一、服务端渲染技术NUXT

1、什么是服务端渲染

服务端渲染又称SSR (Server Side Render)是在服务端完成页面的内容，而不是在客户端通过AJAX获取数据。

服务器端渲染(SSR)的优势主要在于：**更好的 SEO**，由于搜索引擎爬虫抓取工具可以直接查看完全渲染的页面。

如果你的应用程序初始展示 loading，然后通过 Ajax 获取内容，抓取工具并不会等待异步完成后再进行页面内容的抓取。也就是说，如果 SEO 对你的站点至关重要，而你的页面又是异步获取内容，则你可能需要服务器端渲染(SSR)解决此问题。

另外，使用服务器端渲染，我们可以获得更快的内容到达时间(time-to-content)，无需等待所有的 JavaScript 都完成下载并执行，产生更好的用户体验，对于那些「内容到达时间(time-to-content)与转化率直接相关」的应用程序而言，服务器端渲染(SSR)至关重要。

2、什么是NUXT

Nuxt.js 是一个基于 Vue.js 的轻量级应用框架,可用来创建服务端渲染 (SSR) 应用,也可充当静态站点引擎生成静态站点应用,具有优雅的代码结构分层和热加载等特性。

官网网站：

https://zh.nuxtjs.org/

二、NUXT环境初始化

1、下载压缩包

https://github.com/nuxt-community/starter-template/archive/master.zip

2、解压

将template中的内容复制到yygh_site

3、修改package.json

name、description、author（必须修改这里，否则项目无法安装）

```
  "name": "yygh",
  "version": "1.0.0",
  "description": "尚医通",
  "author": "atguigu",
```

4、修改nuxt.config.js

修改title: '{{ name }}'、content: '{{escape description }}'

这里的设置最后会显示在页面标题栏和meta数据中

```
module.exports = {
  /*
  ** Headers of the page
  */
  head: {
    title: 'yygh-site',
    meta: [
      { charset: 'utf-8' },
      { name: 'viewport', content: 'width=device-width, initial-scale=1' },
      { hid: 'description', name: 'description', content: '尚医通' }
    ],
…
```

5、在命令提示终端中进入项目目录

6、安装依赖

```
npm install
```

```
PS F:\0111yygh\yygh_site> npm install
npm WARN deprecated babel-eslint@10.1.0: babel-eslint is now @babel/eslint-parser. This package will no
npm WARN deprecated core-js@2.6.12: core-js@<3.3 is no longer maintained and not recommended for usage d
e of the V8 engine whims, feature detection in old core-js versions could cause a slowdown up to 100x ev
e, upgrade your dependencies to the actual version of core-js.
[ ..................] / fetchMetadata: sill pacote range manifest for inquirer@^7.3.3 fetched in 82ms
```

7、测试运行

```
npm run dev
```

8、NUXT目录结构

（1）资源目录 assets

用于组织未编译的静态资源如 LESS、SASS 或 JavaScript。

（2）组件目录 components

用于组织应用的 Vue.js 组件。Nuxt.js 不会扩展增强该目录下 Vue.js 组件，即这些组件不会像页面组件那样有 asyncData 方法的特性。

（3）布局目录 layouts

用于组织应用的布局组件。

（4）页面目录 pages

用于组织应用的路由及视图。Nuxt.js 框架读取该目录下所有的 .vue 文件并自动生成对应的路由配置。

（5）插件目录 plugins

用于组织那些需要在 根vue.js应用 实例化之前需要运行的 Javascript 插件。

（6）nuxt.config.js 文件

nuxt.config.js 文件用于组织Nuxt.js 应用的个性化配置，以便覆盖默认配置。

9、封装axios

（1）执行安装命令

npm install axios

（2）创建utils文件夹，创建request.js

```
import axios from 'axios'
```

```
import { MessageBox, Message } from 'element-ui'
// 创建axios实例
const service = axios.create({
    baseURL: 'http://localhost',
    timeout: 15000 // 请求超时时间
})
// http request 拦截器
service.interceptors.request.use(
    config => {
    // token 先不处理，后续使用时在完善
    return config
},
  err => {
    return Promise.reject(err)
})
// http response 拦截器
service.interceptors.response.use(
    response => {
        if (response.data.code !== 200) {
            Message({
                message: response.data.message,
                type: 'error',
                duration: 5 * 1000
            })
            return Promise.reject(response.data)
        } else {
            return response.data
        }
    },
    error => {
        return Promise.reject(error.response)
})
export default service
```