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 razviti web aplikaciju koristeći Laravel framework koja omogućava firmama da efikasno upravljaju svojim zaposlenima, dokumentima i zadacima. Glavna svrha aplikacije je da pruži organizovano i centralizovano rešenje za sve aspekte upravljanja ljudskim resursima i dokumentacijom unutar firme.

U segmentu registracije i upravljanja zaposlenima, aplikacija omogućava registraciju zaposlenih sa osnovnim informacijama kao što su pozicija, odeljenje, datum početka rada, datum kraja ugovora, plata i pripadnost firmi. Svaki zaposleni ima mogućnost pristupa svojim zadacima i dokumentima, što omogućava bolju organizaciju i efikasnost u radu.

Što se tiče upravljanja firmama, aplikacija omogućava kreiranje i upravljanje detaljima firmi kao što su naziv, PIB, matični broj, adresa, kontakt telefon i email. Firme kroz ovu aplikaciju mogu imati centralizovan pristup informacijama o svojim zaposlenima i dokumentima koji su sa njima povezani.

Upravljanje dokumentima je još jedna važna funkcionalnost aplikacije. Dokumenti, ili fajlovi, mogu se dodavati, uređivati i pregledavati unutar firme. Svaki dokument sadrži naziv, opis, putanju čuvanja i povezan je sa određenom firmom, što doprinosi boljem upravljanju dokumentacijom i informacijama unutar firme.

Za zaposlene se mogu kreirati zadaci sa specifičnim informacijama kao što su naziv, opis, rok i status, uključujući opcije kao što su završeno, otkazano ili u izradi. Svaki zadatak je direktno povezan sa određenim zaposlenim, što omogućava efikasno praćenje i upravljanje radnim zadacima unutar firme.

Upravljanje privilegijama je ključno za bezbednost i pravilno funkcionisanje aplikacije. Aplikacija pruža mogućnost dodeljivanja različitih nivoa pristupa dokumentima zaposlenima, na osnovu njihovih privilegija. Ovo omogućava da se definira koji zaposleni ima pristup kojim dokumentima, čime se obezbeđuje sigurnost i privatnost podataka.

Na kraju, aplikacija treba da podrži osnovne funkcionalnosti kao što su registracija, prijava i odjava korisnika, sa posebnim naglaskom na bezbednost podataka i pristup sistemu. Ovo uključuje zaštitu podataka korisnika i sigurno upravljanje pristupom, što je od vitalnog značaja za integritet i pouzdanost sistema.

# Implementacija

## Model

Model Firma predstavlja osnovni entitet u aplikaciji. Sadrži osnovne informacije o firmi kao što su naziv, PIB, matični broj, adresa, kontakt telefon i email. Ovaj model je centralna tačka oko koje se grupišu ostali entiteti poput zaposlenih i dokumenata. Na primer, deo koda koji definiše ovaj model može izgleda ovako:

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Firma extends Model

{

    use HasFactory;

    protected $fillable = [

        'naziv',

        'PIB',

        'maticniBroj',

        'adresa',

        'kontaktTelefon',

        'email',

    ];

    public function zaposleni()

    {

            return $this->hasMany(User::class);

    }

    public function fajlovi()

    {

            return $this->hasMany(Fajl::class);

    }

}

U kodu iznad možemo videti i veze koje postoje među modelima. Na primer jedna firma ima više zaposlenih, kao više fajlova.

Na sličan način su implementirani i ostali modeli

Zaposleni, koji je izveden iz User modela, sadrži detalje o zaposlenima unutar firme, uključujući poziciju, odeljenje, datum početka rada, datum kraja ugovora, platu i pripadnost firmi. Model User služi kao osnova za autentifikaciju i autorizaciju, dok Zaposleni dodaje specifične poslovne informacije. Kroz ovaj model, aplikacija može efikasno upravljati korisničkim nalogom i pravima pristupa.

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

class Zaposleni extends User

{

    use HasFactory;

    protected $fillable = [

        'pozicija',

        'odeljenje',

        'datum\_pocetka\_rada',

        'datum\_kraja\_ugovora',

        'plata',

        'firma\_id',

    ];

    public function firma()

    {

        return $this->belongsTo(Firma::class);

    }

}

Model Fajl je namenjen upravljanju dokumentima unutar firme. Svaki fajl ima naziv, opis, putanju na kojoj je sačuvan i povezan je sa firmom. Ovaj model omogućava firmama da organizuju i pristupaju svojim dokumentima na strukturiran i bezbedan način.

