bounteous

AEM integration for Vue Storefront

Installation Guide

v1.0.0

bounteous

Introduction

Vue Storefront is a frontend platform for headless ecommerce that has integrations with a lot of common commerce platforms and headless content management systems. One that has been missing until now is being able to retrieve headless content from Adobe Experience Manager (AEM).

The AEM extension for Vue Storefront is provided as an NPM package so installation is a breeze. Once you have it installed, you simply connect it to your AEM instance by configuring the AEM GraphQL endpoint and you're ready to go.

The Vue Storefront connector for AEM allows you to query AEM Content Fragments via the GraphQL API and render their content in your store. This gives marketers control over the homepage and key landing pages so that they can be authored in AEM. Create your own Vue.js components or leverage the great number of components already available in the Storefront UI library - simply create a Content Fragment model that matches the data required by the Vue components and you're good to go. Additionally, if you have the Commerce Integration Framework (CIF) Add-On installed in AEM, you can drive commerce components like a product carousel or teaser by authoring Content Fragments.

Pre-requisites

The following instructions assume you're already a user of Vue Storefront. You should be familiar with Vue.js and it helps if you have some familiarity with Nuxt.js which Vue Storefront is based upon.

For a fully working end-to-end example of using the AEM extension with Vue Storefront and Magento, please refer to our example git repository.

Note that this integration does not necessarily require any changes in AEM. The extension is merely calling the AEM GraphQL API to retrieve Content Fragment data. That said, if you don't already have Content Fragment models defined that match what's needed by the Storefront UI components, you'll need to create them.

Install Extension

In your Vue Storefront directory, install the module into your app:

```
npm install @bounteous/vue-storefront-aem --save
```

Or

```
yarn add @bounteous/vue-storefront-aem
```

Please refer to the **README** in the npm package for the most up to date information.

Configure Extension

1. Register the module in the `nuxt.config.js` file:

```
export default {
 //...
 modules: [
   '@bounteous/vue-storefront-aem/nuxt',
 ]
 //...
}
```

2. Additionally, in the `nuxt.config.js`, you need to register the AEM module as a rawSource under the @vue-storefront/nuxt module:

```
export default {
 //...
 buildModules: [
   // ...
```

```
  ['@vue-storefront/nuxt', {
    useRawSource: {
      dev: [
        // ...
        '@bounteous/vue-storefront-aem'
        // ...
        ],
      prod: [
        // ...
        '@bounteous/vue-storefront-aem'
        // ...
        ],
    },
  }],
  // ...
]
//...
}
```

3. Next, in `middleware.config.js`, configure the connection to your AEM instance:

```
module.exports = {
 integrations: {
   // ...
   aem: {
     location: '@bounteous/vue-storefront-aem/server',
     configuration: {
       serviceURL: 'http://localhost:4503',
       endpoint: '/content/_cq_graphql/global/endpoint.json',
       // Provide credentials if necessary (for example on Author)
       // See https://github.com/adobe/aem-headless-client-nodejs for more info.
       //auth: [process.env.AEM_USER, process.env.AEM_PW]
     },
   },
 }
}
```

4. With the extension installed, the rest is up to you. Create your own Vue.js components or leverage the great number of components already available in the Storefront UI library - simply create a Content Fragment model that matches the data required by the Vue components and you're good to go.

Refer to our example repository for more details.