

# 黑马头条-数据管理平台



黑马程序员  
[www.itheima.com](http://www.itheima.com)

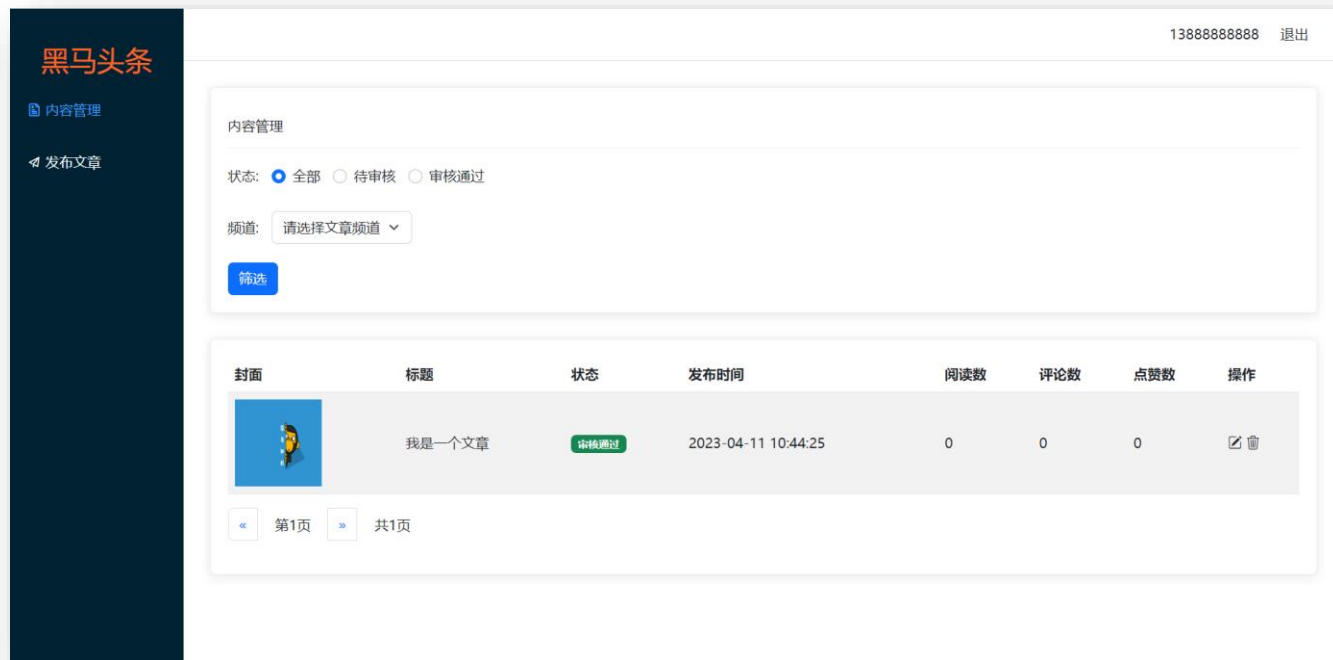
传智教育旗下  
高端IT教育品牌

## 项目介绍

黑马头条-数据管理平台：对IT资讯移动网站的数据，进行数据管理

数据管理平台-演示：配套代码在本地运行

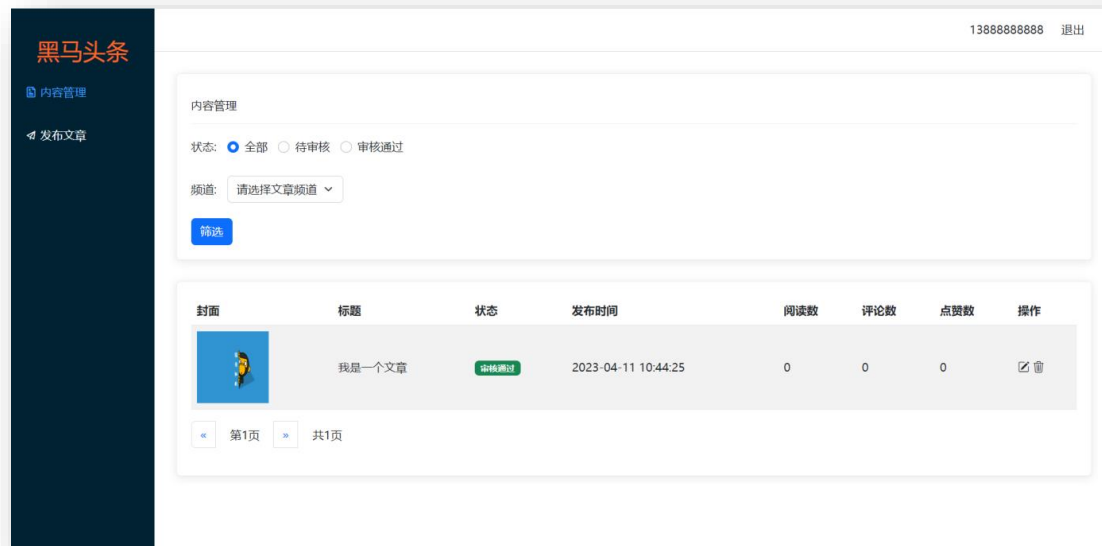
移动网站-演示：<http://geek.itheima.net/>



## 项目介绍

功能:

