今天课程核心内容

1. VUE-CLI 插件创建的 SPA 单页应用调试运行

```
npm run dev
```

2. 美多商城后台项目

美多商城项目的开发分为 2 个部分:

• 美多商城前台项目: 给普通大众用户进行使用,进行商品的浏览以及购物等操作.

● 美多商城后台项目: 给公司内部管理员进行使用,查看、管理相关数据表的数据。

项目架构:

开发模式: 前后端分离

前端框架: Vue.js(SPA 单页应用)

后端框架: Django REST framework

功能部分:管理员登录,数据统计,用户管理,商品管理,订单管理,权限管理

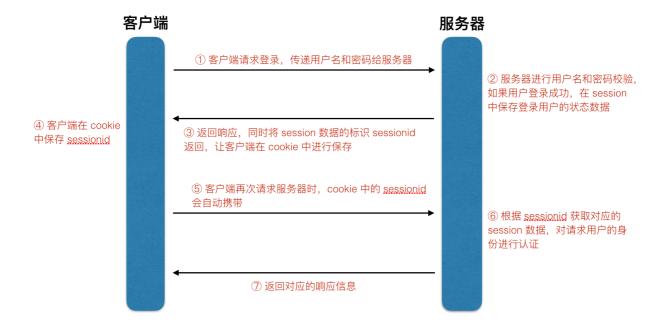
主要技术: IWT 用户认证、自定义 FastDFS 文件存储、权限控制

前端启动端口修改:

```
🤚 manage 🔻 🕨 🍎 🖏 🕟 🗊 📕 📗 Git: 🖌 🗸 🕓 🕤
\textbf{meiduo} \ \rangle \ \texttt{meiduo\_mall\_admin} \ \rangle \ \texttt{config} \ \rangle \ \underset{\textbf{JS}}{\overset{d}{=}} \ \texttt{index.js}
Project • meiduo
               'use strict'
  ▼ meiduo ~/Desktop/code/meiduo
                                            // Template version: 1.3.1
                                                                                                                                        tabase
   ▶ mathematic from t_end_pc
                                             a// see http://vuejs-templates.github.io/webpack for documentation.
   meiduo_mall
    ▼ meiduo_mall_admin
                                                                                                                                        C300
C300
C300
                                              const path = require('path')
Structure
     ▶ 🖿 .idea
     ▶ build
                                              module.exports = {
     ▼ laconfig
                                                dev: {
// Paths
                                                                                                                                        Host
    dev.env.js
                                                  assetsSubDirectory: 'static',
           🚚 prod.env.js
                                                  assetsPublicPath: '/'.
                                                 proxyTable: {
    ▶ mode_modules library root
                                                 // Various Dev Server settings
      ▶ 🖿 src
                                                  host: '127.0.0.1', // can be overwritten by process.env.HOST
     ▶ m static
                                                 👸 .babelrc
         .editorconfig
                                                  errorOverlay: true,
         .gitignore
         apostcssrc.is
                                                  poll: false, // https://webpack.js.org/configuration/dev-server/#devserver-watchoptions-
         favicon.ico
```

3. JWT Token 认证机制

session认证机制回顾:



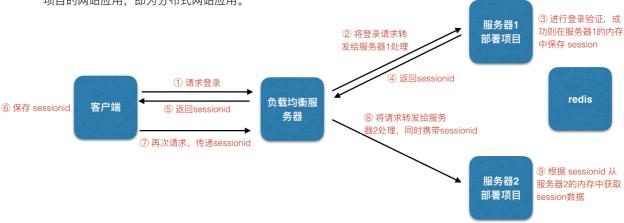
- 1. session数据存储在服务器,如果登录用户过多,会过多占用服务器的存储空间
- 2. session依赖于cookie, session数据标识sessionid存储在cookie中,如果cookie被截获,可能会产生CSRF伪造(解决: CSRF保护)
- 3. 分布式网站应用中,如果session数据存储在服务器内存中,session数据的共享会成为问题(解决:统一存储redis)

单机应用: 只有一台服务器

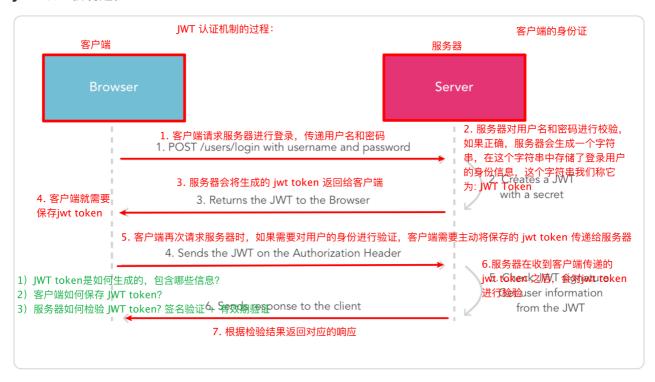


分布式应用:拥有多台服务器

进行项目部署时,当只有一台服务器时,如果客户端的请求量比较大,一台服务器可能支撑不住。鉴于这个原因,通常在进行项目部署时,会将相同的代码一次性部署到多台服务器上,然后通过再配置一个负载均衡服务器来实现客户端请求的转发,以此来减轻每台服务器处理请求的压力。这样以多台服务器同时部署 Web项目的网站应用,即为分布式网站应用。



JWT 认证机制过程:



JWT Token的 3 部分组成:

- 1) 头部(header): JSON 数据,保存 token 类型和签名加密的算法,生成时会进行 Base64 编码
- 2)载荷(payload): JSON 数据,保存有效数据和 token 的有效时间,生成时会进行 Base64 编码
- 3) 签名(signature): 签名字符串数据, 防止 JWT token 被伪造

JWT 认证使用注意点:

• 不要在 jwt 的 payload 部分存放敏感信息,该部分是客户端可解码的部分

- 服务器需要保存好自己使用的签名加密 secret 密钥
- 如果可以,请使用 https 协议(注:此方式不只是针对 JWT 认证机制,其他认证机制要想提高安全性也一样)