Mint UI - Unit03

1.学子问答 -- 首页实践

1.1 数据表结构

数据库名称：`xzqa`

编码方式：`utf8`

1.1.1 xzqa_category表

`xzqa_category` 表用于存储文章分类，其结构如下：

字段名称	数据类型	默认值	是否为空	键	描述
`id`	`SMALLINT UNSIGNED`		`NO`	`PRI AUTO_INCREMENT`	分类 `ID`,主键且唯一
`category_name`	`VARCHAR(30)`		`NO`	`UNIQUE KEY`	分类名称

1.1.2 xzqa_author表

`xzqa_author` 表用于存储作者的相关信息，其结构如下：

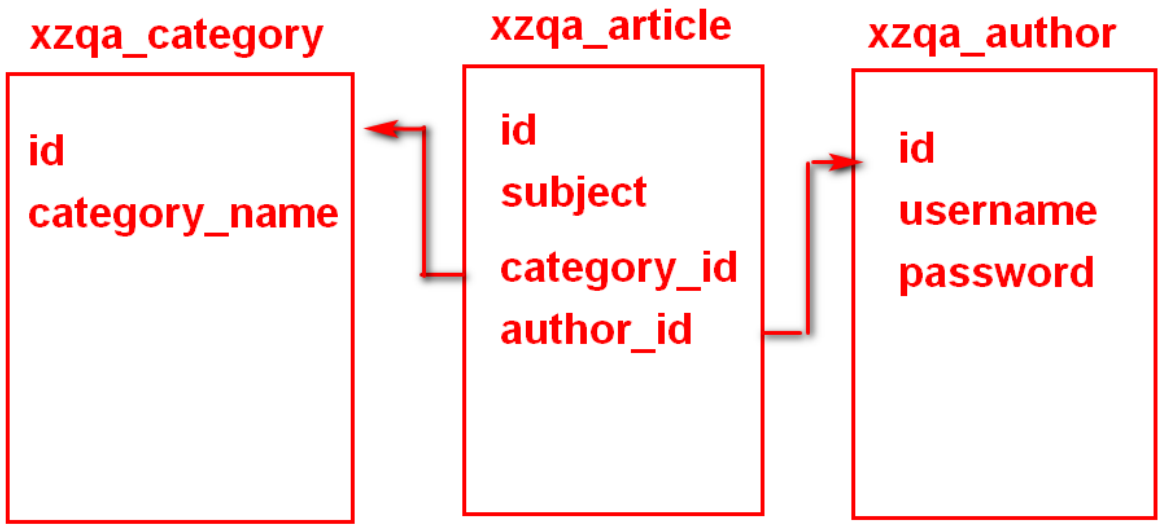
字段名称	数据类型	是否为空	默认值	键	描述
`id`	`MEDIUMINT UNSIGNED`	`NO`		`PRI AUTO_INCREMENT`	作者 `ID`，主键且唯一
`username`	`VARCHAR(30)`	`NO`		`UNIQUE KEY`	用户名，唯一
`password`	`VARCHAR(32)`	`NO`			密码，`MD5` 加密
`nickname`	`VARCHAR(30)`	`YES`	`NULL`		用户昵称
`avatar`	`VARCHAR(50)`	`NO`	`unnamed.jpg`		用户头像
`article_number`	`MEDIUMINT UNSIGNED`	`NO`	`0`		发表的文章数量

