Creating a Production Build

`npm run build` creates a `build` directory with a production build of your app. Inside the `build/static` directory will be your JavaScript and CSS files. Each filename inside of `build/static` will contain a unique hash of the file contents. This hash in the file name enables long term caching techniques.

When running a production build of freshly created Create React App application, there are a number of `.js` files (called *chunks*) that are generated and placed in the `build/static/js` directory:

`main.[hash].chunk.js`

- This is your *application* code. `App.js`, etc.

`[number].[hash].chunk.js`

- These files can either be *vendor* code, or code splitting chunks. *Vendor* code includes modules that you've imported from within `node_modules`. One of the potential advantages with splitting your *vendor* and *application* code is to enable long term caching techniques to improve application loading performance. Since *vendor* code tends to change less often than the actual *application* code, the browser will be able to cache them separately, and won't re-download them each time the app code changes.

`runtime-main.[hash].js`

- This is a small chunk of webpack runtime logic which is used to load and run your application. The contents of this will be embedded in your `build/index.html` file by default to save an additional network request. You can opt out of this by specifying `INLINE_RUNTIME_CHUNK=false` as documented in our advanced configuration, which will load this chunk instead of embedding it in your `index.html`.

If you're using code splitting to split up your application, this will create additional chunks in the `build/static` folder as well.

Static File Caching

Each file inside of the `build/static` directory will have a unique hash appended to the filename that is generated based on the contents of the file, which allows you to use aggressive caching techniques to avoid the browser re-downloading

your assets if the file contents haven't changed. If the contents of a file changes in a subsequent build, the filename hash that is generated will be different.

To deliver the best performance to your users, it's best practice to specify a `Cache-Control` header for `index.html`, as well as the files within `build/static`. This header allows you to control the length of time that the browser as well as CDNs will cache your static assets. If you aren't familiar with what `Cache-Control` does, see this article for a great introduction.

Using `Cache-Control: max-age=31536000` for your `build/static` assets, and `Cache-Control: no-cache` for everything else is a safe and effective starting point that ensures your user's browser will always check for an updated `index.html` file, and will cache all of the `build/static` files for one year. Note that you can use the one year expiration on `build/static` safely because the file contents hash is embedded into the filename.

Profiling

ReactDOM automatically supports profiling in development mode for v16.5+, but since profiling adds some small additional overhead it is opt-in for production mode. You can opt-in by using the `--profile` flag. Use `npm run build -- --profile` or `yarn build --profile` to enable profiling in the production build. See the React docs for details about profiling using the React DevTools.