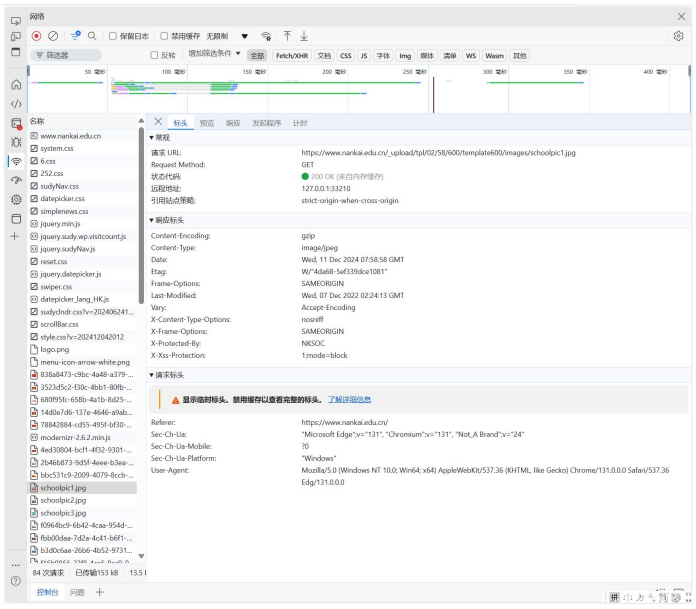
个人作业 1—Web 前端初探

1.针对任意网页，调研其不同方式请求，至少包括 get、post 请求，写出或截图其请求及相应数据包

GET 请求

调研网页：南开大学官网南开大学

找到一个通过 GET 请求获取一张图片，如下：



这个截图展示的是一个 **GET 请求**的详细信息：

请求 URL：
请求的目标资源是 `https://www.nankai.edu.cn/.../schoolpic1.jpg`，即一张 JPEG 图片。

请求方法：
`GET`，用于获取资源。

状态码：
`200 OK`，表示请求成功，服务器返回了图片。

响应内容：

Content-Type： `image/jpeg`，说明返回的是 JPEG 格式的图片。

Content-Encoding： `gzip`，表示数据经过了压缩。

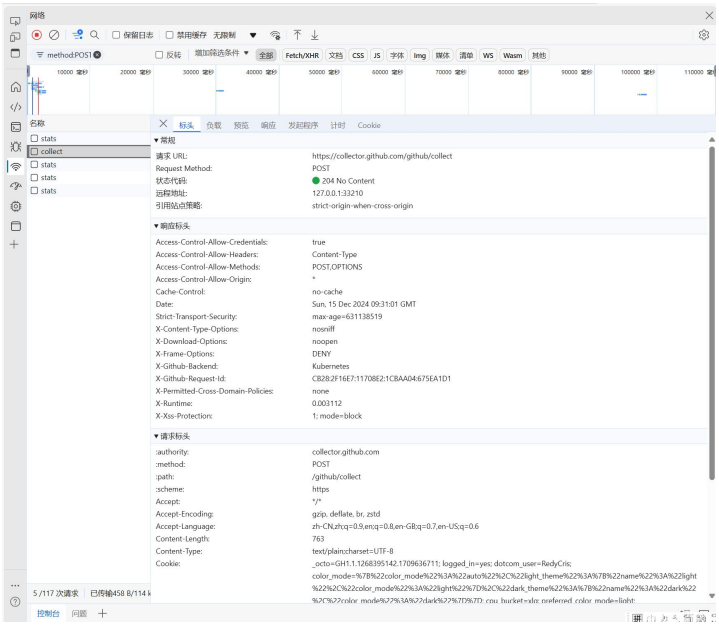
Last-Modified 和 **Etag：**提供缓存验证信息，帮助浏览器判断资源是否更新。