1.1.3 xzqa_article表

`xzqa_article` 表用于存储文章表，其结构如下：

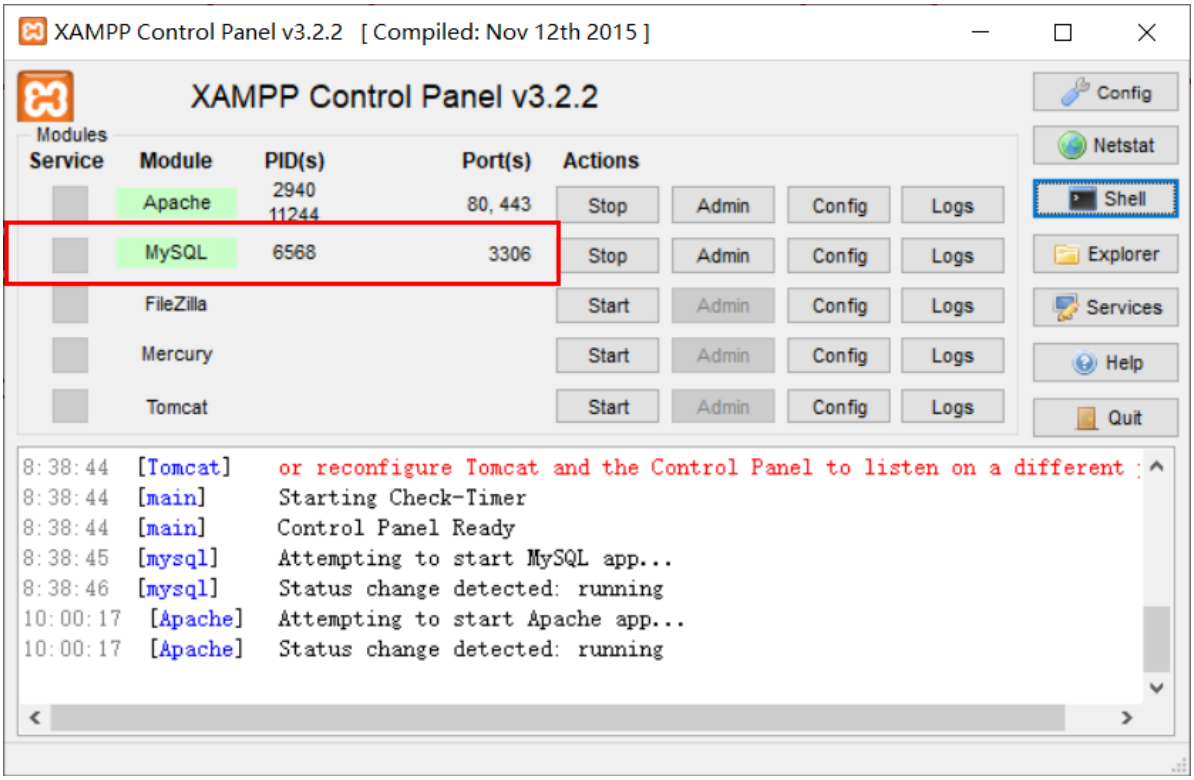
字段名称	数据类型	是否为空	默认值	键	描述
id	INT UNSIGNED	NO		PRI AUTO_INCREMENT	文章ID，主键且自增
subject	VARCHAR(50)	NO			文章标题
description	VARCHAR(255)	NO			文章简介
content	MEDIUMTEXT	NO			文章正文
image	VARCHAR(50)	YES	NULL		文章缩略图
category_id	SMALLINT UNSIGNED	NO			外键，参照xzqa_category表的ID
author_id	INT UNSIGNED	NO			外键，参照xzqa_author表的id
created_at	INT UNSIGNED	NO			文章的发表日期

