测试工具及环境：

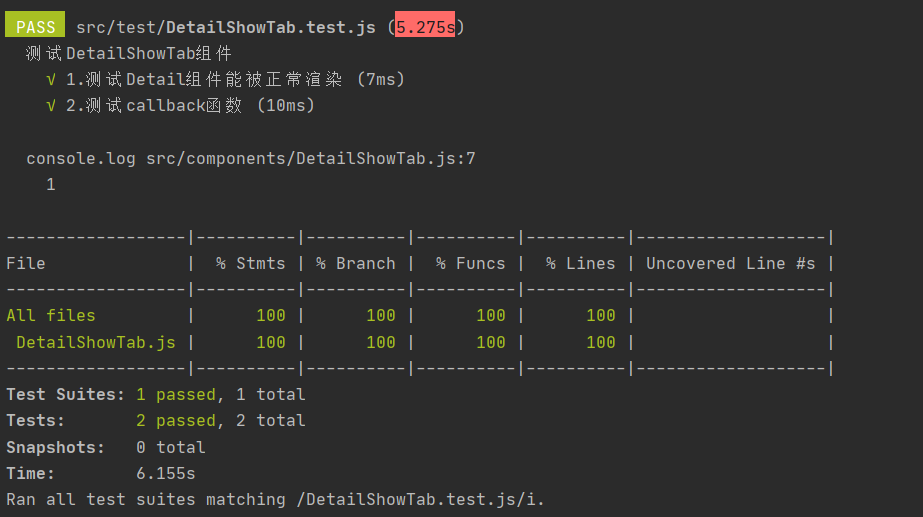
jest+babel-jest+enzyme 测试环境的安装见前的端测试环境搭建文档

