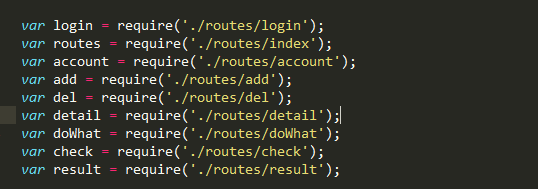
如上，即是需要实现的功能，及各个页面之间的跳转

1. 明确流程，实现基本的跳转
   1. app.js添加相应的映射关系（路由表）





* 1. 按思维导图实现简单的页面跳转
     1. 登陆（login.js）对应一个login.ejs

login.js只要有get和post方法就可以

login.ejs只要有如下的表单即可。



* + 1. 在login页面登陆成功后，跳转到主页面，否则渲染login页面，显示登录失败.（我的方法是，在get和post请求【验证失败的时候】，都传入一个值给页面，只不过前者为空，后者为“密码错误”）。如下：



主页面如下（/index）： 需要index.js和相应的index.ejs

Index.js:这里没有涉及到表单的提交，所以只有get请求。

Index.ejs只要实现如下的样式即可。



* + 1. 逐一分析：我的账户（/account）:需要accout.js和account.ejs

account.js:只要get请求就可以。因为只是为了显示管理员的一些信息。

account.ejs：显示相关信息，如下



* + 1. 加入vip(/add)：需要add.js与add.ejs

add.js: get请求跳转到相应的页面（无参数）;post请求处理表单的提交。

add.ejs:和注册信息的有关的一张表单，如下：



* + 1. 充值/消费页面（/doWhat）:doWhat.js与doWhat.ejs

doWhat.js:get请求，带tel参数。Post请求处理表单的提交

doWhat.ejs: 充值和消费的表单，还有一些当前账户的信息，如下：



* + 1. 会员信息查询页面（/detail）:detail.js与detial.ejs

detail.js:负责get请求就可以，因为只要显示信息就可以了。

detial.ejs：一张表格显示充值记录(同时，有个人信息)，如下：



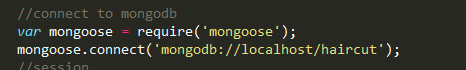
最后一个是删除（/del）：del.js :从数据库中删除数据。只要显示删除成功一个话就可以了。所以暂时不需要del.ejs。