ER图

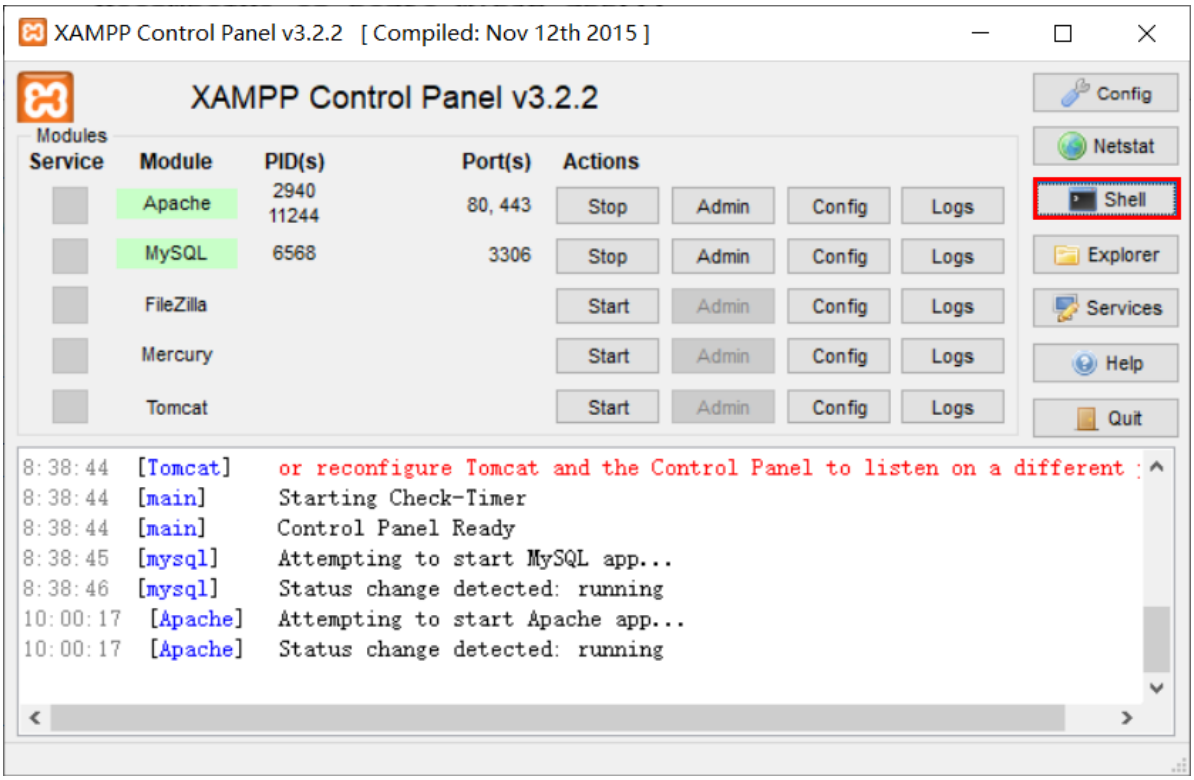


1.1.4 数据导入

A.启动XAMPP中的MySQL



B.单击`Shell`按钮



C.输入以下命令并且回车确认：

```
mysql -uroot -p < SQL脚本文件的位置及名称
```

1.2 首页顶部选项卡动态数据的实现

首页顶部选项卡动态数据应该来源于`xzqa_category`(文章分类)表。也就代表需要向`WEB`服务器发送请求后，获取到该表中的数据，最后在页面中进行显示即可。此时遇到的问题有：

> 1.请问在什么时候发送`HTTP`请求以获取数据呢?
>
> 2.向服务器的哪一个接口发送请求呢?
>
> 3.如何在服务器获取数据呢?
>
> 4.在客户端如何接收并且显示服务器返回的数据呢?

1.请问在什么时候发送`HTTP`请求以获取数据呢?

在什么情况下发送`HTTP`请求以获取服务器的数据？此时引申出`Vue.js`的生命周期的回调函数：

> `beforeCreate`
>
> `created`
>
> `beforeMount`
>
> `mounted`
>
> `beforeUpdate`
>
> `updated`
>
> `beforeDestroy`
>
> `destroyed`

在本案例中选择`mounted`时候来发送`HTTP`请求以获取服务器数据，所以示例代码如下：

```
<script>
    export default{
        mounted(){
            // 此时需要发送HTTP请求以获取服务器数据
        }
    }
</script>
```

现在要发送请求，必须要通过`XHR(XMLHttpRequest)`对象，所以需要`axios`库来实现。故：

第一步：安装`axios`

```
npm install --save axios
```

第二步：配置

配置在`main.js`中完成，示例代码如下：

```
import axios from 'axios';
//此时已经代表WEB服务器的端口已经确定为3000!
axios.defaults.baseURL = 'http://127.0.0.1:3000';
Vue.prototype.axios = axios;
```

