学号 2015301500375

密级_____

（黑体 5 号）

武汉大学本科毕业论文

族谱管理系统平台软件的整合

院（系）名 称：计算机学院

专 业 名 称 ：计算机科学与技术

学 生 姓 名 ：郑斯元

指 导 教 师 ：李蓉蓉 讲师

二〇一九年五月

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名：郑斯元　　日期：2019.4

摘　　要

现有的谱志寻根系统将族谱录入系统和族谱查询系统分离，同时未采用一套统一的用户权限管理，对于系统的使用和推广具有较大的局限性，对族谱的管理和查询修改权限极为不便，移植性也较差。

在本项目中为了解决系统功能分离的问题，进行了族谱录入系统和族谱查询系统的整合并封装成为独立软件，使用一套用户权限管理系统。在合并项目的过程中，使用了 Java web 的 ssh 框架，并且解决了合并过程中的若干问题，包括不同前端框架下网页的整合，网站 jsp 访问的安全机制控制，网站 service 服务合并的依赖关系保持及冲突处理等。

在系统完成后，进行了集成测试和单元测试，以确保系统的稳定性。

关键词：系统整合；权限管理；族谱电子化

ABSTRACT

This project is based on the existing spectrum root - searching system to upgrade. The existing genealogy root-searching system separates the genealogy input system from the genealogy query system, and does not adopt a set of unified user permission management, which has great limitations for the use and promotion of the system, and is extremely inconvenient for the management and query modification of genealogy, and has poor portability.

In this project, in order to solve the problem of separation of system functions, the integration of genealogy entry system and genealogy query system was carried out and packaged into independent software, using a set of user permission management system. In the process of merging projects, the SSH framework of Java web was used, and several problems in the merging process were solved, including the integration of web pages under different front-end frameworks, the security mechanism control of website JSP access, the dependency maintenance of website service merger and conflict handling, etc.

After the completion of the system, integration test and unit test were carried out to ensure the stability of the system.

Key words: System integration; Authority management; Family tree digitization

目　　录

1 绪论

本章内容主要介绍该毕业项目的来源是来自于武汉大学珞珈图腾实验室的科研项目。接着阐述该项目的目的及意义。族谱在当今社会具有重要的研究意义和应用价值，在信息化的过程中遇到一系列的挑战。本实验室在现有族谱平台的基础上进行升级优化，原有系统使用 Java web 进行开发，后台框架为 SSH（struts+spring+hibernate），前端框架包括但不限于 Bootstrap，数据库使用 postgresql。其中，本毕业设计的主要内容是对原有族谱录入系统和族谱查询系统归并整合考虑到原有框架的现状，在合并时存在一定的优点和困难，难点主要包括不同前端框架下网页的整合会导致 UI 风格不统一，jsp 网站存在越权访问的风险，网站 service 服务合并的依赖关系保持及冲突处理复杂等等。

1.1 毕业设计题目的来源

本题目来源于指导教师的科研项目，该项目主要是面向三大传统文献之一的族谱文献进行研究。该项目主要采用大数据文本挖掘技术，分析研究族谱文献的历史人文价值。该项目已有一套专业的族谱数据管理平台软件，但在功能上有需要改善的地方。为了更好的采集族谱数据，同时也面向社会大众提供族谱寻亲交流，需要对该管理平台软件进行完善。

1.2 毕业论文（设计）选题的目的和意义

家谱是一种珍贵的人文材料，是三大历史文献之一。首先，在文化方面，家谱可以促进中国家庭认同感。在学术上，谱系学可以与许多学科交叉，形成有意义的应用。历史方面，家谱可以用于研究家族的迁徙，以作为考据资料考证。医学方面，家谱可以用于调查家族遗传病史的遗传规律，对医学的发展具有重要意义。社会方面，家谱图的建立对社会关系具有重要的促进作用。

谱系学本身有许多重要的意义。就系谱学本身而言，随着大数据的发展和中国传统文化圈的文化传承，将传统纸质系谱学进行编纂、整理和转化为可搜索的结构化数据库和图表已成为当今社会发展的必然。然而，传统的家谱使用纸质媒介书写，难以保存和维护。家族树的时代信息是不均衡的，难以管理和检索。传