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Fajl extends Model

{

    use HasFactory;

    protected $fillable = [

        'naziv',

        'opis',

        'putanja',

    ];

    public function firma()

    {

        return $this->belongsTo(Firma::class);

    }

}

Model Privilegija igra vitalnu ulogu u definisanju prava pristupa i odgovornosti zaposlenih. Ovaj model određuje koji zaposleni ima pristup određenim dokumentima ili zadacima, čime se obezbeđuje bezbednost i integritet poslovnih informacija.

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Privilegija extends Model

{

    use HasFactory;

    protected $fillable = [

        'zaposleni\_id',

        'fajl\_id',

        'permission',

    ];

    public function zaposleni()

    {

            return $this->belongsTo(Zaposleni::class);

    }

    public function fajl()

    {

            return $this->belongsTo(Fajl::class);

    }

}

Task model je ključan za upravljanje zadacima unutar firme. Svaki zadatak je povezan sa zaposlenim i sadrži informacije kao što su naziv, opis, rok i status zadatka (npr. završeno, otkazano, u izradi). To omogućava efikasno praćenje i upravljanje radnim zadacima i projektima.

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Task extends Model

{

    use HasFactory;

    protected $fillable = [

        'zaposleni\_id',

        'naziv',

        'opis',

        'rok',

        'status',  //zavrseno, otkazano, u izradi

    ];

    public function zaposleni()

    {

        return $this->belongsTo(Zaposleni::class);

    }

}

## Migracije

U ovoj aplikaciji kreirane su sledeće migracije

1. 2023\_12\_23\_121010\_create\_firmas\_table.php

Ova migracija kreira tabelu firmas u bazi podataka. Tabela sadrži ključne informacije o firmama, uključujući polja kao što su naziv, PIB, matični broj, adresa, kontakt telefon i email. To je osnovna tabela koja će čuvati sve neophodne podatke o firmama unutar aplikacije.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateFirmasTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->id();

            $table->string('naziv');

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

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

            $table->string('address'); //adresa

            $table->string('phone'); //kontaktTelefon

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

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('firmas');

    }

}

1. 2023\_12\_23\_121020\_add\_columns\_users\_table.php

Ova migracija dodaje nove kolone u postojeću tabelu users. Ove dodatne kolone su verovatno specifične za aplikaciju i služe za čuvanje dodatnih informacija o korisnicima, kao što su pozicija, odeljenje ili druge poslovne informacije relevantne za korisnike aplikacije.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class AddColumnsUsersTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

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

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

            $table->date('datum\_pocetka\_rada')->nullable();

            $table->date('datum\_kraja\_ugovora')->nullable();

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

            $table->unsignedBigInteger('firma\_id')->nullable();

            $table->foreign('firma\_id')->references('id')->on('firmas')->onDelete('cascade');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

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

            $table->removeColumn('pozicija');

            $table->removeColumn('odeljenje');

            $table->removeColumn('datum\_pocetka\_rada');

            $table->removeColumn('datum\_kraja\_ugovora');

            $table->removeColumn('datum\_kraja\_ugovora');

            $table->removeColumn('plata');

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

            $table->removeColumn('firma\_id');

        });

    }

}

1. 2023\_12\_23\_121103\_create\_fajls\_table.php

Ovom migracijom se kreira tabela fajls, koja je namenjena čuvanju informacija o fajlovima ili dokumentima povezanim sa firmama. Tabela bi obično sadržala kolone poput naziva fajla, opisa, putanje na kojoj je fajl sačuvan i referencu na firmu s kojom je fajl povezan.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateFajlsTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->id();

            $table->string('naziv');

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

            $table->string('putanja');

            $table->unsignedBigInteger('firma\_id')->nullable();

            $table->foreign('firma\_id')->references('id')->on('firmas')->onDelete('cascade');

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('fajls');

    }

}

1. 2023\_12\_23\_121134\_create\_tasks\_table.php

Ova migracija stvara tabelu tasks za upravljanje zadacima unutar aplikacije. Tabela će tipično sadržavati informacije o zadacima kao što su naziv zadatka, opis, rok za izvršenje, status zadatka i povezanost sa zaposlenim koji je odgovoran za taj zadatak.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateTasksTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->id();

            $table->unsignedBigInteger('zaposleni\_id')->nullable();

            $table->foreign('zaposleni\_id')->references('id')->on('users')->onDelete('cascade');

            $table->string('naziv');

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

            $table->date('rok')->nullable();

            $table->string('status');

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('tasks');

    }

}

1. 2023\_12\_23\_121249\_create\_privilegijas\_table.php

Kroz ovu migraciju kreira se tabela privilegijas koja služi za definisanje prava pristupa ili privilegija unutar aplikacije. Ova tabela će sadržavati informacije koje definišu ko ima pristup određenim resursima, kao što su specifični dokumenti ili zadaci.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreatePrivilegijasTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->id();