测试流程：

手动写test文件，对前端主要部件进行测试，主要涉及演出浏览、详情、拍卖、推荐等部件，没有测试非核心功能如登录、注册等

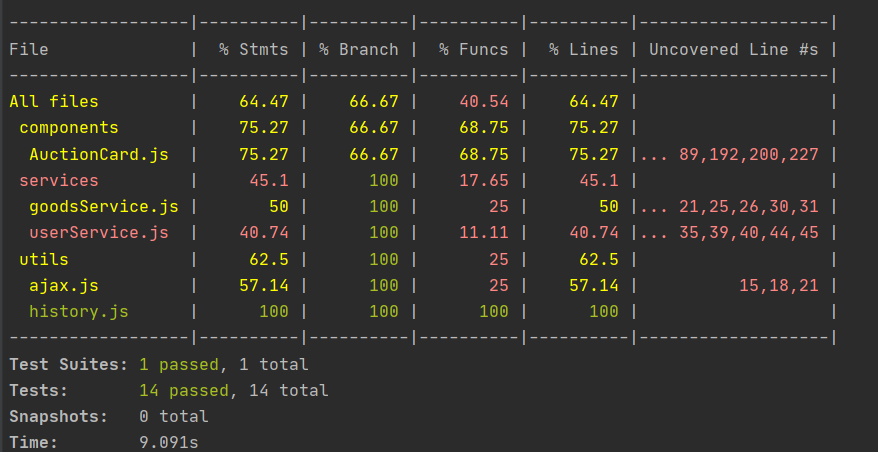
测试详细结果：

1.测试DetailShowTab

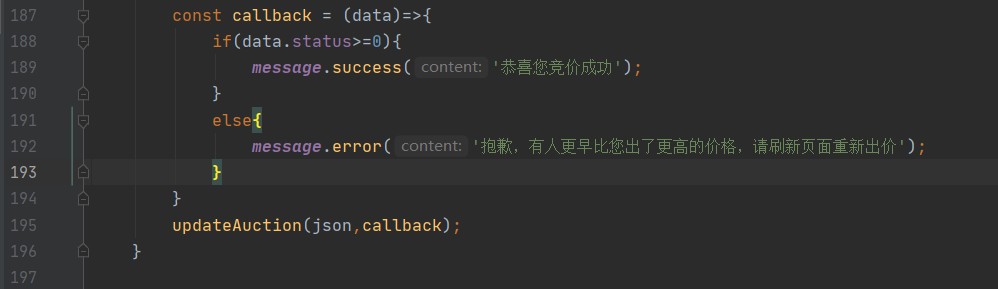


由于这个控件相对简单，而且没有为了处理特殊情况而产生的分支，所以覆盖率可以达到100%

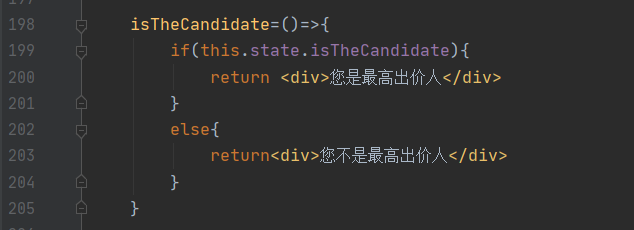
2.测试auctionCard



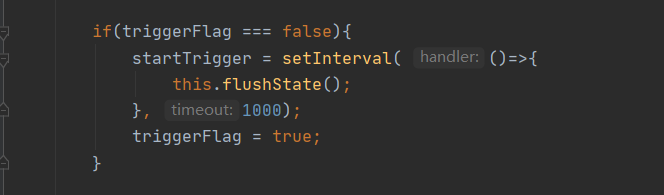
我测试了组件的渲染和基本的函数，但是可以看到覆盖率还是不高，基本都是一些防止用户骚操作或者并发情况下的特殊情况。由于单独一个前端无法测试并发情况，所以有些不能覆盖。我把没覆盖的代码罗列到下边并分析原因：



第192行没覆盖到的原因是，此段代码是防止在同一个interval中有多名用户同时拍卖，后端会给出价较低的用户发送一个message，因为我这里只有一个前端，所以覆盖不到



第200行覆盖不到的原因是单独一个前端只有一个测试用户，两种情况没法完全覆盖



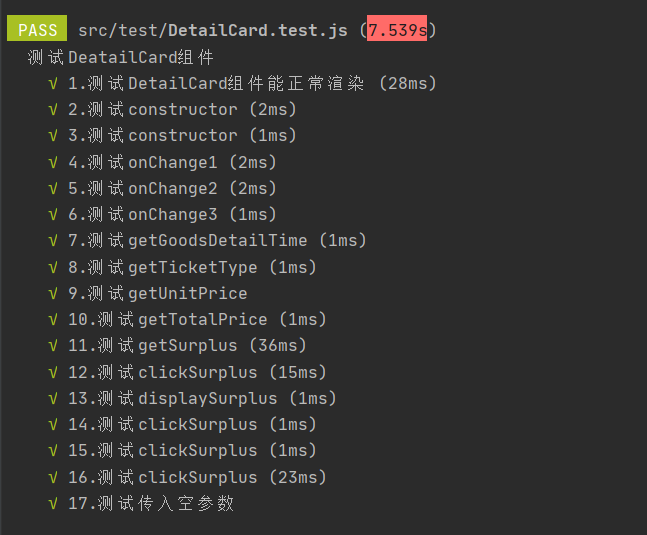
第227行没法覆盖的原因是在测试时，我只生成了一次AuctionCard，没法利用setInterval动态刷新数据。flushState的作用就是动态刷新数据，我进行了单独测试。