统家谱记载内容繁多，编撰格式复杂。传统宗谱学存在着宗谱多、版本不同、分支不完整、条目重复等问题。在当今信息时代，随着信息技术和文档数据化技术的发展，传统的家谱信息与现代信息技术相结合，产生了具有重要意义的电子家谱。许多图书馆、博物馆和其他公共家谱收藏机构，以及商业公司，都已开始将家谱数字化，并取得了不同程度的进展。一些专业系谱修订机构在数字系谱衍生电子系谱、网络系谱等数字产品的基础上，可以快速实现对字符和关系的查询。

电子族谱由于诞生时间较晚，如今发展还不够成熟。在业内还没有谱例的规范，不同电子族谱系统之间也没有共享数据信息，即当前还没有一个大型的开源的可用于推广的族谱平台供大众使用。数字化家谱在当今发展仍然面临一些问题，诸如修改经费成本巨大，修改族谱难度过高等等。我们的目标就是建立一个使用最新技术的大数据平台，用来承载传统的族谱信息，可以方便地进行族谱的录入，查找，和生成等等功能。我们目前的工作是完善现有的某族谱管理系统，该系统已经初步实现了预期的某些功能，但是仍然具有一些不足，所以我们后续的升级开发人员仍然要与时俱进地进行完善改进。

原有族谱网站在族谱的录入、管理、展示功能上已经十分的完善，由于前期主要面向族谱制作商业领域，该系统主要功能集中在谱志的编辑和生成。现在需要面向普通大众，使广大用户在该系统上能够进行方便、友好、快捷的访问。但是，原有族谱管理系统将族谱录入系统和族谱查询系统分离，十分不便。两个网站原本使用相同的 web 框架，具有类似的 Dao 层和持久层操作，却分别使用各自分离的数据库，浪费了大量存储和计算资源。同时，两个网站具有不同的前端框架，在用户感官上难以和谐统一。除此之外，两个网站采用了各自的用户认证系统，在安全性上存在隐患。出于安全性，在设计之初数据录入平台和姓氏网站展示平台是两个独立的平台。不能实现单点登录。但是在当前环境要求下，需要使用一次登录就能够访问两台平台，即将两个平台整合在一起。姓氏网站的访问需要通过索取号来查看族谱信息。在整合录入平台和姓氏网站的同时，用户需要能够访问自己录入的族谱，并根据开放权限来访问其他的族谱。最后，兼容平台版本较为陈旧，不能适应当下信息化检索的要求。本着与时俱进的要求，从用户体验考虑的角度出发，以如无必要勿增实体为原则，现拟对原有的族谱录入网站和族谱查询网站的功能进行整合归并，交互界面进行 UI 风格的统一，以期完成

一个功能较为完善，划分科学人性化，界面较为友好的族谱综合管理系统，在族谱电子化的进程中实现里程碑式进展。

我的毕业设计的主要工作就是修改现有的族谱数据采集平台后的族谱综合管理系统，将姓氏网站集成到族谱录入采集平台。当用户登录时，既能根据需要进行族谱的录入，也能查看自己录入的族谱，还能查看公开的族谱信息以及有权限访问的族谱信息。当没有权限时，可以向族谱创建者申请访问。该系统应能够实现以下功能： 1）用户能够方便快捷的进行登录。用户以唯一的账户登录系统，根据用户自身角色实现对平台的操作。2）整合数据录入和姓氏网站平台，实现单点登录访问。整合族谱的 web 平台，实现统一用户访问。3）修改姓氏网站功能，使其不需要通过索取号来访问族谱信息，而是根据权限的定义来访问姓氏网站。4、用户能够在手机、平板电脑等移动平台上进行方便的操作和访问。

1.3 国内外关于该选题的研究现状和发展趋势

1.3.1 Java web 后台发展历史

在三十年前， 因特网和浏览器技术获得了巨大进步发展，由于用户访问需求复杂，为实现根据用户的不同请求，动态地处理并返回资源，servlet 技术应运而生。在发展初期，用户使用浏览器向服务器发送的请求的资源大多数都是 HTML，css 这样静态的资源。此时，Web 开发还比较简单。图 1.1 表示表示了最简单的浏览器与 web 服务器之间数据交换的关系。

