    return response()->json(['access\_token' => $token, 'token\_type' => 'Bearer']);

}

    public function login(Request $request)

    {

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

            'email' => 'required|email',

            'password' => 'required',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 401);

        }

        if (!Auth::attempt($request->only('email', 'password'))) {

            return response()->json(['message' => 'Unauthorized'], 401);

        }

        $user = User::where('email', $request['email'])->firstOrFail();

        $token = $user->createToken('auth\_token')->plainTextToken;

        return response()->json(['access\_token' => $token, 'token\_type' => 'Bearer']);

    }

    public function logout(Request $request)

    {

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

        return response()->json(['message' => 'Logged out successfully']);

    }

    public function searchUsers(Request $request)

    {

        $query = User::query();

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

            $query->where('opstinaPrebivalista', $request->opstinaPrebivalista);

        }

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

            $query->where('mestoRodjenja', $request->mestoRodjenja);

        }

        if ($request->has('datumRodjenjaStart') && $request->has('datumRodjenjaEnd')) {

            $query->whereBetween('datumRodjenja', [$request->datumRodjenjaStart, $request->datumRodjenjaEnd]);

        } elseif ($request->has('datumRodjenjaStart')) {

            $query->where('datumRodjenja', '>=', $request->datumRodjenjaStart);

        } elseif ($request->has('datumRodjenjaEnd')) {

            $query->where('datumRodjenja', '<=', $request->datumRodjenjaEnd);

        }

        $users = $query->get();

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

    }

    public function exportUsersToCsv()

    {

        $fileName = 'users\_' . date('Y-m-d\_His') . '.csv';

        $filePath = storage\_path('app/public/' . $fileName);

        $users = User::all();

        $columns = array\_keys($users->first()->toArray());

        $file = fopen($filePath, 'w');

        fputcsv($file, $columns);

        foreach ($users as $user) {

            fputcsv($file, $user->toArray());

        }

        fclose($file);

        return response()->json(['message' => 'File saved successfully', 'file' => $fileName]);

    }

    public function getCurrentUser(Request $request)

    {

        $user = $request->user();

        return new UserResource($user);

    }

}

## Api rute

U ovom radu definisane su različite rute koje omogućavaju interakciju sa apijem. Rute su organizovane tako da pruže korisnicima različite mogućnosti u skladu sa njihovim pravima i potrebama.

Prvo, imamo rute za prikazivanje podataka, kao što su "/usluge" i "/zahtevi". Ove rute omogućavaju korisnicima da pregledaju dostupne usluge i zahteve bez potrebe za prijavom, čime se olakšava istraživanje resursa sistema.

Za registraciju i prijavu korisnika, postoje rute "/register" i "/login". Kroz ove rute, korisnici mogu da se registruju i prijave na aplikaciju kako bi dobili pristup dodatnim funkcionalnostima i personalizovanim sadržajima.

Za zaštitu podataka i resursa, implementirane su rute "/logout" za odjavljivanje korisnika i "/user" za dobijanje informacija o trenutno prijavljenom korisniku. Ove rute su zaštićene "auth:sanctum" middleware-om kako bi se obezbedila sigurnost i privatnost korisničkih podataka.

Dodatno, postoji mogućnost pretrage korisnika putem "/searchUsers" rute i izvoza korisničkih podataka u CSV formatu putem "/exportUsersToCsv" rute. Ovo pruža dodatne funkcionalnosti za upravljanje i analizu korisničkih informacija.