此时代表发送`HTTP`请求的工具 -- `axios`已经安装并且配置完毕，可以发送请求了，所以`Index.vue`中的`mounted`的代码修改为如下结构：

```
mounted(){
    // 此时需要发送HTTP请求以获取服务器数据
    //this.axios.请求方式(URL地址,参数).then(res=>{

    //});
}
```

在上述代码必须要确定请求方式、`URL`地址及请求参数才可以真正来发送请求。

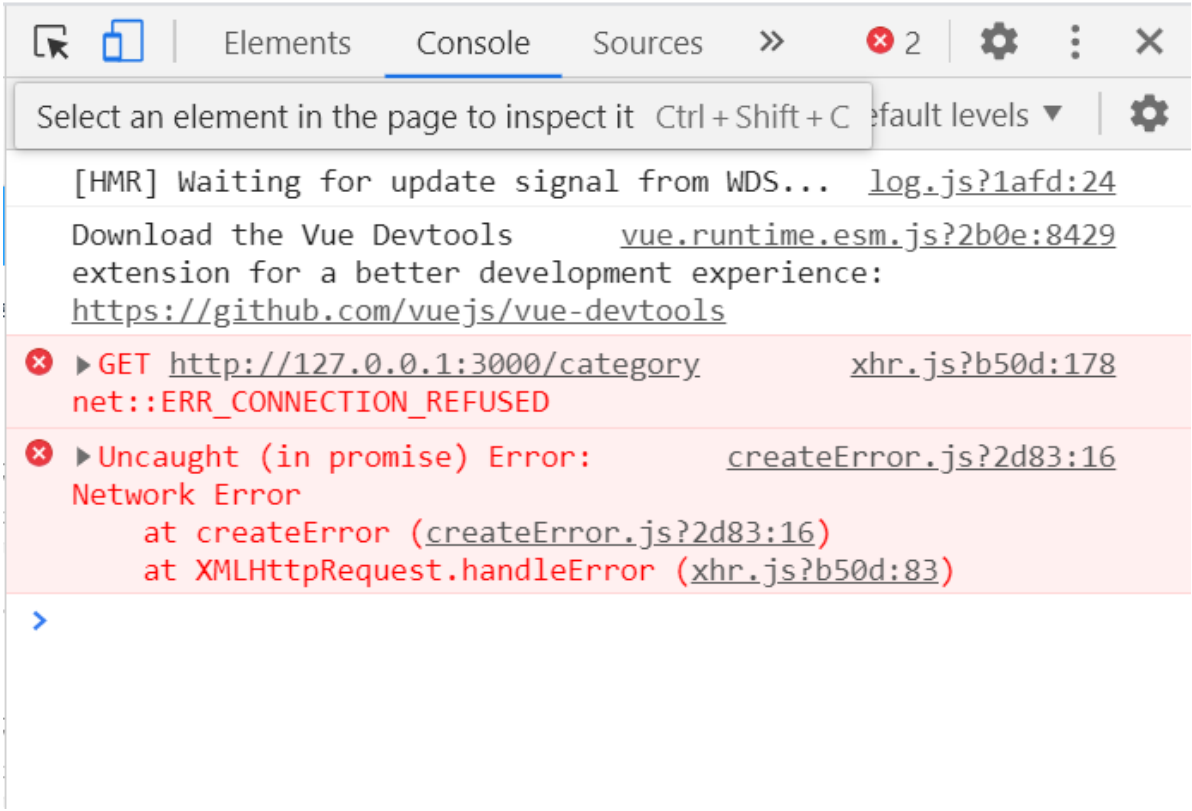
因为现在是获取服务器的数据，所以请求方式一般情况为`GET`，故上述代码需要修改为：

```
mounted(){
    // 此时需要发送HTTP请求以获取服务器数据
    //this.axios.get(URL地址,参数).then(res=>{

    //});
}
```

假设请求的`URL`地址为`http://127.0.0.1:3000/category`，且无参数，所以上述代码可以修改为：

```
mounted(){
    // 此时需要发送HTTP请求以获取服务器数据
    this.axios.get('/category').then(res=>{

    });
}
```

