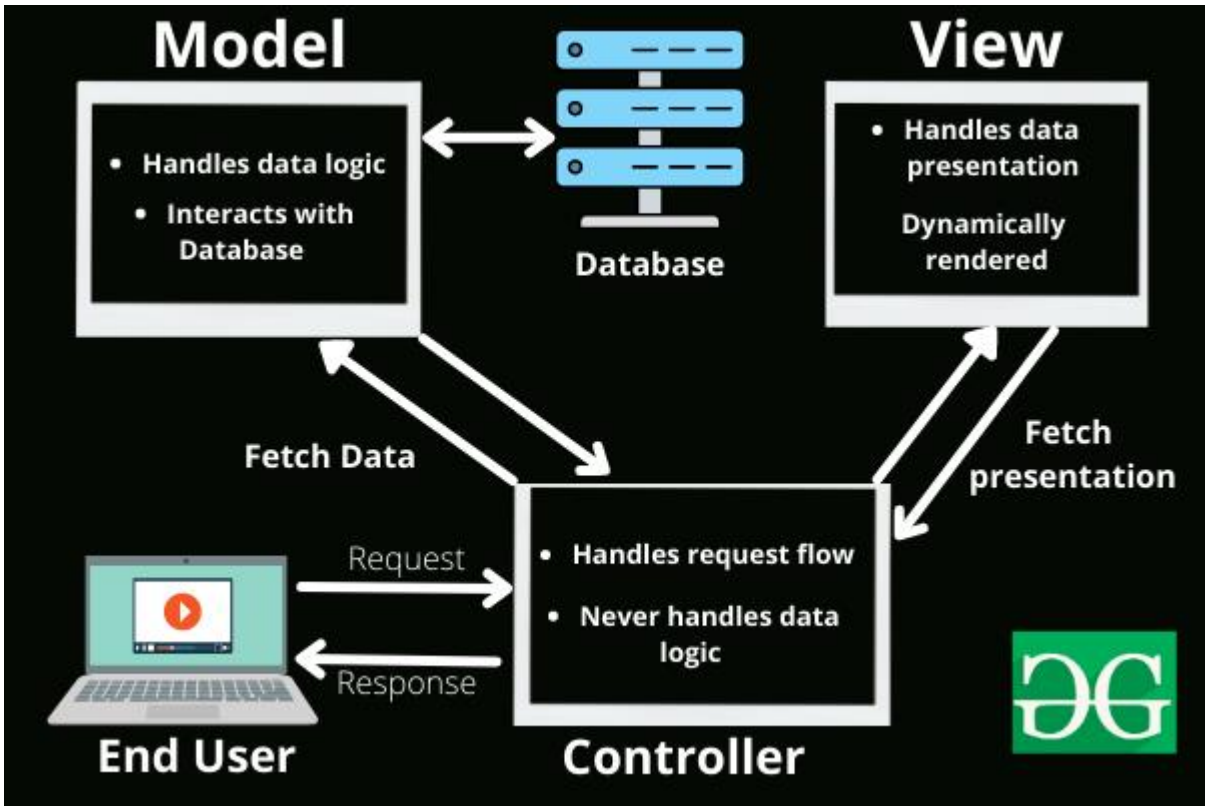
Vježbe 01 (podešavanja)

Arhitektura (MVC):

- Jedan od najpolularnijih arhitektura
- Model-View-Controller – logički razdvaja palikaciju na Modele, Poglede i Kontrolere



Slika 1. https://media.geeksforgeeks.org/wp-content/uploads/20220224160807/Model1.png

Instalacija

Osnovne instalacije:

- composer> https://getcomposer.org/download/
- xampp (php 8.2.0)> https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/8.2.0/xampp-windows-x64-8.2.0-0-VS16-installer.exe/download
- visual studio code> https://code.visualstudio.com/download

Napomene:

- dodati na environment variblu Path vrijednosti:
  - C:\xampp\php
  - C:\ProgramData\ComposerSetup\bin

Instalacija Laravela:

U cmd konzoli izvršiti naredbu: `composer global require laravel/installer`

Kreiranje praznog projekta

U cmd konzoli izvršiti naredbu: `laravel new vjezbe01`

Napomena: Projekat će biti kreiran na lokaciji gdje se izvrši naredba.