// Zaštita svih ruta koje izvršavaju delete, update i store akcije

Route::middleware(['auth:sanctum'])->group(function () {

    Route::resource('/usluge', UslugaController::class)->only(['store', 'update', 'destroy']);

    Route::resource('/zahtevi', ZahtevController::class)->only(['store', 'update', 'destroy']);

    Route::post('/logout', [AuthController::class, 'logout']);

});

// Ostale rute koje su nezaštićene

Route::get('/usluge', [UslugaController::class, 'index']);

Route::get('/zahtevi', [ZahtevController::class, 'index']);

Route::post('/register', [AuthController::class, 'register']);

Route::post('/login', [AuthController::class, 'login']);

Route::get('/searchUsers', [AuthController::class, 'searchUsers']);

Route::get('/exportUsersToCsv', [AuthController::class, 'exportUsersToCsv']);

Route::middleware('auth:sanctum')->get('/user', [AuthController::class, 'getCurrentUser']);

# REST API

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

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih usluga |
| HTTP metoda | GET |
| URL | /api/usluge |
| URL parametri | nema |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | [      {          "id": 1,          "naziv": "cupiditate maxime quis",          "opis": "Voluptatibus sit fugiat et reprehenderit tempora dolores et.",          "cena": "538.00",          "kategorija": "commodi",          "vreme\_obrade": 9,          "dokumentacija\_potrebna": "Qui ex corrupti minus alias vitae sit non.",          "digitalni\_potpis\_potreban": 1,          "prioritet": "low"      },      {          "id": 2,          "naziv": "excepturi iure facilis",          "opis": "Quo eum aliquam et aperiam et fugiat culpa.",          "cena": "52.00",          "kategorija": "unde",          "vreme\_obrade": 7,          "dokumentacija\_potrebna": "Recusandae ut tenetur autem aut qui voluptatem.",          "digitalni\_potpis\_potreban": 0,          "prioritet": "high"      },  …  **}** |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje usluge |
| HTTP metoda | POST |
| URL | /api/usluge |
| URL parametri | ?naziv=dadsa&opis=dsada&cena=111&kategorija=dsadas&vreme\_obrade=5&digitalni\_potpis\_potreban=0&prioritet=low |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "id": 9,      "naziv": "dadsa",      "opis": "dsada",      "cena": "111",      "kategorija": "dsadas",      "vreme\_obrade": "5",      "dokumentacija\_potrebna": **null**,      "digitalni\_potpis\_potreban": "0",      "prioritet": "low"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje usluge |
| HTTP metoda | PUT |
| URL | /api/usluge |
| URL parametri | 9?naziv=USLUGA 1&opis=dsada&cena=111&kategorija=dsadas&vreme\_obrade=5&digitalni\_potpis\_potreban=0&prioritet=low |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "id": 9,      "naziv": "USLUGA 1",      "opis": "dsada",      "cena": "111",      "kategorija": "dsadas",      "vreme\_obrade": "5",      "dokumentacija\_potrebna": **null**,      "digitalni\_potpis\_potreban": "0",      "prioritet": "low"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Brisanje usluge |
| HTTP metoda | DELETE |
| URL | /api/usluge |
| URL parametri | 12 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Usluga deleted successfully"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih zahteva |
| HTTP metoda | GET |
| URL | /api/zahtevi |
| URL parametri | nema |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | {          "id": 1,          "status": **null**,          "submitted\_at": "2024-01-01 04:44:06",          "processed\_at": **null**,          "request\_priority": "medium",          "additional\_notes": "Placeat hic eum consequatur illum ea ex.",          "processing\_deadline": "2024-01-05 17:54:10"      },      {          "id": 2,          "status": **null**,          "submitted\_at": "2024-01-07 12:06:29",          "processed\_at": "2024-01-06 06:26:15",          "request\_priority": "high",          "additional\_notes": "Quas blanditiis magnam non ab quod dolore.",          "processing\_deadline": "2024-01-06 11:39:25"      },  … |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje zahteva |
| HTTP metoda | POST |
| URL | /api/zahtevi |
