爱彼迎

配置

别名配置

craco
安装：
`npm install @craco/craco@alpha -D`

1. 自己在根目录下创建一个craco.config 在里面配置别名
2. 在pack.json里面的scripts 换成craco启动 换成craco 他会把之前的reac配置和自己刚刚配置的进行合并再启动，才可以使别名配置生效

```
const path = require('path')
const resolve = pathname => path.resolve(__dirname, pathname)
module.exports = {
  webpack: {
    // 要传绝对路径 所以在上面const了一下方便下面导入
    alias: {
      '@': resolve('src'),
      'components':resolve('src/components'),
      'utils': resolve('src/utils')
    }
  }
}
```

less配置

安装：`npm i craco-less@2.1.0-alpha.0`

修改：`craco.config.js` 文件如下

```
const CracoLessPlugin = require('craco-less');

module.exports = {
```

```
  plugins: [
    {
      plugin: CracoLessPlugin,
      options: {
        lessLoaderOptions: {
          lessOptions: {
            modifyVars: { '@primary-color': '#1DA57A' },
            javascriptEnabled: true,
          },
        },
      },
    },
  ],
};
```

CSS样式重置

1. `npm install normalize.css`

下载后在index里面引用它

`import ‘normalize.css;`

2. reset.less

在src/assets/css 里自己创建

```
* {
  padding: 0;
  margin: 0;
}

//这段 CSS 代码的作用是为所有 HTML 元素（包括所有子元素）设置统一的
//padding 和 margin 样式，将它们的值全部设为 0。
```

一些其他的变量我们在当前的文件夹下创建了variables.css

```
@textColor: #484848;
@textColorSecondary:#222;
//创建了两个变量分别为对应的颜色
```

在编辑好这些 我们要在重置的les文件中引入这些预先修改的变量，所以要在对应的less上面引入他

`@import'./variables.less'`

3. 把编辑好的reset引入到主less中 方法同上

Router 路由配置

1. 安装：`npm install react-router-dom`
2. 然后在index.js中导入这个第三方包 `import { HashRouter } from 'react-router-dom'`
3. 在render里面写上标签

```
root.render(
  <React.StrictMode>
   <HashRouter>
    <App/>
   </HashRouter>
  </React.StrictMode>
);
```

针对爱彼迎的项目 Header 和 Footer是固定不变的 所以路由的切换至需要配置其余里面的内容

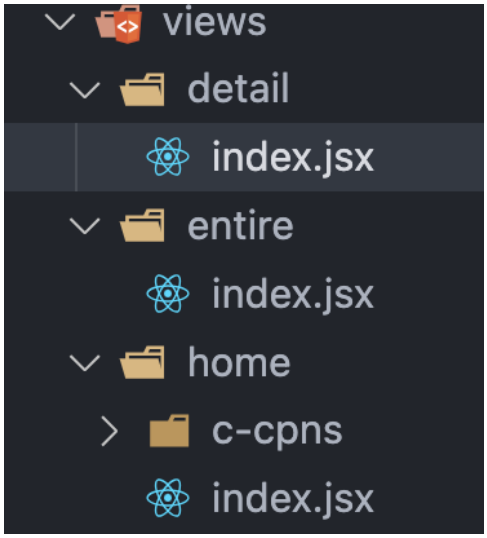
在App.js中配置大概，然后其余的配置在router文件夹下再引入

```
import React, { memo } from 'react'
import { useRoutes } from 'react-router-dom'
import routes from './router'

const App = memo(() => {
  return (
    <div className='app'>
```

```
      <div className='header'>header</div>
      <div className='page'>
        {useRoutes(routes)}
      </div>
      <div className='footer'>footer</div>
    </div>
  )
})

export default App
```

views文件夹是用来存放编辑的界面目录如下 (c-cpns = children-components)

jsx是为了后面导出这个组件



开始在router下配置

配置的过程其实就是编辑对应的映射关系

```
import React from 'react'
import { Navigate } from "react-router-dom"

//配置懒加载
const Home = React.lazy(() => import('@/views/home'))
const Entire = React.lazy(() => import('@/views/entire'))
```

```
const Detail = React.lazy(() => import('@/views/detail'))
//这些组件在实际渲染时会被懒加载。
//这意味着在路径匹配时，这些组件会在需要时才异步加载，而不是在应用启动时全部


const routes = [
  { //当默认进来的时候 我们希望他直接进入home页面
    path:'/',
    element: <Navigate to='/home' />
  },
  {
    path:'/home', //路径
    element:<Home/> ,//组件
    //这个组件可以这样写是因为配置了懒加载 懒加载的时候给他const了什么名字
    //这里就写什么
  },
  {
    path:'/entire',
    element:<Entire/>
  },
  {
    path:'/detail',
    element:<Detail/>
  }
]

export default routes
```

下一步，在src下的index.js配置 suspense 因为是异步懒加载，来确保在没有加载出来的时候显示一些内容

```
import React, { Suspense } from 'react'; //从react中导入suspense
import ReactDOM from 'react-dom/client';
import { HashRouter } from 'react-router-dom'

import App from '@/App';
```

```
import './assets/css/index.less'
import 'normalize.css'



const root = ReactDOM.createRoot(document.getElementById('root'
root.render(
  <React.StrictMode>
   <Suspense fallback='loading'> //把内容用suspense包裹 fallback中
    <HashRouter>
      <App/>
    </HashRouter>
    </Suspense>
  </React.StrictMode>
);
```

Redux的状态管理

方式：`@reduxjs/toolkit` 或 自己搭建

安装：`npm install @reduxjs/toolkit` 、 `npm install react-redux`

在src/store 下新建 index.js进行搭建，代码如下 reducer里面的内容是后面创建完代码后导入进来的

```
import { configureStore } from "@reduxjs/toolkit";
import homeReducer from './modules/home'
import entireReducer from './modules/entire'

const store = configureStore({
  reducer: {
    home: homeReducer, //rtk配置
    entire: entireReducer //手动配置
  }
})
```

```
export default store
```

2. 在store文件夹下新建一个modules用来存放具体的文件 home采用rtk的方法进行搭建，entire采用手动搭建

home的搭建

在module文件夹下新建home.js，代码初始化内容如下

```
import { createSlice } from '@reduxjs/toolkit' //帮助我们创建对应的

const homeSlice = createSlice({
  name: 'home',
  initialState: { //初始化数据，但是现在还不知道需要放什么数据进去

  },
  reducers: {

  }

})

export default homeSlice.reducer
```

在src下的index.js中引入`import { Provider } from 'react-redux'`并在下方对内容进行包裹

```
root.render(
  <React.StrictMode>
   <Suspense fallback='loading'>
    <Provider store={store}> //这个store就是上面所配置的store
      <HashRouter>
```

```
          <App/>
        </HashRouter>
      </Provider>
      </Suspense>
    </React.StrictMode>
);
```