Pokretanje projekta

U cmd konzoli izvršiti naredbu: `php artisan serve`
U pretraživaču tražimo adresu: http://localhost:8000

Napomena: Potrebno se nalaziti na lokaciji …/vjezbe01 kako bi izvršili prethodnu naredbu.

Gašenje projekta

Pritisnite Ctrl+C.

Napomena: Potrebno se nalaziti u istoj komandnoj liniji kao i kada ste pokrenuli projekat.

Brisanje projekta

Obrisati direktorijum vjezbe01

Literatura:

https://laravel.com/docs/10.x

https://www.geeksforgeeks.org/mvc-framework-introduction/

https://laravel.com/docs/10.x/routing

Vježbe 02 (rutiranje)

Rutiranje

- Return string
- Return php
- Return blade.php

Blade

- Print parametra
- Primjer sa Script

- Foreach(list as elem) – endforeach
- If - else– endif
- Laravel blade paket za Code
- Yield – extends – section

Parameters

- /oglas/{id}
- Route::input('id')
- Key_exists(id, lista)
- Abort(404)
- Function($id){…}

Literatura:

https://www.youtube.com/watch?v=nVzKhNCeT-4&t=449s

https://www.youtube.com/watch?v=kFfFsww6N2Y&t=576s

https://www.youtube.com/watch?v=kT3Wb6C-hs4

https://laravel.com/docs/10.x

https://www.geeksforgeeks.org/mvc-framework-introduction/

https://laravel.com/docs/10.x/routing

Vježbe 03

Ekstenzije za Vizual Studio Code:

- Laravel blade
- Php namespace

Debagovanje: Metode dd() i ddd()

Primjer:

```
Route::get('/test', function(Request $request){
    //dd($request);
    return ($request->ime);
});
```

Eloquent

Laravel uključuje Eloquent, objektno-relacioni maper (ORM) koji čini interakciju sa vašom bazom podataka prijatnom. Kada koristite Elokuent, svaka tabela baze podataka ima odgovarajući „Model“ koji se koristi za interakciju sa tom tabelom. Pored preuzimanja zapisa iz tabele baze podataka, Elokuent modeli vam omogućavaju da ubacite, ažurirate i izbrišete zapise iz tabele.

Model možemo kreirati pomoću naredbe:

```
php artisan make:model Flight
```

Uz model možemo da kreiramo mograciju:

```
php artisan make:model Flight --migration
```
ili

```
php artisan make:model Flight -m
```

Možemo da kreiramo i controler, migraciju i factory

```
php artisan make:model Flight -mfc
```

Da bi migracija bila vidljiva na bazi potrebno je izvršiti naredbu:

```
php artisan migrate
```
Ili sledeću u slučaju da bazu izgradite ponovo iz nule.

```
php artisan migrate:refresh
```

Facker

https://github.com/fzaninotto/Faker

Tipovi podataka u tabeli:

https://laravel.com/docs/4.2/schema

Neke važne komande:

