CMS MINDS

How to Connect Headless Drupal With React

By Jayesh Makwana Feb 26, 2025

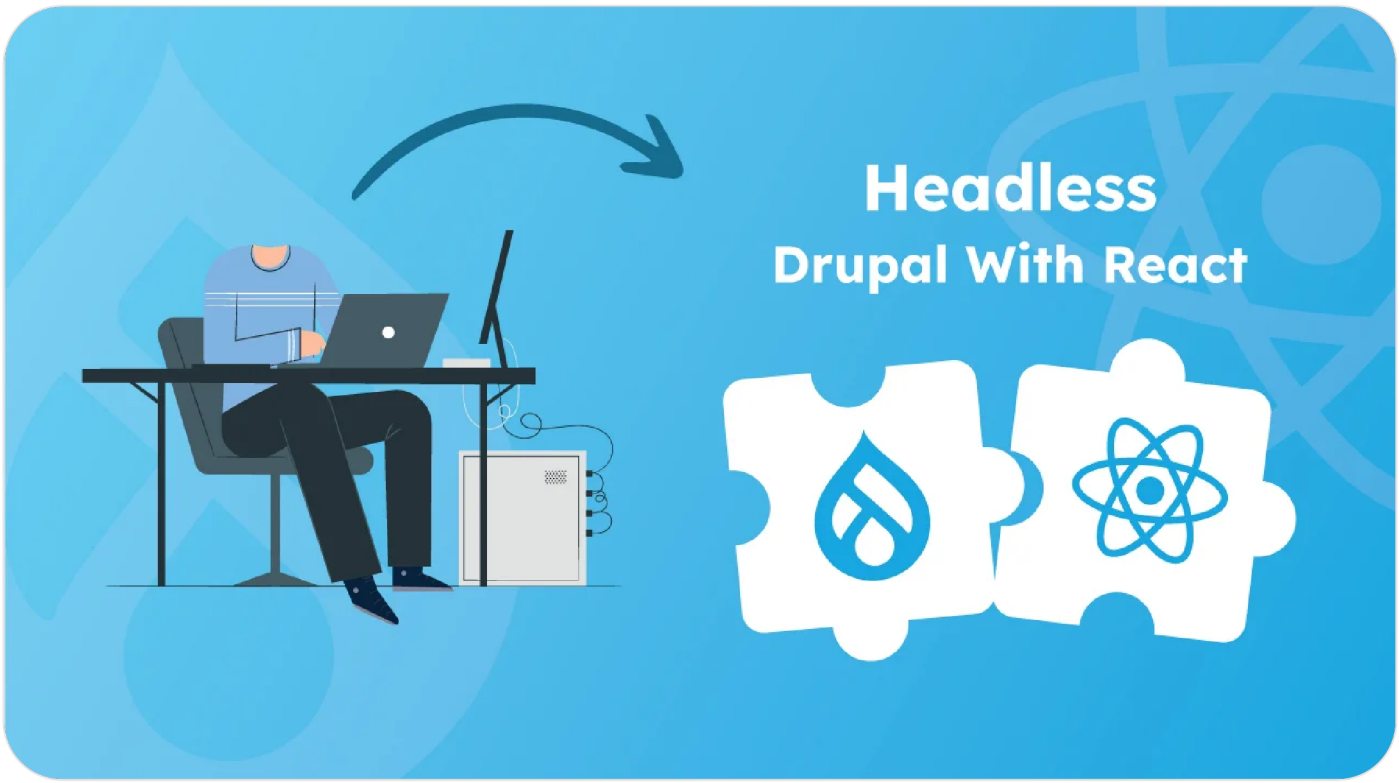


Table of content

CMS MINDS

In web development, decoupled or headless CMS architectures have gained significant traction lately.

But what does a headless CMS do?

A headless CMS allows you to manage your content in one place while using any technology stack to display it. This flexibility is particularly beneficial when leveraging modern JavaScript frameworks like React for building dynamic user interfaces.

Then what does Drupal do in this context?

Drupal, a robust and flexible CMS, can operate in a headless mode, serving content via APIs.

On the other hand, React, a popular JavaScript library for building user interfaces, pairs exceptionally well with a headless CMS setup.

So this guide will walk you through the process of **connecting headless Drupal with React**, enabling you to create powerful, dynamic web applications.

Prerequisites

Before diving in, ensure you have the following:

- Basic knowledge of Drupal and React
- Drupal 9 or 10 installed
- Node.js and npm installed
- A local development environment set up

CMS MINDS

Install and Configure Drupal

Start by installing Drupal. You can download it from the official Drupal website or use Composer:

```
composer create-project drupal/recommended-project my_site_name_dir
```

Navigate to your Drupal installation directory and run the setup wizard. Once installed, log in to the admin panel.

