

编 号：_____

审定成绩：_____

重庆邮电大学 毕业设计（论文）

| | |
|----------------|--|
| 中文题目 | 医生信息网站前端设计与开发 |
| 英文题目 | The design and development of the website front-end of the doctor's information |
| 学院名称 | 生物信息学院 |
| 学生姓名 | 刘洋 |
| 专 业 | 生物医学工程 |
| 班 级 | 0611402 |
| 学 号 | 2014212066 |
| 指导教师 | 赵志强 教授 |
| 答 辩 组 负 责 人 | |

二零一八年 六 月

重庆邮电大学教务处制

学院本科毕业设计(论文)诚信承诺书

本人郑重承诺：

我向学院呈交的论文《医生信息网站前端设计与开发》，是本人在指导教师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明并致谢。本人完全意识到本声明的法律结果由本人承担。

年级

专业

班级

承诺人签名

年 月 日

摘要

在互联网飞速发展的今天，通过网络连接患者与医生成为一种趋势，设计开发医生信息的网站具有重要的意义。本文主要包含医生信息网站的设计以及其开发过程中相关技术的说明和解析。论文前部分对 `vue`，`webpack`，`Ajax`，移动端适配等项目中使用到技术特性以及相关原理进行了相关介绍。同时对最终设计方案以及完成的成果进行了展示，按照具体的使用流程对整个项目所具备的功能进行了介绍，对为什么要如此设计进行了简要的分析和说明。文中重点介绍了项目的具体代码实现，对主要的核心代码如实现 `rem` 计算的 `JavaScript` 代码、切割组件实现优化首次加载速度等方面从思路或者原理上进行了解释，同时文中还提及了本次项目一些基于实际测试的结果而做出的优化的点，如 `CDN` 分发网络的应用，缓存组件以及手动卸载等。

关键词：医生信息，`vue.js`，手机网站

Abstract

With the rapid development of the Internet, patients and doctors have become a trend with network connectivity. Designing and developing websites for doctors' information is of great significance. This thesis includes the design of doctor information website and the explanation and analysis of related technologies in its development process. The first part of the thesis introduces the technical characteristics and related principles of Vue, webpack, Ajax, mobile terminal adaptation and other projects. At the same time, the final design scheme and the completed results are displayed, the function of the whole project is introduced according to the approximate use process, and the reason for such design is briefly analyzed and explained. In this thesis, the specific code implementation of the project is introduced, and the main core code, such as the realization of the JavaScript code for REM calculation and the optimization of the first loading speed of the cutting component, is explained from the idea or principle. At the same time, the optimization of the results of the project based on the actual test is also mentioned in the thesis. Points, such as the application of CDN distribution network, cache components and manual unloading.

Keywords: doctor information, vue.js, mobile website

目录

| | |
|---------------------------|----|
| 第 1 章 引言..... | 1 |
| 1.1 研究背景和意义..... | 1 |
| 1.2 国内外发展现状..... | 1 |
| 1.2.1 国外..... | 1 |
| 1.2.2 国内..... | 2 |
| 1.3 主要内容和工作安排..... | 2 |
| 第 2 章 相关技术及工具..... | 5 |
| 2.1 Vue.js..... | 5 |
| 2.2 Ajax 实现数据交互..... | 6 |
| 2.2.1 axios..... | 6 |
| 2.2.2 浏览器中 Ajax 实现原理..... | 6 |
| 2.2.3 分离开发时的跨域问题..... | 8 |
| 2.3 Node.js..... | 8 |
| 2.4 Webpack..... | 8 |
| 2.5 手机端适配与 SASS..... | 9 |
| 2.5.1 手机端网站适配..... | 9 |
| 2.5.2 SASS..... | 9 |
| 2.6 CDN..... | 9 |
| 2.7 技术可行性..... | 10 |
| 2.8 开发工具..... | 10 |
| 2.9 本章小结..... | 10 |
| 第 3 章 设计与实现..... | 11 |
| 3.1 功能设计..... | 11 |
| 3.1.1 问题分析..... | 11 |
| 3.1.2 需求分析..... | 11 |
| 3.1.3 使用流程设计..... | 11 |
| 3.1.4 页面设计..... | 12 |

| | |
|-----------------------------------|----|
| 3.2 代码实现..... | 15 |
| 3.2.1 入口 HTML..... | 15 |
| 3.2.2 计算 rem 的 JavaScript 代码..... | 16 |
| 3.2.3 前端路由配置..... | 19 |
| 3.2.4 项目 JavaScript 代码入口..... | 20 |
| 3.2.5App 组件..... | 20 |
| 3.2.6 列表页核心代码..... | 22 |
| 3.2.7 详情页核心代码..... | 24 |
| 3.3 本章小结..... | 26 |
| 第 4 章 总结与展望..... | 27 |
| 4.1 主要工作与创新点..... | 27 |
| 4.2 后续研究工作展望..... | 27 |
| 参考文献..... | 29 |
| 致谢..... | 31 |
| 附录 A 其它核心源码..... | 33 |
| 列表页..... | 33 |
| 详情页..... | 35 |
| 预约挂号页..... | 38 |
| 附录 B 英文翻译..... | 41 |
| 英文原文..... | 41 |
| 原文翻译..... | 42 |

第 1 章 引言

1.1 研究背景和意义

现如今，现在的在人们生活的社会当中，可以清晰地发现人类现在正在逐渐迈向信息化社会，慢慢的互联网网络的广泛运用和计算机技术以及计算机网络技术，信息数据库技术、信息管理技术的发展，结合互联网的技术对各种信息的处理和利用已经深入到了我们日常生活中的各行各业，还有深入到人类生活中的各个方面。医院网站是各医院诊所对外宣传中不可缺少的工具，它的内容对于医院诊所的宣传和医院的项目开发的决策起着非常重要的作用。

网上医疗信息在满足网络用户实际需求还存在相当大的距离。广大医疗机构利用自身的医疗服务信息资源建立面向公众的医疗信息服务网站，及时发布医疗保健信息，提供专业、全面的医疗保健信息服务，已成为社会和医疗机构自身发展的迫切要求^[1]。因此，开发一个医院网站是很有必要的事情，人们对医院信息了解的越多，对医院的信任程度就越高，特别是一些特殊的理疗服务，所以，网络就是一个传播信息和宣传医院的平台。

随着移动互联网得发展，移动互联网与医疗服务也进入了高速的发展阶段，移动端的主要优势是患者触手可及，服务更接近患者。

下面各章节中我以近两年兴起的 Vue.js 为主要技术框架结合 webpack 以前后端分离开发的方式来开发一个医院医生信息网站系统为例，谈谈其开发过程和所涉及到的问题及解决方法。通过这次设计，我熟练地掌握了 webstorm 的使用，进一步学习了 Vue.js，webpack 等主流前端开发技术，能够独立的实现项目的设计与开发。在设计中遇到的一系列问题，通过请教老师，查阅资料来解决。这样不但培养了我虚心请教，互助团结的精神，而且也为我以后的工作生活积累了大量的宝贵经验。