php artisan route:list

php artisan make:controller CourseController –resource

php artisan db:seed

php artisan migrate:refresh

php artisan make:model Course –mfc

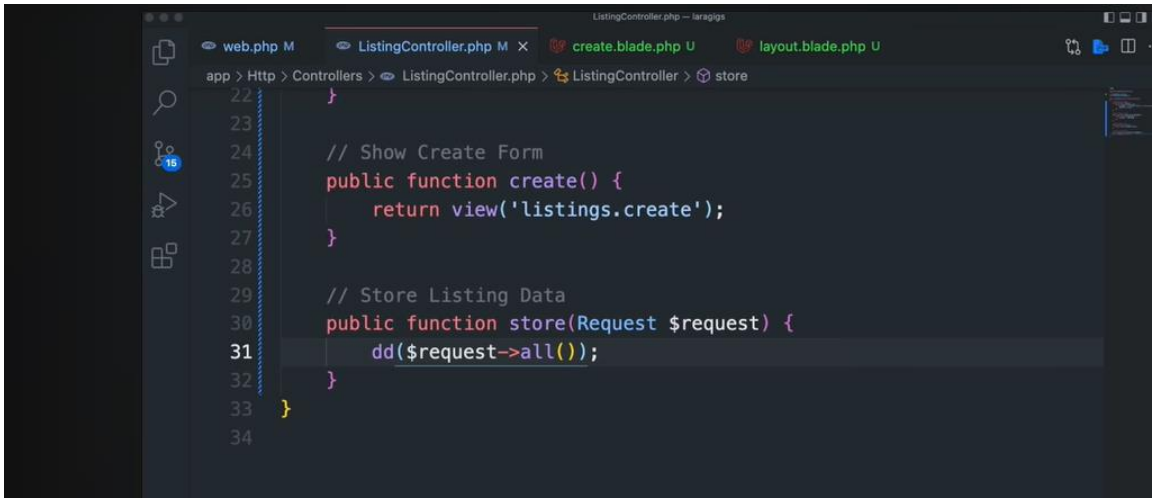
Vježbe 04

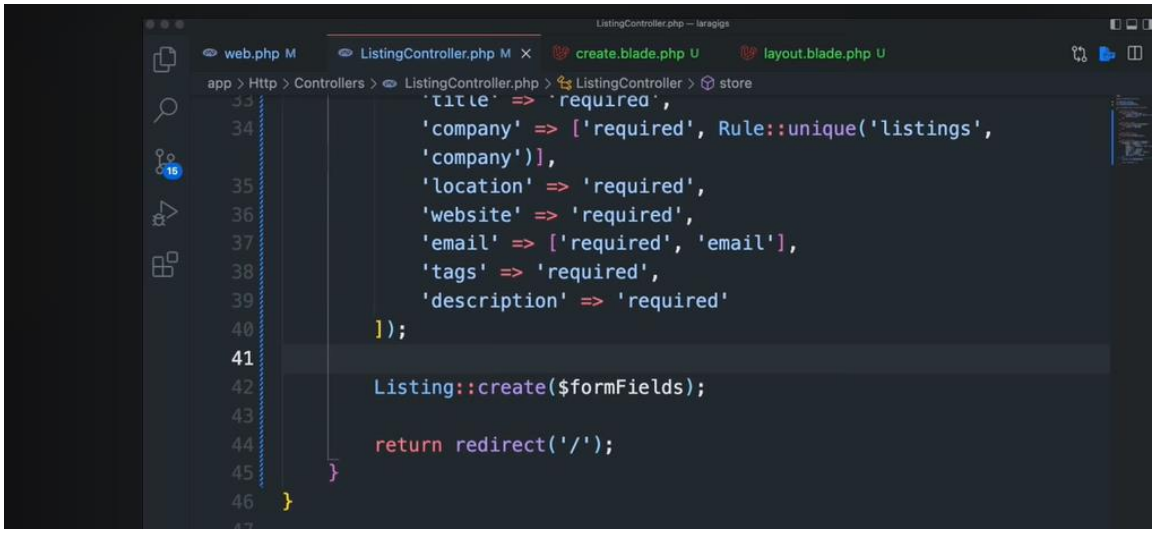
- Poziv kontrolera pomoću get metode:
  - `Route::get('/predmeti', [CourseController::class, 'index']);`
- Dodavanje teme:
  - Tema> https://www.free-css.com/free-css-templates/page277/kidkinder
  - https://stackoverflow.com/questions/13433683/using-css-in-laravel-views
  - asset()
- Blade components:
  - https://tighten.com/insights/extensible-blade-components/
- Create form
  - https://www.youtube.com/watch?v=MYyJ4PuL4pY&t=5425s

