Folder and model specific controller: php artisan make:controller Api/UserController -r -m User

Controller ke jevabe route e fetch kore : Route::resource('users',"Api\UserController");

[Tips] View route : Route::view("/users","users.showAll")->name('users.all');

Step 0 : **laravel\_pusher\_docx** ta follow korte hobe

Step 1 : First e ekta controller create korte hobe,

php artisan make:controller Api/UserController -r -m User

Step 2 : Route resource kora ,

Route::resource('users',"Api\UserController");

Step 3 : 3ta event create kora holo

1. UserCreated
2. UserUpdated
3. UserDelated

**UserCreated** event er code, **Channel** type public rakha holo

<?php  
  
namespace App\Events;  
  
use App\User;  
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 UserCreated implements ShouldBroadcast  
{  
 use Dispatchable, InteractsWithSockets, SerializesModels;  
  
 public $user;  
  
 */\*\*  
 \* Create a new event instance.  
 \*  
 \** ***@return*** *void  
 \*/* public function \_\_construct(User $user)  
 {  
 $this->user = $user;  
 }  
  
 */\*\*  
 \* Get the channels the event should broadcast on.  
 \*  
 \** ***@return*** *\Illuminate\Broadcasting\Channel|array  
 \*/* public function broadcastOn()  
 {  
 \Log::*debug*("User Created ". $this->user->name);  
 return new Channel('users');  
 }  
}

**UserUpdated** code,

<?php  
  
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;  
use App\User;  
  
class UserUpdated implements ShouldBroadcast  
{  
 use Dispatchable, InteractsWithSockets, SerializesModels;  
  
  
 public $user;  
  
 */\*\*  
 \* Create a new event instance.  
 \*  
 \** ***@return*** *void  
 \*/* public function \_\_construct(User $user)  
 {  
 $this->user = $user;  
 }  
  
 */\*\*  
 \* Get the channels the event should broadcast on.  
 \*  
 \** ***@return*** *\Illuminate\Broadcasting\Channel|array  
 \*/* public function broadcastOn()  
 {  
 \Log::*debug*("User Updated ". $this->user->name);  
  
 return new Channel('users');  
 }  
}

**UserDelated** er trait theke **SerializesModels** trait kore remove kore dite hobe. **SerializesModels** Queue er system holo latest/fresh instance retrieve kore. Jehetu delete route korle Model er kono fresh instance thake na tai **SerializesModels** trait thakle delete korle socket ar route back korte pare na.

<?php  
  
namespace App\Events;  
  
use App\User;  
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 UserDeleted implements ShouldBroadcast  
{  
 use Dispatchable, InteractsWithSockets;  
  
 public $user;  
  
 */\*\*  
 \* Create a new event instance.  
 \*  
 \** ***@return*** *void  
 \*/* public function \_\_construct(User $user)  
 {  
 $this->user = $user;  
 }  
  
 */\*\*  
 \* Get the channels the event should broadcast on.  
 \*  
 \** ***@return*** *\Illuminate\Broadcasting\Channel|array  
 \*/* public function broadcastOn()  
 {  
 \Log::*debug*("User Deleted ". $this->user->name);  
  
 return new Channel('users');  
 }  
}

Step 4 : model er moddhe dispatched event gula add korte hobe.

protected $dispatchesEvents = [  
 'created' => UserCreated::class,  
 'updated' => UserUpdated::class,  
 'deleted' => UserDeleted::class  
];

Step 5 : **blade** file er moddhe part gulake jevabe shajano hoise,

@extends('layouts.app')  
  
@section('content')  
 <div class="container">  
 <div class="row justify-content-center">  
 <div class="col-md-8">  
 <div class="card">  
 <div class="card-header">{{ \_\_('Users') }}</div>  
  
 <div class="card-body">  
 <ul id="users">  
  
 </ul>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
@endsection

Then **api** use kore data fetch kora hoise,

<script>  
 ***window***.axios.get('/api/users').then(  
 response => {  
 const userElement = ***document***.getElementById("users");  
  
 let users = response.data;  
  
 users.forEach((user, index) => {  
 let elemnt = ***document***.createElement("li");  
 elemnt.setAttribute('id', user.id);  
 elemnt.innerText = user.name;  
 userElement.appendChild(elemnt);  
  
 });  
 }  
 );  
</script>

And porer step e **Broadcast** kore data update kora hoise,

<script>  
 Echo.channel('users').listen("UserCreated", (e) => {  
  
 const userElement = ***document***.getElementById("users");  
  
 let element = ***document***.createElement("li");  
 element.setAttribute('id', e.user.id);  
 element.innerText = e.user.name;  
 userElement.appendChild(element);  
  
 }).listen("UserUpdated", (e) => {  
  
 const element = ***document***.getElementById(e.user.id);  
 element.innerText = e.user.name;  
  
 }).listen("UserDeleted", (e) => {  
  
 const element = ***document***.getElementById(e.user.id);  
 element.parentNode.removeChild(element);  
 });  
</script>

Full **blade** file code,

@extends('layouts.app')  
  
@section('content')  
 <div class="container">  
 <div class="row justify-content-center">  
 <div class="col-md-8">  
 <div class="card">  
 <div class="card-header">{{ \_\_('Users') }}</div>  
  
 <div class="card-body">  
 <ul id="users">  
  
 </ul>  
 </div>  
 </div>  
 </div>  
 </div>  
 </div>  
@endsection  
  
@push('scripts')  
 <script>  
 ***window***.axios.get('/api/users').then(  
 response => {  
 const userElement = ***document***.getElementById("users");  
  
 let users = response.data;  
  
 users.forEach((user, index) => {  
 let elemnt = ***document***.createElement("li");  
 elemnt.setAttribute('id', user.id);  
 elemnt.innerText = user.name;  
 userElement.appendChild(elemnt);  
  
 });  
 }  
 );  
 </script>  
  
 <script>  
 Echo.channel('users').listen("UserCreated", (e) => {  
  
 const userElement = ***document***.getElementById("users");  
  
 let element = ***document***.createElement("li");  
 element.setAttribute('id', e.user.id);  
 element.innerText = e.user.name;  
 userElement.appendChild(element);  
  
 }).listen("UserUpdated", (e) => {  
  
 const element = ***document***.getElementById(e.user.id);  
 element.innerText = e.user.name;  
  
 }).listen("UserDeleted", (e) => {  
  
 const element = ***document***.getElementById(e.user.id);  
 element.parentNode.removeChild(element);  
 });  
 </script>  
@endpush

Ekhane **broadcast (Echo)** er moddhe ja kora hoise,

1. **users** naam er channel ta hocche **broadcast channel** name jeta **UserCreated, UserUpdated, UserDelated** channel er **broadcastOn()** function er moddhe set kora hoyechlio.

public function broadcastOn()  
{  
 \Log::*debug*("User Created ". $this->user->name);  
 return new Channel('users');  
}

1. listeners er moddhe baki event gula ke set kora hoise chain kore.