            $table->string('permission');

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('privilegijas');

    }

}

1. 2023\_12\_23\_152350\_create\_zaposlenis\_table.php

Ova migracija stvara tabelu zaposlenis koja je namenjena za čuvanje informacija o zaposlenima u firmama. Tabela obično sadrži detalje poput pozicije zaposlenog, odeljenja, datuma početka i kraja rada, plate i informacije o firmi u kojoj su zaposleni.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateZaposlenisTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->id();

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

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

            $table->date('datum\_pocetka\_rada')->nullable();

            $table->date('datum\_kraja\_ugovora')->nullable();

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

            $table->foreignId('firma\_id')->nullable()->constrained('firmas');

            $table->timestamps();

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

        Schema::dropIfExists('zaposlenis');

    }

}

1. 2023\_12\_23\_152535\_remove\_zaposlenis\_table.php

Ovom migracijom se uklanja tabela zaposlenis, budući da nije potrebna u aplikaciji.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class RemoveZaposlenisTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

        Schema::dropIfExists('zaposlenis');

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

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

            $table->id();

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

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

            $table->date('datum\_pocetka\_rada')->nullable();

            $table->date('datum\_kraja\_ugovora')->nullable();

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

            $table->foreignId('firma\_id')->nullable()->constrained('firmas');

            $table->timestamps();

        });

    }

}

1. 2023\_12\_23\_152715\_rename\_columns\_zaposlenis\_table.php

Ova migracija se koristi za preimenovanje kolona u tabeli zaposlenis. Ovo je korisno kada se vrši reorganizacija ili optimizacija strukture baze podataka, i potrebno je osigurati da imena kolona bolje odražavaju njihovu svrhu ili sadržaj.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class RenameColumnsZaposlenisTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->renameColumn('address','adresa');

            $table->renameColumn('phone','kontaktTelefon');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

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

            $table->renameColumn('adresa','address');

            $table->renameColumn('kontaktTelefon','phone');

        });

    }

}

1. 2023\_12\_23\_224308\_add\_foreign\_key\_privilegijas\_table.php

Ovom migracijom dodaju se strani ključevi u tabelu privilegijas, što omogućava povezivanje ove tabele s drugim tabelama u bazi. Strani ključevi su ključni za održavanje integriteta baze podataka i omogućavaju bolje upravljanje relacijama između različitih entiteta u aplikaciji.

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class AddForeignKeyPrivilegijasTable extends Migration

{

    /\*\*

     \* Run the migrations.

     \*

     \* @return void

     \*/

    public function up()

    {

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

            $table->unsignedBigInteger('zaposleni\_id')->nullable();

            $table->foreign('zaposleni\_id')->references('id')->on('users')->onDelete('cascade');

            $table->unsignedBigInteger('fajl\_id')->nullable();

            $table->foreign('fajl\_id')->references('id')->on('fajls')->onDelete('cascade');

        });

    }

    /\*\*

     \* Reverse the migrations.

     \*

     \* @return void

     \*/

    public function down()

    {

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

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

            $table->removeColumn('zaposleni\_id');

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

            $table->removeColumn('fajl\_id');

        });

    }

}

1. 2014\_10\_12\_000000\_create\_users\_table.php

Ova osnovna Laravel migracija kreira tabelu users koja je standardna za sve aplikacije koje koriste Laravelov sistem autentifikacije. Tabela obično sadrži osnovne informacije o korisnicima, uključujući ime, email, lozinku i druge standardne kolone potrebne za upravljanje korisničkim nalozima.

## Kontroleri

U ovom delu ćemo opisati samo najznačajnije kontrolere.

1. FajlController.php

Ovaj kontroler upravlja akcijama vezanim za dokumente (fajlove) unutar aplikacije. To uključuje funkcionalnosti kao što su dodavanje novih dokumenata, ažuriranje, dohvatanje detalja o specifičnim dokumentima, te brisanje dokumenata.

FajlController koristi DropBox API za upravljanje fajlovima ,a dizajniran je da omogući korisnicima da izvršavaju različite operacije sa fajlovima kao što su pregledanje, preuzimanje, otpremanje, ažuriranje i brisanje fajlova u DropBox-u.