1.2 国内外发展现状

1.2.1 国外

关于医疗相关网站的评价，国外研究者提出了一些准则，对于评价体系的构建尚不完整。如网络卫生基金会提出的可靠性和可用性8条准则；美国医学会(AMA)提出医疗卫生信息网站指南^[2]；牛津大学卫生科学研究所提出的 DISCERN 等。国外的研究注重网站的可用性和互动性，侧重于从用户使用的角度来选取指标，关注网站内容大于形式，这些方面值得肯定，但是提出的某些指标可操作性却不强，影响了评价结果的可信度^[3]。

1.2.2 国内

由于经营观念陈旧，许多医院还没有建立自己的网站，或者在别的网上挂了一个网页建了网站的医院，往往缺乏管理，很长时间都不去更新，形同虚设，功能设置不明确，病人只能了解一些表面上的东西，没有互动，网络最大的特点就是互动，没有互动，就会失去很多的机会^[4]。

阻碍医院上网原因有很多，人才的匮乏，医院网站建设需要一批精通计算机网络的人才，特别是既了解网络又了解医学的复合性人才^[5]。这在绝大多数医院里是不具备的。资金的不足，建设网站需要投入一定的资金，而一般医院不会在网络上投入太多的资金。

国内的相关研究比国外起步晚，相关研究多借鉴国外的研究成果。目前国内多以应用软件的方式实现移动医疗，患者需要下载并且安装对应的应用程序来获得移动医疗的便捷服务，并且在手机挂号，医疗咨询等方面已经有了比较成熟的应用^[6]。患者不用到医院排队挂号，排队支付费用，可以远程得到医生优质的健康质询服务。就目前移动医疗带的如减少患者排队，优化就诊流程，患者更方便的得到医疗咨询服务等方面的好处来看，医疗信息网站在手机移动端的确是有巨大的发展潜力^[7]。

1.3 主要内容和工作安排

全文共分为4章，内容结构安排如下：

第1章是引言部分；

第2章是相关技术与开发工具的介绍；

第 3 章项目设计与实现的阐述；

第 4 章是对本次毕业设计的总结和展望；

第2章 相关技术及工具

随着移动端应用的高速发展，当前移动端应用类型也发展得多种多样，每一个行业每一个领域都是最适合自己的应用模式。做技术选型时，充分考虑做出最适合的选择才能让之后的运营发展不至于耗时耗力，手忙脚乱。在如今随着互联网信息技术的高速普及和高速发展，尤其是微信公众平台和应用程序的出现，医院的医疗的服务正受到颠覆性的影响和冲击^[8]。现在由于法律法规等的限制，必须在医院实施，除了医疗和配药。此外，其余的在线注册账号、专家提前预定、检查报告、移动支付等功能能够实现在线完成，大大提高了医院的服务水平，大大提高了患者的医疗体验。

2.1 Vue.js

随着互联网的迅猛发展，用户对 Web 前端的使用体验、交互操作流程、外观有了更高的要求。特别是 Web 系统中越来越多的数据处理和业务逻辑开始偏向前端，导致 Web 前端工作量扩大，代码量增加^[9]。如果仍然采用传统的方式开发设计 Web 前端，会导致前期开发度和后期维护难度增大，可扩展性变差。为了提高开发效率和代码复用率，越来越多的网页开发框架开始流行。于是先后提出了 MVC、MVVM 模式，方便了构建基于事件的 Web 前端开发平台^[10]。本次项目主要使用 Vue 作为项目的技术框架，利用 Vue 框架实现了前后端分离开发，简化了 Web 前端开发流程。

为了提高开发效率，对 Web 前端基于框架进行了开发。对比目前比较流行的 React、Angular、Ploymer 框架，最终选择了 Vue.js 框架。与其他重量级框架不同的是，Vue 是一套构建用户界面的渐进式框架，采用自底向上增量开发的设计方式，是更加灵活、开放的解决方案，架构更加简单，适合开发人员快速掌握其全部特性并投入使用，还便于与第三方库或既有项目整合。结合 Vue 生态系统支持库 Vuex、Vue-router，能够为复杂的应用程序提供驱动。

Vue 的核心是响应式原理，把一个普通 JavaScript 对象传给 Vue 实例的 data 选项，同时每个 Vue 实例都有相应的 watcher 实例对象。如果 data 的属性发生变

化，会通知 `watcher` 重新计算，从而致使它关联的组件得以更新。Vue 异步执行 DOM 更新。只要观察到数据变化，Vue 将开启一个队列，并缓冲在同一事件循环中发生的所有数据改变。如果同一个 `watcher` 被多次触发，只会一次推入到队列中，在缓冲去除了重复数据，避免了不必要的计算和 DOM 操作^[5]。Vue 的响应为双向绑定数据，实时反映数据的真实变化，并映射到数据源上，避免了前端页面开发中 DOM 选择器繁杂的操作，简化了 Web 前端开发流程，降低了开发难度，提高了前端开发效率，缩短了开发成本和周期。

2.2 Ajax 实现数据交互

Ajax 即 “Asynchronous Javascript And XML”（异步 JavaScript 和 XML），是指一种创建交互式网页应用的网页开发技术。Ajax 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。传统的网页（不使用 Ajax）如果需要更新内容，必须重载整个网页页面。

2.2.1 axios

本次项目使用 axios 作为实现 Ajax 技术的函数库。axios 是一个基于 promise 的 HTTP 库。axios 具有从浏览器中创建 XMLHttpRequest、从 node.js 发出 http 请求、支持 Promise API、拦截请求和响应、转换请求和响应数据、取消请求、自动转换 JSON 数据、客户端支持防止 CSRF/XSRF 的特点。利用 axios 请求和响应 JSON 格式的数据进行后台的数据交换。

前后端分离开发的核心是前后端之间通过接口对接数据，而 ajax 技术则是前后端数据连接的桥梁，axios 通过对 XMLHttpRequest 进行封装，提供了很多优秀的 API，是前端进行数据请求的一个优秀的函数库。

2.2.2 浏览器中 Ajax 实现原理

在现代浏览器中 Ajax 最核心的依赖是浏览器提供的 XMLHttpRequest 对象，正是这个对象使得浏览器可以发出 HTTP 请求与接收 HTTP 响应^[6]。

XMLHttpRequest 一开始只是微软浏览器提供的一个接口，后来各大浏览器纷纷效仿也提供了这个接口，再后来 W3C 对它进行了标准化，提出了 XMLHttpRequest 标准^[1]。XMLHttpRequest 标准又分为 Level 1 和 Level 2。

XMLHttpRequest Level 1 主要存在以下缺点：

- 1、受同源策略的限制，不能发送跨域请求；
- 2、不能发送二进制文件（如图片、视频、音频等），只能发送纯文本数据；
- 3、在发送和获取数据的过程中，无法实时获取进度信息，只能判断是否完成；

