trazer informações em ajax em datatable em laravel usando Yajra DataTables

Para trazer informações em AJAX em DataTables usando o pacote Yajra DataTables no Laravel, siga os seguintes passos:

1. Instale o pacote Yajra DataTables via Composer usando o comando:

```
composer require yajra/laravel-datatables-oracle
```

2. Em seguida, adicione o provedor de serviços DataTables no seu arquivo **config/app.php**:

php Copy code

```
'providers' => [ // ... Yajra\DataTables\DataTablesServiceProvider::class, ],
```

3. Depois disso, publique os arquivos de configuração e os recursos necessários executando o comando:

```
php artisan vendor:publish --tag=datatables
```

4. Em seguida, crie um controller que irá gerenciar a exibição dos dados em DataTables. No exemplo abaixo, chamaremos o controller de **UserController**:

php Copy code

```
namespace App\Http\Controllers; use App\Models\User; use DataTables; class
UserController extends Controller { public function index() { return
view('users.index'); } public function getUsers() { $users = User::query();
return DataTables::of($users) ->addColumn('action', function ($user) { //
Adicione as colunas de ação aqui }) ->rawColumns(['action']) ->toJson(); } }
```

5. Na view **users.index**, adicione o seguinte código:

html Copy code

```
<table id="users-table"> <thead> <tr> <th>Name</th> <th>Email</th>
<th>Action</th> </tr> </thead> </table> <script> $(function() { $('#users-
```

```
table').DataTable({ processing: true, serverSide: true, ajax: '{{
route('users.data') }}', columns: [ { data: 'name', name: 'name' }, { data:
'email', name: 'email' }, { data: 'action', name: 'action', orderable: false,
searchable: false } ] }); }); </script>
```

Observe que o **ajax** do DataTables está apontando para a rota **users.data**, que deverá ser definida no arquivo **routes/web.php**.

6. Por fim, adicione a rota **users.data** no arquivo **routes/web.php**:

php Copy code

```
Route::get('users/data', 'UserController@getUsers')->name('users.data');
```

Pronto! Agora, os dados dos usuários serão exibidos na tabela DataTables usando AJAX. O Yajra DataTables se encarregará de fazer as consultas no banco de dados e retornar os dados para a view.

adicione colunas para ação

Para adicionar colunas de ação em um DataTables usando o pacote Yajra DataTables no Laravel, você pode usar a função **addColumn** do Yajra DataTables no seu controller. Aqui está um exemplo de como adicionar colunas de ação:

php Copy code

```
use DataTables; class UserController extends Controller { public function
getUsers() { $users = User::query(); return DataTables::of($users) -
>addColumn('action', function ($user) { $btn = '<a href="/users/' . $user->id
. '/edit" class="edit btn btn-primary btn-sm">Editar</a>'; $btn .= '<a
href="/users/' . $user->id . '" class="delete btn btn-danger btn-sm ml-
2">Excluir</a>'; return $btn; }) ->rawColumns(['action']) ->toJson(); } }
```

Observe que a função **addColumn** recebe dois parâmetros: o nome da coluna e uma função de fechamento que retorna o conteúdo da coluna. Nesse caso, estamos criando

uma coluna "Ação" e adicionando dois botões: um para editar o usuário e outro para excluí-lo.

Além disso, a função **`rawColumns`** é usada para dizer ao DataTables que a coluna "Ação" contém HTML, para que ele possa renderizá-la corretamente.

Por fim, na view onde você exibe a tabela DataTables, basta adicionar a coluna "Ação" ao array de colunas:

```js
$('#users-table').DataTable({ // ... columns: [ { data: 'name', name: 'name' }, { data: 'email', name: 'email' }, { data: 'action', name: 'action', orderable: false, searchable: false } ] });
```

Observe que a propriedade **`orderable`** está definida como **`false`** para impedir que o DataTables ordene a coluna "Ação" e **`searchable`** está definida como **`false`** para impedir que o DataTables adicione um campo de pesquisa para a coluna "Ação".