config文件夹：

routes.php：路由配置

$route[‘default\_controller’]=”welcome”;

配置默认的控制器为welcome.php

Controllers文件夹：放控制器，一般情况下只负责跳转

语法：控制器类名首字母必须大写，必须继承CI\_Controller类

方法：index方法总是在URI的第二段为空时被调用

URI的第二段：php页面名称为第0段，之后开始计算

如example.com/index.php/blog/index/中，index为第二段，表示blog类中的index方法

Model文件夹：负责各种逻辑实现

表单提交时调用的php方法：

<form action=index.php/类名（控制器）/方法名 method=”get/post”>

（注意要给每个form表单中的input元素添加name属性）

如何获取到提交的数据：

$num1=$this->input->post(‘num1’);//此处num1为提交表单中input的name属性对应的输入值

若要返回POST中的多个元素，将所有需要的键值作为数组传给它

$this->input->post(array(‘field1’,’field2’));

CI中关于调用其他文件里的方法

$this->其他php文件名->方法

CI中给视图添加动态数据

$data = array(‘num1’=>1, ‘num2’=>2, ‘op’=> +, ‘result’=>3)

$this->load->view(‘result\_view’,$data)

在视图中，只需要知道键名，就可以取得相对应的数据

CI默认路由规则：



Example.com/index.php/floder/class/function/id/

Example.com指网站的域名或IP地址

Floder指控制器所处的文件夹（将不同的模块对应代码放入不同的文件夹中，对直接放在控制器根目录下的类（文件）不需要这一级路由））

Class是控制器的类名，注意不是文件名

Function是控制器中对应的函数名，如果是index函数，可以不写

Id是要传入function的参数（多个参数之间用/分隔）

函数：

File\_exists()原声PHP函数，用于检查某个文件是否存在



Routes数组使用了通配符(:any)，可以匹配所有的请求，然后将参数传递给Page类的view()方法

CI默认支持的数据库为mysql，要想让其支持sqlite3，需要：

修改application/config/database.php



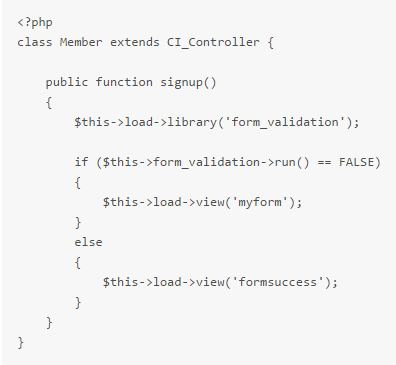
高亮部分为修改的地方，database应该修改为根目录下数据库文件的地址



APPPATH，代表项目中的application文件夹

Form\_validation辅助类中的run方法：

1.当一组规则的名称和控制器类/方法名称完全一样时，它会在该控制器类/方法中自动被run()方法调用

Eg: 

这个例子所对应的规则名称为member/signup

2.也可以指定进行验证的规则

eg: if($this->form\_validation->run(‘signup’) ==FALSE)

validation\_errors(起始标签，结束标签)

返回在起始标签和结束标签之间的表单验证类生成的错误信息

Login功能的实现：

1. controllers/user\_authentication.php

session\_start();

需要用到form、form\_validation、session、自己建立的database

需要实现的是：登录和注册

登录：

1. 在index方法中（这是默认执行的方法）加载login的界面
2. 进行登录功能的实现：检测输入的值是否符合要求（这里需要加入session的判断，如果浏览器session中存有用户的login信息，虽然form\_validation->run()方法的运行结果为false，但是由于之前的登录信息，可以直接跳转到登陆后的界面），通过login的model查看用户名密码是否正确，

若用户名密码正确，将用户的数据加入到浏览器的session中去，并加载登录后的界面；

如果用户名密码错误，提示用户错误并要求重新输入

注册：

1.加载注册的界面

2.进行注册功能的实现:检测输入的值是否符合要求（form\_validation）,若不符合要求，要求用户重新输入，如果符合要求的话，获取输入的值，并将之存入到数据库对应的表中（需要用到model的insert方法），根据插入结果(TRUE/FALSE)显示不同界面

登出：

1. 移除session数据
2. 显示login界面

2.views/login\_form.php

1.如果浏览器的session有之前的login数据，直接跳转到之前的登录功能验证（controller中）？？？？

2.php动态加载页面（加载消息，消息在前面的controller中设置过了，并且在加载视图model的时候传入了消息）

3.加载登录验证的控制器函数(form\_open(控制器函数))

4.输出错误消息

5.登录表单的界面(按钮的监听可以直接加到登陆后界面的展示，因为错误都在前面处理过了)

注意使用了form\_open要配套使用form\_close()

3.views/registration\_form.php

1.同2(1)

2.首先打印表单错误信息(validation\_errors)

3.form\_open(控制器的注册函数)

4.打印具体注册过程中的错误函数（具体同2.2）

5.注册表单的界面

3.views/admin\_page.php(登陆成功后显示的界面)

1.如果浏览器的session中有之前的login数据，可以直接从session中拿到对应的username并显示，如果没有的话header(“location:login”)???

1. 主页面显示，要提供一个logout连接？？

3.models/login\_database.php

1.注册时需要验证用户名是否已经被占用，registration\_insert()

2.登录信息

3.读取用户信息