XMLHttpRequest Level 2 中新增了以下功能：

- 1、可以发送跨域请求，在服务端允许的情况下；
- 2、支持发送和接收二进制数据；
- 3、新增 formData 对象，支持发送表单数据；
- 4、发送和获取数据时，可以获取进度信息；
- 5、可以设置请求的超时时间；

常规的一次请求流程如下：

首先，新建一个 XMLHttpRequest 的实例。

```
var xhr = new XMLHttpRequest();
```

然后，向远程主机发出一个 HTTP 请求。

```
xhr.open('GET', 'example.php');
```

```
xhr.send();
```

接着，就等待远程主机做出回应。这时需要监控 XMLHttpRequest 对象的状态变化，指定回调函数。

```
xhr.onreadystatechange = function(){
    if ( xhr.readyState == 4 && xhr.status == 200 ) {
        alert( xhr.responseText );
    } else {
        alert( xhr.statusText );
    }
};
```

以上代码中使用的核心属性表示如下：

xhr.readyState: XMLHttpRequest 对象的状态，等于 4 表示数据已经接收完毕。

xhr.status: 服务器返回的状态码，等于 200 表示一切正常。

xhr.responseText: 服务器返回的文本数据

xhr.responseXML: 服务器返回的 XML 格式的数据

xhr.statusText: 服务器返回的状态文本。

2.2.3 分离开发时的跨域问题

XMLHttpRequest 无论是 Level 1 还是 Level 2 在后台服务器不配置的情况下，都是不支持跨域请求的，因为浏览出于对安全的考虑。

本次项目开发使用的后台是没有进行相关跨域配置的，而开发时的跨域问题是通过 webpack 的相关配置来解决的。核心原理如下：

进行了相关配置之后，在启动项目的时候，webpack 会利用 Node.js 在本地启动一个临时的代理服务器，我们对后台接口发起的所有请求都可以通过这个代理服务器进行转发，通过这种代理服务器，跨域问题便不再存在^[12]。

2.3 Node.js

Node.js 是一个 Javascript 运行环境，发布于 2009 年 5 月，由 Ryan Dahl 开发，实质是对 Chrome V8 引擎进行了封装，Node 使 JavaScript 代码可以脱离浏览器环境运行。在本次项目中，Node 主要用于开发时启动项目，以及前文提到的搭建临时代理服务器解决跨域问题。

2.4 Webpack

Webpack 是当下最热门的前端资源模块化管理和打包工具。它可以将许多松散的模块按照依赖和规则打包成符合生产环境部署的前端资源^[13]。还可以将按需加载的模块进行代码分隔，等到实际需要的时候再异步加载。通过 loader 的转换，任何形式的资源都可以视作模块，比如 CommonJs 模块、AMD 模块、ES6 模块、CSS、图片、JSON、CoffeeScript、SASS 等。目前 Webpack 已经有 4.0+ 版本。

2.5 手机端适配与 SASS

2.5.1 手机端网站适配

随着高端手机的盛行，移动互联应用开发也越来越受到人们的重视，用 html5 开发移动应用是最好的选择。然而，每一款手机有不同的分辨率，不同屏幕大小。本次项目采用的是当下流行的适配解决方案，运用 js 计算结合 CSS3 的新单位 rem 来进行适配^[14]。

2.5.2 SASS

Sass 是 CSS 语言的预处理器，由于移动端需要考虑适配问题采用的 rem 单位，所有设计稿上标注的 px 单位都需要转换成 rem，所以本次项目采用了 SASS 作为 css 的预处理语言。通过依赖的“node-sass”，“sass-loader”两个插件，结合 webpack 可以直接将用 sass 语法模式书写的代码直接转化成 css 代码。

2.6 CDN

CDN 全称:Content Delivery Network 或 Content Ddistribute Network，即内容分发网络。CDN 的基本思路是尽可能避开互联网上有可能影响数据传输速度和稳定性的瓶颈和环节，使内容传输的更快、更稳定。通过在网络各处放置节点服务器所构成的在现有的互联网基础之上的一层智能虚拟网络，CDN 系统能够实时地根据网络流量和各节点的连接、负载状况以及到用户的距离和响应时间等综合信息将用户的请求重新导向离用户最近的服务节点上。CDN 的目的是解决因分布、带宽、服务器性能带来的访问延迟问题，适用于站点加速、点播、直播等场景。使用户可就近取得所需内容，解决 Internet 网络拥挤的状况，提高用户访问网站的响应速度和成功率。控制时延无疑是现代信息科技的重要指标，CDN 的意图就是尽可能的减少资源在转发、传输、链路抖动等情况下顺利保障信息的连贯性^[15]。

本次项目的相关第三方 JS 框架全部通过 CDN 引入，使用 CDN 加速在移动端的应用中的优势尤为突出，当用户处于网络环境不是特别好的情况下可以最大化的提升项目的加载速度，优化用户体验。

2.7 技术可行性

采用前后端分离式开发的优势明显，前后端分离，使得前后端能够各司其职，后端更侧重于服务的提供，而前端更注重服务的使用，前端通过 JS 可以做非常多的数据处理工作，所以一定程度上也能够降低服务器的压力；后端的处理异常也不用直接反映到前端，通常分离可将异常处理变得更友好，比如以炫丽的页面效果展示错误消息。随着技术的发展，前后端技术的差异性也日益明显，如果仍然以传统 web 开发模式来实现，短时间也不能确保每一个人都能精通全栈开发，进行前后端分离，后端更注重的是服务提供，而不用考虑前端的终端情况，至于如何布局，如何实现数据渲染展示交由前端完成，分工更明确，减少了前后端的耦合，降低了合作难度。通过我个人最近做过的一些项目来看，我也更愿意采用前后端分离的模式。

2.8 开发工具

本次项目采用 WebStorm 作为编辑器，Git 作为版本管理工具。

2.9 本章小结

本章对项目中用到的一些技术做出了相关介绍，对几个重点的技术的核心原理进行了阐述同时介绍了开发时用到的相关工具。

第3章 设计与实现

3.1 功能设计

医院网站是医院利用互联网发布信息、扩展业务范围、提高服务质量的重要工具和平台，是医院利用互联网的最常见形式，发挥了越来越重要的作用。一方面可以提供各方面的医疗服务和专家信息，方便指导群众就医，促进医患沟通与交流，并对公众进行网上健康教育，延伸和拓展医疗服务；另一方面作为医院信息传播平台，可以发布医疗教研成果及医院工作动态等信息，展示和宣传医院品牌，促进医院内部员工之间及医院与国内外同行之间的沟通交流。随着医院网站网民群体的逐步壮大，对医院网站的定位、功能、特征及网民上网目的、习惯、需求等相关问题进行调研非常重要，能够进一步完善医院网站，更好的为患者服务。

3.1.1 问题分析

目前已存的各种相关医生信息网站存在的问题大概有，互动性能不够完善，网站布局架构比较乱，层次比较混乱，主题信息不突出，很容易导致访问者出现迷茫，从而不仅分散了访问者的注意力，而且也占用了访问者宝贵的时间，栏目设置不合理，不能全方位向公众提供医院的各种信息服务。