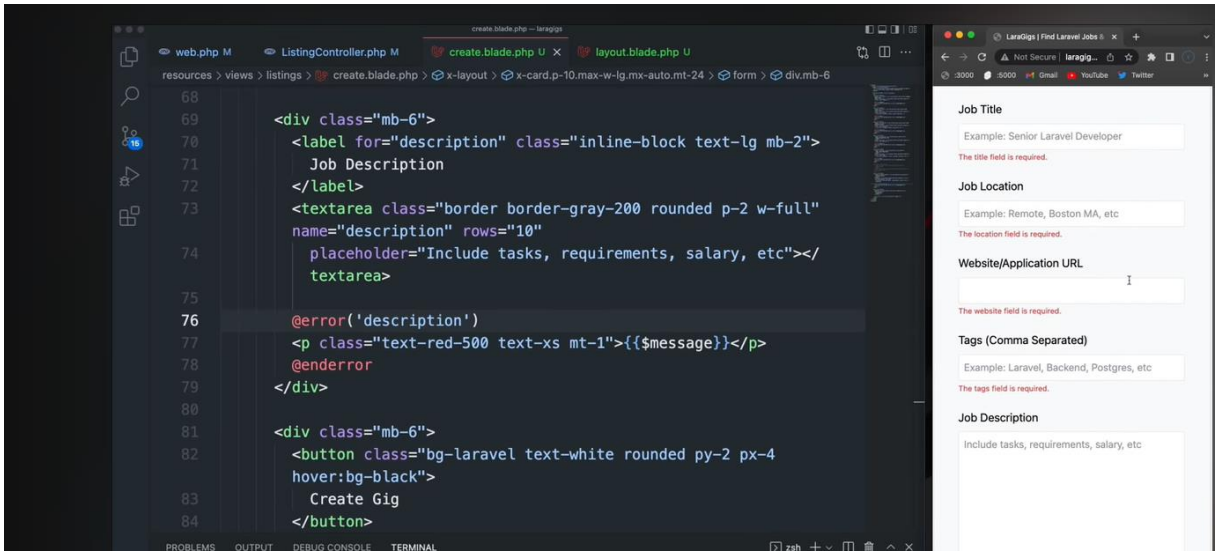
--- za predavača----

Pomocni kod:

```
@include('inc.meni')

<link href="{{asset('img/favicon.ico')}}" rel="icon">

   - Asset img rand ////////////////////////////////////////////

<img class="card-img-top mb-2" src="{{asset('img/class-'.rand(1,3).'.jpg')}}"
alt="">

Componenst ////////////////////////////////////////////////

<x-kurs-kartica :predmet="$predmet" />
@props(['predmet'])
```

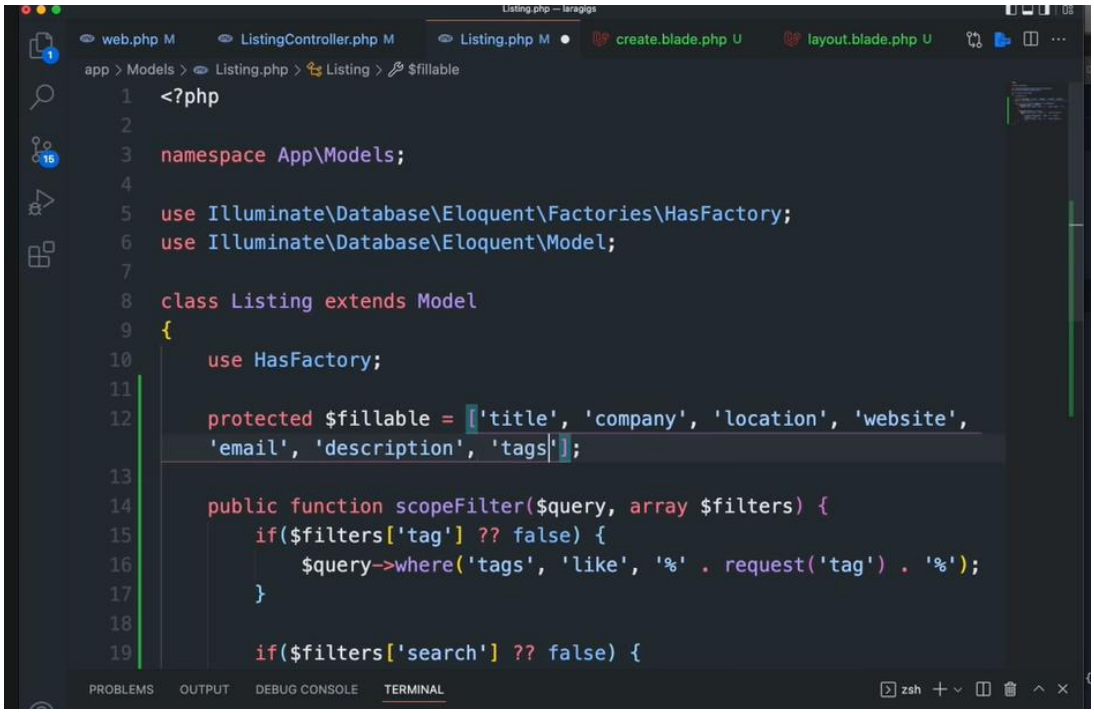



ListingController.php — laragigs
web.php M
ListingController.php M
create.blade.php U
layout.blade.php U
app > Http > Controllers > ListingController.php > ListingController > store
'title' => 'required',
'company' => ['required', Rule::unique('listings', 'company')],
'location' => 'required',
'website' => 'required',
'email' => ['required', 'email'],
'tags' => 'required',
'description' => 'required'
]);
Listing::create($formFields);
return redirect('/');
}
}


