Using Cloudinary image service for media optimization

Cloudinary is a cloud-based end-to-end media management platform that provides solutions to help site creators serve optimized media files (images and videos) to their audiences. It also provides a lot of optional transformations that can be carried out on these media assets.

In this guide you will take a look at the gatsby-source-cloudinary and gatsby-transformer-cloudinary plugins which you can use to improve the experience of handling images on Gatsby sites.

Plugins are generally used to abstract functionality in Gatsby. In this case, the `gatsby-source-cloudinary` plugin is a source plugin which helps to connect Cloudinary media storage capabilities to your site.

> Here's a demo site that uses the gatsby-source-cloudinary showcasing optimized images in a masonry grid, served from Cloudinary.

The problem with handling images on the web

Dealing with images on the web has always been a problem as unoptimized images can slow down your site. The processes put in place to create the best media experience can take a lot of time to implement.

Solutions Cloudinary provides

Cloudinary provides a couple of amazing solutions to this problem, namely:

- Remote storage and delivery of images via CDN
- Offers a wider range of transformations than gatsby-plugin-image.
- Digital Asset Management for enterprise assets

Gatsby-source-cloudinary

This plugin fetches media assets from Cloudinary that are specified in a folder. It then transforms these images into Cloudinary file nodes, which can be queried with GraphQL in a Gatsby project. `gatsby-source-cloudinary` applies f_auto

and q_auto transformation parameters which aid in automatic optimization of format and quality for media assets by over 70 percent.

Prerequisites

Before using the `gatsby-source-cloudinary` plugin you should do the following:

- Upload your images to a folder on Cloudinary. This folder can have any name of your choosing.
- Obtain your API key and API secret from your Cloudinary dashboard.
- Have the dotenv module installed for loading environment variables from a `.env` file.

Using gatsby-source-cloudinary

Add the plugin to your project.

1. Install `gatsby-source-cloudinary`

```
npm install gatsby-source-cloudinary
```

2. In the root of your project, create an environment file called `.env` and add your Cloudinary credentials and their values

```
CLOUDINARY_API_KEY=xxxxxxxxxxxxx
CLOUDINARY_API_SECRET=xxxxxxxxxxxxxxxxxxx
CLOUDINARY_CLOUD_NAME=xxxxx
```

3. Configure `gatsby-config.js`

```
require('dotenv').config(); // highlight-line
module.exports = {
  ...
  plugins:[
  ...
  {
      resolve: `gatsby-source-cloudinary`,
      options: {
        cloudName: process.env.CLOUDINARY_CLOUD_NAME,
        apiKey: process.env.CLOUDINARY_API_KEY,
        apiSecret: process.env.CLOUDINARY_API_SECRET,
        resourceType: `image`,
        prefix: `gatsby-source-cloudinary/`
      }
    }
  ]
}
```

Note that `gatsby-source-cloudinary` takes the following options:

- **`cloudName`**, **`apiKey`**, and **`apiSecret :`** These are credentials from your Cloudinary console, stored as three separate environment variables for security.
- **`resourceType :`** This is the resource type of the media assets - either an image or a video.
- **`prefix :`** This is the folder (in your Cloudinary account) in which the files reside. In the example above, the folder is called `gatsby-source-cloudinary`. Assign a name of your choice. Other optional options are `type`, `tags`, and `maxResult`.

Here's a link to the README for more information.

Gatsby-transformer-cloudinary

After sourcing media files from Cloudinary, you will be able to leverage Cloudinary's media transformation capabilities. To do so, use `gatsby-transformer-cloudinary` which is a type of transformer plugin that is used to change image formats, styles and dimensions. It also optimizes images for minimal file size alongside high visual quality for an improved user experience and minimal bandwidth.

Here's a demo site that uses the gatsby-transformer-plugin

Prerequisites

Before using the `gatsby-transformer-cloudinary` plugin you should do the following:

- Upload your images to a folder on Cloudinary. This folder can have any name of your choosing.
- Have the `gatsby-source-cloudinary` plugin installed and configured.
- Obtain your API key and API secret from your Cloudinary dashboard.
- Have the dotenv module installed for loading environment variables from a `.env` file.

Using gatsby-transformer-cloudinary

1. Install `gatsby-transformer-cloudinary` and `gatsby-source-filesystem` which creates the File nodes that the Cloudinary transformer plugin works on.

```
npm install gatsby-transformer-cloudinary gatsby-source-filesystem
```

2. In the root of your project, create an environment file called `.env` to which to add your Cloudinary credentials and their values.

```
CLOUDINARY_API_KEY=xxxxxxxxxxxxx
CLOUDINARY_API_SECRET=xxxxxxxxxxxxxxxxxxx
CLOUDINARY_CLOUD_NAME=xxxxx
```

3. Configure `gatsby-config.js`

```
require('dotenv').config({
  path: `.env.${process.env.NODE_ENV}`,
});

module.exports = {
  plugins: [
    {
      resolve: `gatsby-source-filesystem`,
      options: {
        name: `images`,
        path: `${__dirname}/src/images`,
      },
    },
    {
      resolve: 'gatsby-transformer-cloudinary',
      options: {
        cloudName: process.env.CLOUDINARY_CLOUD_NAME,
        apiKey: process.env.CLOUDINARY_API_KEY,
        apiSecret: process.env.CLOUDINARY_API_SECRET,

        // This folder will be created if it doesn't exist.
        uploadFolder: 'gatsby-cloudinary',

    },
  ],
};
```

In `gatsby-config.js`, responsive breakpoints can be created for each image, use the `fluidMaxWidth` and `fluidMinWidth` options to set them. Take a look at the plugin documentation for more information on how these parameters can be set.

Additional resources

- Faster Sites with Optimized Media Assets by William Imoh
- Gatsby Transformer Cloudinary
- Gatsby Source Cloudinary
- Aspect ratio parameter