U konstruktoru kontrolera, zavisnost DropBoxService se injektira, omogućavajući pristup funkcionalnostima DropBox API-ja. Ovaj servis je zatim dostupan kroz instancu $this->dropBox, koja se koristi u svim metodama kontrolera.

class FajlController extends Controller

{

    protected $dropBox;

    public function \_\_construct(DropBoxService $dropBox)

    {

        $this->dropBox = $dropBox;

    }

Metoda index koristi listFiles funkciju iz DropBoxService da dohvati listu svih fajlova sa DropBox naloga. Rezultat, koji je lista fajlova, vraća se kao JSON odgovor.

    public function index()

    {

        $files = $this->dropBox->listFiles();

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

    }

update metoda omogućava ažuriranje postojećeg fajla. U ovoj metodi, novi fajl se dohvata iz zahteva, a zatim se koristi updateFile metoda iz DropBoxService-a da se ažurira odabrani fajl na DropBox-u. Rezultat operacije se vraća kao JSON odgovor.

    public function upload(Request $request)

    {

        $file = $request->file('file');

        if (!$file) {

            return response()->json(['error' => 'Nema fajla za otpremanje.'], 400);

        }

        $path = '/public/files/' . $file->getClientOriginalName();

        $contents = file\_get\_contents($file->getRealPath());

        $result = $this->dropBox->uploadFile($path, $contents);

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

    }

Metoda destroy omogućava brisanje fajla sa DropBox naloga. ID fajla se dohvata iz URL-a, a deleteFile metoda DropBoxService-a se koristi za brisanje fajla. Metoda vraća JSON odgovor koji potvrđuje brisanje fajla.

    public function destroy($fileId)

    {

        $this->dropBox->deleteFile($fileId);

        return response()->json(['message' => 'Fajl obrisan']);

    }

1. FirmaController.php

FirmaController se bavi upravljanjem firmama unutar aplikacije. Osnovne operacije koje ovaj kontroler podržava uključuju kreiranje novih firmi, izmenu postojećih informacija o firmama, pregled detalja o firmama, i brisanje firmi iz sistema.

<?php

namespace App\Http\Controllers;

use App\Models\Firma;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class FirmaController extends Controller

{

    /\*\*

     \* Display a listing of the resource.

     \*

     \* @return \Illuminate\Http\Response

     \*/

    public function index() //GET

    {

        $sveFirme = Firma::all();

        return $sveFirme;

    }

    /\*\*

     \* Show the form for creating a new resource.

     \*

     \* @return \Illuminate\Http\Response

     \*/

    public function create()

    {

        //

    }

    /\*\*

     \* Store a newly created resource in storage.

     \*

     \* @param  \Illuminate\Http\Request  $request

     \* @return \Illuminate\Http\Response

     \*/

    public function store(Request $request) //POST

    {

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

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

            'PIB' => 'required|string|max:12|unique:firmas,PIB',

            'maticniBroj' => 'required|string|max:12|unique:firmas,maticniBroj',

            'adresa' => 'required|string',

            'kontaktTelefon' => 'required|string',

            'email' => 'required|email|unique:firmas,email',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 422);

        }

        $firma = new Firma;

        $firma->naziv = $request->naziv;

        $firma->PIB = $request->PIB;

        $firma->maticniBroj = $request->maticniBroj;

        $firma->adresa = $request->adresa;

        $firma->kontaktTelefon = $request->kontaktTelefon;

        $firma->email = $request->email;

        $firma->save();

        return response()->json(['message' => 'Firma uspešno kreirana', 'firma' => $firma], 201);

    }

    /\*\*

     \* Display the specified resource.

     \*

     \* @param  \App\Models\Firma  $firma

     \* @return \Illuminate\Http\Response

     \*/

    public function show($id) //GET

    {

        $f=Firma::find($id);

        if($f){

            return $f;

        }

        return response()->json(['error' => 'Ne postoji ta firma'], 404);

    }

    /\*\*

     \* Show the form for editing the specified resource.

     \*

     \* @param  \App\Models\Firma  $firma

     \* @return \Illuminate\Http\Response

     \*/

    public function edit(Firma $firma)

    {

        //

    }

    /\*\*

     \* Update the specified resource in storage.

     \*

     \* @param  \Illuminate\Http\Request  $request

     \* @param  \App\Models\Firma  $firma

     \* @return \Illuminate\Http\Response

     \*/

    public function update(Request $request, $id)//PUT

    {

        $firma = Firma::find($id);

        if($firma){

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

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

                'PIB' => 'required|string|max:12',

                'maticniBroj' => 'required|string|max:12',

                'adresa' => 'required|string',

                'kontaktTelefon' => 'required|string',

                'email' => 'required|email',

            ]);

            if ($validator->fails()) {

                return response()->json($validator->errors(), 422);

            }

            $firma->naziv = $request->naziv;

            $firma->PIB = $request->PIB;

            $firma->maticniBroj = $request->maticniBroj;

            $firma->adresa = $request->adresa;

            $firma->kontaktTelefon = $request->kontaktTelefon;

            $firma->email = $request->email;

            $firma->save();

            return response()->json(['message' => 'Firma uspešno azurirana', 'firma' => $firma], 200);

        }else{

            return response()->json(['error' => 'Ne postoji ta firma'], 404);

        }

    }

    /\*\*

     \* Remove the specified resource from storage.