create.blade.php — laragigs
web.php M
ListingController.php M
create.blade.php U
layout.blade.php U
resources > views > listings > create.blade.php > x-layout > x-card.p-10.max-w-lg.mx-auto.mt-24 > form > div.mb-6

<label for="description" class="inline-block text-lg mb-2">
Job Description
</label>
<textarea class="border border-gray-200 rounded p-2 w-full"
name="description" rows="10"
placeholder="Include tasks, requirements, salary, etc"></
textarea>
@error('description')
{{$message}}
@enderror

<button class="bg-laravel text-white rounded py-2 px-4
hover:bg-black">
Create Gig
</button>
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
LaraGigs | Find Laravel Jobs
Job Title
Example: Senior Laravel Developer
The title field is required.
Job Location
Example: Remote, Boston MA, etc
The location field is required.
Website/Application URL
The website field is required.
Tags (Comma Separated)
Example: Laravel, Backend, Postgres, etc
The tags field is required.
Job Description
Include tasks, requirements, salary, etc


Listing.php — laragigs
web.php M
ListingController.php M
Listing.php M
create.blade.php U
layout.blade.php U
app > Models > Listing.php > Listing > $fillable
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
class Listing extends Model
{
use HasFactory;
protected $fillable = ['title', 'company', 'location', 'website', 'email', 'description', 'tags'];
public function scopeFilter($query, array $filters) {
if($filters['tag'] ?? false) {
$query->where('tags', 'like', '%' . request('tag') . '%');
}
if($filters['search'] ?? false) {
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
zsh

Vježbe 05

Create Form

```
Course::create($request->all());

protected $fillable = ['naziv', 'ects', 'status', 'opis'];
protected $guarded = [];
```

Validation &Store

- Lista pravila> https://laravel.com/docs/4.2/validation#rule-integer
- https://laravel.com/docs/10.x/validation

----------controler

```
$dataFrame = $request->validate([
      'naziv'=>'required',
      'opis'=>'required|min:10',
      'status'=>'required|in:o,i',
      'ects'=>'required|numeric'
  ]);
```

----------Blade

```
@error('ects')
      <br><p>{{$message}}</p><br>
  @enderror
```

Flash message

----------controler

```
return redirect('/predmeti')->with('message', "Uspješno ste dodali kurs.");
```

----------Blade

```
@if (session()->has('message'))

<p>{{session('message')}}</p>

@endif
```

Keep old data in form

----------Blade

```
 value="{{old('naziv')}}"
```

Search filter

Update

…

Literatura:

https://www.youtube.com/watch?v=MYyJ4PuL4pY&t=8671s

Vježbe 06

Update

----------Kontroler:

```
 return back()->with('message', "Uspjesno je izmjenjen predmet");
```

Brisanje

----------Kontroler:

```
public function destroy(string $id)
{
    $course = Course::where('id', $id)->get()[0]; //first() <==> get()[0]
    $course->delete();
    return redirect('/predmeti')->with("message", "Predmet je obrisan");
}
```

----------Pogled:

```
<form action="/predmeti/{{$prom->id}}" method="post">
    @csrf
    @method('DELETE')
    <button class="btn btn-destroy px-4 mb-4">Obrisite</button>
</form>
```

Registracija

- php artisan make: controller UserController –resource
- Podesimo rutu (Route) na create() funkciju u kontroleru
- Napišemo funkciju create()
- Podesimo link na stranici (meni.blade.php)
- Kreiramo formu u na lokaciji.fajlu 'user.register'
    - Polja: name, email, password (name=”password”), password2 (name=”password_confirmation”)
- Podesimo rutu (Route) na store() funkciju u kontroleru
- Napišemo funkciju store()
    - `'password'=>'required|confirmed'`
    - `$dataFrame['password'] = bcrypt($dataFrame['password']);`
    - $user = User::create($dataFrame);
    - Login: auth()->login($user);
    - return redirect('/')->with(…