1. 登录和权限判断
2. 查看文章内容列表（筛选，分页）
3. 编辑文章（数据回显）
4. 删除文章
5. 发布文章（图片上传，富文本编辑器）





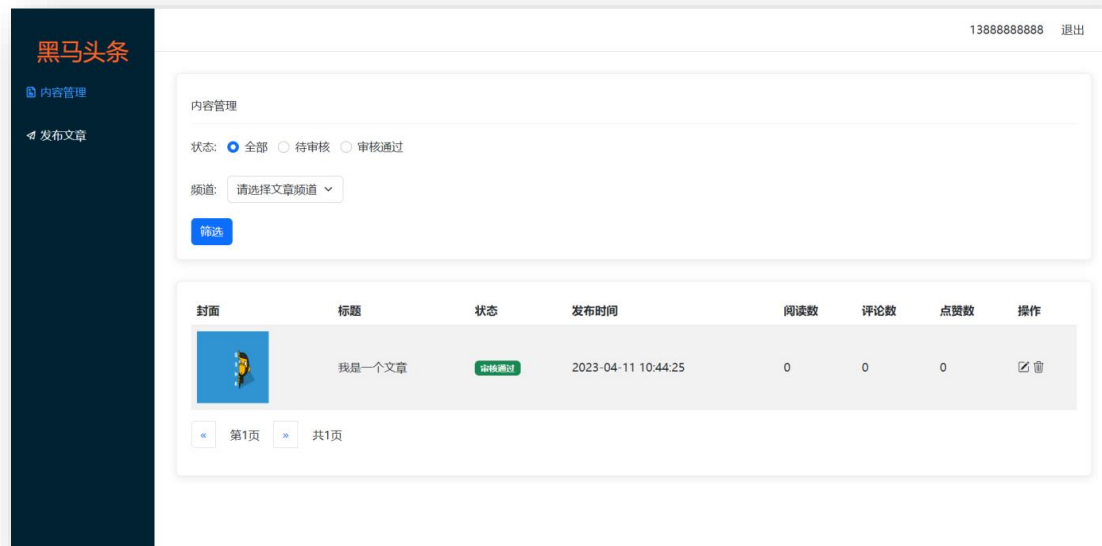
# 总结

1. 黑马头条-数据管理平台，是什么样网站，要完成哪些功能？
  - ✓ **数据**管理网站，登录后对数据进行**增删改查**
2. 数据管理平台，未登录能否管理数据？
  - ✓ 不能，数据是公司内部的，需账号登录后管理

## 项目准备

技术:

- 基于 Bootstrap 搭建网站标签和样式
- 集成 wangEditor 插件实现富文本编辑器
- 使用原生 JS 完成增删改查等业务
- 基于 axios 与黑马头条线上接口交互
- 使用 axios 拦截器进行权限判断



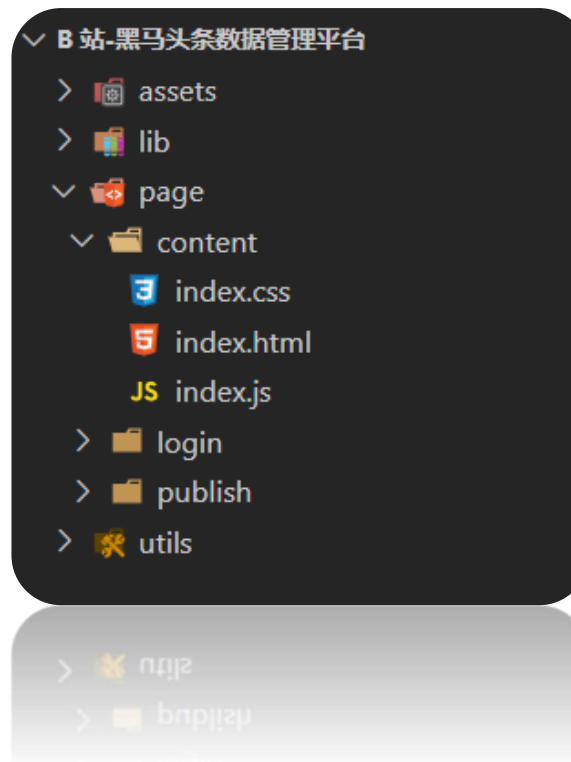
## 项目准备

项目准备：准备配套的素材代码

包含：html, css, js, 静态图片, 第三方插件等等

目录管理：建议这样管理，方便查找

- assets：资源文件夹（图片，字体等）
- lib：资料文件夹（第三方插件，例如：form-serialize）
- page：页面文件夹
- utils：实用程序文件夹（工具插件）





