membuat form persetujuan dengan 2 role yang berurutan ppk dahulu aprove, baru pptk bisa approve di laravel, jika keduanya sudah approve baru setuju

**ChatGPT said:**

ChatGPT

Untuk membuat form persetujuan dengan dua role yang berurutan (PPK dan PPTK) di Laravel, di mana PPK harus menyetujui terlebih dahulu, baru PPTK bisa memberikan persetujuan, dan setelah keduanya menyetujui baru status bisa dianggap "setuju", Anda bisa mengikuti langkah-langkah berikut.

**1. Struktur Database**

Anda perlu memiliki tabel yang bisa menyimpan status persetujuan dari PPK dan PPTK. Misalnya, Anda dapat menambahkan tabel approvals dengan struktur seperti berikut:

**Migrasi untuk Tabel approvals**

php

Copy code

php artisan make:migration create\_approvals\_table --create=approvals

Di dalam file migrasi (database/migrations/xxxx\_xx\_xx\_create\_approvals\_table.php), buat struktur tabel seperti berikut:

php

Copy code

public function up()

{

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

$table->id();

$table->foreignId('user\_id')->constrained()->onDelete('cascade'); // ID user yang menyetujui

$table->enum('role', ['ppk', 'pptk']); // Role (PPK atau PPTK)

$table->boolean('is\_approved')->default(false); // Status persetujuan (false = belum disetujui, true = disetujui)

$table->timestamps();

});

}

Setelah itu jalankan migrasi:

bash

Copy code

php artisan migrate

**2. Model Approval**

Buat model untuk tabel approvals.

php

Copy code

php artisan make:model Approval

Di dalam model (app/Models/Approval.php), tambahkan relasi dan logika yang dibutuhkan:

php

Copy code

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Approval extends Model

{

use HasFactory;

protected $fillable = ['user\_id', 'role', 'is\_approved'];

// Relasi ke User (PPK atau PPTK)

public function user()

{

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

}

}

**3. Controller untuk Mengelola Persetujuan**

Buat controller untuk menangani logika persetujuan oleh PPK dan PPTK.

php

Copy code

php artisan make:controller ApprovalController

Di dalam controller (app/Http/Controllers/ApprovalController.php), tambahkan logika untuk persetujuan.

php

Copy code

namespace App\Http\Controllers;

use App\Models\Approval;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class ApprovalController extends Controller

{

public function index()

{

// Tampilkan form persetujuan berdasarkan role user

$approvals = Approval::where('user\_id', Auth::id())->get();

return view('approvals.index', compact('approvals'));

}

public function approve($id)

{

$approval = Approval::findOrFail($id);

// Pastikan hanya PPK yang bisa approve pertama kali

if (Auth::user()->role != 'ppk' && $approval->role == 'ppk') {

return redirect()->back()->with('error', 'Hanya PPK yang bisa menyetujui pertama.');

}

// Jika PPK sudah approve, maka PPTK bisa approve

if ($approval->role == 'ppk' && $approval->is\_approved == true) {

// Jika PPK sudah setujui, berikan kesempatan untuk PPTK approve

$approval->is\_approved = true;

$approval->save();

// Setelah keduanya approve, update status persetujuan akhir

$this->updateStatusSetuju($approval);

}

// Tandai approval sebagai approved

$approval->is\_approved = true;

$approval->save();

return redirect()->route('approvals.index')->with('success', 'Persetujuan berhasil.');

}

private function updateStatusSetuju(Approval $approval)

{

// Cek apakah keduanya sudah approve

$ppkApproval = Approval::where('role', 'ppk')

->where('user\_id', $approval->user\_id)

->first();

$pptkApproval = Approval::where('role', 'pptk')

->where('user\_id', $approval->user\_id)

->first();

if ($ppkApproval && $pptkApproval && $ppkApproval->is\_approved && $pptkApproval->is\_approved) {

// Status persetujuan telah disetujui kedua pihak

$approval->status = 'setuju';

$approval->save();

}

}

}

**4. Routing**

Tambahkan routing untuk controller di file routes/web.php.

php

Copy code

use App\Http\Controllers\ApprovalController;

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

Route::get('/approvals', [ApprovalController::class, 'index'])->name('approvals.index');

Route::post('/approvals/approve/{id}', [ApprovalController::class, 'approve'])->name('approvals.approve');

});

**5. View untuk Menampilkan Form Persetujuan**

Buat view untuk menampilkan form persetujuan sesuai role yang sedang login. Buat file resources/views/approvals/index.blade.php.