此时脚手架的运行结果如下图所示：



错误的根本原因是：根本没有服务器，所以也就是没有接口！！！

2.安装并配置Node.js HTTP服务器

既然客户端向服务器发送HTTP请求，也就代表服务器必须要提供HTTP服务，而在Node.js环境中实现HTTP服务的方式有两种：

A. Node.js的HTTP模块

B. Express框架

安装Express

```
npm install --save express
```

配置Express

```
// 加载Express模块
const express = require('express');

// 创建WEB服务器
const server = express();

// 指定 WEB服务器监听的端口
server.listen(3000);
```

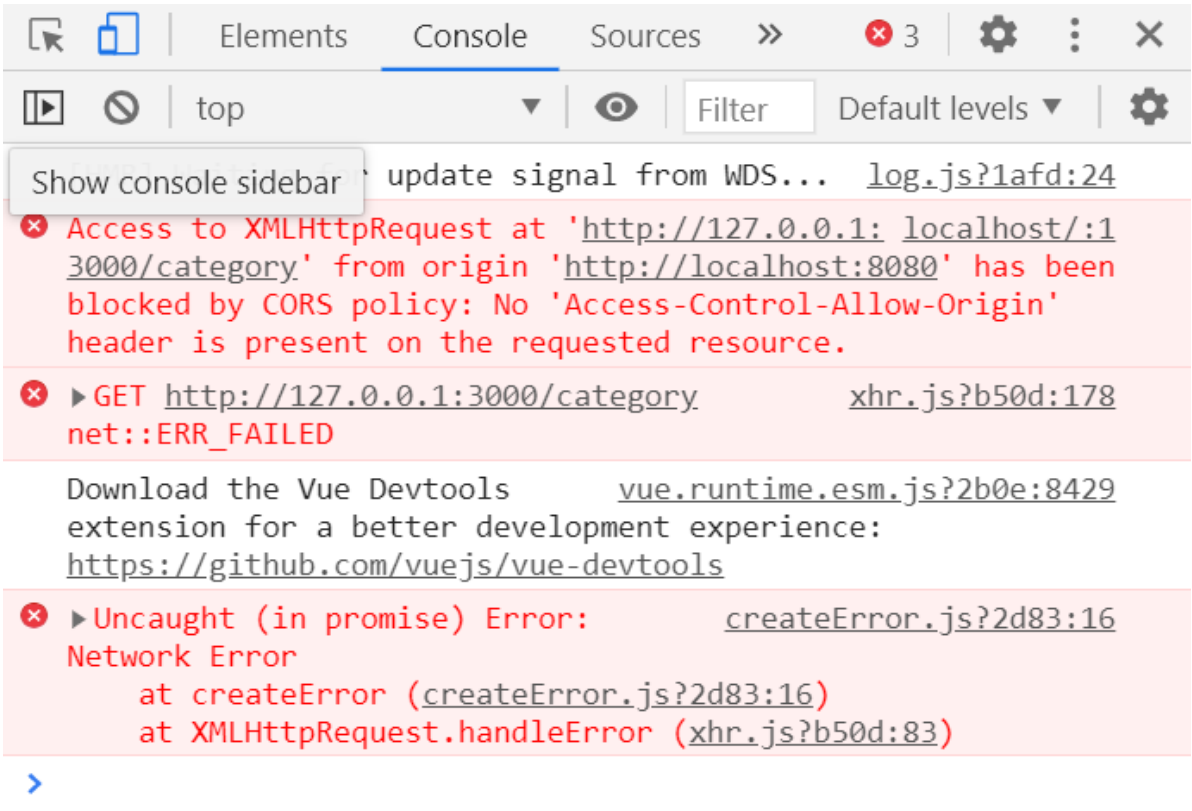
创建名称为/category的GET请求方式的接口，示例代码如下：

```
// 获取所有的文章分类
server.get('/categorty',(req,res)=>{
  res.send('ok');
});
```