# 总结

1. 为什么要按照一定的结构，管理代码文件？
  - ✓ 方便以后的查找和扩展

## 验证码登录

目标：完成验证码登录，后端设置验证码默认为 246810

原因：因为短信接口不是免费的，防止攻击者恶意盗刷

步骤：

1. 在 utils/request.js 配置 axios 请求**基地地址**

作用：提取公共前缀地址，配置后 axios 请求时都会 baseURL + url

2. 收集手机号和验证码数据
3. 基于 axios 调用验证码登录接口
4. 使用 Bootstrap 的 Alert 警告框反馈结果给用户

登录成功

手机号或验证码不正确

```
axios.defaults.baseURL = 'http://geek.itheima.net'
```

黑马头条

请输入手机号

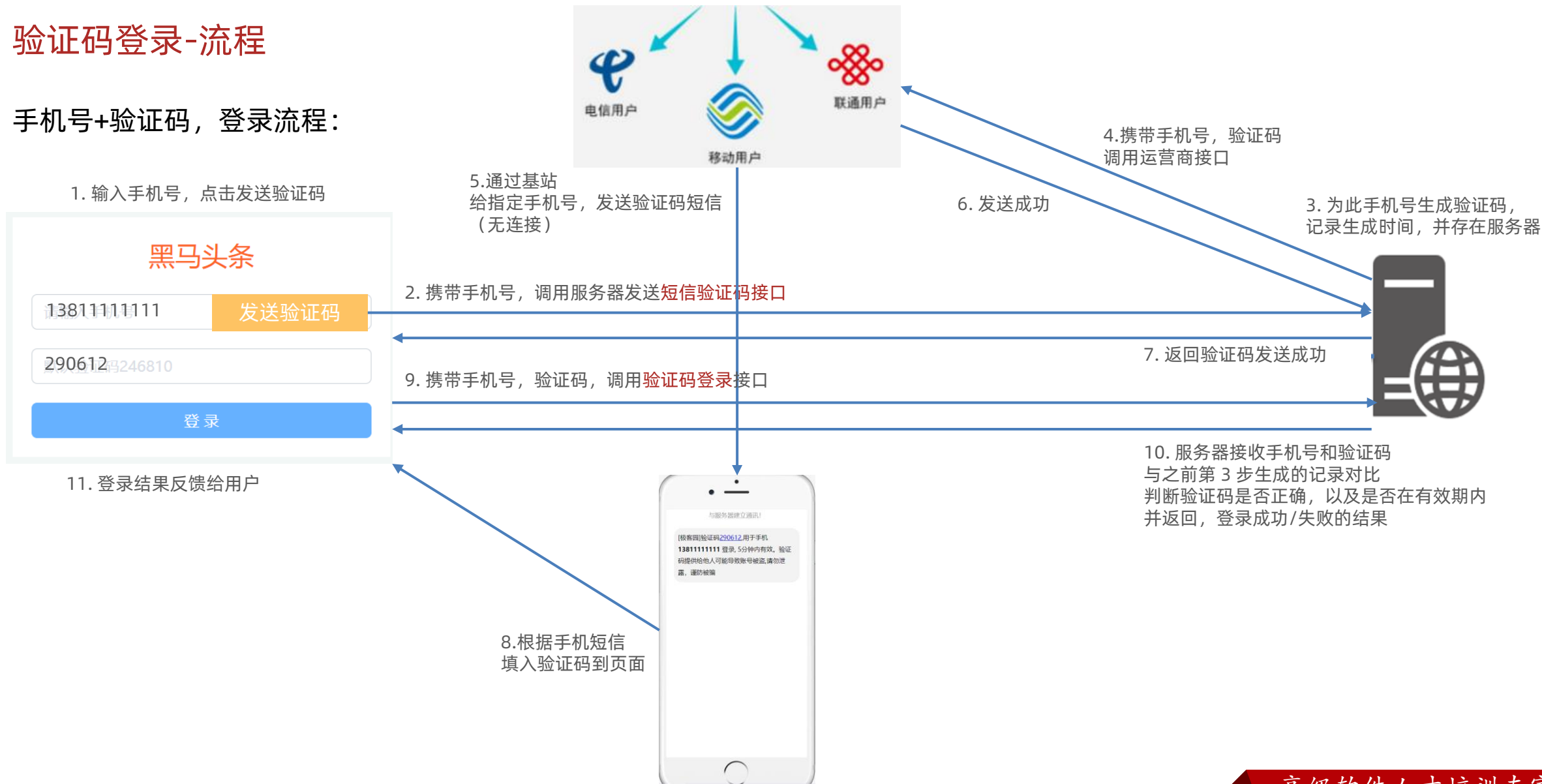
默认验证码246810

登录



## 验证码登录-流程

手机号+验证码，登录流程：



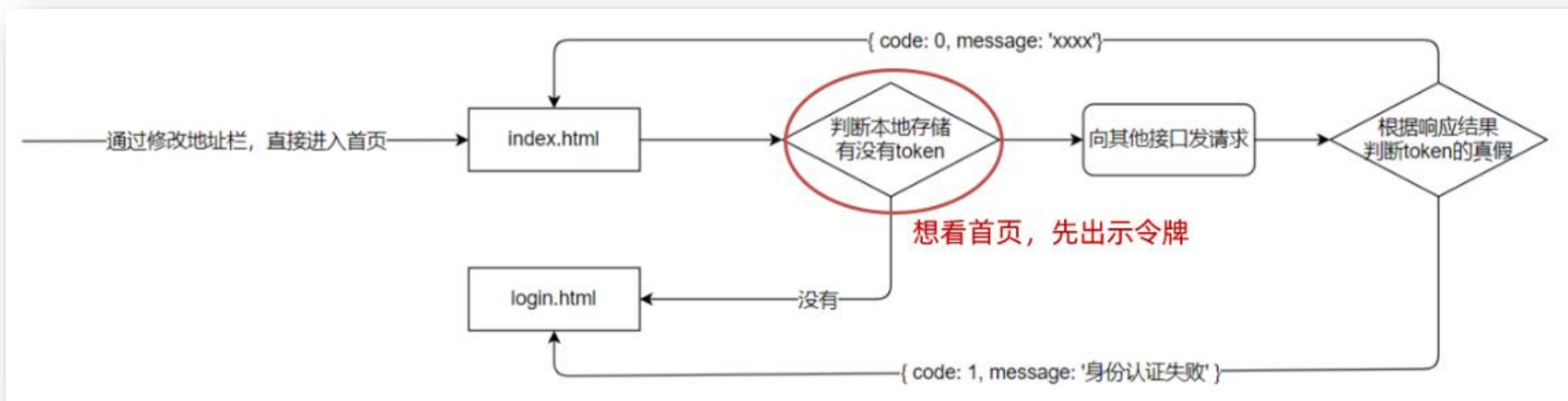
## token 的介绍

概念：访问权限的令牌，本质上是一串字符串

创建：正确登录后，由后端签发并返回

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwczovL3d3dy5pdGNhc3QuY24vIiwic3ViIjoiaWYxZThhNGEtYjNjOC00ZDVlTlIOWUtOGYzN2ZiZjU2YjcxIiwianRpIjoiaWUzZmM4NjUtY2FhYS00ZTY3LTg4MzAtZmZkMGZkNTg4NDczIiwiaWF0IjoxNjgxNjkxNjkxNjkxLjE4MzE2OTQyOTF9.jKv1GauwpY1zGw_co0nqeXB4SC8GrvUBwouuzPYVw84
```

作用：判断是否有登录状态等，控制访问权限



注意：前端只能判断 token 有无，而后端才能判断 token 的有效性

## token 的使用

目标：只有登录状态，才可以访问内容页面

步骤：

1. 在 utils/auth.js 中判断无 token 令牌字符串，则强制跳转到登录页（手动修改地址栏测试）
2. 在登录成功后，保存 token 令牌字符串到本地，再跳转到首页（手动修改地址栏测试）