请求头：
浏览器通过 `User-Agent` 等头部信息向服务器标识自身，并附带了 `Referer` 表示请求来源页面。

这个 GET 请求成功获取了一张图片资源，服务器返回了压缩后的 JPEG 数据。

POST 请求

调研网页：https://github.com/



请求 URL：
`https://collector.github.com/github/collect` 表示向 GitHub 的收集接口发送数据。

请求方法：
`POST`，表示客户端向服务器提交数据。

状态码：
`204 No Content`，表示请求成功，但服务器没有返回任何内容。

请求头信息：

Content-Type： `text/plain;charset=UTF-8` 提交的数据是纯文本，编码为 UTF-8。

Content-Length： 提交数据的大小是 763 字节。

Accept-Encoding： `gzip, deflate, br, zstd` 表示客户端支持多种压缩格式。

Cookie： 包含用户的身份和会话信息。

响应头信息：

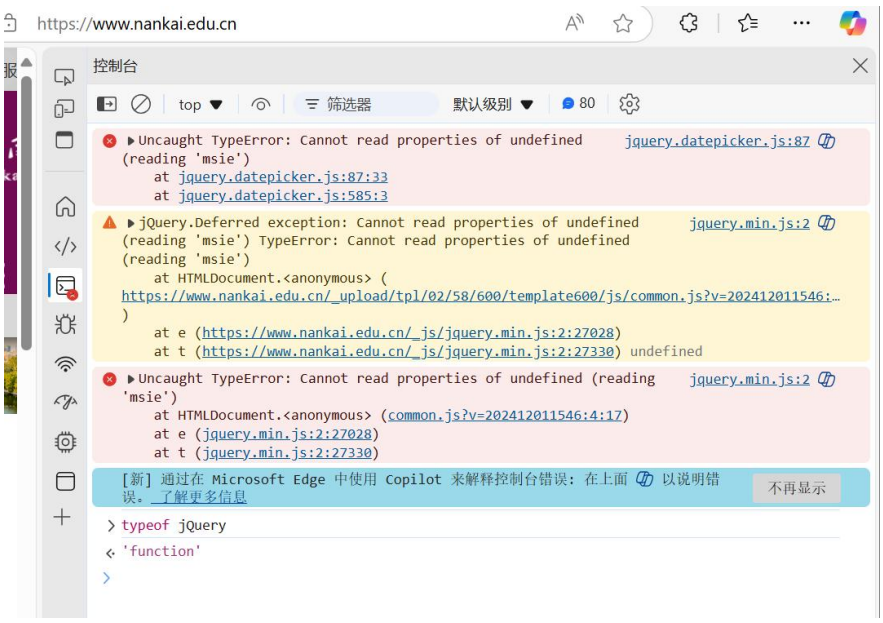
Access-Control-Allow-Methods： `POST, OPTIONS` 服务器允许的请求方法。

Cache-Control： `no-cache` 表示不缓存此次请求的响应结果。

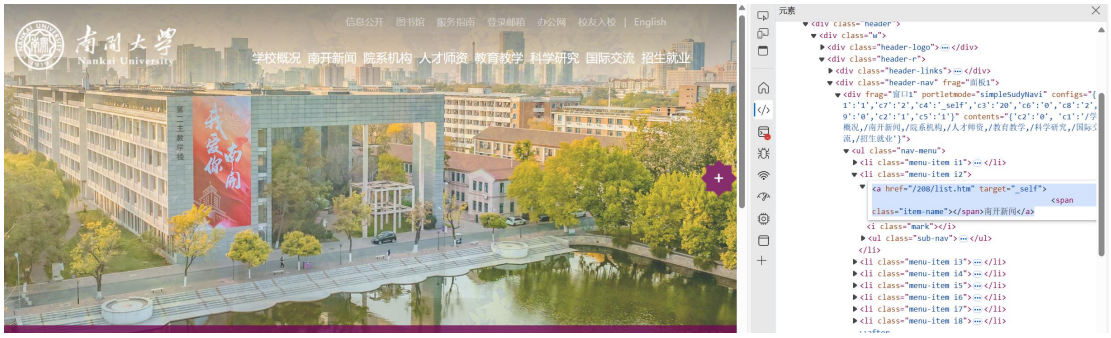
X-Frame-Options： `DENY` 禁止该页面被嵌入到其他页面中，增强安全性。

这个 **POST 请求** 向 GitHub 的服务器提交了一些数据，服务器响应了 **204 No Content**，表示请求成功但无返回内容。请求头和响应头包含了与安全、缓存和数据传输相关的信息。

