Building an E-commerce site with Shopify

In this tutorial, you will set up a new Gatsby website that fetches product data from Shopify. The site displays a list of all products on a product listing page, and a page for every product in the store.

If you're already comfortable with Gatsby and Shopify, you might want to check out our Shopify Starter Demo, a proof of concept showcasing 10,000 products and 30,000 SKUs (variants). You can clone the starter, host it on Gatsby and connect it to your own Shopify data to develop your own proof of concept in as little as an hour.

Setting up your Shopify account

1. Create a new Shopify account and store if you don't have one.
2. Create a private app in your store by navigating to `Apps`, then `Manage private apps`.
3. Create a new private app, with any "Private app name" and add the following under the `Active Permissions for this App` section:
   - `Read access` for `Products`
   - `Read access` for `Product listings` if you want to use Shopify's Product Collections in your Gatsby site
   - `Read access` for `Orders` if you want to use order information in your Gatsby site
   - `Read access` for `Inventory` and `Locations` if you want to use location information in your Gatsby site
4. Enable the Shopify Storefront API by checking the box that says "Allow this app to access your storefront data using Storefront API". Make sure to also grant access to `Read product tags`, `Read and modify checkouts`, and `Read customer tags` by checking their corresponding boxes.
5. Copy the password, you'll need it to configure your plugin below.

Set up the Gatsby Shopify plugin

1. If you do not already have one ready, create a Gatsby site.

2. Install the `gatsby-source-shopify` plugin.

```
npm install gatsby-source-shopify
```

3. Enable and configure the plugin in your `gatsby-config.js` file, replacing [app-password] with the password you copied above and [yourstore.myshopify.com] with the canonical address of your store.

```
plugins: [
    {
      resolve: `gatsby-source-shopify`,
      options: {
        password: [app-password],
        storeUrl: [yourstore.myshopify.com],
      },
    },
]
```

Note: You will likely not want to put your `password` and `storeUrl` directly in your `gatsby-config.js` file but rather, use an environment variable. Check out the Gatsby Shopify starter for an example of how to do that.

4. Run `gatsby develop` and make sure the site compiles successfully.

Querying Shopify data and listing products

Open the Gatsby GraphiQL interface by visiting `http://localhost:8000/___graphql`. With at least one example product added into Shopify you should see several new types of nodes in the Explorer tab, like `allShopifyProduct`. To query all products in your store sorted by title, try running the query:

```
{
  allShopifyProduct(sort: { fields: [title] }) {
    edges {
      node {
        title
        shopifyId
        description
        priceRangeV2 {
          maxVariantPrice {
            amount
          }
          minVariantPrice {
            amount
          }
        }
        status
      }
    }
  }
}
```

To add a simple page listing all products, add a new file at `/src/pages/products.js`.

```
import React from "react"
import { Link, graphql } from "gatsby"

import Layout from "../components/layout"

const ProductsPage = ({ data }) => (
  <Layout>
    <h1>Products</h1>
    <ul>
      {data.allShopifyProduct.edges.map(({ node }) => (
        <li key={node.shopifyId}>
          <h3>
            <Link to={`/products/${node.handle}`}>{node.title}</Link>
            {" - "}${node.priceRangeV2.minVariantPrice.amount}
          </h3>
          <p>{node.description}</p>
        </li>
      ))}
    </ul>
  </Layout>
)

export default ProductsPage

export const query = graphql`
  {
    allShopifyProduct(sort: { fields: [title] }) {
      edges {
        node {
          title
          shopifyId
          description
          handle
          priceRangeV2 {
            minVariantPrice {
              amount
            }
          }
        }
      }
    }
  }
`
```

Generating a page for each product

You can programmatically create pages in Gatsby for every product in your Shopify store.

Create a template for your product pages by adding a new file, `/src/templates/product.js`.

```js
import React from "react"

import Layout from "../components/layout"

const ProductTemplate = ({ pageContext }) => {
  const { product } = pageContext
  return (
    <Layout>
      <h1>{product.title}</h1>
      <div>{product.description}</div>
    </Layout>
  )
}

export default ProductTemplate
```

Edit your `gatsby-node.js` file and add the following code. This queries all Shopify products with GraphQL, then creates a page using the template in `product.js`. Each page gets assigned a URL in the format of `/products/[product handle]`.

```js
const path = require(`path`)
exports.createPages = async ({ graphql, actions }) => {
  const { createPage } = actions
  // Query for all products in Shopify
  const result = await graphql(`
    query {
      allShopifyProduct(sort: { fields: [title] }) {
        edges {
          node {
            title
            images {
              originalSrc
            }
            shopifyId
            handle
            description
            priceRangeV2 {
              maxVariantPrice {
                amount
              }
```

```
                minVariantPrice {
                  amount
                }
              }
              status
            }
          }
        }
      }
  `)
  // Iterate over all products and create a new page using a template
  // The product "handle" is generated automatically by Shopify
  result.data.allShopifyProduct.edges.forEach(({ node }) => {
    createPage({
      path: `/products/${node.handle}`,
      component: path.resolve(`./src/templates/product.js`),
      context: {
        product: node,
      },
    })
  })
}
```

Additional Resources

- Gatsby Shopify Starter