3.1.2 需求分析

本网站主要是为医生信息展示而开发的，患者对医生信息网站的功能需求绝不仅仅只满足于通过这个网站可以了解到医院相关医生的信息，在通过网站看到对应医生的详细信息之后，患者更希望也能够预约对应的医生进行挂号和在线咨询。

3.1.3 使用流程设计

根据问题和需求分析，具体项目使用流程设计如图 3.1 所示。

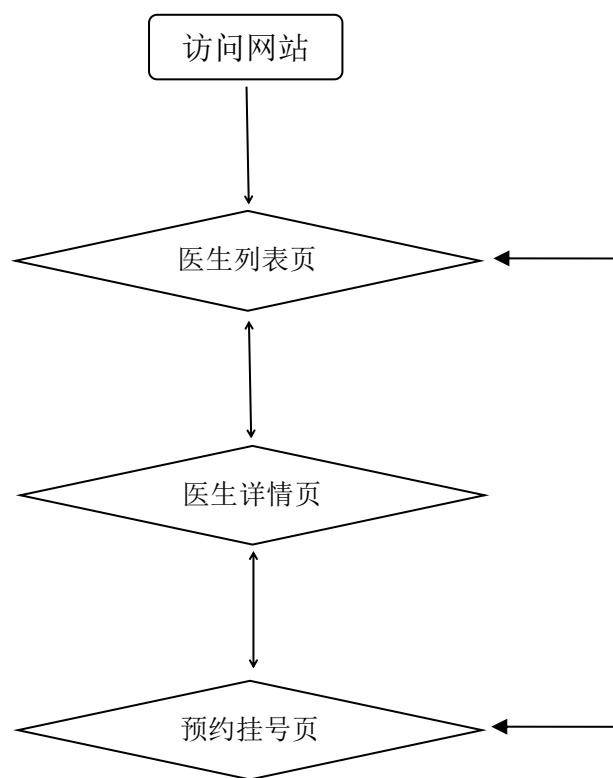


图 3.1 流程设计图

本次项目的具体使用逻辑按照图 3.1 实现。患者访问网站之后首先进入医生列表页，在列表页列出医院医生的信息列表。患者如果确定了某个医生，则可以直接通过链接跳转到预约挂号页进行留言和挂号，如果需要了解具体的详细信息，则可以通过链接跳转到对应医生的详情页。在详情页中，对每一个医生的详细信息进行了详细的介绍，患者通过详情页了解了医生的信息之后，如果认为该医生与自己的病情不符则可以重新返回列表页继续寻找对应的医生，否则也可从详情页链接到对应的预约挂号页进行留言和挂号。

3.1.4 页面设计

1) 医生信息列表页如图 3.2 所示。



图 3.2 医生信息列表页图

通过图 3.2 可以直观的看到医生信息列表页的主要功能。通过列表页患者可以直观的了解到每一个医生姓名，职称简介，所属科室，以及外观长相。这些都是患者比较关注的地方。往往很多的患者更希望能够直观得看到医生的长相，因为它可以直观表现出一个医生的从医经验等方面的东西。如果患者在列表页直接找到了想要预约得医生，就可以直接通过“预约挂号”链接，直接到预约挂号页面进行预约挂号的操作。

2) 医生信息详情页如图 3.3 所示。



图 3.3 医生信息详情页图

在详情页中，可以看到对应的医生的完整个人简介，专业特长，学术任职等信息。同时，通过在此页中的信息，如果患者认为该医生符合自己的需求就可以直接通过上方的预约咨询按钮得到对应的预约咨询服务。

3) 预约挂号的提交页面如图 3.4 所示。



| |
|---|
| 您的姓名 |
| 刘洋 |
| 联系方式 |
| 18883944710 |
| 预约日期 |
| 2018年5月20日 |
| 您的需求 |
| 请简单描述你的病情 |
| <button>确认提交</button> <button>在线咨询</button> |

图 3.4 预约提交页图

在预约提交页中，患者可以将自己的基本信息、预约时间、病情描述提交到医院的后台之中。在这一个页面为了方便患者，同时提供了“在线咨询”的按钮，通过这个按钮，可以直接调用手机的电话拨打功能，呼叫医院的对应部门，进行在线的咨询。

3.2 代码实现

本次项目的完整代码即目录结构见 <https://github.com/1161906592/biyesheji>，下面是对本次项目中的一些核心代码实现进行详细的说明。

3.2.1 入口 HTML

Vue 项目的特性是，整个项目的路由页面都在一个 html 之中完成，所有的更新都是通过 Ajax 进行数据交互局部刷新，所以本次项目只有一个 index.html 文件来作为整个项目的入口。具体代码如下：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>biyesheji</title>
```

```

<link rel="icon" href="data:image/ico;base64,aWNv">
<script src="/static/js/fit.js"></script>
<script src="https://cdn.bootcss.com/vue/2.5.17-beta.0/vue.min.js"></script>
<script
src="https://cdn.bootcss.com/vue-router/3.0.1/vue-router.min.js"></script>
<script src="https://cdn.bootcss.com/axios/0.18.0/axios.min.js"></script>
</head>
<body>
<div id="app"></div>
</body>
</html>

```

在本次项目之中使用 link 标签结合 base64 的方式将网页的 icon 图标写死在了页面之中，因为在项目实际的测试过程中我发现在路由切换的过程中，虽然页面没有整体刷新，但是浏览器会默认发起一次对 icon 图标的请求，这种请求是完全没有必要的，于是我采用了这种写死图标数据的方式来解决。实际结果验证，通过这种方式处理之后，icon 图标的请求便再也没有发起。

第二章中所提及的 CDN 技术便是在 index.html 之中应用，本次项目使用 CDN 主要是对“Vue”，“VueRouter”，“axios”，三个核心依赖库进行 script 标签方式的引入。

定义在网页 body 之中的 div#app 将作为此次项目的页面容器，所有路由切换之后的视图都将渲染在这个标签之中。

3.2.2 计算 rem 的 JavaScript 代码

```

(function (window, document) {
  'use strict'
  var fit = {};
  (function () {
    var viewportEl = document.querySelector('meta[name="viewport"]')
    var fitEl = document.querySelector('meta[name="hotcss"]')
    var dpr = window.devicePixelRatio || 1
    var maxWidht = 540

```