     \*

     \* @param  \App\Models\Firma  $firma

     \* @return \Illuminate\Http\Response

     \*/

    public function destroy($id) //DELETE

    {

        $f = Firma::with('zaposleni','fajlovi')->find($id);

        if($f){

            $f->delete();

            return response()->json(['message' => 'Firma obrisana'], 200);

        }else{

            return response()->json(['error' => 'Ne postoji ta firma'], 404);

        }

    }

}

1. TaskController.php

TaskController je zadužen za upravljanje zadacima u aplikaciji. Ovaj kontroler omogućava kreiranje zadataka, ažuriranje, dohvatanje informacija o zadacima, te brisanje zadataka.

<?php

namespace App\Http\Controllers;

use App\Http\Resources\TaskResource;

use App\Models\Task;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class TaskController extends Controller

{

    public function index() //GET

    {

        $sviTaskovi = Task::all();

        return TaskResource::collection($sviTaskovi);

    }

    public function store(Request $request)

    {

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

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

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

            'opis' => 'nullable|string',

            'rok' => 'required|date',

            'status' => 'required|in:zavrseno,otkazano,u izradi',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 422);

        }

        $task = Task::create($request->all());

        return response()->json(['message' => 'Task uspešno kreiran', 'task' => new TaskResource($task)], 201);

    }

    public function show($id) //GET

    {

        $task = Task::find($id);

        if ($task) {

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

        }

        return response()->json(['error' => 'Task nije pronađen'], 404);

    }

    public function update(Request $request, $id)

    {

        $task = Task::find($id);

        if ($task) {

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

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

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

                'opis' => 'nullable|string',

                'rok' => 'required|date',

                'status' => 'required|in:zavrseno,otkazano,u izradi',

            ]);

            if ($validator->fails()) {

                return response()->json($validator->errors(), 422);

            }

            $task->update($request->all());

            return response()->json(['message' => 'Task uspešno ažuriran', 'task' => new TaskResource($task)], 200);

        } else {

            return response()->json(['error' => 'Task nije pronađen'], 404);

        }

    }

    public function destroy($id)

    {

        $task = Task::find($id);

        if ($task) {

            $task->delete();

            return response()->json(['message' => 'Task obrisan'], 200);

        } else {

            return response()->json(['error' => 'Task nije pronađen'], 404);

        }

    }

}

1. ZaposleniController.php

ZaposleniController upravlja svim aspektima vezanim za zaposlene unutar aplikacije. To uključuje registraciju novih zaposlenih, ažuriranje njihovih podataka, pregled informacija o zaposlenima, i uklanjanje zaposlenih iz sistema.

<?php

namespace App\Http\Controllers;

use App\Http\Resources\ZaposleniResource;

use App\Models\User;

use App\Models\Zaposleni;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Validator;

class ZaposleniController extends Controller

{

    public function index()

    {

        $sviZaposleni = User::paginate(5);

        return  ZaposleniResource::collection($sviZaposleni);

    }

    public function store(Request $request)

    {

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

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

            'email' => 'required|email|unique:users,email',

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

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

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

            'datum\_pocetka\_rada' => 'required|date',

            'datum\_kraja\_ugovora' => 'nullable|date',

            'plata' => 'required|numeric',

            'firma\_id' => 'required|exists:firmas,id',

        ]);

        if ($validator->fails()) {

            return response()->json($validator->errors(), 422);

        }

        $zaposleni = new User;

        $zaposleni->name = $request->name;

        $zaposleni->email = $request->email;

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

        $zaposleni->pozicija = $request->pozicija;

        $zaposleni->odeljenje = $request->odeljenje;

        $zaposleni->datum\_pocetka\_rada = $request->datum\_pocetka\_rada;

        $zaposleni->datum\_kraja\_ugovora = $request->datum\_kraja\_ugovora;

        $zaposleni->plata = $request->plata;

        $zaposleni->firma\_id = $request->firma\_id;

        $zaposleni->save();

        return response()->json(['message' => 'Zaposleni uspešno kreiran', 'zaposleni' => $zaposleni], 201);