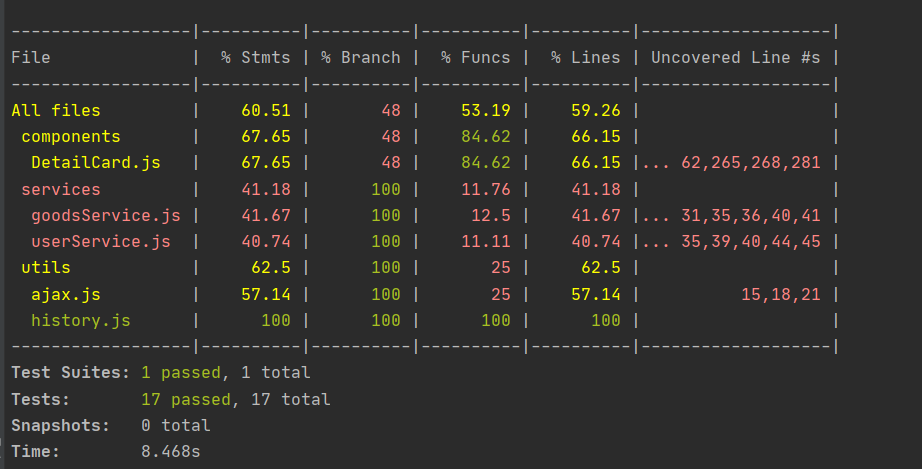
3.测试RecommedList



这个组件目前也相对简单，没有难以覆盖的分支，所以覆盖率也能到100%

4.测试DetailCard

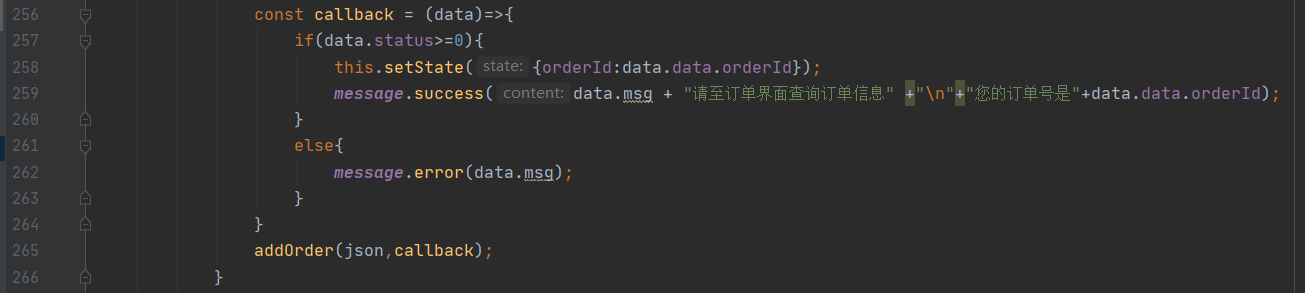




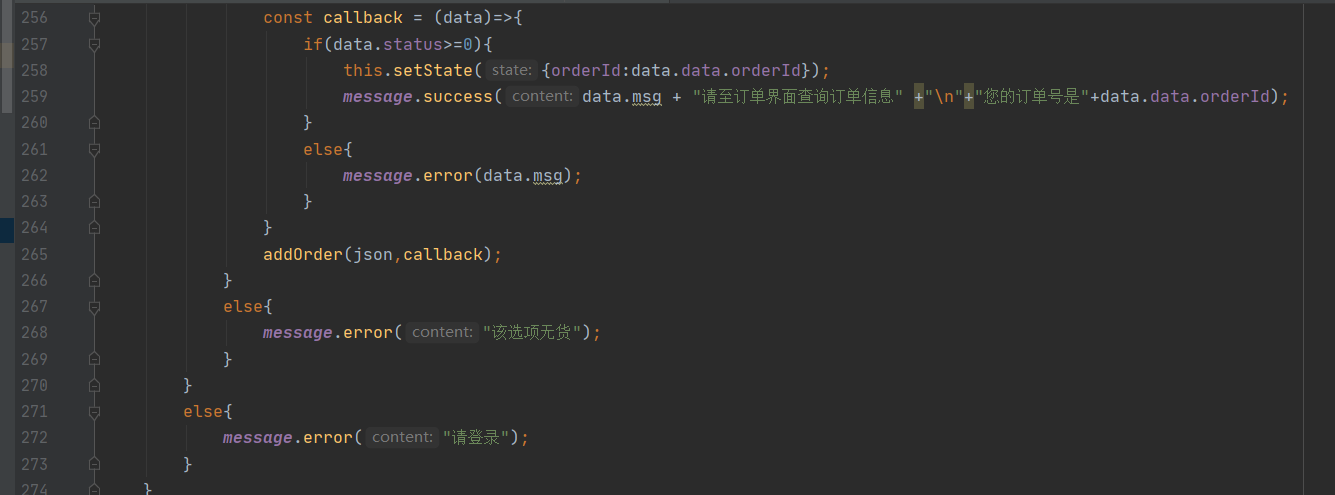
对于详情界面也进行了对渲染和函数的测试，对于有些不能覆盖到的代码，我在下面将进行分析：