```
var designWidth = 0
dpr = dpr >= 3 ? 3 : (dpr >= 2 ? 2 : 1)
if (hotcssEl) {
  var fitCon = fitEl.getAttribute('content')
  if (fitCon) {
    var initialDprMatch = fitCon.match(/initial\dpr=([\d\.]+)/)
    if (initialDprMatch) {
      dpr = parseFloat(initialDprMatch[1])
    }
    var maxWidthMatch = fitCon.match(/max-width=([\d\.]+)/)
    if (maxWidthMatch) {
      maxWidth = parseFloat(maxWidthMatch[1])
    }
    var designWidthMatch = fitCon.match(/design-width=([\d\.]+)/)
    if (designWidthMatch) {
      designWidth = parseFloat(designWidthMatch[1])
    }
  }
}
document.documentElement.setAttribute('data-dpr', dpr)
fit.dpr = dpr
document.documentElement.setAttribute('max-width', maxWidth)
fit.maxWidth = maxWidth
if (designWidth) {
  document.documentElement.setAttribute('design-width', designWidth)
}
fit.designWidth = designWidth
var scale = 1 / dpr
var content = 'width=device-width, initial-scale=' + scale + ',
minimum-scale=' + scale + ', maximum-scale=' + scale + ', user-scalable=no'
if (viewportEl) {
  viewportEl.setAttribute('content', content)
} else {
```

```

        viewportEl = document.createElement('meta')
        viewportEl.setAttribute('name', 'viewport')
        viewportEl.setAttribute('content', content)
        document.head.appendChild(viewportEl)
    }
})();
fit.mresize = function () {
    var innerWidth = document.documentElement.getBoundingClientRect().width
    || window.innerWidth;
    if (fit.maxWidth && (innerWidth / fit.dpr > fit.maxWidth)) {
        innerWidth = fit.maxWidth * fit.dpr
    }
    if (!innerWidth) {
        return false
    }
    document.documentElement.style.fontSize = (innerWidth * 20 / 320) + 'px'
}
fit.mresize()
window.addEventListener('resize', function () {
    clearTimeout(fit.tid)
    fit.tid = setTimeout(hotcss.mresize, 33)
}, false)
window.addEventListener('load', fit.mresize, false)
setTimeout(function () {
    fit.mresize()
}, 333)
})(window, document)

```

上述代码的主要功能是为网页计算根标签的字体大小即 **rem**，由于移动端存在不同的屏幕尺寸，不同的分辨率，不同的 **dpr**，所有的手机网页都必须进行适配才能正常的使用，本次项目我采用的是 **rem** 来实现的适配。上述代码的核心思路是，先通过 **js** 来为页面添加 **viewport** 的 **meta** 标签，通过 **dpr** 与分辨率的关系来指定 **viewport** 的初始缩放比例，并设置用户不可手动缩放网页，然后获取手机屏幕的宽

度，根据宽度来设置 rem 的大小。最终形成的效果是，屏幕越大，分辨率越大，网页上的对应的宽高，字体大小等也同时更大，但是整体的布局等都是是一样的。

同时，在实际测试的过程中，我使用了多款不同的手机进行测试，我发现在某些特定的情况下，直接只执行一次计算设置的方法会出现设置失败的情况，于是我选择了通过多种方式重复调用了该方法。进行了这种处理之后，经过多次没有发现该 Bug 再次发生。

通过上述核心 js 代码结合 css 预处理语言，最终实现了项目多移动终端适配的效果。

3.2.3 前端路由配置

本次项目中，页面都是不存在权限控制的，所以我就直接将几个页面涉及的路由全部直接定义了出来，具体配置如下：

```
import Router from 'vue-router'
export default new Router({
  routes: [
    {
      path: '/',
      name: 'list',
      component: () => import('../views/list')
    },
    {
      path: '/detail',
      name: 'detail',
      component: () => import('../views/detail')
    },
    {
      path: '/form',
      name: 'form',
      component: () => import('../views/form')
    }
  ]
})
```

```
}}
```

上述代码对前端的页面路由分配进行了配置 `path` 和 `name` 用于标记对应的页面组件，`component` 用于指定组件。同时，我使用了回调 `import` 的方式来实现了路由的按需加载，在此之前，我使用的是直接 `import` 的方式，没有实现按需加载，于是在实际测试的过程中，我发现在网速不佳的情况下，整个项目的首次加载速度特别的慢，大大加长了等待的时间，已经严重的影响了用户体验。最终我通过使用回调 `import` 的方式来引入组件，实现了用户点击到某个页面才去加载对应的组件，最终首次加载的速度得到了较明显的优化。

3.2.4 项目 JavaScript 代码入口

本次项目的入口是 `index.html`，同时整个项目在浏览器中的所有行为依赖于的 JavaScript 代码，`main.js` 则是 JavaScript 代码的入口，具体代码实现如下：

```
import Vue from 'vue'
import router from './router'
const App = () => import('./App')
Vue.config.productionTip = false
/* eslint-disable no-new */
new Vue({
  router,
  render: h => h(App)
}).$mount('#app')
```

上述代码的具体功能是实例化一个具有路由的 `Vue` 对象，并且使用 `$mount` 方法将它挂载到 `index.html` 中的 `div#app` 这个 `Dom` 之上。同时，本次项目使用 `CDN` 引入的是 `Vue.js` 是 `vue.runtime.min.js`，该版本是最精简的版本，不具有编译组件的功能，于是我选择使用 `render` 方法来渲染 `App.vue` 组件。

3.2.5 App 组件

`App.vue` 是整个项目的主组件，其具体代码如下：

```
<template>
  <div id="app">
```

```
<!-- 缓存组件 -->
<keep-alive>
  <router-view/>
</keep-alive>
</div>
</template>
<script>
export default {
  name: 'App'
}
</script>
<style>
// 清除默认内外边距
body,div,dl,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,h6,pre,code,form,fieldset,legend,input,textarea,p,blockquote,th,td,hr,button,article,aside,details,figcaption,figure,footer,header,hgroup,menu,nav,section {
  margin:0;
  padding:0
}
body{
  -webkit-text-size-adjust: none; /* 禁止字体自动调整 */
  -webkit-tap-highlight-color: transparent;
}
input,select,textarea {
  font-size:100%
}
input,textarea{
  -webkit-appearance:none; /* 去除输入框默认内阴影 */
}
textarea {
  resize: none;
}
```

```
input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button{
  -webkit-appearance: none !important;
  margin: 0;
}
img {
  border: 0;
  vertical-align: middle;
}
a {
  text-decoration: none;
  color: inherit;
}
</style>
```

在 App 组件中，我定义了<router-view/>标签，该标签的作用是渲染路由中定义的一级页面。同时本次项目我选择了添加 keep-alive 来缓存组件，这样做的目的是让每一个组件不是每一次都刷新，从列表页到详情页再从详情页回到列表页的时候，详情页仍然保持着之前所有的状态不变，也不会重新去请求表页的数据。我的理解是，在患者浏览医生信息这段时间之内，医生列表页信息发生改变的可能性并不大，所有没有太大的必要每次去重新请求数据。同时，如果每次后退都刷新的话用户体验并不是很好，用户更愿意从上次列表的滚动位置继续浏览下面的内容。

在 App 组件的 style 标签之中，我对整个项目可能用到的 css 属性进行了样式重置，清除了一些默认的风格，这样在后面的开发中可以避免很多麻烦。

3.2.6 列表页核心代码

