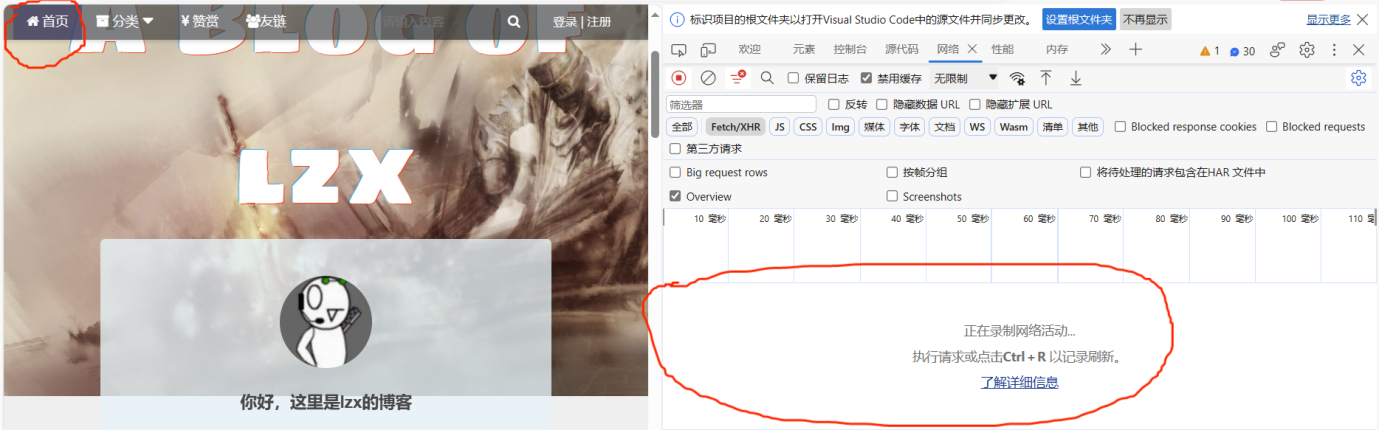
前端优化

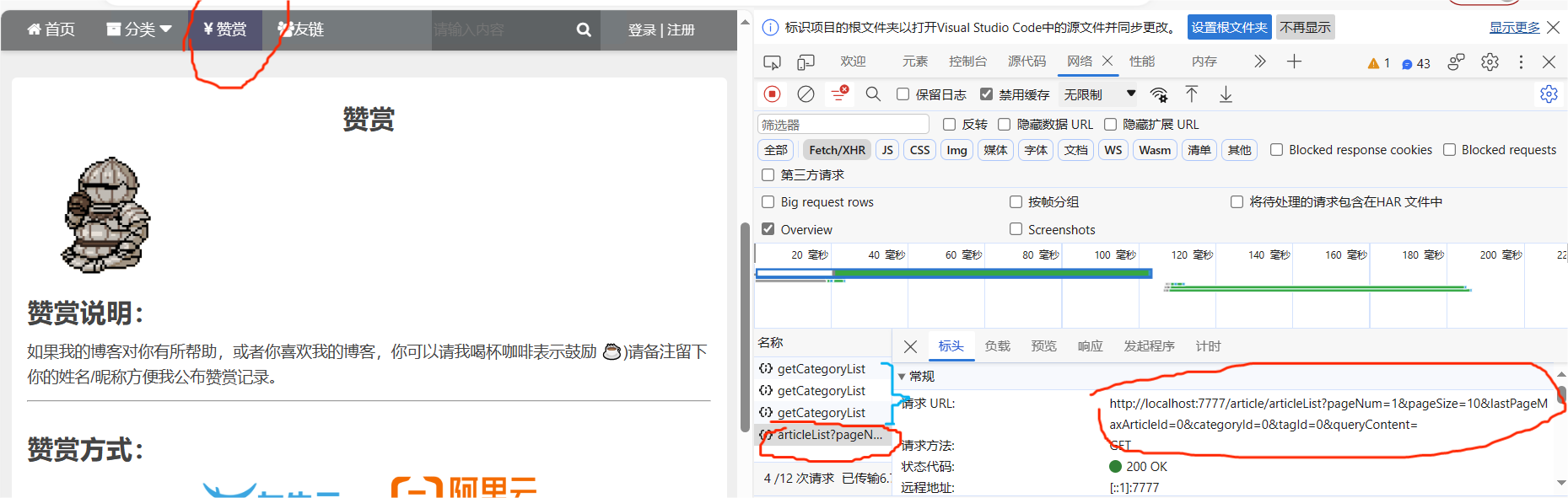
案例1. keep-alive组件和watch监听器引来的多余请求问题

***1.1背景：***

在从首页跳转到悬赏页面时，我发现明明悬赏页面中没有调用getAtricleList接口，但从f12开发者工具中却发现调用了该接口。另外，getCategoryList却被调用了多次。这两个问题如下图所示。



