React App Setup:

Clone this repository, then run the command:

- npm install

Once it's installed, run command:

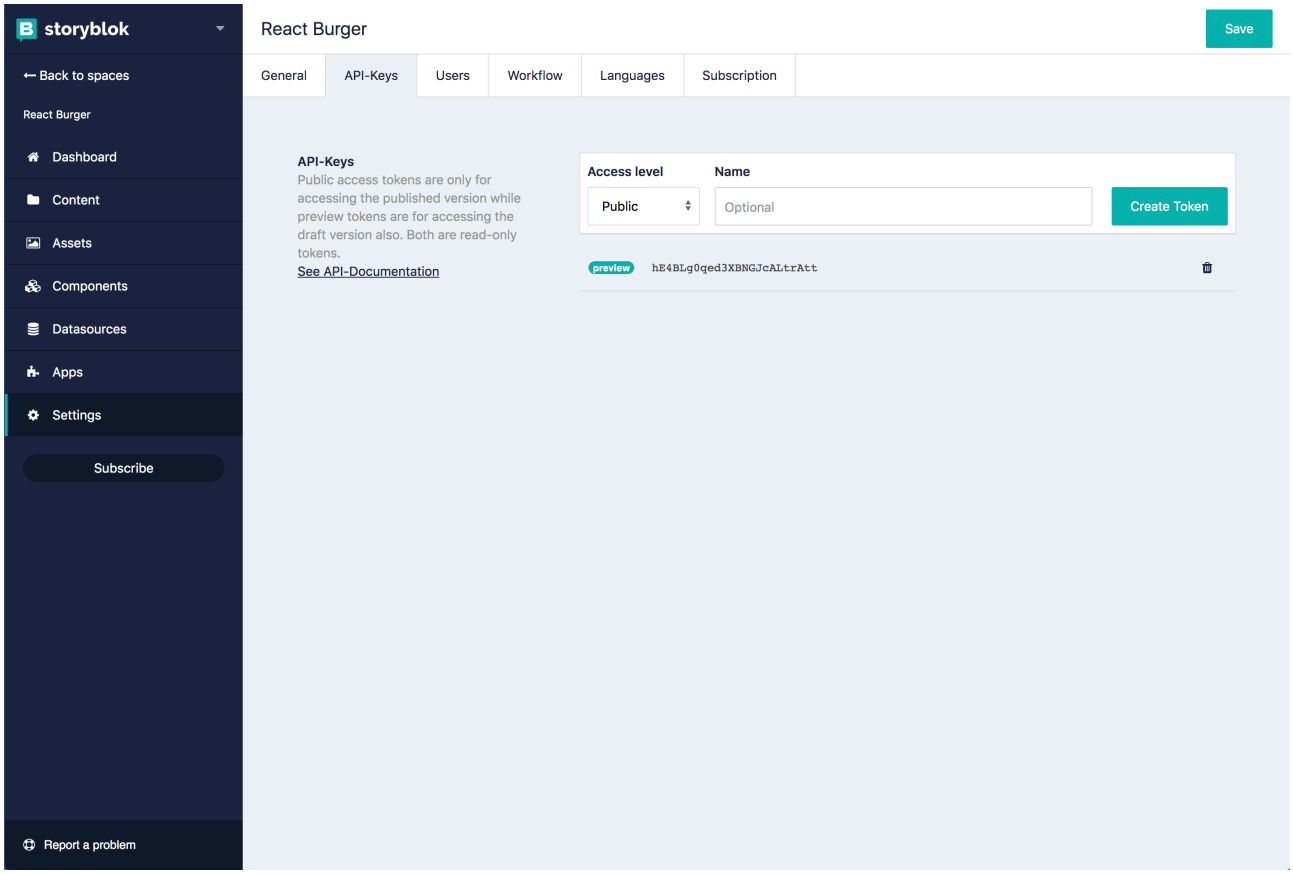
- npm start

Your app is up and running!

1. Add the API Key:

Find your API key in Storyblok:

> settings > API-Keys



In order to connect your react app to *your* Storyblok CMS you will need to add an *your* API Key.

In the *.env* file located in the root of the repo you just cloned, add your API key to the "REACT_APP_STORKYBLOK_API_KEY" variable.

You could just copy the API key into your code, but it would be publicly available. So best practice is to add it your *.env* file to keep it secret when pushing your code to Github.

You would then add the *.env* file to the *.gitignore* file.

2. Checkout the Index.js File to see how we're pulling our data in:

There are 2 main ways of pulling data into an app, either you can query api endpoints the traditional way, then filter through the data returned in you front-end code, or you can just query for the exact data you need from the front-end, so you don't have to do any filtering in your code…

We're going to do it the second way, it's a lot nicer to use, and makes querying data a lot faster and removes a lot of headache.

In order to do this we're going to be using **Apollo**, which is a GraphQL Library for React, I've already set this up in the repo.

Follow this link to see what types of queries you can do with Storyblok + GraphQL:

https://www.storyblok.com/docs/graphql-api

What we are doing in the index.js file is setting-up the Apollo Client and wrapping our app in the <ApolloProvider /> Component, this way we will be able to query any data from our CMS, anywhere in our app. BRILLIANT!