```
const token = localStorage.getItem('token')
// 没有 token 令牌字符串，则强制跳转登录页
if (!token) {
  location.href = '../login/index.html'
}
```



# 总结

## 1. token 的作用?

- ✓ 判断用户是否有登录状态等

## 2. token 的注意:

- ✓ 前端只能判断 token 的有无
- ✓ 后端通过解密可以提取 token 字符串的原始信息，判断有效性

## 个人信息设置和 axios 请求拦截器

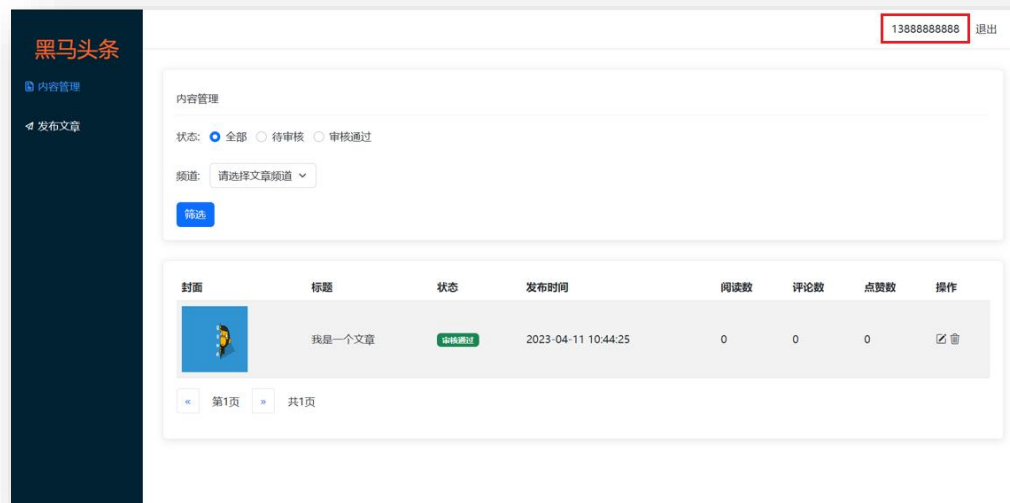
需求：设置用户昵称

语法：axios 可以在 headers 选项传递请求头参数

```
axios({  
  url: '目标资源地址',  
  headers: {  
    Authorization: `Bearer ${localStorage.getItem('token')}`  
  }  
})
```

问题：很多接口，都需要携带 token 令牌字符串

解决：在[请求拦截器](#)统一设置公共 headers 选项



### 获取-用户个人资料

GET /v1\_0/user/profile

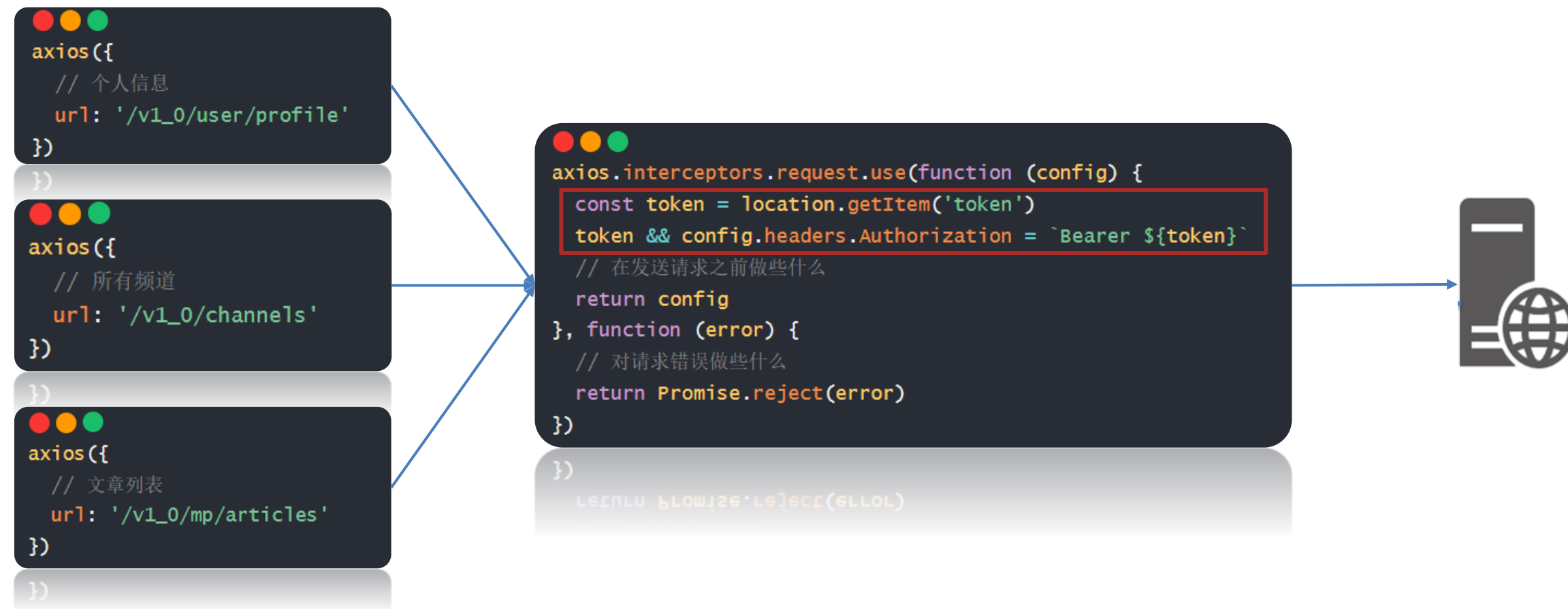
获取-用户个人资料

#### 请求参数

参数名	位置	类型	必填	说明
Authorization	header	string	是	<p>示例值: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwczovL3d3dy5pdGNhc3QuY24vliwiY290ZTY3LTg4MzAtZmZkMGZkNTg4NDczliiaWF0IjoxNjkwNjkwNjkwLjE4eHAiOiJlbnV1GauwpY1zGw_co0nqeXB4SC8GrvUBwouuzPYVw84</p> <p>格式: Bearer token值</p>

## 个人信息设置和 axios 请求拦截器

axios 请求拦截器：发起请求之前，触发的配置函数，对请求参数进行额外配置





# 总结

1. 什么是 axios 请求拦截器?
  - ✓ 发起请求之前，调用的一个函数，对请求参数进行设置
2. axios 请求拦截器，什么时候使用?
  - ✓ 有公共配置和设置时，统一设置在请求拦截器中

## axios 响应拦截器和身份验证失败

axios 响应拦截器：响应回到 then/catch 之前，触发的拦截函数，对响应结果统一处理

例如：身份验证失败，统一判断并做处理

