星宿网站框架设计总结：

# 一、搭建

首先在nodejs中安装express模块；

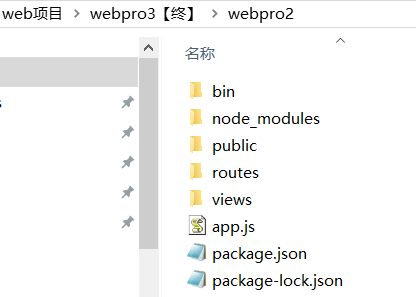
Npm install -g express

接着安装 express-generator；

Npm install -g express-genertor

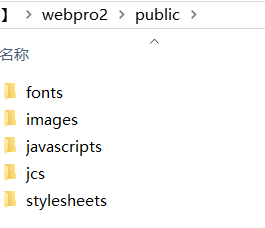
接着使用命令创建一个项目

Express -e webpro2

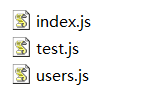


bin –存放连接数据库的操作

public –存放静态资源文件，css，图片等



routes –路由文件，存放js程序，用来分析客户端发来的post和get命令



views –ejs模板文件，存放ejs模板编写的前端页面



app.js –程序的入口，里面主要是利用以下代码进行路由分发

var indexRouter = require('./routes/index');

var usersRouter = require('./routes/users');

var te = require('./routes/test');

app.use('/', indexRouter);

app.use('/users', usersRouter);

app.use('/', indexRouter);

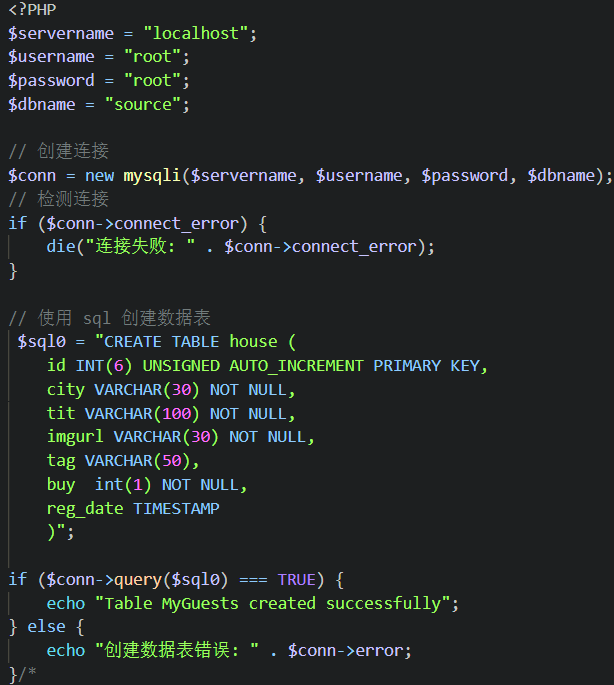
app.use('/test', te);

