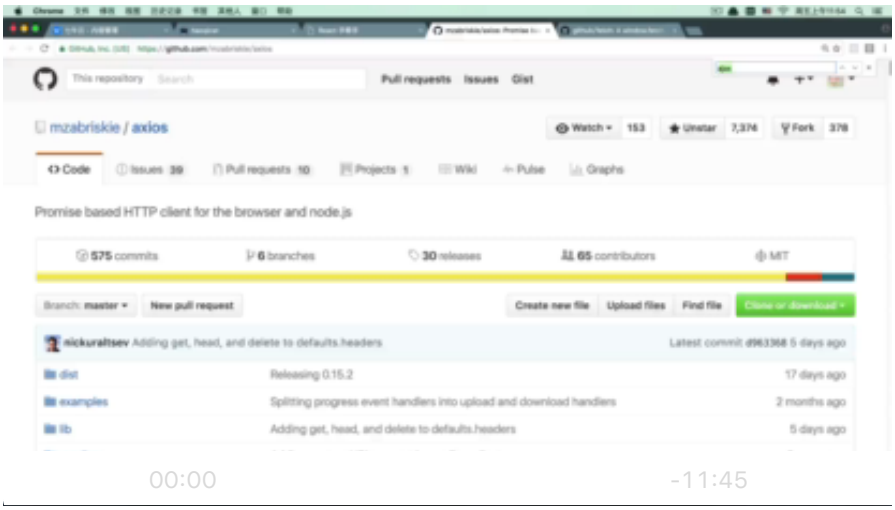

使用 axios 请求后端用户认证接口

安装 axios 包

```
npm install --save axios
```

提交表单获取输入框中的数据

修改文件 `src/ui/auth/LogIn.js`，首先导入 `axios` 模块：

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

然后，分别给两个 TextField 组件设置不同的 `ref` 属性值：

```
<TextField ref="username" ... />
<TextField ref="password" ... />
```

这样输入框中的数据，比如说用户输入的用户名，就可以通过下面的代码获得：

```
this.refs.username.getValue()
```

然后给 `form` 组件添加 `onSubmit` 事件：

```
<form onSubmit={this.handleSubmit.bind(this)}
```

最后，定义 `handleSubmit()` 事件处理方法：

```
handleSubmit(event) {
  event.preventDefault();
  let username = this.refs.username.getValue(
  let password = this.refs.password.getValue(
  axios.post('http://localhost:3000/auth/logi
  .then(response => {
    console.log(response.data.token)
  })
  .catch(error => {
    console.log(error);
  });
}
```

打开用户登录表单页面，输入用户名和密码，提交表单的时候会遇到错误，在浏览器中向服务器端发送

XMLHttpRequest 请求的时候，出现跨域问题。

跨域问题解决

进入服务器端项目目录下，安装 cors 中间件：

```
npm install --save cors
```

然后，打开 `server/index.js` 文件，导入 `cors` 中间件：

```
var cors = require('cors');
```

然后加载 `cors` 中间件：

```
app.use(cors());
```

对于跨域问题，可以参考好奇猫上的另外一门课程《React-Express 极简 API》，有比较详细的介绍。

重新提交登录表单

回到前端用户登录界面，重新提交登录表单，若用户信息在服务器端通过认证，则会在浏览器控制台中显示出从服务器端返回的认证码，说明用户已经登录成功了。

用户登录成功之后，前端界面应该体现出用户已经处于登录状态了，最起码导航栏的 `登录` 标签应该消

失，替换为 `退出` 标签才是合乎常理的。不过现在有一个问题，LogIn 组件和导航栏 Header 组件之间并没有血缘关系，LogIn 组件不能通过 `props` 机制给 Header 组件传递信息，让 Header 组件及时更改标签状态。从下节课程开始，我们将使用 Redux 管理项目中的 state，让各个组件之间能够数据同步。