```
axios({
  // 个人信息
  url: '/v1_0/user/profile'
}).then(result => {

}).catch(error => {

})
```

```
axios({
  // 文章列表
  url: '/v1_0/mp/articles'
}).then(result => {

}).catch(error => {

})
```

```
// 添加响应拦截器
axios.interceptors.response.use(function (response) {
  // 2xx 范围内的状态码都会触发该函数。
  // 对响应数据做点什么
  return response;
}, function (error) {
  // 超出 2xx 范围的状态码都会触发该函数。
  // 对响应错误做点什么，例如：判断响应状态为 401 代表身份验证失败
  if (error?.response?.status === 401) {
    alert('登录状态过期，请重新登录')
    localStorage.clear()
    location.href = '../login/index.html'
  }
  return Promise.reject(error);
});
```







# 总结

## 1. 什么是 axios 响应拦截器?

- ✓ 响应回到 then/catch 之前，触发的拦截函数，对响应结果统一处理

## 2. axios 响应拦截器，什么时候触发成功/失败的回调函数?

- ✓ 状态为 2xx 触发成功回调，其他则触发失败的回调函数

## 优化-axios 响应结果

目标：axios 直接接收服务器返回的响应结果

```
axios({
  // 个人信息
  url: '/v1_0/user/profile'
}).then(result => {
  // result: 服务器响应数据对象
}).catch(error => {
})
```

```
axios({
  // 文章列表
  url: '/v1_0/mp/articles'
}).then(result => {
  // result: 服务器响应数据对象
}).catch(error => {
})
```

```
// 添加响应拦截器
axios.interceptors.response.use(function (response) {
  // 2xx 范围内的状态码都会触发该函数。
  // 对响应数据做点什么，例如：提取服务器返回的数据对象，返回到发起请求的位置
  const result = response.data
  return result;
}, function (error) {
  // 超出 2xx 范围的状态码都会触发该函数。
  // 对响应错误做点什么，例如：判断响应状态为 401 代表身份验证失败
  if (error?.response?.status === 401) {
    alert('登录状态过期，请重新登录')
    localStorage.clear()
    location.href = '../login/index.html'
  }
  return Promise.reject(error);
});
```

```
{data: {...}, status: 200, statusText: 'OK', headers: i, config: {...}, ...}
  config: {transitional: {...}, adapter: Array(2), transformRequest: Array(1),
  data:
    data:
      birthday: "1990-11-20"
      gender: 1
      id: "f521c589-53e5-44f2-95b3-de00ad92b599"
      mobile: "13888888888"
      name: "13888888888"
      photo: "http://geek.itheima.net/images/user_head.jpg"
      [[Prototype]]: Object
    [[Prototype]]: Object
  headers: i {content-length: '203', content-type: 'application/json; charse
  request: XMLHttpRequest {onreadystatechange: null, readyState: 4, timeout:
  status: 200
  statusText: "OK"
  [[Prototype]]: Object
```

## 发布文章-富文本编辑器

富文本：带样式，多格式的文本，在前端一般使用标签配合内联样式实现

富文本编辑器：用于编写富文本内容的容器