    }

    public function show($id)

    {

        $zaposleni = User::find($id);

        if ($zaposleni) {

            return new ZaposleniResource($zaposleni);

        }

        return response()->json(['error' => 'Zaposleni nije pronađen'], 404);

    }

    public function update(Request $request, $id) //PUT

    {

        $zaposleni = User::find($id);

        if ($zaposleni) {

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

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

                'email' => 'required|email',

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

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

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

                'datum\_pocetka\_rada' => 'required|date',

                'datum\_kraja\_ugovora' => 'nullable|date',

                'plata' => 'required|numeric',

                'firma\_id' => 'required|exists:firmas,id',

            ]);

            if ($validator->fails()) {

                return response()->json($validator->errors(), 422);

            }

            $zaposleni->update($request->all());

            return response()->json(['message' => 'Zaposleni uspešno ažuriran', 'zaposleni' => new ZaposleniResource($zaposleni)], 200);

        } else {

            return response()->json(['error' => 'Zaposleni nije pronađen'], 404);

        }

    }

    public function destroy($id)

    {

        $zaposleni = User::find($id);

        if ($zaposleni) {

            $zaposleni->delete();

            return response()->json(['message' => 'Zaposleni obrisan'], 200);

        } else {

            return response()->json(['error' => 'Zaposleni nije pronađen'], 404);

        }

    }

}

1. AuthController.php

AuthController je odgovoran za autentifikaciju korisnika u aplikaciji. Ovaj kontroler upravlja procesima kao što su registracija korisnika, prijava i odjava, te promena lozinki. AuthController je ključan za održavanje sigurnosti i integriteta korisničkih naloga unutar aplikacije, obezbeđujući da samo ovlašćeni korisnici imaju pristup određenim funkcionalnostima i podacima.

<?php

namespace App\Http\Controllers;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

use Illuminate\Support\Facades\Hash;

class AuthController extends Controller

{

    public function register(Request $request)

    {

        $validatedData = $request->validate([

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

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

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

        ]);

        $user = User::create([

            'name' => $validatedData['name'],

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

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

        ]);

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

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

    }

    public function login(Request $request)

    {

        $request->validate([

            'email' => 'required|email',

            'password' => 'required',

        ]);

        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', 'user' => $user]);

    }

    public function logout(Request $request)

    {

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

        return response()->json(['message' => 'Successfully logged out']);

    }

}

## Api rute

Kratak pregled api ruta

1. Nezaštićene rute - rute su dostupne svima, bez potrebe za autentifikacijom.

GET /firme: Vraća listu svih firmi.

GET /firme/{id}: Vraća detalje o specifičnoj firmi na osnovu ID-ja.

POST /register: Omogućava registraciju novog korisnika.

POST /login: Omogućava prijavu korisnika.

1. Zaštićene rute - rute zahtevaju autentifikaciju, što znači da korisnik mora biti prijavljen i imati validan token da bi ih koristio.

DELETE /firme/{id}: Omogućava brisanje firme na osnovu ID-ja.

POST /firme: Omogućava kreiranje nove firme.

PUT /firme/{id}: Omogućava ažuriranje informacija o firmi na osnovu ID-ja.

POST /logout: Omogućava odjavu korisnika.

GET /fajlovi: Vraća listu svih fajlova.

POST /fajlovi/upload: Omogućava otpremanje novog fajla.

DELETE /fajlovi/delete/{fileId}: Omogućava brisanje fajla na osnovu ID-ja.

Rute za zaposleni koje nisu index i show: Ove rute omogućavaju kreiranje, ažuriranje i brisanje informacija o zaposlenima.