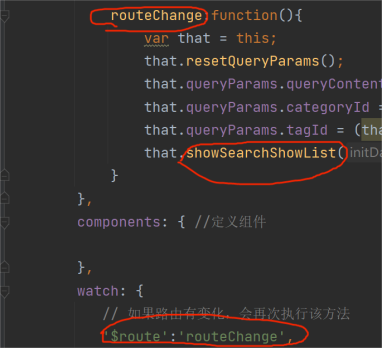
（在首页，此时f12开发者工具没有记录任何请求）



（跳转到赞赏页，可以看到f12记录中有getArticleList请求的记录，还有3个getCategoryList）

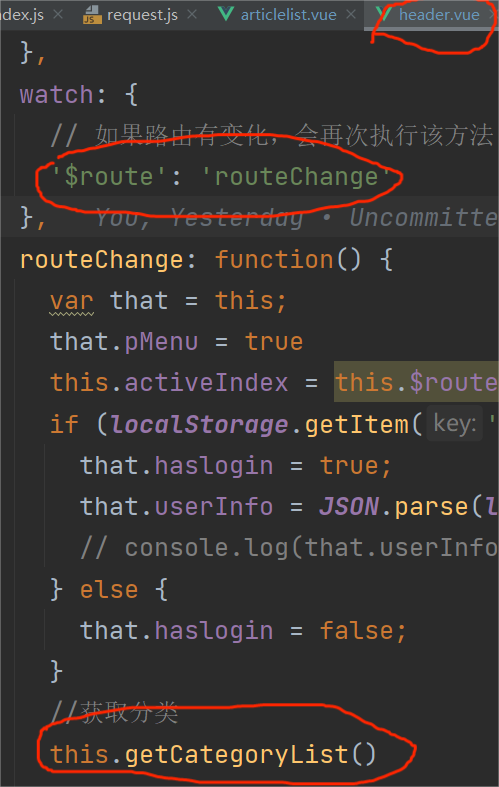
***1.2原因分析：***

不难想到，在跳转页面时，vue实例的$route变量发生了变化，而在Home组件中用watch监听器监听着这个变量，如果发生变化，则调用getArticleList接口，如下图所示。



（Home组件监听着$route,如若变化则会执行routeChange->showSearchShowList进行getArticleList的调用）

而keep-alive组件又使得Home组件在跳转后没有被销毁，所以会在跳转时调用getArticleList接口。至于getCategoryList为什么有3个,是因为有3个header组件都没有被销毁，处于失活或激活状态（没有被卸载），所以这3个组件都监听到了$route变量，都执行了包含调用getCategoryList接口的相关方法。Header组件的相关代码，外观如下图所示：