```
<p><span style="color: rgb(225, 60, 57);"><u><em><strong>我是富文本：多格式多类型的文本内容</strong></em></u></span></p><p>特点：</p><ul><li>方便，快捷</li><li>开箱即用</li></ul><p>例如：<a href="https://www.wangeditor.com/" target="_blank">wangEditor</a></p><p><br></p><table style="width: auto;"><tbody><tr><th colSpan="1" rowSpan="1" width="auto">我</th><th colSpan="1" rowSpan="1" width="auto">是</th><th colSpan="1" rowSpan="1" width="auto">表</th><th colSpan="1" rowSpan="1" width="auto">格</th></tr></tbody></table><pre><code>const { createEditor, createToolbar } = window.wangEditor</code></pre><p><br></p>
```

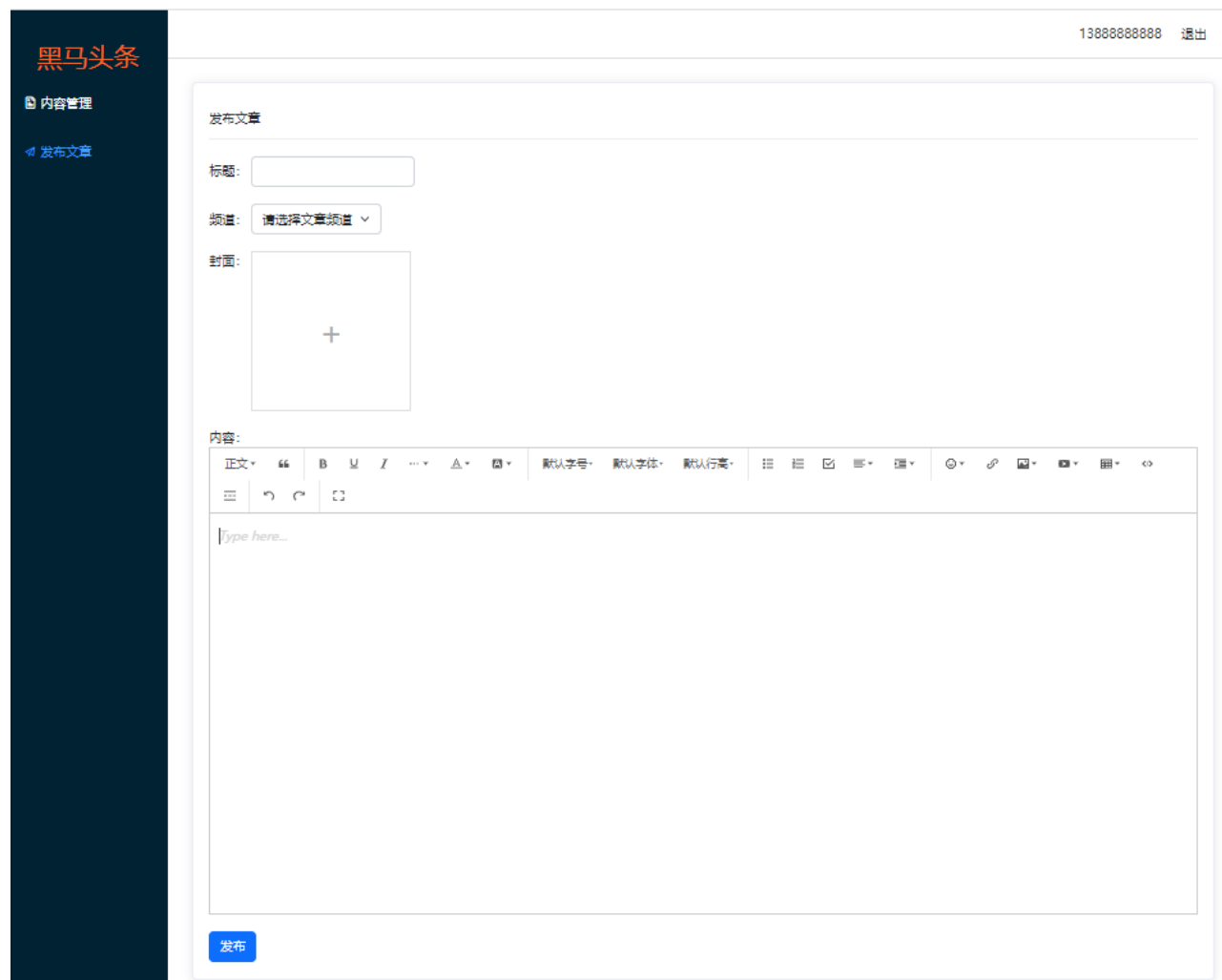
## 发布文章-富文本编辑器

目标：发布文章页，富文本编辑器的集成

使用：wangEditor 插件

[步骤](#)：参考文档

1. 引入 CSS 定义样式
2. 定义 HTML 结构
3. 引入 JS 创建编辑器
4. 监听内容改变，保存在隐藏文本域（便于后期收集）



黑马头条

内容管理

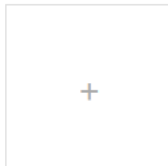
发布文章

13888888888 退出


发布文章

标题:

频道:

封面: 

内容:

正文 

Type here...

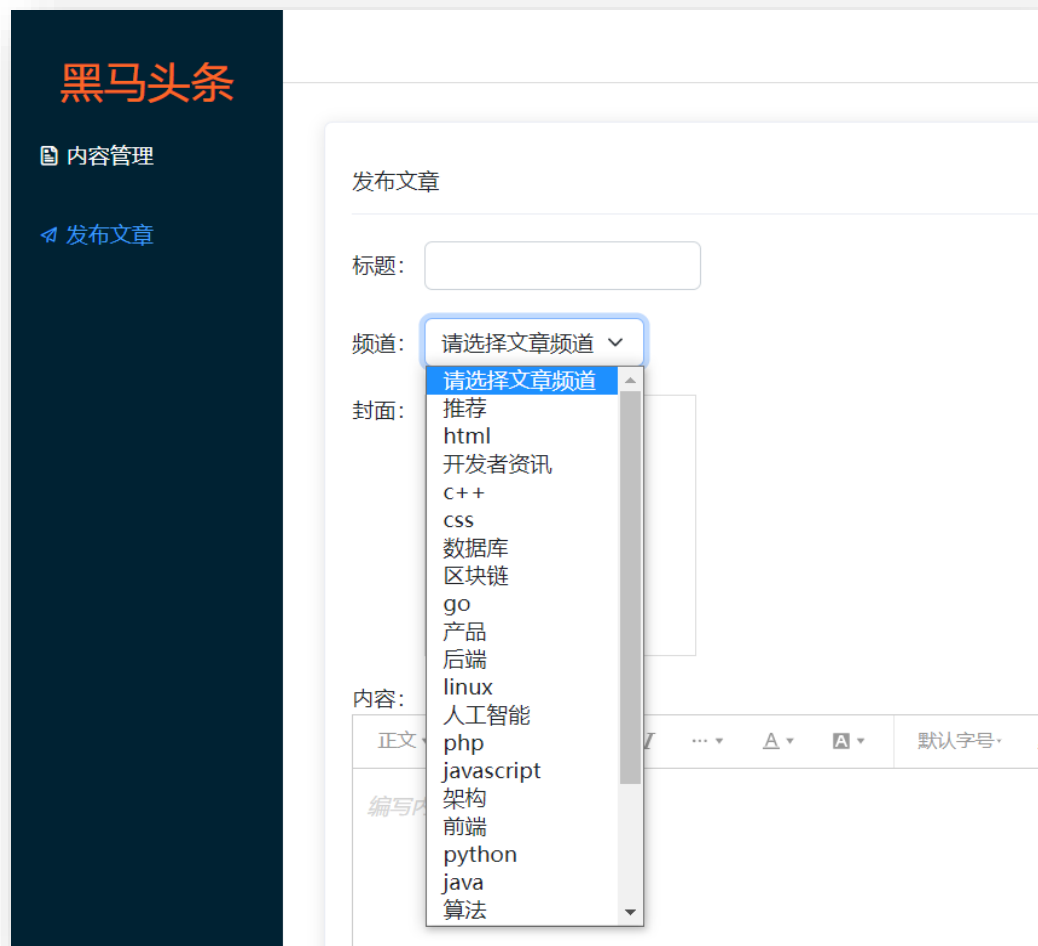
发布

## 发布文章-频道列表

目标：展示频道列表，供用户选择

步骤：

1. 获取频道列表数据
2. 展示到下拉菜单中



## 发布文章-封面设置

目标：文章封面的设置

步骤：

1. 准备标签结构和样式
2. 选择文件并保存在 FormData
3. 单独上传图片并得到图片 URL 地址
4. 回显并切换 img 标签展示（隐藏 + 号上传标签）

注意：图片地址临时存储在 img 标签上，并未和文章关联保存



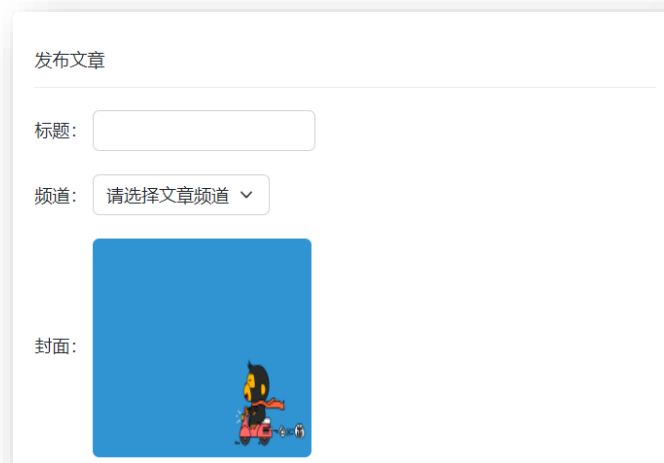
发布文章

标题:

频道: 请选择文章频道 ▾

封面: 

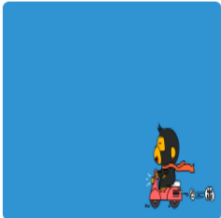
+



发布文章

标题:

频道: 请选择文章频道 ▾

封面: 

## 发布文章-收集并保存

目标：收集文章内容，并提交保存

步骤：

1. 基于 form-serialize 插件**收集**表单数据对象
2. 基于 axios 提交到服务器**保存**
3. 调用 Alert 警告框**反馈**结果给用户
4. **重置表单**并跳转到列表页

文章发布成功

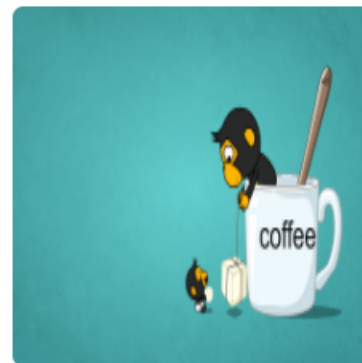
文章标题和文章内容不能为空

发布文章

标题：

频道：

封面：



内容：

正文 ▾

“

B

U

I

...

A ▾

A ▾

默认字号 ▾

默认字体 ▾

学习 css 要做好**基本功**，并多加练习，加油!!!

## 内容管理-文章列表展示

目标：获取文章列表并展示

步骤：

1. 准备查询**参数对象**
2. **获取**文章列表数据
3. **展示**到指定的标签结构中

