CRUD OPERATION IN LARAVEL REACTJS USING INERTIA

1. Create your migration (database/migrations)
   - Use this command: php artisan make:migration create_custom_contents_table
   - After creating, create the fields for your table

```
 public function up(): void
    {
        Schema::create('custom_contents', function (Blueprint $table) {
            $table->id();
            $table->unsignedBigInteger('user_id')->nullable();
            $table->string('content_type');
            $table->string('content_title');
            $table->string('content_text');
            $table->string('subject')->nullable();
            $table->timestamps();

            $table->foreign('user_id')->references('id')->on('users')-
>onDelete('cascade');

        });
    }
 /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('custom_contents');
    }
```

   - If you want to change something in your table like a field name or data type, create a new migration like this:

```
public function up(): void
    {
        Schema::table('custom_contents', function (Blueprint $table) {
            $table->text('content_title')->change();
            $table->longText('content_text')->change();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
```

```
    {
        Schema::table('custom_contents', function (Blueprint $table) {
            $table->string('content_text')->change();
            $table->string('content_title')->change();
        });
    }
```

2. Create a model
   - Use this command php artisan make:model CustomContent
   - There are other ways to create the model and migration simultaneously, php artisan make:model CustomContent -m
   - After creating, define fields and relationships in the model(if there are any)

```
- class CustomContent extends Model
- {
-     use HasFactory;
-
-     protected $table = 'custom_contents';
-
-     protected $fillable = [
-         'user_id',
-         'content_type',
-         'content_title',
-         'content_text',
-         'subject',
-         'content_status'
-     ];
-
-     //This is to format the date for the modified_at
-     public function getUpdatedAtAttribute($value)
-     {
-         return Carbon::parse($value)->format('Y-m-d');
-     }
-
-     //This defines the relationship of this table to the user table
- public function user()
-     {
-         return $this->belongsTo(User::class, 'user_id');
-     }
-
-
- }
```

In the user model file we also have this:

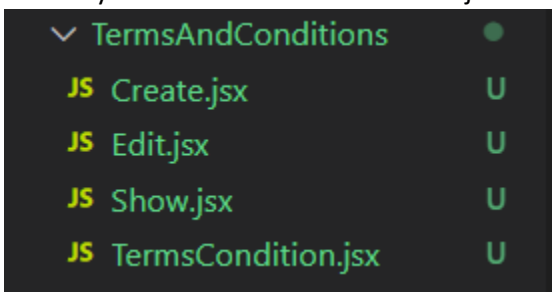
```php
public function custom_content(): HasOne
    {
        return $this->hasMany(CustomContent::class, 'user_id');
    }
```

Which defines the one to many relationship of the table user and custom contents. This is Eloquent in Laravel

3. Create a controller resource
   - Use this command php artisan make:controller TermsAndConditionController --resource
4. Add the route of your controller in the web.php

```php
Route::resource('manage-terms-and-conditions',
TermsAndConditionController::class);
```

5. Create files for your index, edit, create, and show
   - It is recommended to create separate files for different crud functionalities for readability and easy to maintain. TermsCondition.jsx is the Index



   -

6. Create.js

```jsx
import InputError from '@/Components/InputError';
import InputLabel from '@/Components/InputLabel';
import LongTextInput from '@/Components/LongTextInput';
import Modal from '@/Components/Modal';
import PrimaryButton from '@/Components/PrimaryButton';
import TextInput from '@/Components/TextInput';
import { Head, useForm } from '@inertiajs/react';
import { useState } from 'react';

export default function Create({ isOpen, onClose }) {
    const { data, setData, post, processing, errors, reset } = useForm({
        content_title: '',
        content_text: '',
    });
```

```
    const submit = (e) => {
        e.preventDefault();
        post(route('manage-terms-and-conditions.store'), {
            onSuccess: () => {
                onClose();
                alert('Success!');
                reset();
            },
        });
    };

    return (
        <Modal show={isOpen} onClose={onClose}>
                <div className="bg-customBlue p-3" >
                    <h2 className="text-xl text-white font-bold">Add Term and
Condition</h2>
                </div>

                <div className="p-6 space-y-5">
                <form onSubmit={submit}>
                <div className='space-y-5'>
                    <div className="flex flex-col">
                        <InputLabel htmlFor="content_title" value="Title" />
                        <TextInput
                            id="content_title"
                            value={data.content_title}
                            onChange={(e) => setData('content_title',
e.target.value)}
                            type="text"
                            className="mt-1 block w-full"
                            placeholder="Title"
                        />
                        <InputError message={errors.content_title} className="mt-
2" />
                    </div>

                    <div className="flex flex-col">
                        <InputLabel htmlFor="content_text" value="Content" />
                        <LongTextInput
                            id="content_text"
                            value={data.content_text}
                            onChange={(e) => setData('content_text',
e.target.value)}
                            className="mt-1 block w-full"
                            placeholder="Content"
```

```
                        />
                        <InputError message={errors.content_text} className="mt-
2" />
                    </div>

                    <div className="mt-6 flex">
                        <PrimaryButton type="submit" disabled={processing}>
                            Save
                        </PrimaryButton>
                    </div>
                    </div>
                </form>

                </div>

                <div className="bg-customBlue p-2 flex justify-end" >
                    <button onClick={onClose} className="text-white text-right
mr-5">Close</button>
                </div>
            </Modal>
    );
}
```