package.json –存放配置信息，保存各个依赖包的版本信息等

# 二、数据储存

## 1.表单创建

用的是php脚本来进行表单创建，因为用mysql命令行进行创建太麻烦了，然后直接放上代码，用的是mysqli函数进行数据库连接，然后创建语句，进行表创建。



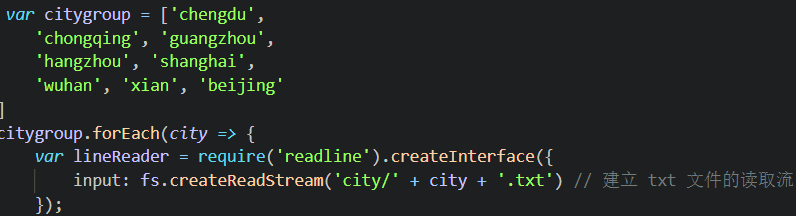
## 2.数据库数据上传

用的是nodejs中的fs模块和mysql模块来对txt文件按行读取，然后用正则来对数据流进行目标数据提取，

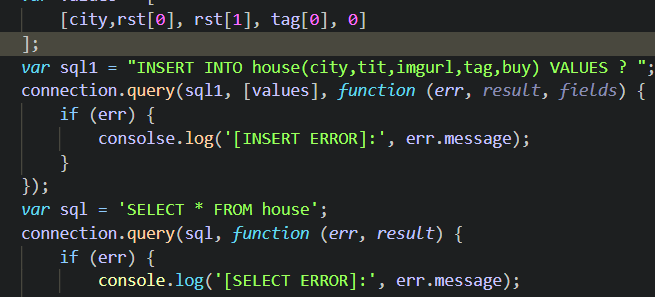




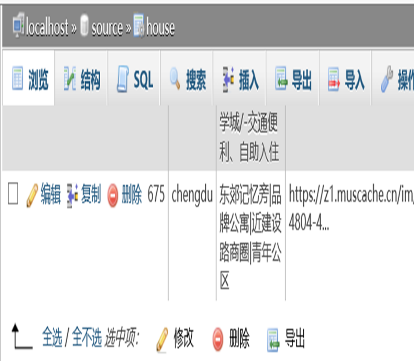
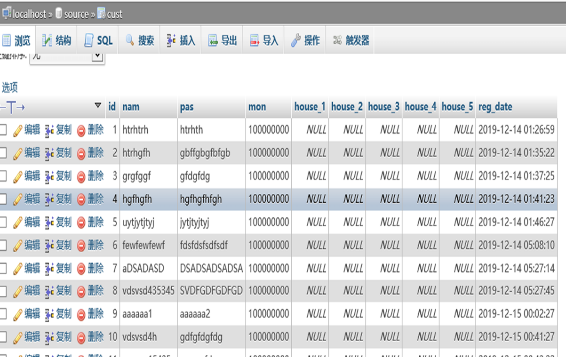
对本地txt文件进行读取