```
const queryObj = {  
  status: '', // 筛选状态  
  channel_id: '', // 频道id  
  page: 1, // 当前页码  
  per_page: 2 // 每页条数  
}
```

### 黑马头条



- 内容管理
- 发布文章

内容管理

状态: ☒ 全部 ☐ 待审核 ☐ 审核通过

频道: 请选择文章频道

筛选

封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	聊聊 ChatGPT	审核通过	2023-04-23 16:43:29	0	0	0	<a href="#">编辑</a> <a href="#">删除</a>
	如何学习 MySQL	审核通过	2023-04-23 16:42:56	0	0	0	<a href="#">编辑</a> <a href="#">删除</a>

[<<](#) 第1页 [>>](#) 共2页

### 获取-文章列表

**GET** /v1\_0/mp/articles

获取-文章列表

#### 请求参数

Query 参数

**status** string 可选  
文章状态: 1-待审核, 2-审核通过, 不传为全部

**channel\_id** string 可选  
频道id, 不传为全部

**page** string 可选  
当前页码

**per\_page** string 可选  
每页条数



## 内容管理-筛选功能

目标：根据筛选条件，获取匹配数据展示

步骤：



1. 设置频道列表数据
2. 监听筛选条件改变，保存查询信息到**查询参数对象**
3. 点击筛选时，传递**查询参数对象**到服务器
4. 获取匹配数据，覆盖到页面展示

内容管理

状态: ☒ 全部 ☐ 待审核 ☐ 审核通过

频道: 请选择文章频道 ▾

筛选

封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	什么是数据库?	审核通过	2023-04-24 11:08:53	0	0	0	<a href="#">✎</a> <a href="#">🗑</a>
	C++ 好学吗?	审核通过	2023-04-24 11:08:35	0	0	0	<a href="#">✎</a> <a href="#">🗑</a>

[<](#) 第1页 [>](#) 共3页

## 内容管理-分页功能

目标：完成文章列表，分页管理功能

步骤：

1. 保存并设置文章**总条数**
2. 点击下一页，做**临界值判断**，并**切换页码参数**请求最新数据
3. 点击上一页，做临界值判断，并切换页码参数**请求最新数据**