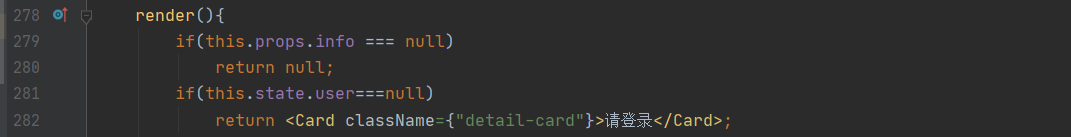
首先62行这里，测试componentDidMount竟然不能调用getDetailTime,我没太想明白是因为什么，还需要继续探索



265行因为没法与后端交互，我也没写对应的mock函数，所以没法向后端发送addOrder的请求



268行因为我没选取无货的测试数据，所以没法覆盖，但是在实际使用中可以看到已经可以处理无货的情况，不影响使用



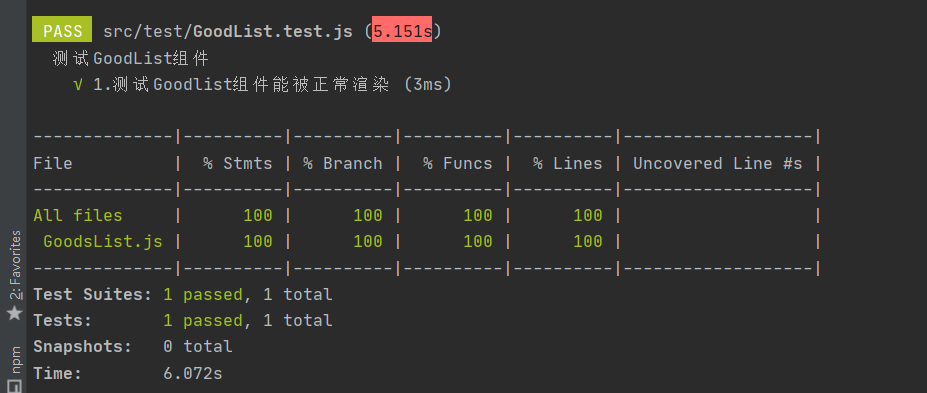
281行没测试到没登录的情况，这一点我们还在商讨如果没登陆是否可以看到详情页面，所以这一点可以先放置一下。

5.测试GoodsList



因为用的是shallow不渲染下层的goods，所以覆盖率很容易到达100%

6.测试Goods



Goods与GoodsList差不多，都是只有一个参数，没有较多的分支结构，所以覆盖率很容易达到100%。

测试总结：

本次测试主要掌握了使用jest+enzyme工具对react前端进行测试，其中涉及到的组件的参数、state等问题有很多坑，都已经写入前端踩坑指南.md文档中了，以后再遇到相关问题可以先查看自己写的经验文档。另外在测试中我发现单纯的前端测试，很难使得代码覆盖率到达100%，因为有很多分支需要后端的支持，不然前端就需要写很复杂的mock函数，而这也是我本次单元测试所欠缺的，还需要继续学习。