Untuk membuat dan menerima notifikasi secara real-time di Laravel menggunakan Redis, kamu bisa menggunakan kombinasi dari Laravel Echo, Pusher (atau Ably), dan Redis sebagai backend untuk pub/sub. Berikut adalah contoh langkah-langkah yang bisa kamu ikuti:

**1. Instalasi Laravel Echo dan Pusher**

Pertama, pastikan kamu telah menginstal paket Laravel Echo dan Pusher. Jika belum, kamu bisa menambahkannya ke dalam proyek dengan perintah berikut:

bash

Copy code

composer require pusher/pusher-php-server

npm install --save laravel-echo pusher-js

**2. Konfigurasi Pusher**

Buka file .env dan tambahkan konfigurasi Pusher:

env

Copy code

PUSHER\_APP\_ID=your-app-id

PUSHER\_APP\_KEY=your-app-key

PUSHER\_APP\_SECRET=your-app-secret

PUSHER\_APP\_CLUSTER=your-app-cluster

BROADCAST\_DRIVER=pusher

CACHE\_DRIVER=redis

QUEUE\_CONNECTION=redis

SESSION\_DRIVER=redis

Di file config/broadcasting.php, pastikan driver pusher sudah dikonfigurasi:

php

Copy code

'connections' => [

'pusher' => [

'driver' => 'pusher',

'key' => env('PUSHER\_APP\_KEY'),

'secret' => env('PUSHER\_APP\_SECRET'),

'app\_id' => env('PUSHER\_APP\_ID'),

'options' => [

'cluster' => env('PUSHER\_APP\_CLUSTER'),

'useTLS' => true,

],

],

// Connection lainnya...

],

**3. Membuat Event dan Menyiarkan Notifikasi**

Buat event yang akan disiarkan ke pengguna ketika ada notifikasi baru. Kamu bisa membuat event dengan perintah artisan:

bash

Copy code

php artisan make:event NewNotification

Pada class event (App\Events\NewNotification), pastikan event tersebut menggunakan ShouldBroadcast:

php

Copy code

namespace App\Events;

use Illuminate\Broadcasting\Channel;

use Illuminate\Broadcasting\InteractsWithSockets;

use Illuminate\Broadcasting\PresenceChannel;

use Illuminate\Broadcasting\PrivateChannel;

use Illuminate\Contracts\Broadcasting\ShouldBroadcast;

use Illuminate\Foundation\Events\Dispatchable;

use Illuminate\Queue\SerializesModels;

class NewNotification implements ShouldBroadcast

{

use Dispatchable, InteractsWithSockets, SerializesModels;

public $message;

public function \_\_construct($message)

{

$this->message = $message;

}

public function broadcastOn()

{

return new Channel('notifications');

}

}

**4. Membuat Notifikasi dan Menyebarkan Event**

Misalkan kamu memiliki controller yang akan mengirimkan notifikasi:

php

Copy code

use App\Events\NewNotification;

public function sendNotification(Request $request)

{

$message = $request->input('message');

// Sebarkan event ke pengguna

broadcast(new NewNotification($message))->toOthers();

return response()->json(['status' => 'Notification sent!']);

}

**5. Mendengarkan Event di Frontend**

Sekarang, di bagian frontend, kamu harus mendengarkan event yang telah disebarkan menggunakan Laravel Echo dan Pusher.

Buat file resources/js/bootstrap.js atau file JavaScript lainnya:

js

Copy code

import Echo from "laravel-echo";

window.Pusher = require('pusher-js');

window.Echo = new Echo({

broadcaster: 'pusher',

key: process.env.MIX\_PUSHER\_APP\_KEY,

cluster: process.env.MIX\_PUSHER\_APP\_CLUSTER,

encrypted: true

});

window.Echo.channel('notifications')

.listen('NewNotification', (e) => {

console.log(e.message);

alert('New notification: ' + e.message);

});

Jangan lupa untuk menjalankan npm run dev atau npm run watch untuk mengompilasi asset.

**6. Menjalankan Redis Server**

Pastikan Redis server berjalan dengan perintah berikut:

bash

Copy code

redis-server

**7. Testing**

Kamu bisa menguji dengan membuat request ke endpoint yang mengirimkan notifikasi dan melihat apakah notifikasi muncul secara real-time di bagian frontend.

Itu adalah contoh dasar untuk membuat dan menerima notifikasi real-time menggunakan Redis di Laravel. Kamu dapat memperluas fungsionalitas ini dengan menambahkan fitur lebih lanjut sesuai kebutuhan.