del.js处理get请求，即可。带参数

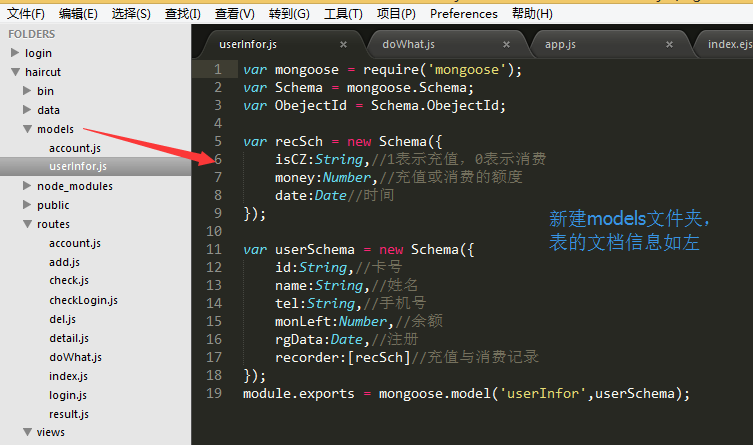
二、实现了基本的跳转之后，就要把数据库加入进去。

1.npm install mongoose//安装模块文件

2.app.js中连接



3. 创建如下



4. mongod -dbpath "./data" 启动数据库（第一次启动会自动创建）。我直接放在项目了

5.测试下是否正常，ok的话。就可以开始行动了。

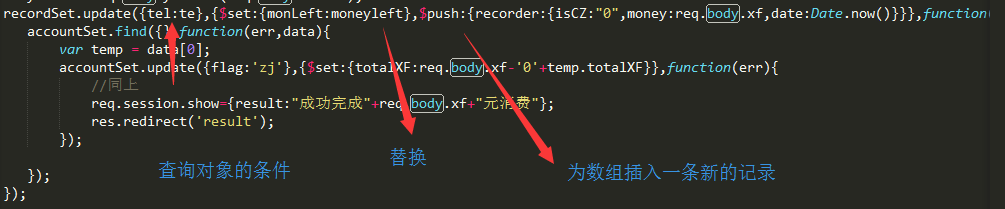
6.想象一下：

a注册页面表单的提交需要(插入数据)，先写

b其余一些需要查询或者更新数据库，让后写

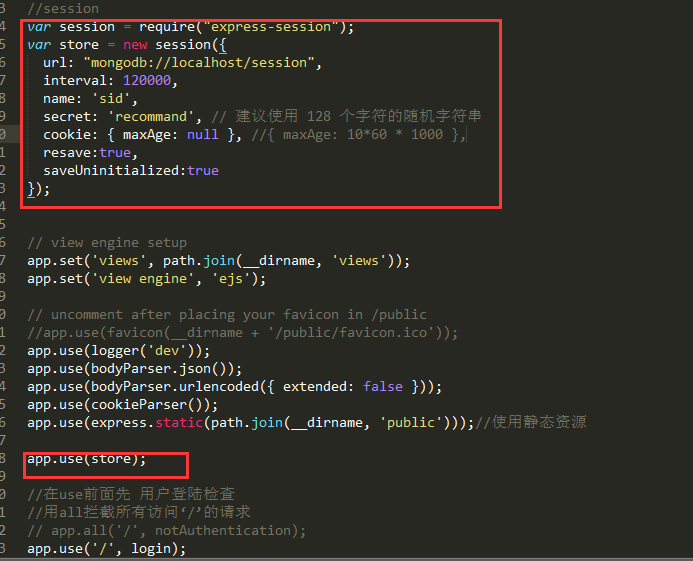
三、遇到的问题：

1. 我们充值或者消费的时候，我们需要修改表的数组。这里可以百度得到。



2路由之间值的传递：session

A npm install express-session.

B 

C 就可以直接使用了。如

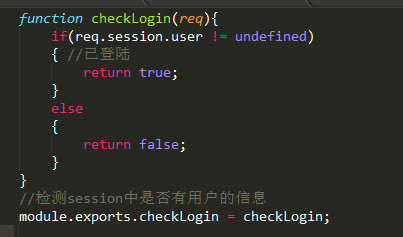
req.session.user = {id:’55’,pwd:’5188888’};

req.session.err = ‘用户名错误’;

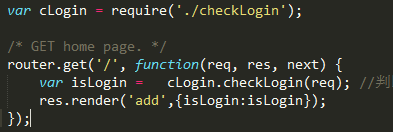
3.如何防止用户直接输入类似localhost/add这样的连接。这里有多个方法。

A .session

我们可以在用户执行每个get请求的时候，先判断req.session.user是否有值，如所以，写一个公共的方法，如下：

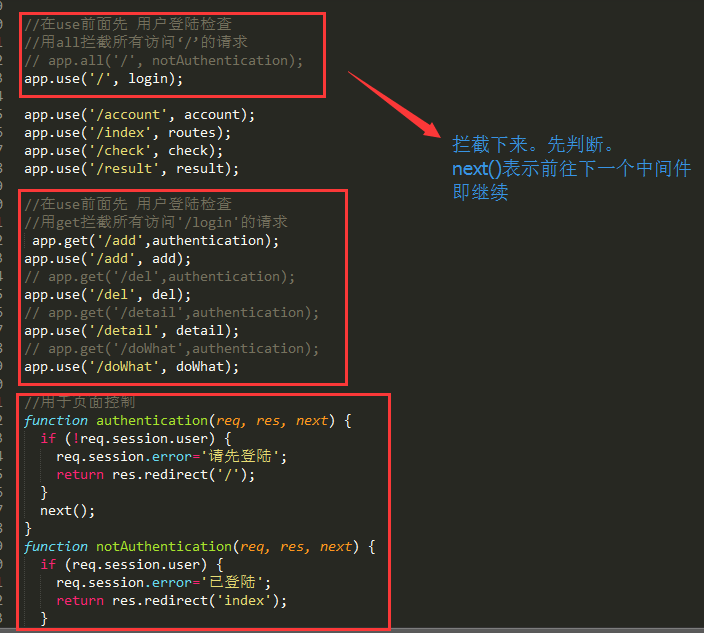


需要的时候，像这样，先导入，就可以使用



利用上面的返回，传递给页面，页面在各个操作之前，在js中那个先判断<%=isLogin%>为true or false。为false的话。Window.location = ‘/’ ;return false;(可以阻止表单的提交)。