启动`Node.js`服务器，在终端输入以下命令：

```
node app.js
```

此时客户端的运行结果如下图所示：



出现错误的原因是：跨域错误（因为服务器地址与端口都不相同）

跨域的解决方案有：

A. `JSONP`

B. `CORS`

现在使用`CORS`来解决跨域错误，所以：

第一步：安装

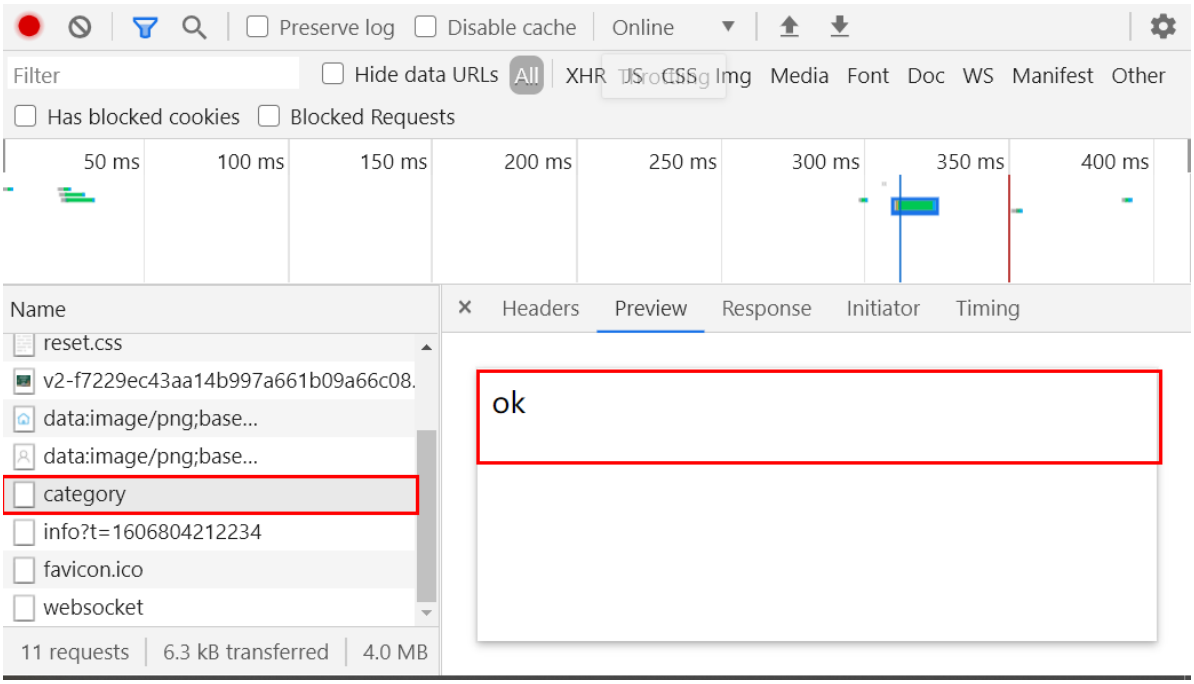
```
npm install --save cors
```

第二步：配置

```
// 加载CORS模块
const cors = require('cors');
// 为所有的HTTP请求使用CORS模块
server.use(cors({
  origin:['http://127.0.0.1:8080','http://localhost:8080']
}))
```