- In the TermsAndConditionController.php

```
/**
     * Show the form for creating a new resource.
     */
    public function create()
    {
        return Inertia::render('SuperAdmin/TermsAndConditions/Create');
    }

    /**
     * Store a newly created resource in storage.
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'content_title' => 'required|string|max:1000',
            'content_text' => 'required|string',
        ]);
```

```
        CustomContent::create([
            'user_id' => Auth::id(),
            'content_type' => 'terms and conditions',
            'content_title' => $request->content_title,
            'content_text' => $request->content_text,
            'subject' => null,
        ]);

        return redirect(route('manage-terms-and-conditions.index'))->with('success', 'Terms and conditions created successfully.');
    }
```

7. Index.js
   - This is how you get all the data from the database in your controller

```
 public function index()
    {
        $termsConditions = CustomContent::with('user')
            ->where('content_type', 'terms and conditions')
            ->get();

        return Inertia::render('SuperAdmin/TermsAndConditions/TermsCondition', [
            'termsConditions' => $termsConditions,
        ]);
    }
```

```
//Then pass the termsConditions as prop so that we can use it

export default function TermsCondition({ auth, termsConditions = [] }) {
       const [isCreateModalOpen, setIsCreateModalOpen] = useState(false);

    const [isShowModalOpen, setIsShowModalOpen] = useState(false);
    const [isEditModalOpen, setIsEditModalOpen] = useState(false);
    const [selectedTerms, setSelectedTerms] = useState(null);


    return (
        <AdminLayout
            user={auth.user}
            header={<h2 className="font-semibold text-xl text-gray-800 leading-tight">Terms And Condition</h2>}
        >
            <Head title="Terms Condition" />
```

```
            <div className="py-12">
                <div className="max-w-7xl mx-auto sm:px-6 lg:px-8">
                    <div className="bg-white overflow-hidden shadow-sm
sm:rounded-lg">
                        <div className="flex flex-row justify-between m-3">
                            <div className="text-gray-900">Terms and
Condition</div>
                            <div>
                                <AddButton onClick={openCreateModal}
className="text-customBlue hover:text-white space-x-1">
                                    <FaPlus /><span>Add T&Cs</span>
                                </AddButton>
                            </div>
                        </div>

                        <div className="overflow-x-auto shadow-md sm:rounded-lg
px-5">
                            <div className="flex items-center justify-between
flex-column md:flex-row flex-wrap space-y-4 md:space-y-0 py-4 bg-white">
                                <label className="sr-only">Search</label>
                                <div className="relative">
                                    <div className="absolute inset-y-0 rtl:inset-
r-0 start-0 flex items-center ps-3 pointer-events-none">
                                        <svg className="w-4 h-4 text-gray-500
dark:text-gray-400" aria-hidden="true" xmlns="http://www.w3.org/2000/svg"
fill="none" viewBox="0 0 20 20">
                                            <path stroke="currentColor"
strokeLinecap="round" strokeLinejoin="round" strokeWidth="2" d="m19 19-4-4m0-7A7
7 0 1 1 1 8a7 7 0 0 1 14 0Z"/>
                                        </svg>
                                    </div>
                                    <input type="text" id="table-search-users"
className="block pt-2 ps-10 text-sm text-gray-900 border border-gray-300 rounded-
lg w-80 bg-gray-50 focus:ring-blue-500 focus:border-blue-500"
placeholder="Search" />
                                </div>
                            </div>
                            <table className="w-full text-sm text-left rtl:text-
right text-gray-500">
                                <thead className="text-xs text-gray-700 uppercase