| URL parametri | ?korisnik\_id=1&usluga\_id=1&status=sda&submitted\_at=2024-01-01 04:44:06&request\_priority=0 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "ime": "ImeKorisnika",      "prezime": "PrezimeKorisnika",      "imeOca": "ImeOcaKorisnika",      "email": "emaila@primer.com",      "password": "sifra123",      "uloga": "korisnik",      "jmbg": "1234167890123",      "brLicneKarte": "0121456789",      "datumRodjenja": "2000-01-01",      "mestoRodjenja": "GradRodjenja",      "br\_lk": "01113456789",      "adresaPrebivalista": "Adresa 123",      "opstinaPrebivalista": "Opstina",      "korisnik\_id": "1",      "usluga\_id": "1",      "status": "sda",      "submitted\_at": "2024-01-01 04:44:06",      "request\_priority": "0"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje zahteva |
| HTTP metoda | PUT |
| URL | /api/ zahtevi |
| URL parametri | /2/?korisnik\_id=1&usluga\_id=1&status\_zahteva=sda&submitted\_at=2024-01-01 04:44:06&request\_priority=0 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "id": 2,      "status": **null**,      "submitted\_at": "2024-01-01 04:44:06",      "processed\_at": "2024-01-06 06:26:15",      "request\_priority": "0",      "additional\_notes": "Quas blanditiis magnam non ab quod dolore.",      "processing\_deadline": "2024-01-06 11:39:25"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Brisanje zahteva |
| HTTP metoda | DELETE |
| URL | /api/ zahtevi |
| URL parametri | 2 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Zahtev deleted successfully"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Registracija korisnika |
| HTTP metoda | POST |
| URL | /api/register |
| URL parametri | nema |
| HTTP body parametri | {      "ime": "ImeKorisnika",      "prezime": "PrezimeKorisnika",      "imeOca": "ImeOcaKorisnika",      "email": "emaila@primer.com",      "password": "sifra123",      "uloga": "korisnik",      "jmbg": "1234167890123",      "brLicneKarte": "0121456789",      "datumRodjenja": "2000-01-01",      "mestoRodjenja": "GradRodjenja",      "br\_lk": "01113456789",      "adresaPrebivalista": "Adresa 123",      "opstinaPrebivalista": "Opstina"  } |
| Format HTTP body parametara | json |
| Izlazni parametri | {      "access\_token": "2|t9brjG2nA34DMf7v6xfYxmYAliNBRDZ00hrdAM2U81135eb8",      "token\_type": "Bearer"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prijava korisnika |
| HTTP metoda | POST |
| URL | /api/login |
| URL parametri | ?email=emaila@primer.com&password=sifra123 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "access\_token": "3|k0Y3REv5XD3P045deOlCsS8uxWqRcMaEGCbX8qjsc1aa8c91",      "token\_type": "Bearer"  } |
| 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 successfully"  } |
| Format izlaznih parametara | application/json |

# Korisničko uputstvo

Prilikom pokretanja aplikacije je potrebno registrovati novog korisnika.

Slika na kojoj se nalazi tekst, broj, linija, softver

Opis je automatski generisan

Nakon registracije je potrebno da se korisnik uloguje

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

Opis je automatski generisan

Kako bi korisnik imao pristup zasticenim rutama, neophodno je da dobijeni token kopira u Authorization

Slika na kojoj se nalazi tekst, snimak ekrana

Opis je automatski generisan

Prikaz svih usluga

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

Opis je automatski generisan

Kreiranje usluge

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

Opis je automatski generisan

Izmena usluge

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

Opis je automatski generisan

Brisanje usluge

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

Opis je automatski generisan

Metoda koja vraca trenutno ulogovanog korisnika

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

Opis je automatski generisan

Slika na kojoj se nalazi tekst, broj, softver, linija

Opis je automatski generisan

Kreiranje zahteva

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

Opis je automatski generisan

Azuriranje zahteva

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

Opis je automatski generisan

Brisanje zahteva

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

Opis je automatski generisan

Metoda za pretragu korisnika

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

Opis je automatski generisan

Metoda za eksportovanje korisnika u csv fajl

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

Opis je automatski generisan