第三步：重新启动服务器

此时脚手架的运行结果如下图所示：



3.在WEB服务器接口中获取数据库的数据

要在WEB服务器中获取数据库的数据需要MySQL模块的支持才可以，所以：

A.安装

```
npm install --save mysql
```

B.配置

```
// 加载MySQL模块
const mysql = require('mysql');
```

C.MySQL连接池

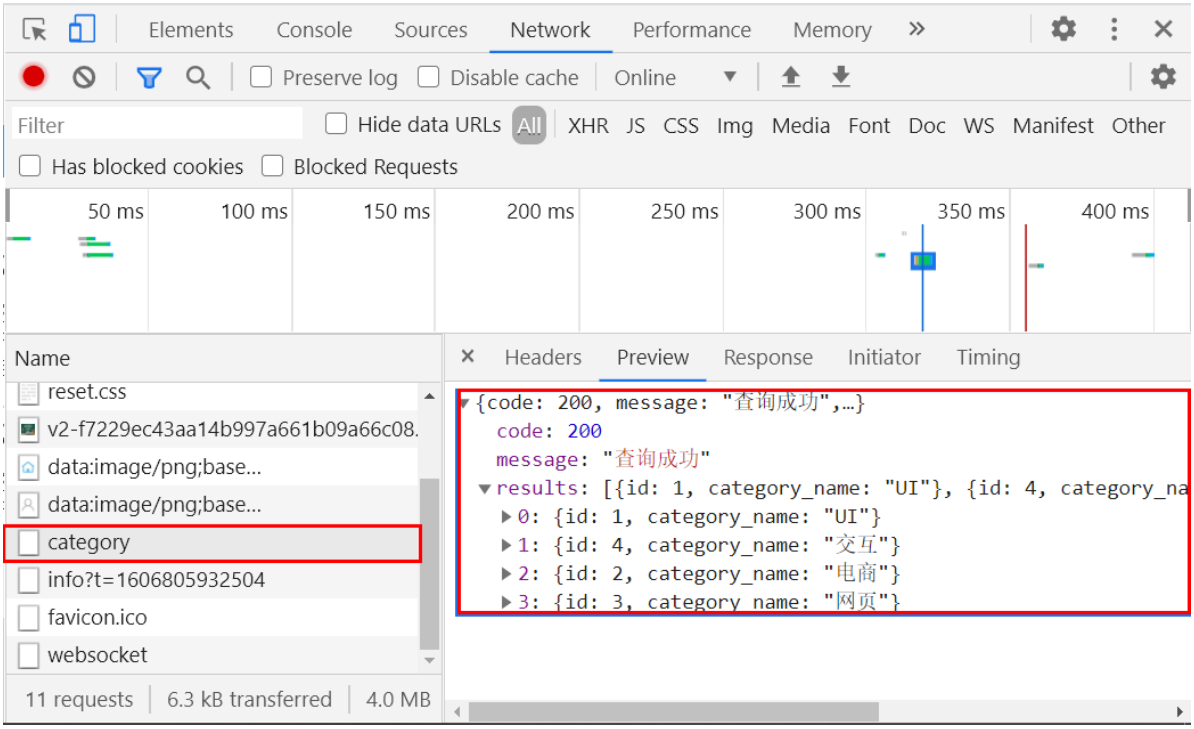
```
//创建MySQL连接池
const pool = mysql.createPool({
    // 数据库服务器的地址
```

```
    host:'127.0.0.1',
    // 数据库服务器的端口号
    port:3306,
    // 数据库用户的用户名
    user:'root',
    // 数据库用户的密码
    password:'',
    // 数据库名称
    database:'xzqa',
    // 最大连接数
    connectionLimit:15
});
```

虽然现在`MySQL`的已经安装并且配置成功了，但是在服务器的`/category`接口中并没有真正从数据库获到到数据，所以，需要重新修改`/category`接口，示例代码如下：

```
// 获取所有的文章分类
server.get('/category',(req,res)=>{
    // 查询xzqa_category数据表的全部记录
    let sql = 'SELECT id,category_name FROM xzqa_category';
    // 执行SQL查询
    pool.query(sql,(error,results)=>{
      if(error) throw error;
      res.send({code:200,message:'查询成功',results:results});
    });
});
```

此时脚手架的运行结果如下图所示：



4.在脚手架接收并且显示数据

需要在`Index.vue`的`mounted`中接收数据，示例代码如下：

```
mounted(){
    // 此时需要发送HTTP请求以获取服务器数据
    this.axios.get('/category').then(res=>{
        let results = res.data.results;
    });
}
```

在 `mounted` 回调函数中接收到数据，但还需要将数据存储到 `data()` 中去，只有存储到 `data()` 中去后，才可以在页面中通过 `v-for` 指令进行循环输出，所以：