B.app.js中对页面的控制。

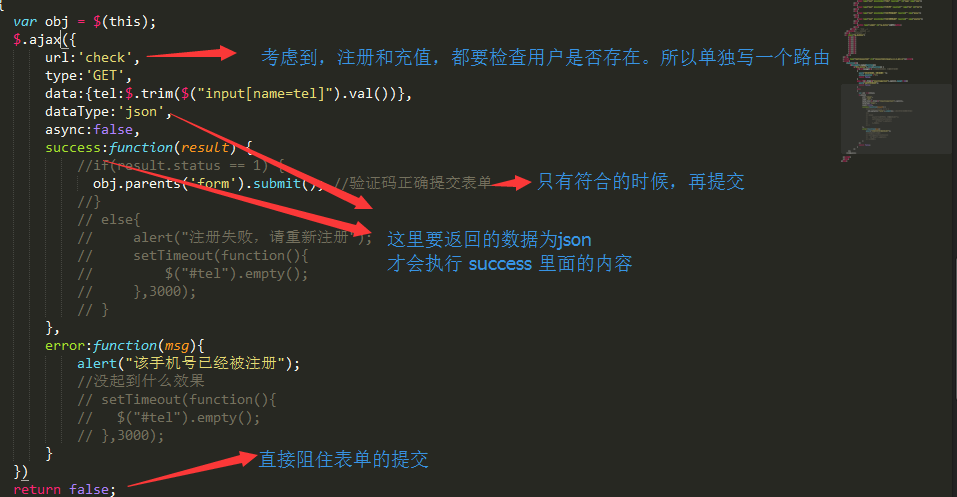


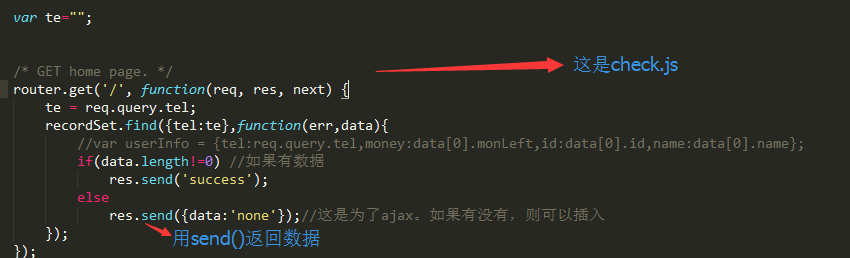
C 上面明显很麻烦，每个路由都需要添加。

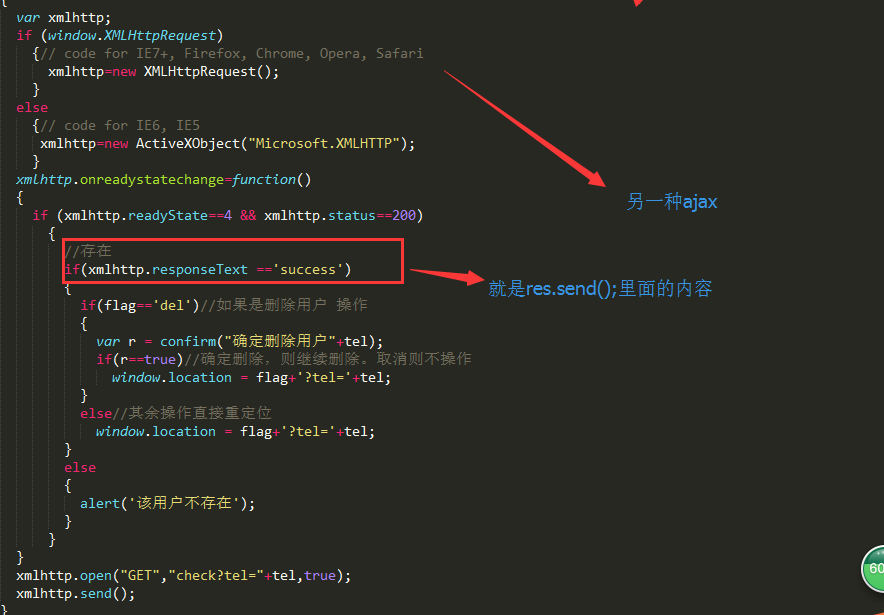


4. 如何在表单提交前，先判断下。

比如，注册的时候，我们需要先判断用户是否存在。







5我们会发现，我们这里注册成功之后，或者充值成功，删除成功，都需要一个向用户显示的结果的页面。所以也就有了result.js和result.ejs

result.js:操作完成之后，定位到这个页面，可防止刷新，导致数据重复提交。

result.ejs:显示操作结果，并且会自动跳转到主页面。

四、写样式