Napomena: paziti na uključivanje paketa

Auth link

- @auth
  - welcome… {{auth()->user()->name}}
  - logout form
    - method=post, action='/logout'
    - buton submit
- @else
  - Registracija
  - Prijava
- @endauth

Odjava

- Podesimo „post" rutu (Route) na logout() funkciju u kontroleru
- Kreiratu funkciju logout(Request $request) u kontroleru
  - Auth()->logout();
  - $request->session()->invalidate();
  - $request->session()->regenerateToken();
  - Return… with message

Prijava (logovanje)

- Podesimo „get" rutu (Route) na login() funkciju u kontroleru
- Napišemo funkciju login() u kontroleru (nema novih stvari - jednostavno)
- Kreiramo blade stranicu
  - Forma
- Podesimo „post" rutu „/users/autenticate" (Route) na autenticate() funkciju u kontroleru
- Napišemo funkciju autenticate(Request $request) u kontroleru User
  - $dataFrame = …
  - If(auth()->attempt($dataFrame)){
    - $request ->session() -> regenerate();
    - Return redirect(…)->with(…)
  - Return back() -> withErrors([
    - 'email'=>'Email adresa nije dobra!',
  -])->onlyInput('email');
  - Za one koji žele više: https://stackoverflow.com/questions/31081644/how-to-redirect-back-to-form-with-input-laravel-5

GOST i Autentikacija (middlewere)

- App/http/middlewere
- Route::get(…)->Middlewere('auth');
- Route::get('/login',…)->name(' login ');

Testirati

```
Route::resource('/resource', 'Controller', [
        'except' => [
            'index',
            'show'
        ]
    ])
    ->middleware(['auth']);
```

- Testirati nakon logovanja adresu '/register'
    - Route::get('/register', …)->middlewere('guest');
- Provjeriti HOME konstantu na lokaciji 'app\Providers\RouteServiceProvider.php '

Literatura:

https://www.youtube.com/watch?v=MYyJ4PuL4pY&t=8671s

Vježbe 07

Veze

- Kreiramo novi laravel projekat:

laravel new veze

- Kreiramo modele:

php artisan make:model Phone –m

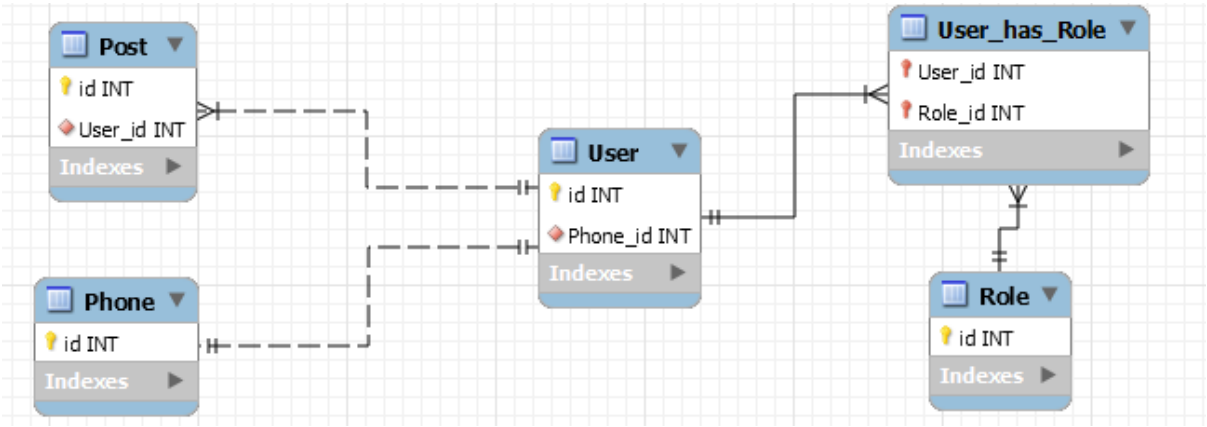
php artisan make:model Post –m

php artisan make:model Role -m

- Na config/database.php postaviti sljedeće:

```
'database' => env('DB_DATABASE', 'veze'),
'username' => env('DB_USERNAME', 'root'),
```

- Definisati tabele unutar migracija na lokaciji database/migrations kao na githubu. Tabele definišemo na osnovu dijagrama tabele.



