# 后台

封装mysql

封装mysql的时候注意 有sql 参数 还有一个数组（根据不同查询设置值）,第三个参数因为是有两个不同的值，用箭头函数可以根据什么参数去判断比较方便 编程的思想能（mvc）封装更为灵活，api使用的时候更加语义化

检测路由，

1利用数据库查看数据库有没有连接上，在后台查看结果（命令行）

2之后在去用res.render渲染页面

检测用户名是否存在（输入用户名时 触发的ajax）

1获取用户填写的内容（要把body-parser记得引用上）

2之后连接数据库去数据可查询用户对应的数据 如果有错误会提示错误

3获取之后要去进行判断有没有用户名（res.json 将响应的数据返回给页面 用json的形式是因为前台ajax请求的时候方便取，取了之后进行判断，根据判断内容取确定返给客户什么的信息）

注册

1获取用户填写的内容（要把body-parser记得引用上 有username ，password， email， v\_code验证码）

2先去判断邮箱和验证码合不合法

3连接数据库去数据可查询（用户 邮箱 ）对应的数据 如果有错误会提示错误

4获取邮箱 用户之后要去进行判断 分为两种情况一种有数据 一种没有数据 没有数据分为两种 用户不存在 邮箱不存在

登陆

1获取用户填写的内容（username，password ，remember\_me）

2之后连接数据库去数据可查询用户对应的数据 如果有错误会提示错误

3获取之后要去进行判断用户名与密码是否正确 正确之后就存储

4之后将登陆的信息传送给session （用户可以你下次取session取 例如七天之后免登录）先要引入中间件和包espress-session 去官网查询

退出

退出之后就吧session设置为空

之后显示信息

显示登录页

根据跳转链接显示注册页面

跳转页面的连接之后根据不同情况去渲染页面

显示注册页

根据跳转链接显示登录页面formidable

跳转页面的连接之后根据不同情况去渲染页面

验证码

根据随机生成的数据

利用包将数据转化成图片

将验证码吗图片存储在session中（要根据这个图片做对比）

之后img标签相应图片的数据

添加音乐

0先利用session去判断等没登陆（后面可以取消）

1引入formidable这个包，之后先将var form = new formidable.IncomingForm();

2将移动的目标目录复制给变常量，因为下载文件路径是默认的

3用form.parse(req, function(err, fields, files) {}这个方法将数据下载下来

4如果有错将错的信息打印出来

5建立一个数组，将数据放在里面 （后面调用sql语句的时候直接调用，优点 存储比较方便，调用也比较方便）

6之后将公共的地方存给变量（sql，和 路径）进行拼接方便移动

7判断文件的歌词和歌曲（files.file，files.filelrc）是否有 ，有的话就拼接 因为是文件所以SQL语句不一样，路径不一样

8进行路径拼接的时候要把文件的名字取出来 之前在session存储的数据id 就是musics里面的uid 获取数据里面需要用这个数据 所以得去获取

9将拼接好的路径 放入data数组里面 将这个数据放入数据库，

10添加成功返回对应的数据

编辑音乐

和添加是一样的 sql语句不一样

分离

配置文件

App。Js

先将路由文件抽取出来

路由调用控制器（）

后端接口

登陆注册音乐的增删改查

登陆:验证用户密码一致

后端权限验证

在中间件通过正则区分哪些验证

如果session上没有user代表没有登陆

音乐的增删改查

添加/更新音乐

存储文件路径生成网络url的请求路径， 存储db音乐方便音乐列表的使用

配置静态文件的的中间件

查询/删除音乐

url 传递参数

利用path的方式查询字符串的方式

根据查询的东西要判断是否登陆

音乐列表（express）

利用{{each}}去渲染页面

进行判断

整体套路

前台注册

、 用户名input节流

密码强度

后台注册获取数据

获取表单数据

Ajax的请求

接收参数

url参数

请求体

处理数据

将数据保存到DB(数据库)

响应数据给客户端

根据响应对象的code进行判断响应页面

头部中不需要处理因为是公共的

Musiccontrol 里面是需要引入

配置config文件

用到了fromidable

Path核心对象

数据库db文件

注意

// module.exports 默认是一空对象，和exports 是相等的

M - Model 数据模型(DB)

V - View 视图(res.render + 模板引擎 )

C - Controller 控制器

C就是调用M + 调用V + 属于自己的处理业务逻辑

# 前台

添加

·· 获取利用from，获取客户端的数据 （里面有文件所以不能用from里面的serialize，创建var formData = new FormData(); //装数据的篮子，将通过input表单里面的.vauel和.files这两个属性获取，直接获取里面的值的结果是盘符的绝对路径，这是固定用法）

发起ajax请求，。

成功之后判断是否成功，成功渲染页面，失败提示信息

登陆

获取利用from，获取客户端的数据 var data = $(this).serialize();

发起ajax请求，。

成功之后判断是否成功，成功渲染页面，失败提示信息

注册

添加input事件，防止用户频繁操作造成的过多请求

失去焦点事件之后出发ajax请求

进行判断

密码强度

当按下按键时，改变文本域的颜色（keyup）

分为三种强度，根据什么样的强度显示几个盒子，

之后遍历有几个盒子，对应显示第几个的颜色

显示下一个之前要清空之前的

注册功能

获取利用from，获取客户端的数据

发起ajax请求，。

成功之后判断是否成功，成功渲染页面，失败提示信息

歌词，

渲染页面

渲染页面在后端直接渲染，将数据返回，直接用模板引擎

将获取的file和filelrc用自定义属性的方法写在标签里面，方便后面获取

解析歌词

将需要的数据进行正则（后面会进行正则匹配）

将切割的字符串进行遍历，

进行里面的正则匹配 符合继续下一步 不符合就结束本次循环（continue）

统计时间作为对象的key存在到对象后面方便操作取

将歌词作为对象的value

之后将遍历的obj 时间放在time属性中 将歌词的剩下的内容放在标签里面之后要动态添加到页面

滚动事件（timeupdate当前播放位置改变）

当前播放位置改变的时候触发事件（e.target是利用浏览器的bom获取当前时间的）

获取当前时间，通过e.target.currentTime获取当前时间

进行当前时间与在ajax获取数据的对象中的key判断

之后找到与点击的时间相对应

之后设置top值让歌词显示中间的话就要将盒子的top值减去p标签的高度

之后给当前的标签给标签他的同级移除盒子

点击事件

点击播放的按钮

获取在span标签里面的自定义属性

进行ajax请求 url获取歌词

成功之后解析那个获取的结果（将解析歌词的函数引用）

将遍历好的数据给span标签加进去

之后将滚动事件引入

再给audio的src加入文件播放