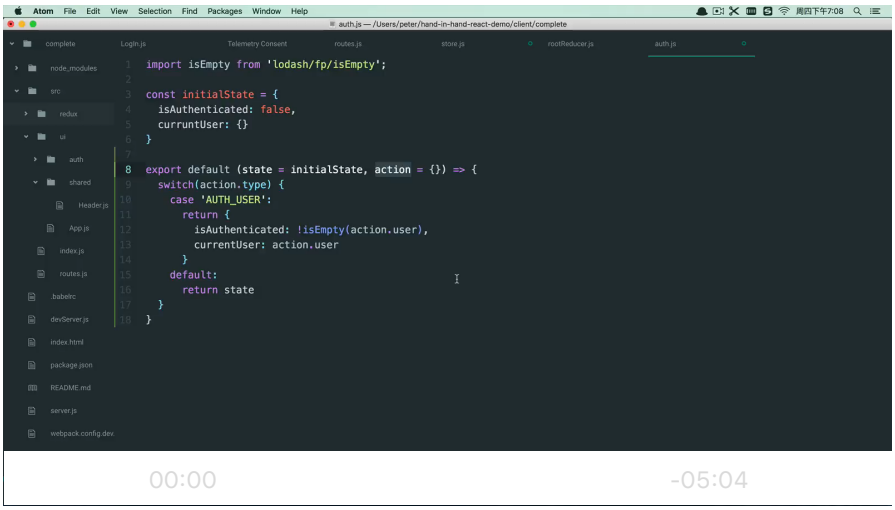

通过 React Router 添加前端页面的路由功能

第四章：前端用户登录功能实现

编写用户登录表单

借助 Redux Thunk 实现异步 Action Creator

同步的 Action Creator

在 Redux 中，一个 action 创建函数只是简单的返回一个 action，如下所示：

```
function login(data) {
  return { type: 'AUTH_USER' }
}
```

然后，我们就可以调用 Redux store 提供的 dispatch 方法，分发刚才创建的 action，代码如下：

```
store.dispatch(login(data))
```

上面一行代码执行后，会立即触发相应的 reducer，更新 store 存储的 state。整个过程是一个同步执行的操作，这个 `login` 函数是一个同步的 action creator，但是对于有网络请求的异步操作来说，同步的 action creator 就不适合了，不能直接返回一个 action，需要借助 Redux Thunk 中间件定义一个异步的 action creator。请参考文档异步 Action

异步的 Action Creator

借助 Redux Thunk，可以让一个 action creator 返回一个函数，在这个返回函数中可以执行 API 请求操作。当 action creator 返回函数时，这个函数会被 Redux Thunk 中间件执行。

新建 `src/redux/actions/authActions.js` 文件，首先导入 `axios`：

```
import axios from 'axios';
```

然后，定义一个 `login` 方法，如下：

```
export function login(data) {
  return dispatch => {
    axios.post('http://localhost:3000/api/aut
      const token = response.data.token;
      const user = response.data.user
```

```
        sessionStorage.setItem('jwtToken', toke
        sessionStorage.setItem('user', JSON.str
        dispatch(setCurrentUser(user));
        console.log('登录成功了！')
      }).catch(error => {
        handleError(error);
      });
    }
  }
```

sessionStorage

定义 setCurrentUser 方法

```
export function setCurrentUser(user) {
  return {
    type: 'AUTH_USER',
    user
  };
}
```

`setCurrentUser` 方法就是一个 action creator

定义 handleError 方法

```
function handleError(error) {
  if (error.response) {
    console.log(error.response.data.error);
  } else {
    console.log(error);
  }
}
```

参考 axios 文档处理错误信息