Enable Necessary Modules

To make Drupal headless, enable the JSON

and RESTful Web Services modules. These modules allow Drupal to expose content via RESTful APIs.

```
drush en jsonapi -y

drush en rest -y
```

Configure Permissions

Ensure that appropriate permissions are set so that the API endpoints are accessible. Go to “People” > “Permissions” and configure permissions for anonymous and authenticated users as needed.

Create Content Types and Fields

Create content types and fields to structure your content. For example, you might create an “Article” content type with fields like “Title,” “Body,” and “Image.”

Add Sample Content

Populate your content types with some sample data. This will be useful for testing API endpoints.

CMS MINDS

JSON is a specification for building APIs in JSON. Drupal's JSON module automatically provides RESTful endpoints for all content entities.

Accessing Content Through JSON

Endpoints

You can access your content via JSON endpoints. For example, to fetch all articles, you can use the following endpoint:

```
http://your-drupal-site.com/jsonapi/node/article
```

Example API Requests

Here's an example of fetching articles:

```
curl http://your-drupal-site.com/jsonapi/node/article
```

You should see a JSON response containing your articles.

Step 3: Setting Up a React Project

Create a New React Project

Use Create React App to bootstrap your React project:

```
npx create-react-app my-react-app

cd my-react-app
```

Project Structure Overview

Create React App provides a basic project structure. Familiarize yourself with the key files and directories.

CMS MINDS

Install axios for making API requests and react-router for handling routing:

```
npm install axios react-router-dom
```

Step 4: Fetching Data from Drupal in React

Introduction to Axios for API Requests

Axios is a promise-based HTTP client for the browser and Node.js. It makes API requests easier to manage.

Fetching Data from Drupal in React

Set up axios in your React project. Create a new file src/api.js to configure axios:

```
import axios from ‘axios’;

const api = axios.create({

baseURL: ‘http://your-drupal-site.com/jsonapi’,

});

export default api;
```

Making GET Requests to JSON

Endpoints

In your React components, you can now use axios to fetch data. Here’s an example in a component:

```
import React, { useEffect, useState } from ‘react’;
```

CMS MINDS

```
const Articles = () => {

const [articles, setArticles] = useState([]);

useEffect(() => {

api.get('/node/article')

.then(response => {

setArticles(response.data.data);

})

.catch(error => {

console.error('Error fetching articles:', error);

});

}, []);

return (

<div>

<h1>Articles</h1>

{articles.map(article => (

<div key={article.id}>

<h2>{article.attributes.title}</h2>

<p>{article.attributes.body.value}</p>

</div>

))}
```

CMS MINDS

```
);

};

export default Articles;
```

Handling Responses and Errors

Ensure to handle errors appropriately to improve user experience.

Step 5: Displaying Drupal Content in React Components

Creating Components for Different Content Types

Create React components for each content type. For example, an Article component to display individual articles.

Mapping Drupal Data to React State

Use React's useState and useEffect hooks to manage and display data fetched from Drupal.

Rendering Data in Components

Render the fetched data within your components. Iterate over arrays to display lists of content.

Step 6: Implementing Routing in React

Introduction to React Router

React Router is a standard library for routing in React. It enables navigation among views of various components.

Setting Up Routes for Different Content Types and Items

In your src/App.js:

CMS MINDS

```
import { BrowserRouter as Router, Route, Switch } from 'react-router-dom';

import Articles from './Articles';

import Article from './Article';

const App = () => (

<Router>

<Switch>

<Route path="/" exact component={Articles} />

<Route path="/article/:id" component={Article} />

</Switch>

</Router>

);

export default App;
```

Example Routes and Components

Define routes for your components, passing necessary parameters through URL.

Step 7: Handling Authentication and Permissions

Overview of Authentication Methods

Drupal supports various authentication methods, including Basic Auth and OAuth.

Setting Up Authentication in Drupal

CMS MINDS

Handling Authenticated Requests in React

Include authentication tokens in your API requests in React.

```
api.get('/node/article', {
headers: {
'Authorization': 'Bearer your-auth-token'
}
})
```

Example of Protected Routes and Conditional Rendering

Implement protected routes in React, rendering components conditionally based on authentication state.

Step 8: Adding Interactivity and Enhancing User Experience

Adding Forms and Handling Form Submissions

Create forms in React for creating and updating content. Use controlled components to manage form state.

Creating New Content

Example form submission to create new content:

```
const createArticle = (title, body) => {
api.post('/node/article', {
```

CMS MINDS

```
type: 'node--article',

attributes: {

title,

body: {

value: body,

},

},

},

});

};
```

Implementing Search Functionality

Add search functionality to filter content based on user input.