2. 针对任意网页，使用 Jquery，能够出发某一事件，写出至少三条语句，截图响应前后不同的状态



首先检查网页是否引入了 jQuery



找到南开新闻的字体

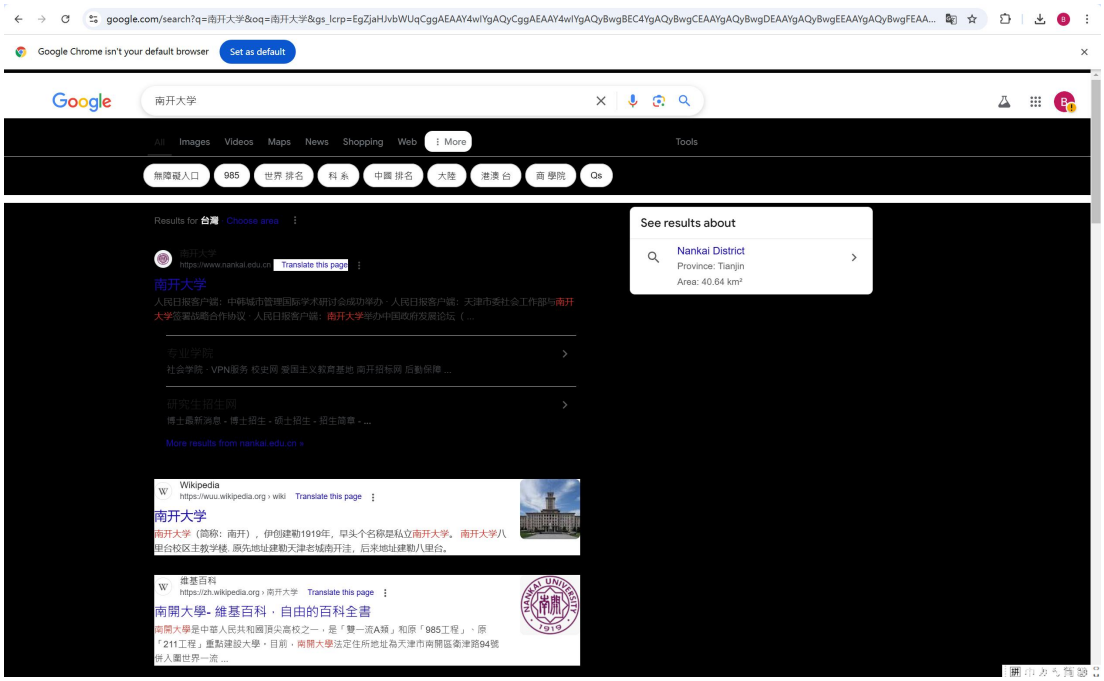
我们为这个加一个红色边框

将 ″南开新闻″ 修改为 ″最新新闻″：

为链接绑定一个点击事件，点击时弹出提示框，并直接用 jQuery 触发点击事件

3. 完成一个浏览器插件，功能不限，文档中写明功能及代码

插件功能：**将网页背景变成黑暗模式**



插件目录结构

```
dark-mode-plugin/
├── manifest.json        // 插件配置文件
├── background.js        // 后台脚本
└── content.js           // 注入的内容脚本
```

---

1. manifest.json

```
{
  "manifest_version": 3,
  "name": "Dark Mode Plugin",
  "version": "1.0",
  "description": "切换网页为暗模式。",
  "permissions": ["activeTab"],
  "background": {
    "service_worker": "background.js"
  },
  "content_scripts": [
    {
      "matches": ["<all_urls>"],
      "js": ["content.js"]
    }
  ]}
```

2. background.js

后台脚本可以用于监听插件的安装事件：

```
chrome.runtime.onInstalled.addListener(() => {
    console.log("Dark Mode Plugin 已安装！");
});
```

3. content.js

content.js 中实现暗模式的功能逻辑，简单设置网页的背景和文字颜色：

```
// 切换暗模式 document.body.style.backgroundColor =
"black";document.body.style.color = "white";
```