php

Copy code

@extends('layouts.app')

@section('content')

<div class="container">

<h2>Persetujuan</h2>

@foreach ($approvals as $approval)

<div>

<h4>{{ $approval->role == 'ppk' ? 'PPK' : 'PPTK' }}: {{ $approval->is\_approved ? 'Disetujui' : 'Belum Disetujui' }}</h4>

@if (!$approval->is\_approved)

<form action="{{ route('approvals.approve', $approval->id) }}" method="POST">

@csrf

<button type="submit" class="btn btn-primary">Setujui</button>

</form>

@endif

</div>

@endforeach

</div>

@endsection

**6. Sistem Keamanan dan Middleware**

Pastikan bahwa setiap pengguna hanya bisa mengakses atau melakukan tindakan berdasarkan peran mereka. Misalnya, hanya PPK yang bisa menyetujui pertama, dan hanya PPTK yang bisa menyetujui setelah PPK.

Anda dapat menggunakan middleware untuk memverifikasi role pengguna. Sebagai contoh:

php

Copy code

// Middleware untuk memverifikasi peran

public function handle($request, Closure $next, $role)

{

if (Auth::check() && Auth::user()->role == $role) {

return $next($request);

}

return redirect()->route('home')->with('error', 'Access Denied');

}

**7. Testing dan Validasi**

Pastikan untuk menguji seluruh alur persetujuan:

* PPK harus memberikan persetujuan terlebih dahulu.
* Setelah PPK menyetujui, PPTK baru bisa memberikan persetujuan.
* Jika keduanya sudah menyetujui, status bisa diubah menjadi "setuju".

Dengan langkah-langkah ini, Anda akan memiliki sistem persetujuan dengan dua peran berurutan di Laravel.

4o

**You said:**

tambahkan polimorfis untuk table aprovable

**ChatGPT said:**

ChatGPT

Untuk menambahkan fitur polimorfisme ke tabel persetujuan (approval) di Laravel, kita dapat memanfaatkan hubungan polimorfis yang memungkinkan kita untuk menggunakan satu tabel untuk menyimpan hubungan ke berbagai model yang berbeda. Ini sangat berguna jika kita ingin agar persetujuan (approval) tidak hanya berlaku untuk satu entitas tertentu (misalnya, proyek), tetapi juga untuk berbagai entitas yang berbeda (misalnya, tugas, permintaan pengadaan, dll).

Berikut adalah langkah-langkah untuk menambahkan fungsionalitas polimorfis pada tabel approvals.

**1. Modifikasi Migrasi approvals untuk Mendukung Polimorfisme**

Tabel approvals harus dimodifikasi agar bisa menyimpan relasi polimorfis. Untuk itu, kita akan menambahkan dua kolom: approvable\_id dan approvable\_type yang digunakan untuk menyimpan referensi ke model yang berbeda.

Buat migrasi baru untuk menambahkan kolom polimorfis:

bash

Copy code

php artisan make:migration add\_approvable\_to\_approvals\_table --table=approvals

Kemudian, dalam file migrasi tersebut, tambahkan kolom approvable\_id dan approvable\_type:

php

Copy code

public function up()

{

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

$table->unsignedBigInteger('approvable\_id'); // ID dari model yang disetujui

$table->string('approvable\_type'); // Nama model yang disetujui

});

}

Jalankan migrasi:

bash

Copy code

php artisan migrate

**2. Modifikasi Model Approval**

Selanjutnya, kita akan mengubah model Approval untuk mendukung relasi polimorfis.

Di dalam model Approval (app/Models/Approval.php), tambahkan relasi morphTo() untuk menunjuk ke model yang berbeda-beda.

php

Copy code

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Approval extends Model

{

use HasFactory;

protected $fillable = ['user\_id', 'role', 'is\_approved', 'approvable\_id', 'approvable\_type'];

// Relasi ke User (PPK atau PPTK)

public function user()

{

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

}

// Relasi polimorfis

public function approvable()

{

return $this->morphTo();

}

}

**3. Menambahkan Relasi Polimorfis pada Model yang Dapat Disetujui**

Misalkan Anda memiliki model Project, Task, atau model lain yang dapat disetujui, Anda perlu menambahkan relasi polimorfis di model-model tersebut. Sebagai contoh, untuk model Project:

php

Copy code

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Project extends Model

{

use HasFactory;