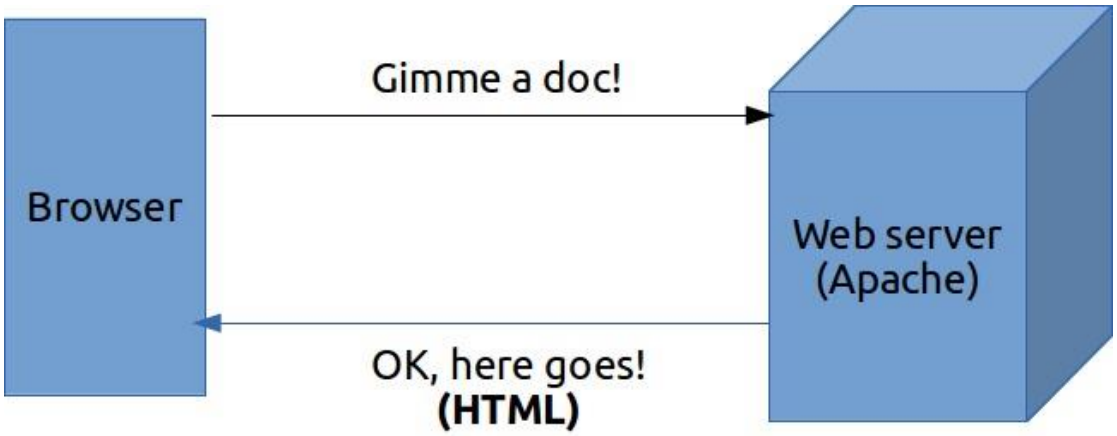

图 1.1 B/S 系统的数据交换

这种 web 界面只能实现最简单的功能，比如说只能获取到静态内容无法实现统计流量，查看访客，填写包含有姓名和邮件地址的表单等等。CGI 和 Perl 脚本的出现很大程度上解决了上述问题，它们与文件系统或者数据库进行交互只需要通过在 web 服务器端运行一段短小的代码。其流程结构如图 1.2 所示。