You will also notice, on line 15, this:

```
token: process.env.REACT_APP_STORKYBLOK_API_KEY,
```

This is the code pulling in your API key from the *.env* file.

3. Go to App.js:

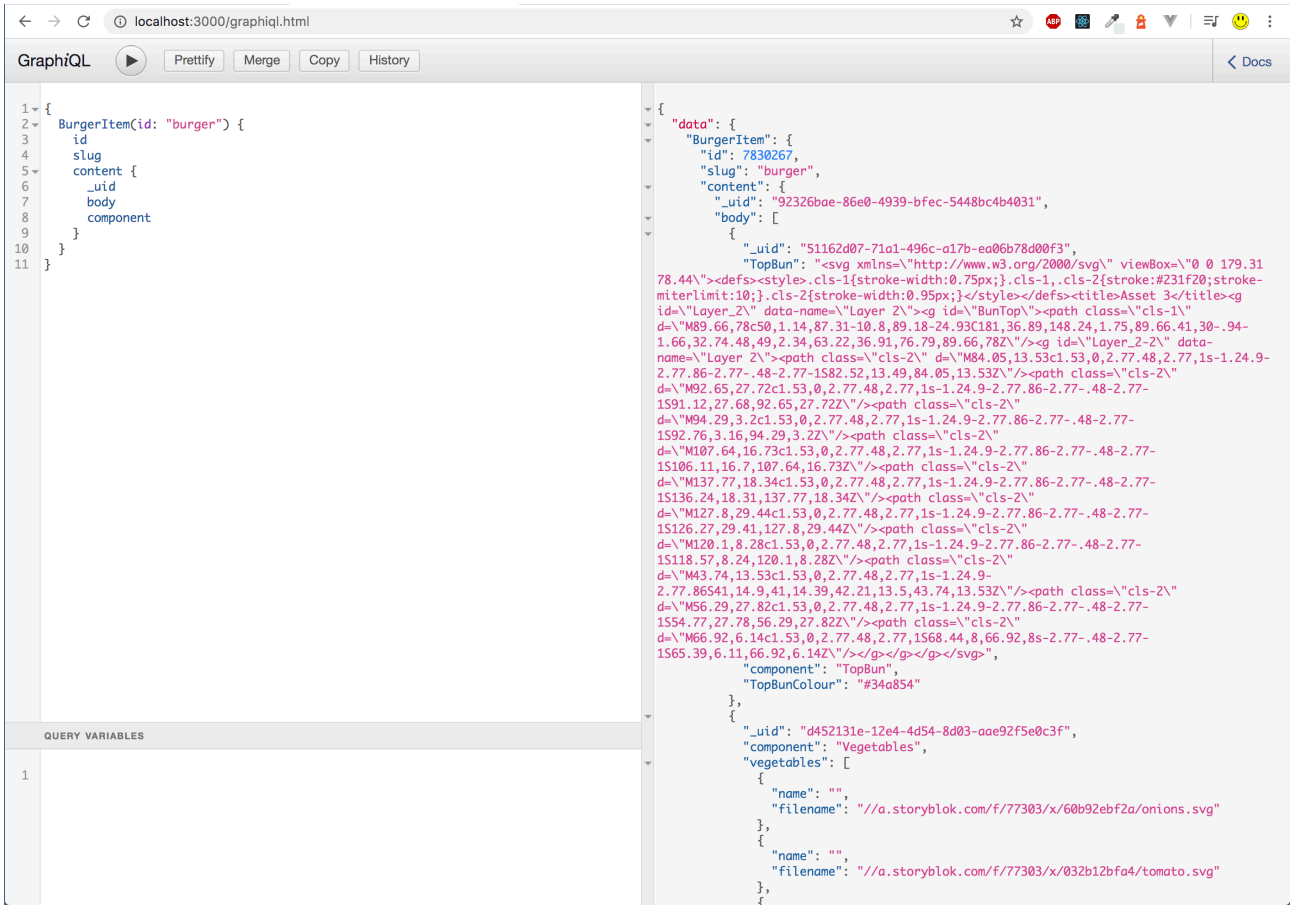
This is where we're going to be querying the API.

on line 9 we have our graphql query, which is querying our burger.

It's always nice to be able to visualise the data returned by APIs so you can quickly see which keys you want to reference in your code to make data appear.

GraphQL has this wonderful tool called GraphiQL which enables you to write queries and see their output. I've included this too.

All you need to do is add "/graphiql.html" to the end of the url in your browser, like so:



You can mess about with it by adding queries, in the left column. If you're not sure what to query, press "shift+ spacebar" to bring up a list of all the available keys to query! MAGIC!

Now let's go back to the *App.js* file.

All the data we will be needing will be present in the datavariable.

And more precisely in the data variable on line 24, which is where we store the data returned by the GraphQL query we made.

This data is being passed into **Components,** a component I created.

What is happening?

App.js is calling the **Components** function which is being exported from the ./components/ index.js file.
We are also passing "data BurgerItem content" into the **Components** function.