```
import axios from 'axios'
export default {
  name: 'list',
  data () {
    return {
```

```
      data: null
    }
  },
  methods: {
    toDetail (item) {
      this.$router.push({
        name: 'detail',
        query: {
          id: item.id
        }
      })
    },
    toForm () {
      this.$router.push({
        name: 'form'
      })
    }
  },
  created () {
    axios
      .get('/bysj/data')
      .then(res => {
        console.log(res)
        this.data = res.data.data
      })
      .catch(() => {
      })
  }
}
```

上述代码是列表页的核心 JavaScript 代码，该页面引入了 axios 库作为数据请求的功能库，在组件被创建时即 created 生命周期函数被触发时，我通过 get 方式向服务器的“/bysj/data”接口发起请求，同时此次项目使用的是阿里云服务器和

mysql 数据库，该接口是我使用 Node.js 编写的。该请求成功之后的数据返回如图 3.5 所示。



图 3.5 列表页数据返回图

在 `methods` 中定义的 `toDetail` 方法用于响应用户点击对应的医生跳转到对应医生的详情页面，`toForm` 用于在用户选择对某个医生进行预约时跳转到对应的预约挂号页面。上述方法均是通过页面上的点击事件触发，用户通过点击对应的 `Dom` 即可实现链接到对应的页面。

3.2.7 详情页核心代码

```

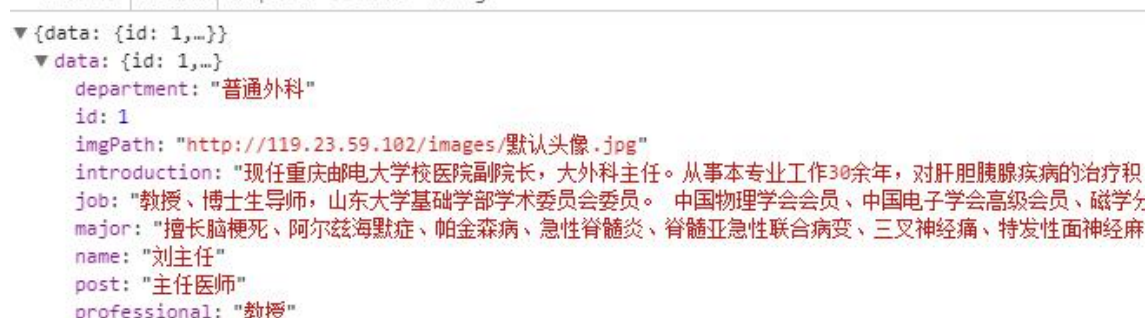
import axios from 'axios'
export default {
  name: 'detail',
  data () {
    return {
      data: {
        introduction: "",
        job: "",
        major: "",

```



```
        name: "",
        post: "",
        professional: "",
        department: "",
        imgPath: ""
      }
    }
  },
  methods: {
    toForm () {
      this.$router.push({
        name: 'form'
      })
    }
  },
  created () {
    axios
      .get('/bysj/detail', {
        params: {
          id: this.$route.query.id
        }
      })
      .then(res => {
        console.log(res)
        this.data = res.data.data
      })
  },
  deactivated () {
    /* 手动销毁组件 */
    this.$destroy()
  }
}
```

详情页的数据请求同样使用 `get` 方式，来自于“`/bysj/detail`”接口，通过向该接口传递每一个医生的 `id` 信息来获取到对应医生的详细信息，具体返回数据如图 3.6 所示。