# REST API

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

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih firmi |
| HTTP metoda | GET |
| URL | api/firme |
| URL parametri |  |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | [  {  "id": 1,  "naziv": "Jacobi-Goodwin",  "PIB": "308249669",  "maticniBroj": "399498864",  "adresa": "72998 Lonie Island\nJudahview, RI 28740-8878",  "kontaktTelefon": "623-456-7536",  "email": "sadie75@bode.net",  "created\_at": "2024-01-05T20:01:09.000000Z",  "updated\_at": "2024-01-05T20:01:09.000000Z"  },  {  "id": 2,  "naziv": "Kreiger-Johns",  "PIB": "877651191",  "maticniBroj": "458700029",  "adresa": "87566 Myah Island Apt. 579\nSouth Francesfurt, OH 17132",  "kontaktTelefon": "1-682-349-3990",  "email": "audrey.bogan@gerlach.org",  "created\_at": "2024-01-05T20:01:09.000000Z",  "updated\_at": "2024-01-05T20:01:09.000000Z"  },  ..  ] |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje firme |
| HTTP metoda | POST |
| URL | /api/firme |
| URL parametri | ?naziv=firma1&PIB=1234232&maticniBroj=45432&adresa=adresa 1&kontaktTelefon=1111111&email=firma1@gmail.com |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Firma uspešno kreirana",      "firma": {          "naziv": "firma1",          "PIB": "1234232",          "maticniBroj": "45432",          "adresa": "adresa 1",          "kontaktTelefon": "1111111",          "email": "firma1@gmail.com",          "updated\_at": "2024-01-05T21:06:58.000000Z",          "created\_at": "2024-01-05T21:06:58.000000Z",          "id": 6      }  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje firme |
| HTTP metoda | PUT |
| URL | /api/ firme |
| URL parametri | 6?naziv=firma11111&PIB=1234232&maticniBroj=45432&adresa=adresa 1&kontaktTelefon=1111111&email=firma1@gmail.com |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Firma uspešno azurirana",      "firma": {          "id": 6,          "naziv": "firma11111",          "PIB": "1234232",          "maticniBroj": "45432",          "adresa": "adresa 1",          "kontaktTelefon": "1111111",          "email": "firma1@gmail.com",          "created\_at": "2024-01-05T21:06:58.000000Z",          "updated\_at": "2024-01-05T21:09:19.000000Z"      }  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Brisanje firme |
| HTTP metoda | DELETE |
| URL | /api/ firme |
| URL parametri | 7 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Firma obrisana"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prikaz svih taskova |
| HTTP metoda | GET |
| URL | /api/task |
| URL parametri | nema |
| HTTP body parametri | nema |
| Format HTTP body parametara | JSON |
| Izlazni parametri | "data": [          {              "id": 1,              "naziv": "Reprehenderit magnam assumenda blanditiis rem est est occaecati amet.",              "opis": "Sapiente totam sunt et labore voluptatem laboriosam dolorum. Totam omnis et aliquid. Voluptatibus ea expedita esse animi animi ut.",              "rok": "1991-03-06",              "status": "u izradi",              "zaposleni": {                  "id": 4,                  "name": "Jameson Pfeffer",                  "email": "paula.leannon@example.com",                  "pozicija": "Benefits Specialist",                  "odeljenje": "repellat",                  "datum\_pocetka\_rada": "1988-11-22",                  "datum\_kraja\_ugovora": "1970-05-17",                  "plata": "8915.90",                  "firma": {                      "id": 1,                      "naziv": "Jacobi-Goodwin",                      "PIB": "308249669",                      "maticniBroj": "399498864",                      "adresa": "72998 Lonie Island\nJudahview, RI 28740-8878",                      "kontaktTelefon": "623-456-7536",                      "email": "sadie75@bode.net",                      "created\_at": "2024-01-05T20:01:09.000000Z",                      "updated\_at": "2024-01-05T20:01:09.000000Z"                  }              }          },          {              "id": 2,              "naziv": "Unde optio et iste itaque consequatur.",              "opis": "Ut id qui laudantium. Alias rerum necessitatibus qui aperiam. Fugiat omnis atque est placeat. Sint id eos non incidunt.",              "rok": "2005-11-08",              "status": "otkazano",              "zaposleni": {                  "id": 7,                  "name": "Brando Lebsack",                  "email": "owitting@example.net",                  "pozicija": "Business Development Manager",                  "odeljenje": "dolorem",                  "datum\_pocetka\_rada": "2009-07-21",                  "datum\_kraja\_ugovora": "1990-06-14",                  "plata": "5064.61",                  "firma": {                      "id": 3,                      "naziv": "Kunde, Gerlach and O'Keefe",                      "PIB": "768579133",                      "maticniBroj": "686542762",                      "adresa": "592 Rippin Land Suite 917\nNew Barney, NM 87843-3733",                      "kontaktTelefon": "339.670.4321",                      "email": "isom75@kilback.com",                      "created\_at": "2024-01-05T20:01:09.000000Z",                      "updated\_at": "2024-01-05T20:01:09.000000Z"                  }              }          }, |
| Format izlaznih parametara | Application/json |
|  |  |
| Opis funkcije | Kreiranje taska |
| HTTP metoda | POST |
| URL | /api/task |
| URL parametri | ?zaposleni\_id=1&naziv=task1&rok=2024-01-10&status=zavrseno |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Task uspešno kreiran",      "task": {          "id": 11,          "naziv": "task1",          "opis": **null**,          "rok": "2024-01-10",          "status": "zavrseno",          "zaposleni": {              "id": 1,              "name": "Mr. Hugh Stroman PhD",              "email": "yvolkman@example.com",              "pozicija": "Photographic Restorer",              "odeljenje": "illo",              "datum\_pocetka\_rada": "1995-08-24",              "datum\_kraja\_ugovora": "2015-07-10",              "plata": "2566.68",              "firma": {                  "id": 1,                  "naziv": "Jacobi-Goodwin",                  "PIB": "308249669",                  "maticniBroj": "399498864",                  "adresa": "72998 Lonie Island\nJudahview, RI 28740-8878",                  "kontaktTelefon": "623-456-7536",                  "email": "sadie75@bode.net",                  "created\_at": "2024-01-05T20:01:09.000000Z",                  "updated\_at": "2024-01-05T20:01:09.000000Z"              }          }      }  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Ažuriranje taska |
| HTTP metoda | PUT |
| URL | /api/task |
| URL parametri | 11?zaposleni\_id=1&naziv=task11111&rok=2024-01-10&status=zavrseno |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Task uspešno ažuriran",      "task": {          "id": 11,          "naziv": "task11111",          "opis": **null**,          "rok": "2024-01-10",          "status": "zavrseno",          "zaposleni": {              "id": 1,              "name": "Mr. Hugh Stroman PhD",              "email": "yvolkman@example.com",              "pozicija": "Photographic Restorer",              "odeljenje": "illo",              "datum\_pocetka\_rada": "1995-08-24",              "datum\_kraja\_ugovora": "2015-07-10",              "plata": "2566.68",              "firma": {                  "id": 1,                  "naziv": "Jacobi-Goodwin",                  "PIB": "308249669",                  "maticniBroj": "399498864",                  "adresa": "72998 Lonie Island\nJudahview, RI 28740-8878",                  "kontaktTelefon": "623-456-7536",                  "email": "sadie75@bode.net",                  "created\_at": "2024-01-05T20:01:09.000000Z",                  "updated\_at": "2024-01-05T20:01:09.000000Z"              }          }      }  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Brisanje taska |
| HTTP metoda | DELETE |
| URL | /api/task |
| URL parametri | 11 |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "message": "Task obrisan"  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Registracija korisnika |
| HTTP metoda | POST |
| URL | /api/register |
| URL parametri | name=pera&email=peric@gmail.com&password=peraperapera |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "access\_token": "1|pV79bHyuYLO5UpcORAEouBBqGvCYZ0z3FBnuOoMn",      "token\_type": "Bearer",      "user": {          "name": "pera",          "email": "peric@gmail.com",          "updated\_at": "2024-01-05T21:18:36.000000Z",          "created\_at": "2024-01-05T21:18:36.000000Z",          "id": 11      }  } |
| Format izlaznih parametara | application/json |