（header组件，可以看到如果检测到路由有变化，就会调用routeChange->getCategoryList进行调用）



（header组件外观）

Header组件明显应该只有1个。这个相关的优化将会在案例2中讲述

***1.3优化：***

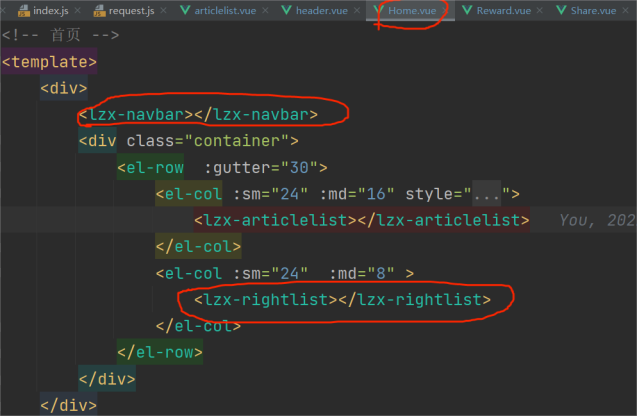
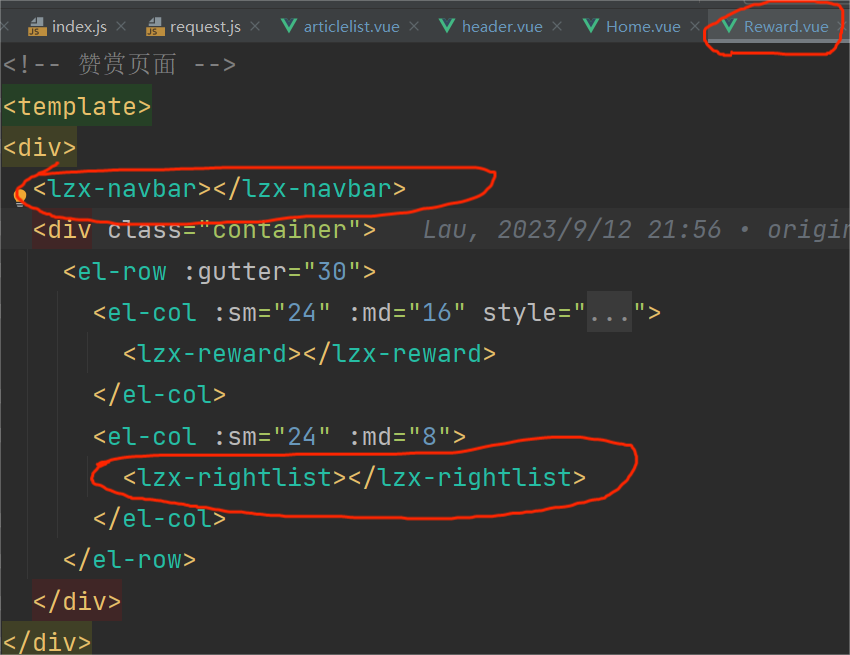
那么,keep-alive组件要去除吗。为了避免用户可能的重复请求导致的服务器资源浪费，先尽可能保留他。至于多余的getArticleList请求，是因为articleList组件用watch监听着$route这个变量。因此我们删除对这个变量的监听。而读者因此可能担心从其他页面跳转到首页无法显示数据。但不会有这样的问题：如果是显示之前首页的数据，keep-alive会帮我们解决；如果显示新的数据（这里特指从右侧的标签云查询数据或者从上方的内容查询框搜索数据），这里作者使用的是组件之间的事务通信，比较简单，略去代码。对于getCategoryList，因为相关数据变化频率比较低，而且实时性要求也比较低，所以只需要在创建时获取即可，至于监听器，我们选择保留，因为其中调用着其他方法。代码同样比较简单，因此略去。

优化结果：使用keep-alive避免了用户可能的重复请求。也避免了上述getArticleList,getCategoryList的多余请求。

案例2. 页面未能公用组件导致的多余请求

***2.1背景与原因***

在优化案例1中，笔者提到会有3个header组件，至于为什么会有3个header组件，是因为原项目中没能共用该组件，这就导致一个页面一个header组件。在案例1中，我们得知，如果这些页面都被keep-alive组件包围，则会导致这些组件都会监听$route变量，从而导致多个不必要的getCategoryList调用，代码如下图所示。

（如上图所示，可以看到，不同页面的相同组件并没有被共用，这就导致如果点击了多个有lzx-navbar组件（即header）的页面，而这些页面又被keep-alive组件包围，就会导致多个相同的header组件同时被缓存）



（这是路由文件的代码，可以看到都是一级路由）

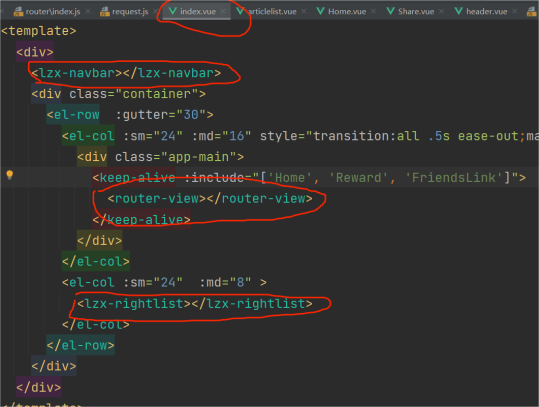
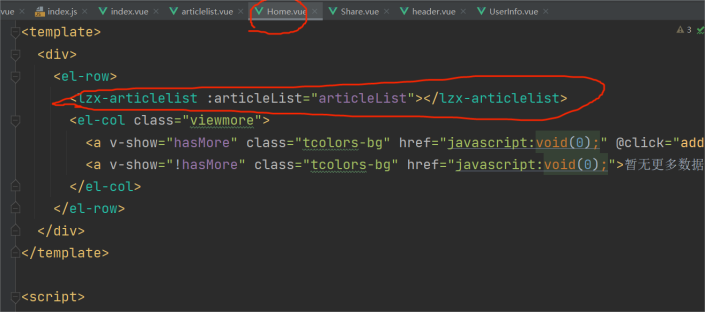
尽管在案例1中我们解决了多余调用getCategoryList的问题，但如果调用多个页面，则还会请求多次header资源。期望的是只需有1个header组件被缓存。

***2.2优化***

优化后的相关代码如下图所示。



（路由文件，可以看到，从只有一级路由变成了有二级路由）

（左图为index组件，包含了lzx-navbar，lzx-rightlist组件；右图为Home组件，包含了lzx-articlelist组件。可以看到成功实现了组件共用）

优化结果：一般情况下，用户只会请求一次lzx-navbar等组件。