```
▼ {data: {id: 1,...}}
  ▼ data: {id: 1,...}
    department: "普通外科"
    id: 1
    imagePath: "http://119.23.59.102/images/默认头像.jpg"
    introduction: "现任重庆邮电大学校医院副院长，大外科主任。从事本专业工作30余年，对肝胆胰腺疾病的治疗积"
    job: "教授、博士生导师，山东大学基础学部学术委员会委员。 中国物理学会会员、中国电子学会高级会员、磁学"
    major: "擅长脑梗死、阿尔兹海默症、帕金森病、急性脊髓炎、脊髓亚急性联合病变、三叉神经痛、特发性面神经麻"
    name: "刘主任"
    post: "主任医师"
    professional: "教授"
```

图 3.6 详情页数据返回图

在 `data` 中定义的各项属性用于绑定 `dom` 并且接受接口返回的数据渲染到对应的 `dom` 之中，`methods` 中定义的 `toForm` 方法用于用户点击按钮式跳转到预约挂号页面。详情页的重点代码处理在于，我在 `deactivated` 钩子函数之中调用了 `$destroy` 方法。由于在本次项目使用了 `keep-alive` 来让组件得到了缓存，在最初的实际测试中，我发现不管进入的是哪一个医生的详情页都不会发生改变，于是我选择了运用这种调用 `$destroy` 的方式来手动销毁 `detail` 组件。最终，我实现了预期的返回列表页不刷新进入详情页重新请求数据的效果。

3.3 本章小结

在本章中，我对本次毕业设计项目的设计和分别的开发实现做出来详细的说明和介绍。

第4章 总结与展望

4.1 主要工作与创新点

本次毕业设计的设计与开发没有经过他人指点完全由本人经过学习之后独立完成。本次毕业设计的项目的开发采用了与传统网站开发不同的前后端分离开发的模式，在项目的开发和实际测试中遇到了很多意想不到的问题，在经过自己查阅相关的资料最终独立的解决了相关的问题，并且对项目的一些方面做出了优化。在本科学习过程中是一次全新的尝试，在项目开发中，我的独立解决问题的能力得到了锻炼，学习到了很多新的知识。同时，整个毕业论文设计的过程中我学到了做任何事情所要有的态度和心态，首先我明白了做学问要一丝不苟，对于出现的任何问题和偏差都不要轻视，要通过正确的途径去解决，在做事情的过程中要有耐心和毅力，不要一遇到困难就打退堂鼓，只要坚持下去就可以找到思路去解决问题的。在工作中要学会与人合作的态度，认真听取别人的意见，这样做起事情来就可以事半功倍。通过本次毕业设计，学到了很多相关的新兴技术，同时也对此类网站的架构积累了宝贵的经验。

4.2 后续研究工作展望

伴随着科技水平持续良好发展，中国已然进入了数字化、信息化时代，网络成为人们进行信息交流的主要途径。在医疗领域，智能手机、平板电脑、移动医护推车、床旁信息终端的普及渗透率日渐提高，医疗信息领域的移动互联化需求迅猛增加。对于移动医疗，美国医院信息管理系统协会(HIMSS)给出的定义是：通过使用移动通信技术，例如智能手机、3G/4G 移动网络和卫星通信等，提供医疗服务和信息。移动医疗以它使用便利、成效明显、易普及等特点，推动了医院信息化进程，获得了大众的认可，得到了普遍的使用。医院积极推进信息化建设可以更好地实现临床科室与患者的交流与互动，实现医生与医生交流、医生与患者交流、患者与患者交流。网络医疗的目的是通过各个行业相互联系，最终实现以个人为中心或以家庭为中心的健康管理模式，使每个人都有一个医疗保健系统。

健康管理平台，记录个人身体的各个方面的健康指标等信息，然后通过对健康数据的大数据的分析，给个人健康管理的意见和建议，让每个人都有自己的健康管理模式。

参考文献

- [1] 段跃润. 基于 HTML5 的移动电商系统前端的设计与实现[D].南京大学,2017.
- [2] 王海燕,郭珍军.海外移动医疗信息化进展[J].现代电信科技,2014,41(04):10-14.
- [3] 刘星,王德安.以专题为中心的网站架构设计探析[J].电脑编程技巧与维护,2017(21):68-70.
- [4] 麦冬,陈涛,梁宗湾.轻量级响应式框架 Vue.js 应用分析[J].信息与电脑(理论版),2017(07):58-59.
- [5] 旷志光,纪婷婷,吴小丽.基于 Vue.js 的后台单页应用管理系统的研究与实现[J].现代计算机(专业版),2017(30):51-55.
- [6] 唐永瑞,张达敏.基于 Ajax 与 MVC 模式的信息系统的研究与设计[J].电子技术应用,2014,40(02):128-131.
- [7] 江庆,叶浩荣.Vue+Webpack 框架在银行 App 前端开发的应用[J].金融科技时代,2016(11):15-19.
- [8] 吴吉义,李文娟,黄剑平,章剑林,陈德人.移动互联网研究综述[J].中国科学:信息科学,2015,45(01):45-69.
- [9] 张立,王丽莎.移动互联网研究综述[J].现代国企研究,2016(04):200-201+204.
- [10] 王迪,王汉生.移动互联网的崛起与社会变迁[J].中国社会科学,2016(07):105-112.
- [11] 何琳,魏雅雯,茆意宏.移动互联网用户阅读利用行为研究[J].图书情报工作,2014,58(17):23-30.
- [12] 王建国.Ajax 技术在网站开发中的应用研究[J].湖南城市学院学报(自然科学版),2016,25(01):155-156.
- [13] 靖伟.Ajax 技术的研究与应用[J].中国传媒大学学报(自然科学版),2015,22(06):50-55.
- [14] 贺涛,缪淮扣,钱忠胜.基于 Ajax 技术的 Web 应用的建模与测试用例生成[J].计算机科学,2014,41(08):219-223+244.
- [15] 周柱,郎朗.Ajax 技术在 B/S 架构中的数据传输应用研究[J].新余学院学报,2016,21(03):109-113.

致谢

经过了这段时间的学习与努力，我最后完成了论文的写作。从开始接到论文题目到完成论文写作，每走一步对我来说都是一种新的尝试与探索，这也是我第一个独立构思完成的项目。在这段时间里，我学到了很多知识也有很多感受，从对前后端分离等技术和医疗信息化的一无所知，我开始了从头开始学习和尝试，查看相关的资料和书籍，开始对本次自己的构思有了一个大概明确的方向，使自己十分稚嫩作品一步步完善起来，每一次改善都是我学习的收获。最后完成了本次毕业设计的全部工作，收获了很多，从最终的结果来看自己当初的设想的确也有一些可取之处。

最后，我要感谢参与我论文评审和答辩的各位老师，他们给了我一个审视几年来学习成果的机会，让我能够明确今后的发展方向，他们对我的帮忙是一笔无价的财富。我将在今后的工作、学习中加倍努力，以期能够取得更多成果回报他们、回报社会。再次感谢他们，祝他们一生幸福、安康!

附录 A 其它核心源码

列表页

```
<template>
  <div class="list">
    <div v-for="(item, index) in data" :key="index" class="item"
      @click="toDetail(item)">
      <div class="img">
        
        <div>{{item.department}}</div>
      </div>
      <div class="info">
        <div class="title">
          <div class="name">{{item.name}}</div>
          <div>{{item.post}}</div>
          <div>{{item.professional}}</div>
        </div>
        <div class="introduction">
          {{item.introduction}}
        </div>
        <div class="item-nav">详细介绍</div>
        <div class="item-nav" @click.stop="toForm">预约挂号</div>
      </div>
    </div>
  </div>
</template>
<style lang="scss">
  @import "../assets/px2rem";
  .list{
    background: #efefef;
    .item{
```

```
display: flex;
margin-top: px2rem(10);
background: #fff;
}
.item:first-child{
margin-top: 0;
}
.img{
position: relative;
flex: none;
width: px2rem(230);
height: px2rem(230);
img{
width: px2rem(230);
height: px2rem(230);
}
div{
position: absolute;
width: 100%;
bottom: 0;
line-height: px2rem(48);
background: rgba(blue, 0.3);
color: #fff;
font-size: px2rem(28);
text-indent: px2rem(20);
}
}
.info{
height: px2rem(230);
padding: px2rem(8) px2rem(8) px2rem(8) px2rem(16);
font-size: px2rem(24);
}
.title{
```

```
margin-bottom: px2rem(8);
div{
  flex: none;
  display: inline-flex;
  margin-right: px2rem(10);
  justify-items: flex-end;
}
}
.name{
  font-size: px2rem(32);
}
.introduction{
  margin-bottom: px2rem(8);
  color: #666;
  height: px2rem(120);
  line-height: px2rem(40);
  text-align: justify;
}
.item-nav{
  display: inline-flex;
  font-size: px2rem(28);
  color: blue;
  margin-right: px2rem(40);
}
}
</style>
```

详情页

```
<template>
<div class="detail">
  <div class="person">
    
```

[illegible]

```
    width: px2rem(230);
    height: px2rem(230);
}
.person{
    display: flex;
    background: #ffffff;
    font-size: px2rem(24);
    div{
        height: 1.5em;
    }
    .info{
        padding: px2rem(10) 0 0 px2rem(20);
        div{
            margin-bottom: px2rem(10);
        }
    }
    .name{
        font-size: px2rem(32);
    }
    .call{
        display: inline-block;
        padding: px2rem(8) px2rem(16);
        border-radius: px2rem(8);
        background: #432eff;
        color: #fff;
    }
}
.section{
    margin-top: px2rem(12);
    background: #fff;
    padding: px2rem(20) px2rem(10);
}
.introduction{
```

```
        text-indent: 2em;
        font-size: px2rem(28);
        line-height: px2rem(48);
        min-height: px2rem(80);
    }
}
```

预约挂号页

```
<template>
  <div class="form">
    <div class="label">您的姓名</div>
    <div class="input"><input type="text" placeholder="请输入您的姓名"
"></div>
    <div class="label">联系方式</div>
    <div class="input"><input type="text" placeholder="请输入电话或手机"
"></div>
    <div class="label">预约日期</div>
    <div class="input"><input type="text" placeholder="请输入预约日期"
"></div>
    <div class="label">您的需求</div>
    <div class="input"><input type="text" placeholder="请简单描述你的病情"
"></div>
    <div class="btn">
      <div>确认提交</div>
      <a href="tel:023-123456"><div>在线咨询</div></a>
    </div>
  </div>
</template>
<style lang="scss">
  @import "../assets/px2rem";
```