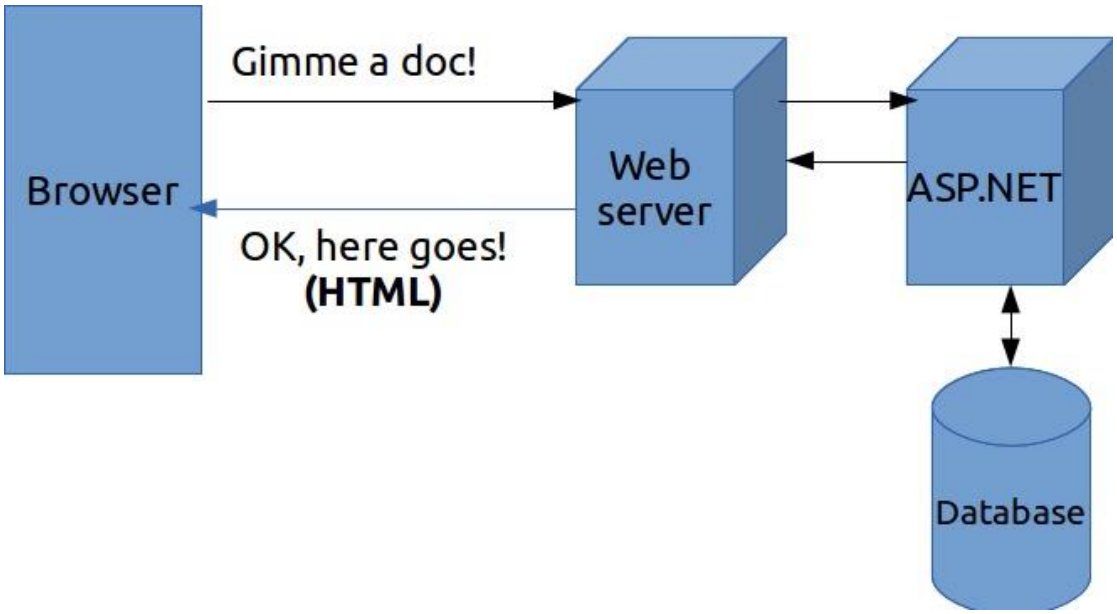

图 1.3 CGI 结构的数据交换过程

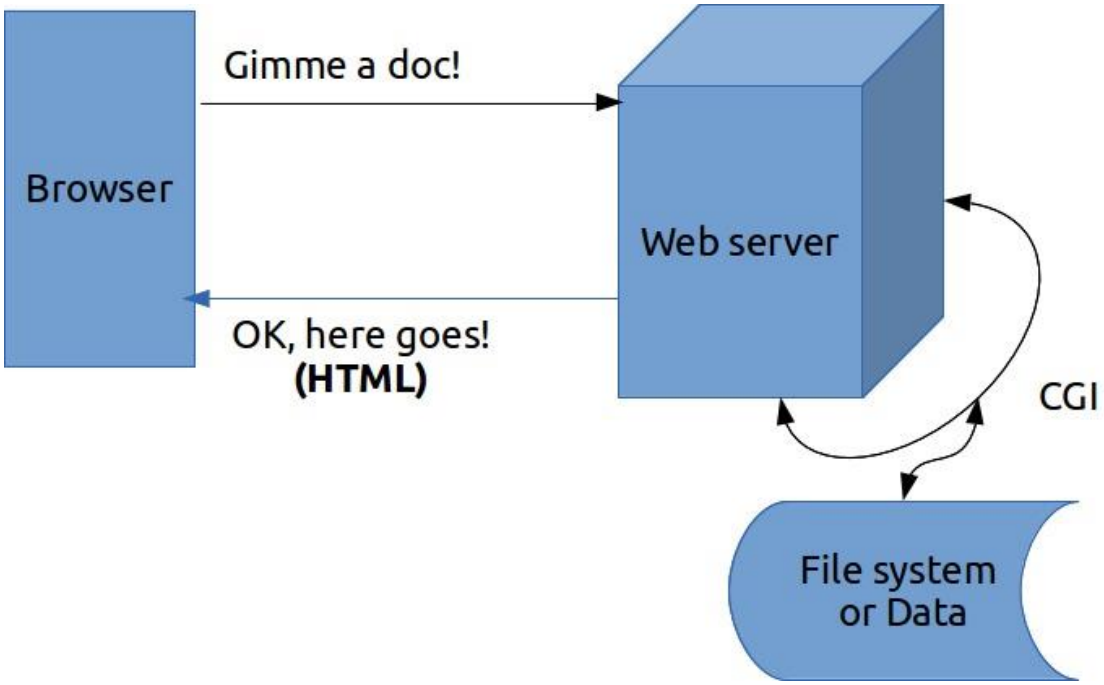

图 1.2 ASP.net 结构的数据交换过程

而 CGI/Perl 仍然存在一系列问题。首先，CGI 较为混乱并且不具备良好的伸缩性。由于直接使用文件系统或者环境变量，安全性无法保证。之后，出现了一系列新的用于 web 开发的语言，包括 Java Server Pages(JSP)，微软的 ASP，以及 PHP。Jsp 始于 sun 公司，它们认为 servlet 编程很繁琐，同时开发者需要具备一定的前端和美工知识，同时需要在 Servlet 中写前端代码，不够直观。在这种情况下，sun 公司开发出了 jsp。该方法在界面中使用<% %>HTML 脚本标识嵌入所运行的 Java 代码。这一阶段的架构可以比作 IIS 和 ASP.NET，web 开发者可以用 Visual Studio 等工具平台快速构建一个可伸缩并且安全的应用程序。其流程结构如图 1.3 所示。

Jsp 出现后，仍然存在相关问题，在这种背景下， JSP+JavaBean+Servlet 横空出世，它使用了 MVC 思想。C 指 Servlet，它可以根据用户的请求调用相关的业务，并返回对应的 jsp 页面给用户。M 对应 JavaBean，它完成服务器所需的处理操作。V 指 JSP，它主要使用 HTML 标签和 JSP 标签完成输入和输出。其结构如图 1.4 所示。