进行数据提交到数据库



左 客户表；右 城市表（675条数据）

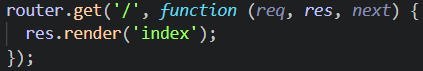


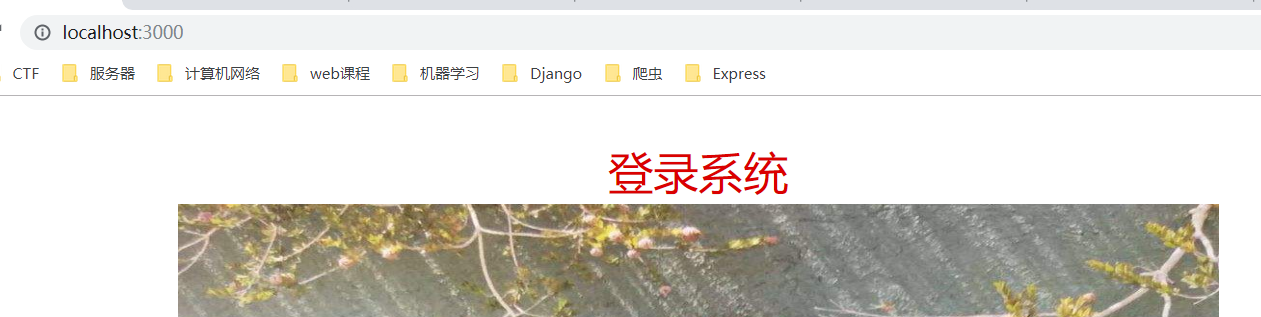
# 三、路由处理

## 1.Index.js

1. 5个get请求

**第一个是登录界面**，会返回index.ejs

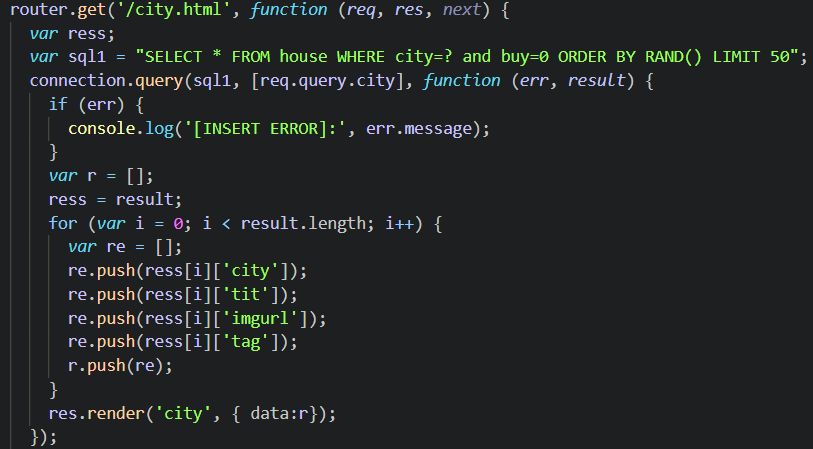




**第二个是城市界面请求**，客户端发送<http://localhost:3000/city.html?city=chengdu>请求后，服务端会收到该请求，然后通过req.query.city提取里面的参数，通过数据库查询语句

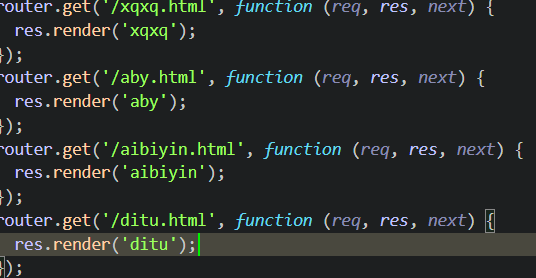
*var* sql1 = "SELECT \* FROM house WHERE city=? and buy=0 ORDER BY RAND() LIMIT 50";connection.query(sql1, [req.query.city], *function* (*err*, *result*) {

返回相应的城市数据，并通过拆分组成数组和city.ejs模板一起返回给客户端。





剩下其他是相关页面的请求



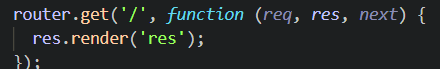
2.一个post请求，该post请主要是登录功能，所以会对数据库进行查询，然后根据查询结果返回相应的操作。将数据库pass与req.body.pass进行比较。



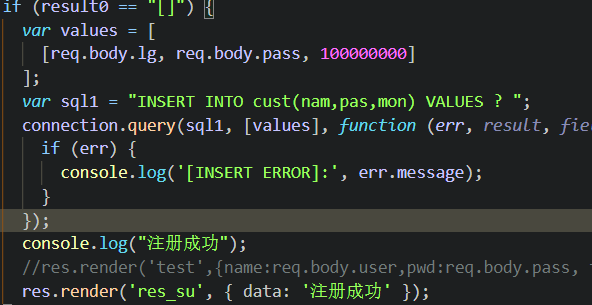
## 2.users.js

该路由js里面只有有个get’请求处理和post请求处理；

首先get请求，是返回res.ejs注册界面



然后post请求，该请求是处理注册的数据请求，这里使用中间件body-parse，用req.body.参数名来得到客户端发送的注册数据，在对数据进行提交时，首先会进行判断，主要是判断是否该用户名被使用过，如果没有被使用过，就进行提交到数据库，然后返回注册成功。