bg-gray-50">
                                    <tr>
                                        <th scope="col" className="p-4">
                                            <div className="flex items-center">
```

```
                                        <input id="checkbox-all-search"
type="checkbox" className="w-4 h-4 text-blue-600 bg-gray-100 border-gray-300
rounded focus:ring-blue-500" />
                                        <label htmlFor="checkbox-all-
search" className="sr-only">checkbox</label>
                                    </div>
                                </th>
                                <th scope="col" className="px-6 py-3">
                                    Title
                                </th>
                                <th scope="col" className="px-6 py-3">
                                    Content
                                </th>
                                <th scope="col" className="px-6 py-3">
                                    Modified By
                                </th>
                                <th scope="col" className="px-6 py-3">
                                    Date Modified
                                </th>
                                <th scope="col" className="px-6 py-3">
                                    Action
                                </th>
                            </tr>
                        </thead>
                        <tbody>
                            {termsConditions.map((tc) => (
                                <tr key={tc.id} className="bg-white
border-b dark:bg-gray-800 dark:border-gray-700 hover:bg-gray-50">
                                    <td className="w-4 p-4">
                                        <div className="flex items-
center">
                                            <input id="checkbox-table-
search-1" type="checkbox" className="w-4 h-4 text-blue-600 bg-gray-100 border-
gray-300 rounded focus:ring-blue-500" />
                                            <label htmlFor="checkbox-
table-search-1" className="sr-only">checkbox</label>
                                        </div>
                                    </td>
                                    <th scope="row" className="flex
items-center px-6 py-4 text-gray-900">
                                        <div className="pl-3">
                                            <div className="text-base
font-semibol max-w-44 truncate">{tc.content_title}</div>
                                        </div>
                                    </th>
```

```
                                        <td className="px-6 py-4 max-w-60
truncate">{tc.content_text}</td>
                                        <td className="px-6 py-
4">{tc.user.name}</td>
                                        <td className="px-6 py-
4">{tc.updated_at}</td>
                                        <td className="px-6 py-4 flex flex-
col space-y-1">
                                            <a onClick={() =>
openIndexModal(tc)} className="font-medium text-blue-600 dark:text-blue-500
hover:underline cursor-pointer">View</a>
                                            <a onClick={() =>
openEditModal(tc)} className="font-medium text-blue-600 dark:text-blue-500
hover:underline cursor-pointer">Edit</a>
                                            <a onClick={() =>
deleteTermCondition(tc.id)} className="font-medium text-blue-600 dark:text-blue-
500 hover:underline cursor-pointer">Delete</a>
                                        </td>
                                    </tr>
                                ))}
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </div>

        <Create isOpen={isCreateModalOpen} onClose={closeCreateModal} />
        {selectedTerms && <Show isOpen={isShowModalOpen}
onClose={closeIndexModal} termConditions={selectedTerms} />}
        {selectedTerms && <Edit isOpen={isEditModalOpen}
onClose={closeEditModal} termConditions={selectedTerms} />}

    </AdminLayout>
  );
}
```

