React Storefront

The React Storefront headless connector for the Kibo Composable Commerce Platform (KCCP) enables you to integrate your storefront site with the following pages:

- Product listing pages (PLP)
- Product detail pages (PDP)
- Cart

This guide will show you how to get up and running with the connector.

Prerequisites

You will need a KCCP back end to try out the connector. For instructions on how to obtain access to a dev account and configure a sandbox, see Getting Started.

Create an Application

Create a new React Storefront app using version 8.14.0 or later:

```
npm create react-storefront my-kibo-app
```

Next, `cd` into your created application and install the Kibo connector:

```
cd my-kibo-app
 npm install @kibocommerce/react-storefront-kibo-connector
```

Configure the Environment

Configure the `KIBO_CONFIG_HOST` environment variable in the `.env` file to point to your KCCP back end. See the `.env.sample` file as an example of adding the `env` variable via dotenv. You can also reference this guide for more info about Node.js environment variables.

For example, your `.env` file may look like:

```
KIBO_ACCESS_TOKEN_URL=https://t00000.sandbox.mozu.com/api/platform/applications/authtickets/oauth
KIBO_CLIENT_ID: KIBO_APP.1.0.0.Release
KIBO_SHARED_SECRET: 12345_Secret
KIBO_API_HOST: https://kibo-site.com
HOME_DOCUMENT_TYPE=cms_content_type
PRODUCT_DOCUMENT_TYPE=cms_content_type
```

These parameters are:

- `KIBO_API_HOST` : Link to your GraphQL API instance.
- `KIBO_ACCESS_TOKEN_URL` : Link to the Authentication Server, used to request an access

token from Kibo's OAuth 2.0 service.

- `KIBO_CLIENT_ID`: Unique Application (Client) ID of your application, found in your Developer Console.
- `KIBO_SHARED_SECRET`: Secret API key used to authenticate your application, found in your Developer Console.
- `HOME_DOCUMENT_TYPE`: Custom developer defined content type name for Kibo CMS HTML blocks.
- `PRODUCT_DOCUMENT_TYPE`: Custom developer defined content type name for Kibo CMS HTML blocks.

Based on the config, this integration will handle authenticating your application against the API using your Client ID and Shared Secret. These can be found in your Dev Center.

Set the Connector

Finally, set the connector in your `next.config.js` file. By default this file is set to use the `react-storefront/mock-connector` as shown below:

```
module.exports = withReactStorefront({
  // ... Some code

  connector: 'react-storefront/mock-connector',
  // ... More code
```

Change this line to use the `@kibocommerce/react-storefront-kibo-connector`:

```
module.exports = withReactStorefront({
  // ... Some code

  connector: '@kibocommerce/react-storefront-kibo-connector',
  // ... More code
```

Now you can run your project locally and visit http://127.0.0.1:3000 in your browser to view it.

```
npm start
```

Deploying to Layer0

The front-end React Storefront can be hosted anywhere that supports Node and Express, but it works great on Layer0. You can try Layer0 for free by signing up here. Once you have an account you can deploy it by running:

```
Layer0 deploy
```

Refer to the Layer0 deployment guide for more information.

Development

To begin development:

1. In the first terminal window (this repo), run `yalc publish` and `npm run watch`.
2. In the second terminal window, open RSF starter app (commercial branch).
3. Go to `next.config.js` and change connector field value to `react-storefront-kibo-connector`.
4. Run `yalc add react-storefront-kibo-connector`.
5. Run `npm i`.
6. Run `npm run start`.