```
const queryObj = {  
  status: '', // 筛选状态  
  channel_id: '', // 频道id  
  page: 1, // 当前页码  
  per_page: 2 // 每页条数  
}  
let totalCount = 0 // 总条数
```

内容管理

状态: ☒ 全部 ☐ 待审核 ☐ 审核通过

频道: 请选择文章频道 ▾

筛选

封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	什么是数据库?	审核通过	2023-04-24 11:08:53	0	0	0	☑ ☒
	C++ 好学吗?	审核通过	2023-04-24 11:08:35	0	0	0	☑ ☒





◀ 第1页 ▶ 共3页

## 内容管理-删除功能

目标：完成删除文章功能

步骤：

1. 关联文章 id 到删除图标
2. 点击删除时，获取文章 id
3. 调用删除接口，传递文章 id 到服务器
4. 重新获取文章列表，并覆盖展示

内容管理							
状态: <input checked="" type="radio"/> 全部 <input type="radio"/> 待审核 <input type="radio"/> 审核通过							
频道: 请选择文章频道 ▾							
筛选							
封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	数据库也非常重要	审核通过	2023-04-27 16:08:17	0	0	0	<input checked="" type="checkbox"/> 
	学好 JavaScript	审核通过	2023-04-27 16:07:50	0	0	0	<input checked="" type="checkbox"/> 
« 第1页 » 共3页							

## 内容管理-删除最后一条

目标：在删除最后一页，最后一条时有 Bug

解决：

1. 删除成功时，判断 DOM 元素只剩一条，让当前页码 page--
2. 注意，当前页码为 1 时不能继续向前翻页
3. 重新设置页码数，获取最新列表展示

内容管理

状态: ☒ 全部 ☐ 待审核 ☐ 审核通过

频道: 请选择文章频道

筛选

封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	如何学习 html	审核通过	2023-04-27 16:06:46	0	0	0	<a href="#">编辑</a> <a href="#">删除</a>

[<](#) 第3页 [>](#) 共3页

## 内容管理-编辑文章-回显

目标：编辑文章时，回显数据到表单

步骤：

1. 页面跳转传参 (URL 查询参数方式)
2. 发布文章页面接收参数判断 (共用同一套表单)
3. 修改标题和按钮文字
4. 获取文章详情数据并回显表单

/page/publish/index.html?id=9d07ed29-83dc-4f18-b3c1-fb33b7f84716

```
const params = `?id=1001&name=xiaoli`  
// 查询参数字符串 => 查询参数对象  
const result = new URLSearchParams(params)  
// 需要遍历使用  
result.forEach((value, key) => {  
  console.log(value, key)  
  // 1001 id  
  // xiaoli name  
})
```

封面	标题	状态	发布时间	阅读数	评论数	点赞数	操作
	如何学习 JavaScript	审核通过	2023-04-28 11:14:17	0	0	0	<a href="#">✎</a> <a href="#">🗑</a>
	为何要学习前端	审核通过	2023-04-28 11:07:44	0	0	0	<a href="#">✎</a> <a href="#">🗑</a>

« 第1页 » 共 2 条

修改文章

标题:

频道:

封面:  


内容:  

正文     ...   默认字号· 默认字体· 默认行高·

如何学习 JavaScript

修改

## 内容管理-编辑文章-保存

目标：确认修改，保存文章到服务器

步骤：

1. **判断**按钮文字，区分业务（因为共用一套表单）
2. 调用编辑文章接口，保存信息到服务器
3. 基于 Alert 反馈结果消息给用户

文章发布成功

文章标题和文章内容不能为空

修改文章

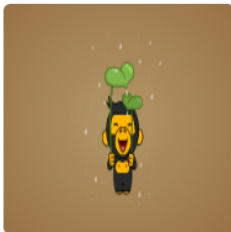
标题：

如何学习 JavaScript

频道：

javascript

封面：



内容：

正文

B

U

I

...

A

A

默认字号

默认字体

默认行高

如何学习 JavaScript

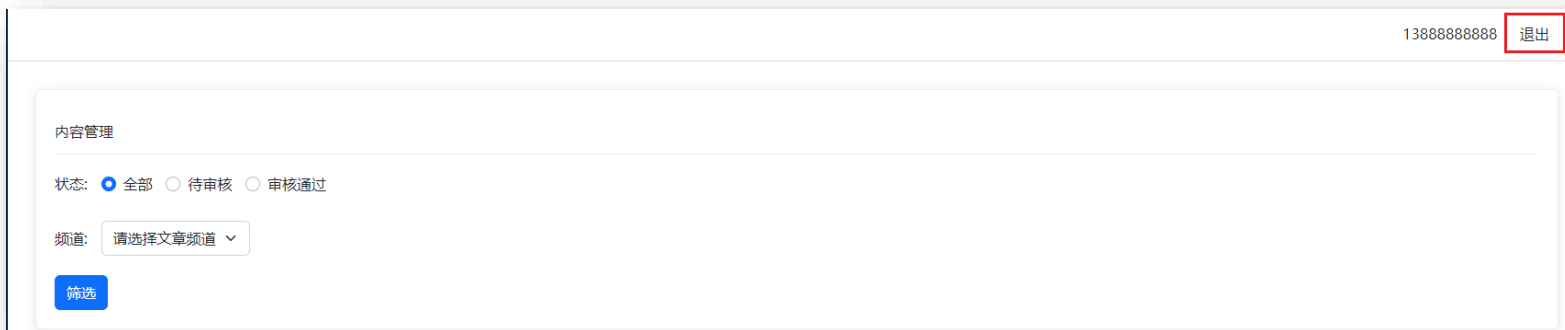
修改

## 退出登录

目标：完成退出登录效果

步骤：

1. 绑定点击事件
2. 清空本地缓存，跳转到登录页面





传智教育旗下高端IT教育品牌