8. Edit.js

```
/**
     * Show the form for editing the specified resource.
     */
    public function edit(CustomContent $termsAndCondition): Response
    {
        return Inertia::render('SuperAdmin/TermsAndConditions/Edit', [
```

```
            'termConditions' => $termsAndCondition->load('user'),
        ]);
    }

    /**
     * Update the specified resource in storage.
     */
    public function update(Request $request, $id): RedirectResponse
    {
        //For debugging. You can see the result in storage/logs/laravel.log
        \Log::info('Update request received for tc ID: ' . $id);
        \Log::info('Update request data: ', $request->all());

        $request->validate([
            'content_title' => 'required|string|max:1000',
            'content_text' => 'required|string',
        ]);

        $termsAndCondition = CustomContent::find($id);
        $termsAndCondition->update([
            'content_title' => $request->content_title,
            'content_text' => $request->content_text,
            'user_id' => Auth::id(),
        ]);

        return redirect(route('manage-terms-and-conditions.index'))-
>with('success', 'Terms and conditions updated successfully.');
    }
```

```
export default function Edit({ isOpen, onClose, termConditions }) {
    const { data, setData, put, processing, errors, reset } = useForm({
        content_title: termConditions.content_title,
        content_text: termConditions.content_text,
    });

    const submit = (e) => {
        e.preventDefault();
        put(route('manage-terms-and-conditions.update', termConditions.id), {
            ...data,
            preserveScroll: true,
            onSuccess: () => {
                onClose();
                alert('Successfully Updated!');
            },
```

```
            onError: (errors) => {
                console.error('Update failed', errors);
            },
        });
    };

    return (
        <Modal show={isOpen} onClose={onClose}>
            <div className="bg-customBlue p-3">
                <h2 className="text-xl text-white font-bold">Update Term and
Condition</h2>
            </div>

            <div className="p-6 space-y-5">
                <form onSubmit={submit}>
                    <div className='space-y-5'>
                        <div className="flex flex-col">
                            <InputLabel htmlFor="content_title" value="Title" />
                            <TextInput
                                id="content_title"
                                value={data.content_title}
                                onChange={(e) => setData('content_title',
e.target.value)}
                                type="text"
                                className="mt-1 block w-full"
                                placeholder="Title"
                            />
                            <InputError message={errors.content_title}
className="mt-2" />
                        </div>

                        <div className="flex flex-col">
                            <InputLabel htmlFor="content_text" value="Content" />
                            <LongTextInput
                                id="content_text"
                                value={data.content_text}
                                onChange={(e) => setData('content_text',
e.target.value)}
                                className="mt-1 block w-full"
                                placeholder="Content"
                            />
                            <InputError message={errors.content_text}
className="mt-2" />
                        </div>
```

```
                        <div className="mt-6 flex">
                            <PrimaryButton type="submit" disabled={processing}>
                                Save
                            </PrimaryButton>
                        </div>
                    </div>
                </form>
            </div>

            <div className="bg-customBlue p-2 flex justify-between">
                <button className="text-white text-right ml-5 hover:text-gray-
300">Make Unavailable</button>
                <button onClick={onClose} className="text-white text-right mr-
5">Close</button>
            </div>
        </Modal>
    );
}
```

9. Show.js
   - This is just to view the details

```
/**
     * Display the specified resource.
     */
    public function show(CustomContent $termsAndCondition): Response
    {
        return Inertia::render('SuperAdmin/TermsAndConditions/Show', [
            'termConditions' => $termsAndCondition,
        ]);
    }
```

```
export default function Show({ isOpen, onClose, termConditions }) {
    if (!termConditions) return null;

    return (
        <Modal show={isOpen} onClose={onClose}>
            <div className="bg-customBlue p-3">
                <h2 className="text-xl text-white font-bold">View Term and
Condition</h2>
            </div>

            <div className="p-6 space-y-5">
                <div className="space-y-5">
                    <div className="flex flex-col">
```

```
                        <label className="block text-sm font-medium text-
gray-700">Title</label>
                        <div className="mt-1 p-2 border border-gray-300
rounded-md">
                            {termConditions.content_title}
                        </div>
                    </div>

                    <div className="flex flex-col">
                        <label className="block text-sm font-medium text-
gray-700">Content</label>
                        <div className="mt-1 p-2 border border-gray-300
rounded-md break-words">
                            {termConditions.content_text}
                        </div>
                    </div>

                    <div className="flex flex-row justify-between">
                        <div>
                            <label className="block text-sm font-medium
text-gray-700">Date Created: {termConditions.created_at}</label>
                        </div>

                        <div>
                            <label className="block text-sm font-medium
text-gray-700">Date Modified: {termConditions.updated_at}</label>
                        </div>
                    </div>
                </div>
            </div>

            <div className="bg-customBlue p-2 flex justify-end">
                <button onClick={onClose} className="text-white text-right
mr-5">Close</button>
            </div>
        </Modal>
    );
}
```

10. For the delete functionality.

```
/**
    * Remove the specified resource from storage.
    */
   public function destroy($id): RedirectResponse
```

```
    {
        CustomContent::find($id)->delete();

        return redirect(route('manage-terms-and-conditions.index'))-
>with('success', 'Terms and conditions deleted successfully.');
    }
```

```
<a onClick={() => deleteTermCondition(tc.id)} className="font-medium text-
blue-600 dark:text-blue-500 hover:underline cursor-pointer">Delete</a>
```

- In the TermsCondition.js/Index, this is the method for the delete functionality

```
const deleteTermCondition = (id) => {
        if (confirm("Are you sure you want to delete this term and
condition?")) {
            router.delete(route('manage-terms-and-conditions.destroy',
id), {
                preserveScroll: true,
                onSuccess: () => {
                    alert('Successfully deleted!');
                },
            });
        }
    };
```