如果账号已经存在，则返回已经存在；



## 3.test.js

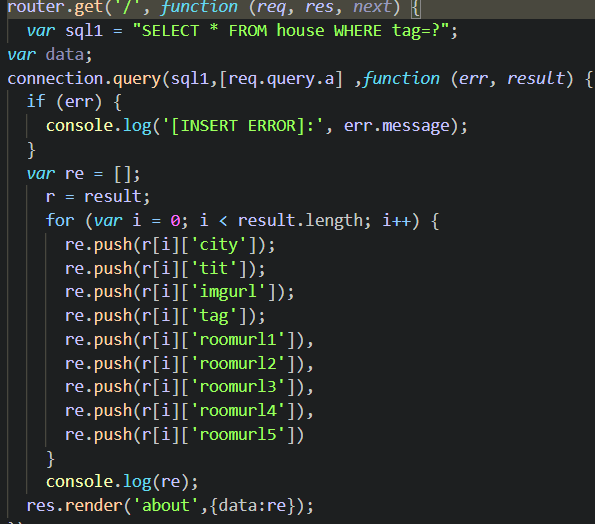
该路由主要是来对每个房子的信息进行处理，如果客户端请求

<http://localhost:3000/test?a=40106113>

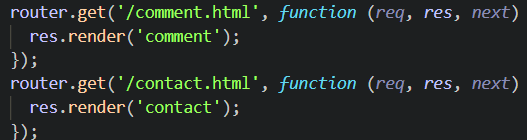
则test.js会对参数a进行提取，然后在数据库中进行查询tag=req.query.a的数据，将得到

的数据进行数组化处理后，与about.ejs一同返回给客户端，进行页面渲染。

另外的该路由还会处理来自/test下的其他get请求，即与about.ejs有关的页面链接。



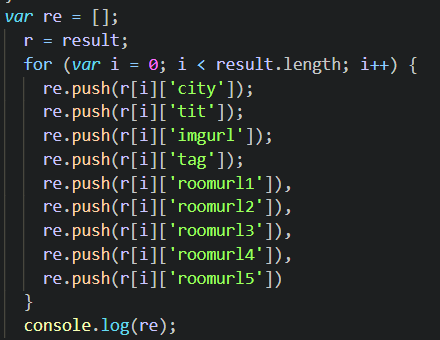
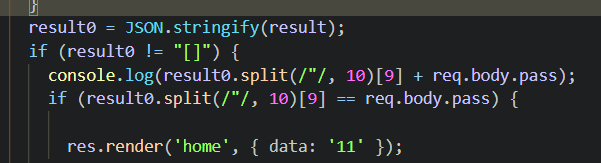
然后对其他请求，



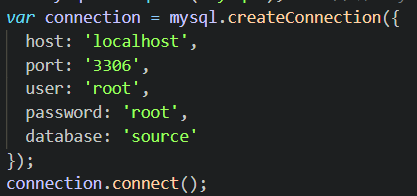
## 4.其他重要的地方

1.对于数据库提取出来的数据，由于是object对象，直接传给前端是无法使用的（反正我没使用上，可能是我太菜了），所以通常对每个select语句的地方都会使用for循环进行数组化，方便前端页面的使用。在查询是否存在账户时，由于不会使用exist，所以就使用其他麻烦的操作进行替代，将查询结果进行stringfy，如果结果为空，在string化后是[]

故通过if res==[]；将新判断。

2.在每个路由js中都会有可连接数据库的语句块



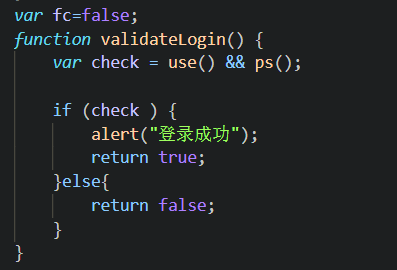
# 四、登录.ejs和注册.ejs

## 1.登录.ejs

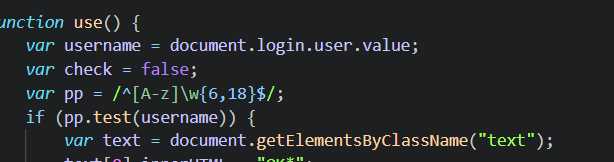
里面有个轮播图，然后就是表单提交，在提交时，使用js进行验证，这部分就是利用正则表达式进行判断，首先利用dom提取表单的值，然后验证，如果进行所有验证后，均能通过，则设定一个哨兵==true，在表单form中使用属性onsubmit进行限制，当onsubmit==true时，才允许提交。