```
.form{
  height: 100vh;
  background: #efefef;
  font-size: px2rem(28);
  .label{
    padding-left: px2rem(20);
    height: px2rem(60);
    line-height: px2rem(60);
  }
  .input{
    background: #fff;
    height: px2rem(60);
    line-height: px2rem(60);
    input{
      width: 100%;
      height: 100%;
      border: none;
      outline: none;
      box-sizing: border-box;
      padding-left: px2rem(20);
    }
  }
  .btn{
    display: flex;
    padding: 0 px2rem(170);
    margin-top: px2rem(30);
    justify-content: space-between;
    div{
      width: px2rem(180);
      height: px2rem(50);
      line-height: px2rem(50);
      text-align: center;
      background: #ff9242;
```

```
border-radius: px2rem(8);
font-size: px2rem(24);
color: #ffffff;
}
a div{
background: #432EFF;
}
}
}
</style>
```


附录 B 英文翻译

英文原文

DNA sequencing is critical in understanding the basic structure and function of the human genome by determining the exact nucleotides and their order in a DNA molecule . It also provides the foundational view of the genetic basis for human diseases including cancer . The automated and parallelized approaches of Sanger sequencing directly led to the success of the Human Genome Project and the sequencing projects of other important model organisms for biomedical research (e.g., the mouse genome). The availability of these whole genomes has provided scientists with unprecedented opportunities to make novel discoveries for genome architecture and function, genome evolution, and molecular bases of variations in the human population and disease mechanisms .

In the past decade, the development of faster, cheaper, and higher-throughput sequencing technologies has dramatically expanded the reach of genomic studies. These “ next-generation sequencing ” (NGS) technologies (as opposed to the Sanger sequencing which is considered as first-generation) have been one of the most disruptive technological advances. One demonstration of this exciting technology development is the cost reduction in sequencing. In 2001, the cost of sequencing a million base pairs was about \$5,000; but it only costs about \$0.05 in the mid-2015 (<http://www.genome.gov/sequencingcosts/>). In other words, it costs less than \$5,000 to sequence an entire human genome with 30 % coverage, and this cost continues to drop dramatically, making personalized genomics possible for the general public in the very near future. Illumina sequencing [3] has been the main player in numerous NGS applications, ranging from whole-genome sequencing and whole-exome sequencing, to RNA sequencing, etc. In addition, several other sequencing technologies have been developed in recent years and are sometimes referred as “third-generation sequencing,” with Pacific Biosciences (PacBio) single-molecule real-time (SMRT) technology and the Oxford Nanopore ’ s nanopore sequencing(which produces handheld portable sequencers) being the most promising. It is inevitable that such exciting advancement

will continue in the next few years, with the availability of even higher-throughput, lower-cost, and higher-quality sequencing technologies. Genomics has become one of the most demanding big data disciplines as it has been estimated that we would reach one zettabase (one trillion gigabase) of sequence per year with 100 million to 2 billion human genomes sequenced by the year 2025.

The development of new sequencing technologies has enabled dramatic advancement in our understanding of the human genome and the genetic mechanisms for human diseases. For example, as of March 2016, the NHGRI-EBI GWAS Catalog (<https://www.ebi.ac.uk/gwas/>) contains over 2400 studies and greater than 16,000 unique SNP associations with various human diseases including Alzheimer's, autism, diabetes, obesity, schizophrenia, and many other complex disorders. The 1000 Genomes Project has sequenced over 2,500 individual genomes from various human populations to have a comprehensive resource of human genetic variations. The Cancer Genome Atlas (TCGA) project will characterize the whole genome about 11,000 tumor samples from more than 30 cancer types to understand the genome-wide somatic mutation profiles. However, despite such rapid advancement in genome sequencing and large-scale genomic studies of personal genomes and human disease genomes, our understanding of the genome and genetic variation associated with diseases remains limited.

原文翻译

DNA 测序对于了解人类基因组的基本结构和功能至关重要，因为它决定了 DNA 分子中精确的核苷酸及其顺序。它还提供了包括癌症在内的人类疾病遗传基础的基本观点。Sanger 测序的自动化和并行化方法直接导致了人类基因组计划的成功，以及其他重要的生物医学研究（例如，老鼠基因组）的测序项目。这些完整的基因组的可用性为科学家提供了前所未有的机会，为基因组结构和功能、基因组进化以及人类和疾病机制变化的分子基础做出新的发现。

在过去的十年中，更快、更便宜和更高通量测序技术的发展极大地扩展了基因组研究的范围。这些“下一代测序”（NGS）技术（相对于被认为是第一代的桑格测序）是最具颠覆性的技术进步之一。这种令人兴奋的技术发展的一个例子是测序的成本降低。2001 年，对 100 万对碱基对进行测序的成本约为 5 000 美元；

但在 2015 年年中，它的成本仅为 0.05 美元（<http://www.genome.gov/sequencingcosts/>）。换句话说，用 30% 的覆盖率来测序整个人类基因组的成本还不到 5000 美元，而且这一成本还在继续大幅下降，使得在不久的将来，个性化的基因组学成为可能。Illumina 测序 3 是许多 NGS 应用程序的主要参与者，从全基因组测序和全基因组测序，到 RNA 测序，等等。近年来，已经开发出了其他一些测序技术，有时被称为“第三代测序”，太平洋生物科学（PacBio）单分子实时（SMRT）技术和牛津纳米孔的纳米孔测序技术（生产手持便携式测序仪）是最有前途的。不可避免的是，在未来几年内，这种令人兴奋的进步将会持续下去，因为有更高的吞吐量、低成本和高质量的测序技术。基因组学已经成为最苛刻的大数据学科之一，据估计，我们每年将达到一个 zettabase（1 万亿千兆字节）的序列，到 2025 年，人类基因组测序将达到 1 亿到 20 亿个。

新测序技术的发展使我们对人类基因组和人类疾病遗传机制的理解有了巨大的进步。例如，截至 2016 年 3 月，NHGRI-EBI GWAS 目录（<https://www.ebi.ac.uk/gwas/>）包含超过 2400 项研究，以及超过 16000 个独特的 SNP 协会，这些协会与各种人类疾病有关联，包括阿尔茨海默病、自闭症、糖尿病、肥胖、精神分裂症和许多其他复杂的疾病。千人基因组计划已经对来自不同人类群体的 2500 个个体基因组进行了测序，从而获得了人类遗传变异的综合资源。癌症基因组图谱（TCGA）项目将从 30 多个癌症类型中提取出 11 000 个肿瘤样本，以了解全基因组的体细胞突变图谱。然而，尽管基因组测序和对个人基因组和人类疾病基因组的大规模基因组研究取得了如此迅速的进展，但我们对与疾病相关的基因组和遗传变异的理解仍然有限。