第一步:在 `data()` 中声明变量，示例代码如下:

```
data(){
    return {
      // 存储所有的文章分类数据
      category:[]
    }
}
```

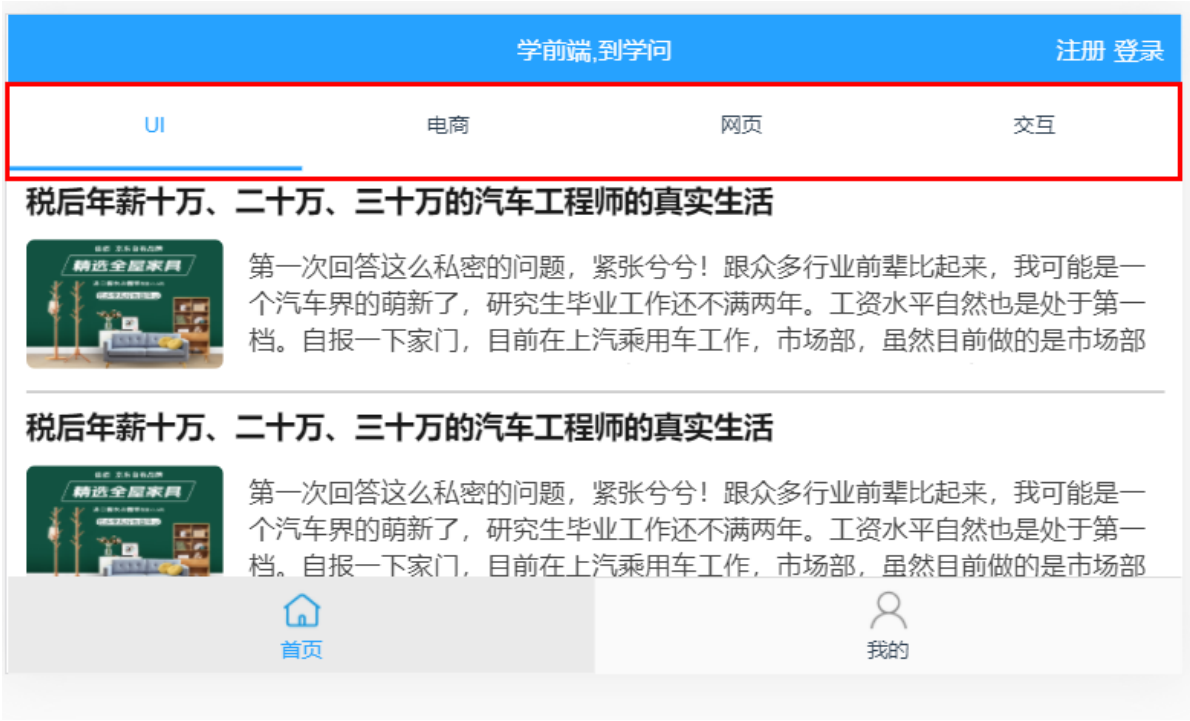
第二步:在 `mounted` 的回调函数中，将服务器返回的结果存储到 `data()` 变量中，示例代码如下:

```
mounted(){
    // 此时需要发送HTTP请求以获取服务器数据
    this.axios.get('/category').then(res=>{
        // 获取服务器返回的结果
        let results = res.data.results;
        // 将服务器返回结果存储到category变量中
        this.category = results;
    });
}
```

第三步:在顶部选项卡中通过 `v-for` 指令进行循环输出：

```
<!-- 顶部选项卡开始 -->
<mt-navbar v-model="active">
    <mt-tab-item
        :id="item.id.toString()"
        v-for="(item,index) of category"
        :key="index">
        {{item.category_name}}
    </mt-tab-item>
</mt-navbar>
<!-- 顶部选项卡结束 -->
```

此时脚手架的运行结果如下图所示：



现在顶部选项卡已经可以正常显示了，但是页面中的数据仍然是静态数据。

所以...

2.首页初始化数据的动态获取