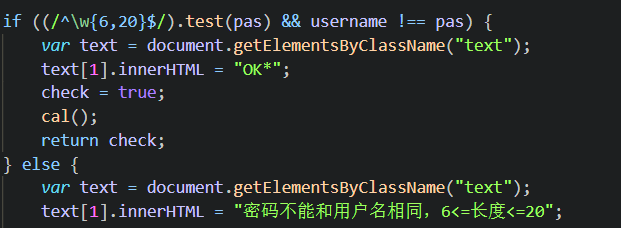
Use()和ps()函数返回true或者false



对账户的限制是用户名只能以字母开始，而不能以数字，\_或$开始，字符长度大于6小于20，然后正则式是 /^[A-z]\w{6,18}$/，利用test函数，

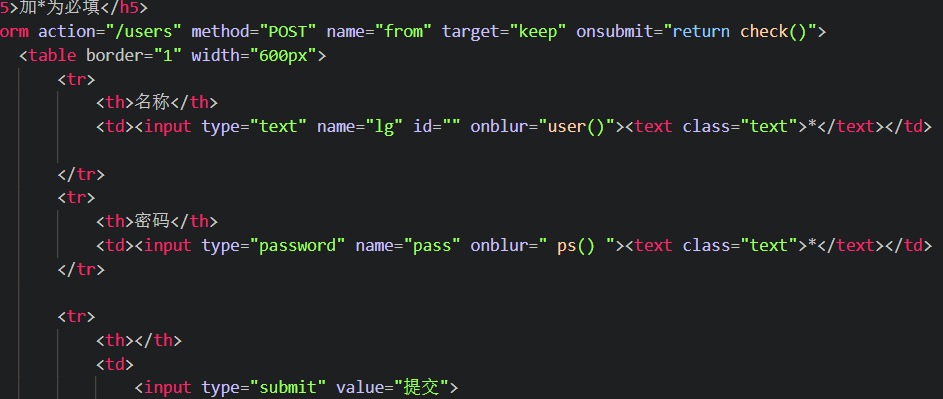


对密码的限制是首先不能和账户相同，然后长度有限制。



## 2.注册.ejs

里面思路和登录的ejs是一样的，这里不再赘述，主要是表达值验证，onsubmit判断，最后提交，服务器判断，最后返回注册结果。



# 五、心得体会

在进行后端搭建的时候遇到了很多很多困难，主要是对express框架的不熟悉，然后对数据库的语法不熟悉，对模型的加载渲染不熟悉。

Express框架中有很多页面模型，，默认是jade模板。然后可以设置-e变为ejs模板，还可以通过ejspress引擎进行改变，变为html模板，我在第一次时，用的html模板，遇到了一个最大的问题，就是在渲染页面并加载附带的json数据时，html页面无法渲染json数据，为了这个事，我几乎浪费了50个小时，几乎差点放弃。最后，还是基于对web的热爱，决定重新设计整个网站框架，在网上看了很多资料后，发现在别人博客中有利用ejs模板进行数据加载的demo，于是决定利用他的示例。

利用ejs的确可以渲染服务器数据，这是是我高兴不已。然后我又遇到了另外一个问题，这个问题我在利用pyhon中的django开发网站时没有遇到（我还是用的我写python开发网站的思路）。最后经过很久的试验，终于发现了差别所在。Express在加载ejs页面时，如果<%=变量%>，就会查询是否有变量从服务器传来，如果没有，就会报错。但是在django中，django加载页面时如果html页面存在{{ 变量 }}，如果服务器有传送相关数据，就会渲染数据，如果没有数据，就会默认不加载。

在对数据库进行操作时，也犯了一个不容易发现的错误。即如果没有将res.render('about',{data:re});放在connection.query(sql1,[req.query.a] ,function (err, result) {}的回调函数中，res.render和connection.query将会同时执行，data中得不到数据库中select的数据，第二就是，function是个函数，如果变量声明在回调函数中，在外界是得不到里面的变量。在连接数据库时，也花了很久的时间，因为我以为是8088端口，结果不是，是3306端口，唉气死了。

学会了在浏览器的开发者模式查看网页的问题，比如某些静态资源没有加载上，某些变量命名错误或者重复，以及可以动态调试页面布局。还学会了用console.log输出变量来检查某些算法或者语法错误。学会了从各个地方查阅资料，用test的方法写demo或者测试函数之类的，总之收获还是蛮大的。下面是参考的资料。

# 六、参考资料