// Relasi ke Approval (polimorfis)

public function approvals()

{

return $this->morphMany(Approval::class, 'approvable');

}

}

Demikian pula, jika Anda memiliki model lain, misalnya Task, Anda dapat menambahkan relasi yang sama:

php

Copy code

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Task extends Model

{

use HasFactory;

// Relasi ke Approval (polimorfis)

public function approvals()

{

return $this->morphMany(Approval::class, 'approvable');

}

}

**4. Menggunakan Polimorfisme di Controller**

Sekarang kita dapat menggunakan hubungan polimorfis ini di controller untuk menangani approval pada entitas yang berbeda. Misalnya, jika Anda ingin memberikan approval pada sebuah Project atau Task, Anda cukup menetapkan approvable ke model yang sesuai.

Misalnya, di dalam ApprovalController, jika Anda ingin menambahkan approval untuk sebuah Project, Anda bisa menulis kode seperti berikut:

php

Copy code

namespace App\Http\Controllers;

use App\Models\Approval;

use App\Models\Project;

use App\Models\User;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class ApprovalController extends Controller

{

public function approve(Request $request, $approvableId, $approvableType)

{

// Pastikan hanya PPK yang bisa menyetujui pertama kali

$approval = Approval::where('approvable\_id', $approvableId)

->where('approvable\_type', $approvableType)

->where('user\_id', Auth::id())

->first();

if ($approval) {

// Lakukan logika persetujuan

$approval->is\_approved = true;

$approval->save();

// Jika ini adalah persetujuan kedua, pastikan status selesai

$this->updateStatusSetuju($approval);

}

return redirect()->route('approvals.index')->with('success', 'Persetujuan berhasil.');

}

private function updateStatusSetuju(Approval $approval)

{

// Cek apakah keduanya sudah approve

$ppkApproval = Approval::where('approvable\_id', $approval->approvable\_id)

->where('approvable\_type', $approval->approvable\_type)

->where('role', 'ppk')

->first();

$pptkApproval = Approval::where('approvable\_id', $approval->approvable\_id)

->where('approvable\_type', $approval->approvable\_type)

->where('role', 'pptk')

->first();

if ($ppkApproval && $pptkApproval && $ppkApproval->is\_approved && $pptkApproval->is\_approved) {

// Status persetujuan telah disetujui kedua pihak

$approvable = $approval->approvable;

$approvable->status = 'setuju'; // Menandakan bahwa proses telah disetujui

$approvable->save();

}

}

}

**5. Menggunakan Form Persetujuan di View**

Di dalam view Anda, pastikan untuk mengirimkan data approvable\_type dan approvable\_id ketika membuat form persetujuan. Misalnya, jika kita ingin persetujuan untuk Project:

php

Copy code

@extends('layouts.app')

@section('content')

<div class="container">

<h2>Persetujuan untuk Proyek</h2>

@foreach ($approvals as $approval)

<div>

<h4>{{ $approval->role == 'ppk' ? 'PPK' : 'PPTK' }}: {{ $approval->is\_approved ? 'Disetujui' : 'Belum Disetujui' }}</h4>

@if (!$approval->is\_approved)

<form action="{{ route('approvals.approve', ['approvableId' => $approvable->id, 'approvableType' => get\_class($approvable)]) }}" method="POST">

@csrf

<button type="submit" class="btn btn-primary">Setujui</button>

</form>

@endif

</div>

@endforeach

</div>

@endsection

Di dalam kode tersebut, kita mengirimkan approvableId dan approvableType yang berfungsi untuk menyatakan model mana yang disetujui (misalnya Project atau Task).

**6. Menyelesaikan Proses Persetujuan**

Setelah keduanya (PPK dan PPTK) menyetujui, Anda dapat melakukan perubahan pada status model yang disetujui, seperti mengubah status proyek menjadi setuju atau status lainnya.

**7. Testing**

Pastikan Anda menguji:

* Penggunaan persetujuan pada model yang berbeda (misalnya Project, Task).
* Alur persetujuan berjalan sesuai dengan urutan yang benar (PPK harus approve dulu, baru PPTK).
* Setelah kedua pihak approve, status model yang disetujui diperbarui dengan benar.

Dengan menggunakan relasi polimorfis, Anda membuat sistem persetujuan yang lebih fleksibel dan dapat digunakan untuk berbagai jenis entitas di aplikasi Anda.

Urutannya pppk, pptk , penanggung jawab final -> selesaikan kontrak