- Izvrsimo migraciju

Veza One-to-one:

- Metoda **belongsTo()**

Dodamo sljedeću metodu u model User

```
public function phone(): BelongsTo
{
    return $this->belongsTo(Phone::class);
}
```

Kreiramo funkciju u user kontroleru (prvo je potrebno kreirati kontroler):

```
public function user_phone1(string $id)
{
    $phone = User::find($id)->phone;
    return $phone;
}
```

Dodamo adredu u web.php:

```
Route::get('/user/{id}/phone1', [UserController::class, 'user_phone1']);
```

Upit koji se izvršava u pozadini je:

*select * from `phones` where `phones`.`id` = X limit 1*, gdje je X odgovarajuća vrijednost, tj. phone_id u tabeli user.

Na adresi http://127.0.0.1:8000/user/1/phone1 prikazaće se telefon korisnika sa ID-em 1.

Veza One-to-one:

- Metoda **hasOne()**

Dodamo sljedeću metodu u model Phone

```
public function user(): HasOne
{
    return $this->hasOne(User::class);
}
```

Kreiramo funkciju u phone kontroleru (prvo je potrebno kreirati kontroler):

```
public function index(Request $r, String $id)
{
    $user = Phone::find($id)->user;
    return $user;
}
```

Dodamo adredu u web.php:

```
Route::get('/phone/{id}', [PhoneController::class, 'index']);
```

Upit koji se izvršava u pozadini je:

*select * from `users` where `users`.`phone_id` = X and `users`.`phone_id` is not null limit 1,* gdje je X odgovarajuća vrijednost, tj. id prosljećen linkom.

Na adresi http://127.0.0.1:8000/phone/1 prikazaće se korisnik čiji telefon ima ID 1.

Veza One-to-Many:

- Metoda **hasMany()**

Dodamo sljedeću metodu u model User

```
public function posts(): hasMany
{
    return $this->hasMany(Post::class);
}
```

Kreiramo funkciju u user kontroleru:

```
public function user_posts(string $id)
{
    $posts = User::find($id)->posts;
    return $posts;
}
```

Dodamo adredu u web.php:

```
Route::get('/user/{id}/posts', [UserController::class, 'user_posts']);
```

Upit koji se izvršava u pozadini je:

*select * from `posts` where `posts`.`user_id` = X and `posts`.`user_id` is not null,* gdje je X odgovarajuća vrijednost, tj. id prosljećen linkom.

Na adresi http://127.0.0.1:8000/user/1/posts prikazaće se postovi korisnika sa ID-em 1.

Veza Many-to-Many

- Metoda **belongesToMany()**

Dodamo sljedeću metodu u model Role

```
public function users()
{
    return $this->belongsToMany(User::class, 'role_user');
}
```

Kreiramo funkciju u role kontroleru:

```
public function index(Request $request, String $id)
{
    $users = Role::find($id)->users;
    return $users;
}
```

Dodamo adredu u web.php:

```
Route::get('/role/{id}', [RoleController::class, 'index']);
```

Upit koji se izvršava u pozadini je:

```
select
  `users`.*,
  `role_user`.`role_id` as `pivot_role_id`,
  `role_user`.`user_id` as `pivot_user_id`
from
  `users`
  inner join `role_user` on `users`.`id` = `role_user`.`user_id`
where
  `role_user`.`role_id` = X
```

, gdje je X odgovarajuća vrijednost, tj. id prosljećen linkom.

Na adresi http://127.0.0.1:8000/role/1 prikazaće prikazaće sve korisnike koji imaju ulogu 1.

Kod za obrnuti prikaz, tj. prikaz svih uloga određenom korisniku je dat na gitu.

Važna napomena: Laravel na osnovu imena modela zna da povezuje tabele. Nazivi tabele sa kojima se vrsi povezivanje se moze mijenjati unutar metoda hasOne, hasMany, belongesTo i belongesToMany kao na primjer `belongsToMany(User::class, 'role_user');` Ovdje se parametar 'role_user' mogao i izostaviti.

Literatura:

https://laravel.com/docs/10.x/eloquent-relationships