Here’s a detailed guide on creating the API for the "Registrasi Event Olahraga" in Laravel, including Read and Insert functionality with MySQL. Then, I'll guide you through creating a simple React.js/Next.js frontend for accessing the API.

### Step 1: Set Up Laravel Project

1. \*\*Create a new Laravel project\*\* by running:

```bash

laravel new event-registration-api

```

Or, if you are using Composer:

```bash

composer create-project laravel/laravel event-registration-api

```

2. \*\*Set up the database connection\*\* in `.env` file:

```plaintext

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=event\_registration

DB\_USERNAME=your\_username

DB\_PASSWORD=your\_password

```

3. \*\*Create a database\*\* named `event\_registration` in MySQL.

### Step 2: Create the Table and Model

1. \*\*Generate a migration and model\*\* for `EventRegistration`:

```bash

php artisan make:model EventRegistration -m

```

2. \*\*Define the table structure\*\* in the generated migration file (`database/migrations/xxxx\_xx\_xx\_create\_event\_registrations\_table.php`):

```php

Schema::create('event\_registrations', function (Blueprint $table) {

$table->id();

$table->string('participant\_name');

$table->string('event\_name');

$table->date('event\_date');

$table->string('registration\_number')->unique();

$table->string('category');

$table->timestamps();

});

```

3. \*\*Run the migration\*\* to create the table:

```bash

php artisan migrate

```

### Step 3: Define API Routes

1. \*\*Open the routes file\*\* at `routes/api.php` and add the following routes:

```php

use App\Http\Controllers\EventRegistrationController;

Route::get('/event-registrations', [EventRegistrationController::class, 'index']);

Route::post('/event-registrations', [EventRegistrationController::class, 'store']);

```

### Step 4: Create Controller for Event Registration

1. \*\*Generate a controller\*\*:

```bash

php artisan make:controller EventRegistrationController

```

2. \*\*Define the `index` and `store` methods\*\* in `EventRegistrationController.php`:

```php

namespace App\Http\Controllers;

use App\Models\EventRegistration;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

class EventRegistrationController extends Controller

{

// GET all data

public function index()

{

$registrations = EventRegistration::all();

return response()->json([

'code' => 200,

'message' => 'Success',

'data' => $registrations

], 200);

}

// POST new data

public function store(Request $request)

{

$validator = Validator::make($request->all(), [

'participant\_name' => 'required|string|max:255',

'event\_name' => 'required|string|max:255',

'event\_date' => 'required|date',

'registration\_number' => 'required|string|unique:event\_registrations',

'category' => 'required|string|max:255',

]);

if ($validator->fails()) {

return response()->json([

'code' => 400,

'message' => 'Failed',

'errors' => $validator->errors()

], 400);

}

EventRegistration::create($request->all());

return response()->json([

'code' => 200,

'message' => 'Success'

], 200);

}

}

```

3. \*\*Enable mass assignment\*\* in the `EventRegistration` model (`app/Models/EventRegistration.php`):

```php

protected $fillable = [

'participant\_name',

'event\_name',

'event\_date',

'registration\_number',

'category',

];

```

### Step 5: Test the API with Postman

- \*\*For `GET` Request\*\*:

- Set method to `GET`, and URL to `http://localhost:8000/api/event-registrations`.

- Expect JSON response like:

```json

{

"code": 200,

"message": "Success",

"data": [...]

}

```

- \*\*For `POST` Request\*\*:

- Set method to `POST`, and URL to `http://localhost:8000/api/event-registrations`.

- Include JSON body:

```json

{

"participant\_name": "Jane Doe",

"event\_name": "Marathon 2024",

"event\_date": "2024-11-10",

"registration\_number": "A123456",

"category": "Adult"

}

```

### Step 6: Create React.js/Next.js Frontend to Access API

1. \*\*Create a new React.js or Next.js project\*\* in a separate folder:

```bash

npx create-react-app event-registration-frontend

```

Or for Next.js:

```bash

npx create-next-app event-registration-frontend

```

2. \*\*Set up Axios\*\* to handle API requests:

```bash

npm install axios

```

3. \*\*Create a form\*\* and table component for adding and displaying registrations, and make requests to your API:

- Use Axios for `GET` and `POST` requests.

- For styling, use a framework like Tailwind CSS:

```bash

npm install -D tailwindcss

npx tailwindcss init

```

4. \*\*Display the Data\*\*:

- Create a `useEffect` hook in your main component to fetch data when the component mounts.

- Map over the data and display it in a table.

5. \*\*Add Data Using Form\*\*:

- Create a form with fields for `participant\_name`, `event\_name`, `event\_date`, `registration\_number`, and `category`.

- Handle form submission to post data to your API and update the list.

6. \*\*Test the Frontend\*\*:

- Run the frontend application:

```bash

npm start

```

- Use the form to add a new registration and verify the data updates dynamically.

This setup will give you a full API integration with a simple frontend interface for event registration. Let me know if you need more detailed instructions on specific parts!