Adding Loading States and Error Handling

Enhance user experience by adding loading indicators and error messages.

Conclusion

In this guide, we explored **how to connect headless Drupal with React**. By setting up Drupal as a headless CMS, exposing content via JSON, and fetching and displaying this content in a React application, you can create powerful and dynamic web applications. The decoupled architecture not only offers flexibility but also allows you to leverage the best tools for each layer of your stack.

CMS MINDS

Author's Bio

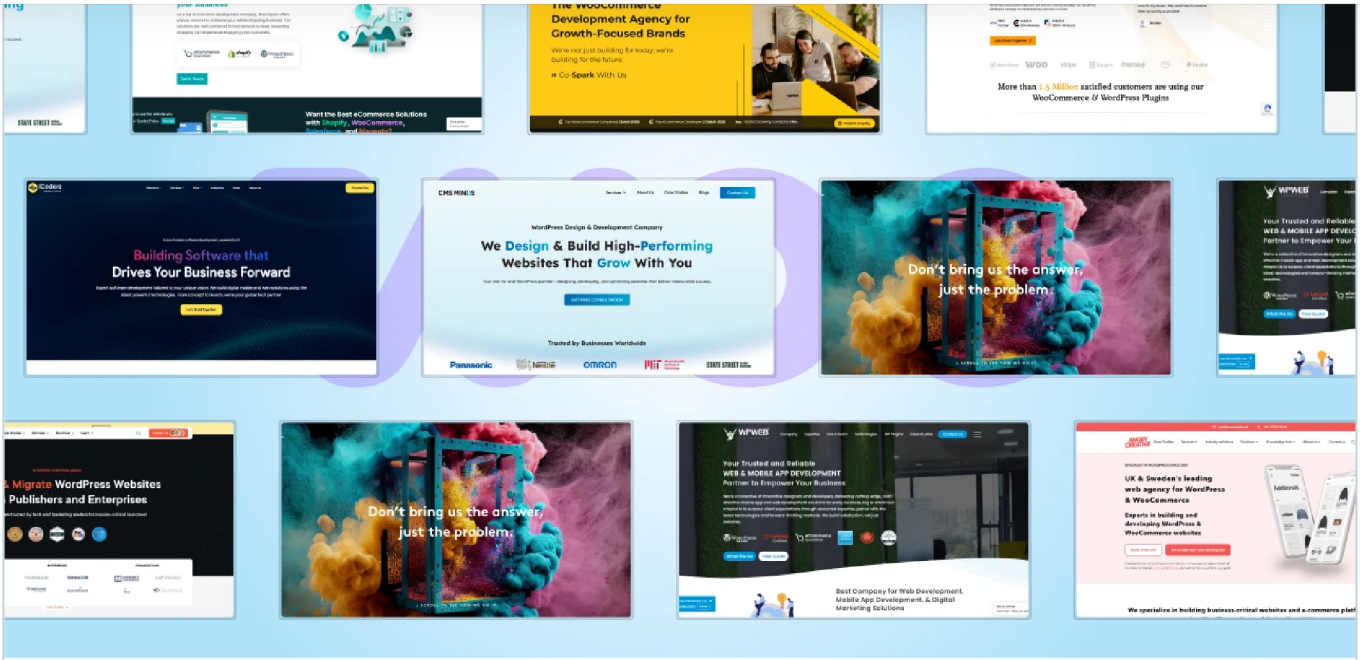
Jayesh Makwana

Jayesh Makwana writes with one goal in mind: to make Drupal easy for everyone. From his first steps with Drupal 6 to mastering the latest updates, he's a true Drupal geek, passionate about sharing his knowledge. Whether it's tips on Drupal migration, upgrading your site, or catching up on the newest features, Jayesh's articles are your friendly guide. He simplifies complex topics, making them accessible to Drupal users of all levels. Follow Jayesh for insights that enlighten and inspire, all delivered with the enthusiasm of someone who loves what they do.

Share This Article:

Recent Blogs

CMS MINDS



Aug 22, 2025

Top WooCommerce Development Companies to Partner With in 2025

Learn to integrate Headless Drupal with React for dynamic web...

CMS MINDS

Top 15 WordPress Website Examples You Need to See

Explore the best WordPress website examples for blogs, businesses, and...

Aug 19, 2025

What is Elementor and How Does It Work with WordPress?

Elementor is a powerful WordPress page builder that lets you...

VIEW ALL BLOGS

CMS MINDS

8404
Six Fork
Road,
Suite
204B
Raleigh,
NC
27615

Reviews PRO Partner

CMS MINDS