|  |  |
| --- | --- |
| Opis funkcije | Prijava korisnika |
| HTTP metoda | POST |
| URL | /api/login |
| URL parametri | email=peric@gmail.com&password=peraperapera |
| HTTP body parametri | (nema) |
| Format HTTP body parametara | (nema) |
| Izlazni parametri | {      "access\_token": "2|2ZHQEDdeds0A0dMjAUgvYbbHnCgPKkHLs4gWnuMi",      "token\_type": "Bearer",      "user": {          "id": 11,          "name": "pera",          "email": "peric@gmail.com",          "email\_verified\_at": **null**,          "created\_at": "2024-01-05T21:18:36.000000Z",          "updated\_at": "2024-01-05T21:18:36.000000Z",          "pozicija": **null**,          "odeljenje": **null**,          "datum\_pocetka\_rada": **null**,          "datum\_kraja\_ugovora": **null**,          "plata": **null**,          "firma\_id": **null**      }  } |
| 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": "Successfully logged out"  } |
| Format izlaznih parametara | application/json |

# Korisničko uputstvo

Registracija korisnika

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Ikona na računaru

Opis je automatski generisan

Prijava korisnika

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

Opis je automatski generisan

Prikaz svih taskova

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

Opis je automatski generisan

Kreiranje taska

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Ikona na računaru

Opis je automatski generisan

Ažuriranje taska

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

Opis je automatski generisan

Brisanje taska

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Ikona na računaru

Opis je automatski generisan

Prikaz svih firmi

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

Opis je automatski generisan

Dodavanje firme

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

Opis je automatski generisan

Ažuriranje firme

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

Opis je automatski generisan

Brisanje firme

Slika na kojoj se nalazi tekst, snimak ekrana, softver, Ikona na računaru

Opis je automatski generisan

Odjava korisnika

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

Opis je automatski generisan