Polyfill Errors

With many web3 tools you may run into Polyfill issues with error messages like 'crypto' not found, or 'buffer' not found. These are caused by Node JS Polyfills that are no longer included with Webpack after version 5. You can read more about the full details of thishere .

To save you some time we have compiled a list of configurations and corresponding packages you will need to install in order to overcome some of these Polyfill errors for different Frontend Frameworks.

Next JS

Next JS is an easier framework to work with as it does not require any additional packages to be installed. You will however need to edit yournext.config.js file to the following:

```
/** @type

{ import ( 'next' ) . NextConfig } */ const nextConfig =

{ reactStrictMode :

true , webpack :

( config ,

{ isServer } )

=>

{ if

( ! isServer )

{ config . resolve . fallback

=

{ fs :

false , net :

false , tls :

false , } ; } return config ; } , } ;

module . exports

= nextConfig ;
```

React with Vite

There are several ways to build out a React application and one of them includes using Vite.

Below are the packages you will need to install before configuring your frontend:

yarn add @vitejs/plugin-react @esbuild-plugins/node-globals-polyfill process stream-browserify util rollup-plugin-polyfill-node or

npm install yarn add @vitejs/plugin-react @esbuild-plugins/node-globals-polyfill process stream-browserify util rollup-plugin-polyfill-node Below is the configuration needed in yourvite.config.ts :

```
import react from

"@vitejs/plugin-react" ; import

{ defineConfig }

from

"vite" ;

export
```

```
default

defineConfig ( { plugins :

[ react ( ) ] , optimizeDeps :

{ esbuildOptions :

{ define :

{ global :

"globalThis" , } , } , } , resolve :

{ alias :

{ process :

"process/browser" , stream :

"stream-browserify" , util :

"util" , } , } , } ) ;
```

Vue JS

The following are packages you will need to install before configuring your frontend with Vue:

yarn add @esbuild-plugins/node-globals-polyfill stream-browserify util rollup-plugin-polyfill-node or

npm install @esbuild-plugins/node-globals-polyfill stream-browserify util rollup-plugin-polyfill-node Below is the configuration needed in yourvite.config.ts

```
import vue from

"@vitejs/plugin-vue" ; import vueJsx from

"@vitejs/plugin-vue-jsx" ; import

{ defineConfig }

from

"vite" ;

export

default

defineConfig ( { plugins :

[ vue ( ) ,

vueJsx ( ) ] , optimizeDeps :

{ esbuildOptions :

{ define :

{ global :

"globalThis" , } , } , } , resolve :

{ alias :

{ stream :

"stream-browserify" , util :

"util" , } , } , } ) ;
```

Create React App

While Create React App is no longer a recommended way of building out a React Project according to the official React documentation many projects may still be using it as a legacy dependency. In order to configure you will need to eject from Create React App and useReact App Rewired .

Below are packages that will be helpful in getting this set up:

yarn add assert browserify-zlib c-kzg crypto-browserify https-browserify net os-browserify path-browserify react-app-rewired stream-browserify stream-http tls url or

npm install assert browserify-zlib c-kzg crypto-browserify https-browserify net os-browserify path-browserify react-app-rewired stream-browserify stream-http tls url Your config-overrides.js will need to look like this:

const webpack =

require ("webpack") ;

module . exports

=

function

override (config)

{ const fallback = config . resolve . fallback

||

{ } ; Object . assign (fallback ,

{ fs :

false , crypto : require . resolve ("crypto-browserify") , stream : require . resolve ("stream-browserify") , assert : require . resolve ("assert") , http : require . resolve ("stream-http") , os : require . resolve ("os-browserify") , https : require . resolve ("https-browserify") , url : require . resolve ("url") , zlib : require . resolve ("browserify-zlib") , path : require . resolve ("path-browserify") , "c-kzg" : require . resolve ("c-kzg") , "process/browser" : require . resolve ("process/browser") , }) ; config . resolve . fallback

= fallback ; config . plugins

=

(config . plugins

||

[ ]) . concat ([new

webpack . ProvidePlugin ({ process :

"process/browser" , Buffer :

["buffer" ,

"Buffer"] , }) ,]) ; config . ignoreWarnings

=

[/ Failed to parse source map /] ; return config ; } Previous Common Errors Next Contract Addresses