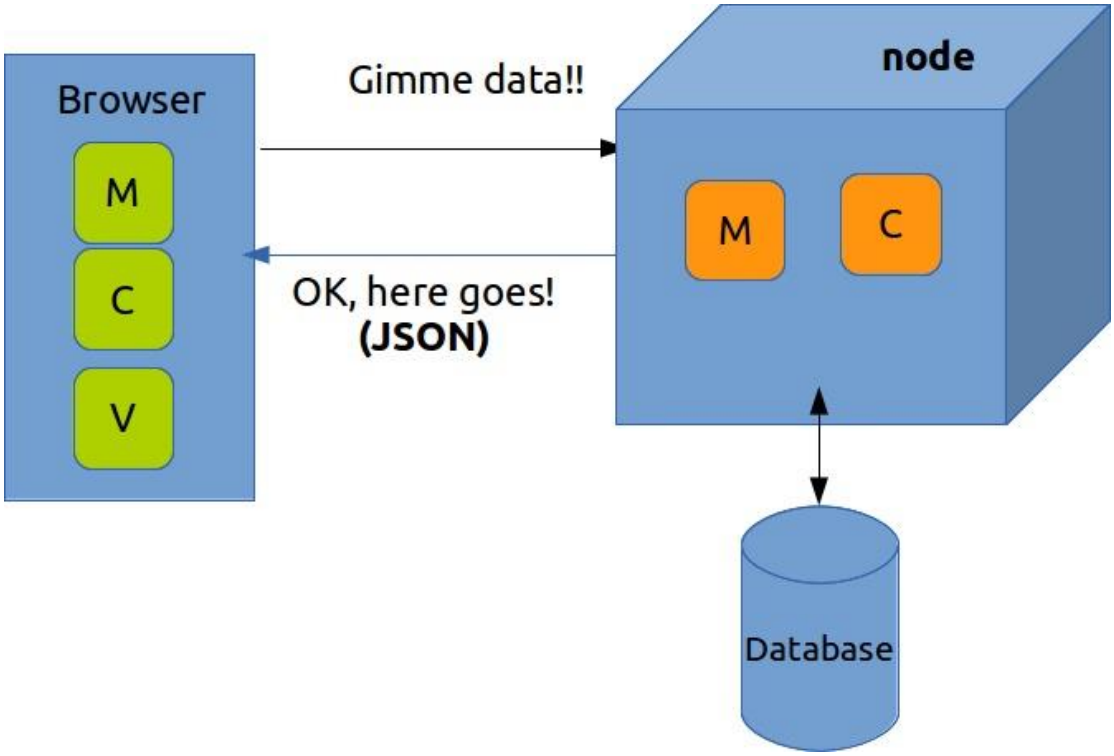

图 1.4 MVC 结构的数据交换过程

在 2001 年 6 月，struts1.0 出现，它拥有一套针对 jsp 的 struts 标签，从而使得 jsp 中没有了 Java 代码，提高了 jsp 的编写效率，使得逻辑更加清晰。在 Struts 下，Action 类可以被用来代替 servlet，它能够进行一些请求过滤和自动转码的原本属于 servlet 的功能。为了实现高内聚、松耦合的思想，Spring 框架[2]出现了，它可以实现 IOC（控制反转）和 AOP（面向切面的编程）两大功能。SpringMVC 通过“基于注解”的方式代替了 struts，并且通过 Controller 类来代替和实现了 Action 的功能。

1.3.2 前端框架发展历史

在最早的网页出现的时候，前端页面都是静态的，所有前端代码和前端数据都是后端生成的。前端只是纯粹的展示功能，脚本的作用只是增加一些特殊效果。而 2004 年 AJAX 的诞生改变了前端开发。就是从这个阶段开始，前端脚本开始变得复杂，不再仅仅是后端的模板，而是实现了从读写数据、处理数据、生成视图等功能的完整业务逻辑。因此迫切需要辅助工具，方便开发者组织代码。这导致了前端 MVC 框架 Backbone.js 在 2010 年的诞生。它把 MVC 模式搬到了前端。后来，更多的前端 MVC 框架出现。而在 SPA 阶段，前端可以做到读写数据、切换视图、用户交互，这意味着，网页其实是一个应用程序，而不是信息的纯展示。所谓 SPA，就是指在一张网页（single page）上，通过良好的体验，模拟出多页面应用程序（application）。用户的浏览器只需要将网页载入一次，然后所有操作都可以在这张页面上完成，带有迅速的响应和虚拟的页面切换。随着 SPA 的兴起，2010 年后，前端工程师从开发页面，逐渐变成了开发“前端应用”。

1.3.3 数据库发展历史

20 世纪 60 年代，在数据管理目的的推动下，数据库技术兴起，成为计算机科学的一个重要分支。数据管理，诸如分类，编码，检索，维护，存储等操作，是数据库的核心任务。随着计算机软硬件的发展，数据库技术也在不断发展。从数据管理的角度看，数据库技术经历了手工管理、文件系统和数据库系统三个阶段。人工管理阶段是指计算机诞生的早期阶段，科学计算是这一阶段计算机的主要用途。文件系统阶段是指计算机可以大量管理数据的阶段。而 60 年代末面向一个应用程序或一个程序，而是面向整个企业或整个应用程序的数据库的出现，

标志着数据库系统阶段的开始。数据库技术的形成过程包括以下三个阶段:

A. 第一代数据库系统 层次和网状数据库管理系统

B. 第二代数据库系统 关系数据库管理系统

C. 新一代数据库技术的研究和发展

1.4 难点分析

1.4.1 不同前端框架下网页的整合

原有的两个 Java web 系统，使用了不同的前端框架，原族谱录入系统的界面使用了原生 jsp，如 restoremanager.jsp 等等，而族谱查询系统的界面则使用了 bootstrap 框架，如 showthepedigree 等等。于是在编写 jsp 时出现了问题。如果不使用 bootstrap 框架，则导入后的页面无法保持原有的界面格式。如果使用 bootstrap 框架，则原有的 header 和导航栏会发生 UI 风格的冲突。所以，在按照上述方法编码时，可能会导致因为框架不同而使界面风格难以统一。

1.4.2 网站 jsp 访问的安全机制控制

当前的族谱管理系统具有严格的用户权限管理机制，不同权限的用户仅仅可以访问有限的内容。而在网页的访问过程中，可能会存在一定的安全隐患。比如某些攻击者可能会获取 jsp 文件的命名方式，从而越过登录校验直接越权访问相关链接。此时，进行网页的防跳转是必要的。

1.4.3 网站 service 服务合并的依赖关系保持及冲突处理

两个系统的 hibernate 文件有所差异，且由于历代的修改，maven 的依赖关系十分复杂，不利于调试。在迁移的过程中，如果一旦破坏了原有的依赖关系，将使得系统的整合工作十分困难。

2 已有系统的介绍

本章主要介绍原有的族谱录入系统以及族谱查询系统均使用了 SSH（struts+spring+hibernate）框架，具有相似的后台框架结构，再合成时应充分考虑到二者的共同点，进行对应层次的整合。本章简单介绍了 ssh 框架的结构以及从功能上说，应对原有功能划分进行整合归并，实现功能的精简化和合理化。最后，本章在第三节对原有族谱录入及查询系统的编码方式进行了分析，为后文的整合建立了基础。

2.1 平台及框架分析

现有的族谱录入系统及族谱查询系统是两个单独分离的不包含离线数据库的 Java web 项目，在 jdk1.8 及配套版本的 jre 环境下可以用 eclipse IDE 打开，可以在 Tomcat8.5 服务器上运行，数据库使用 postgresql10.5。

2.1.1 SSH 框架介绍

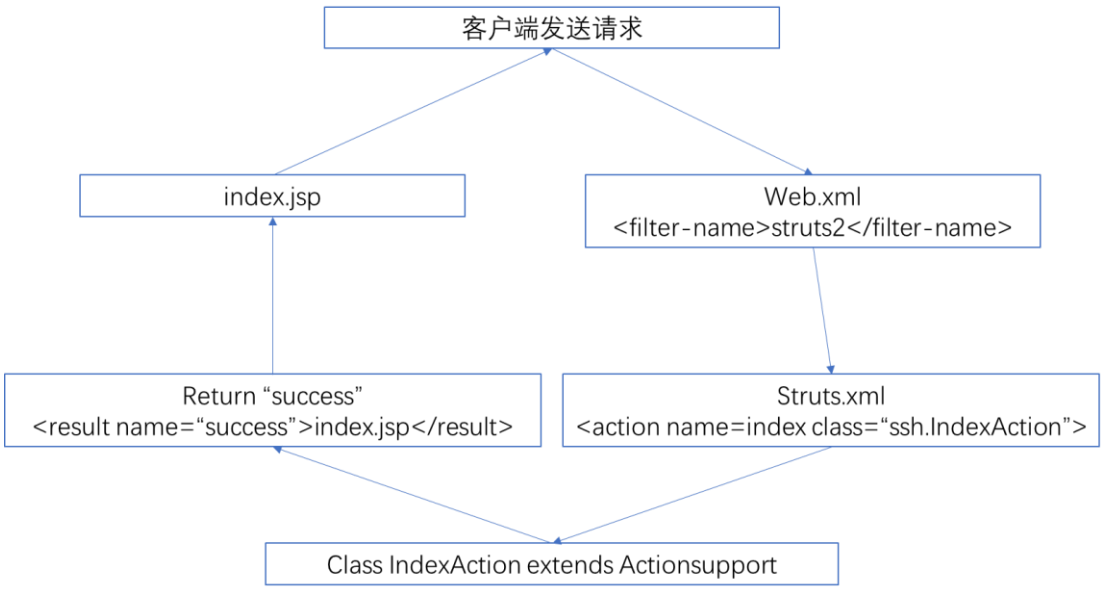

图 2.1 SSH 框架各部分协作处理请求的过程图

两个系统均使用 SSH 框架，是目前一种流行的 web 框架，主要分为四层，

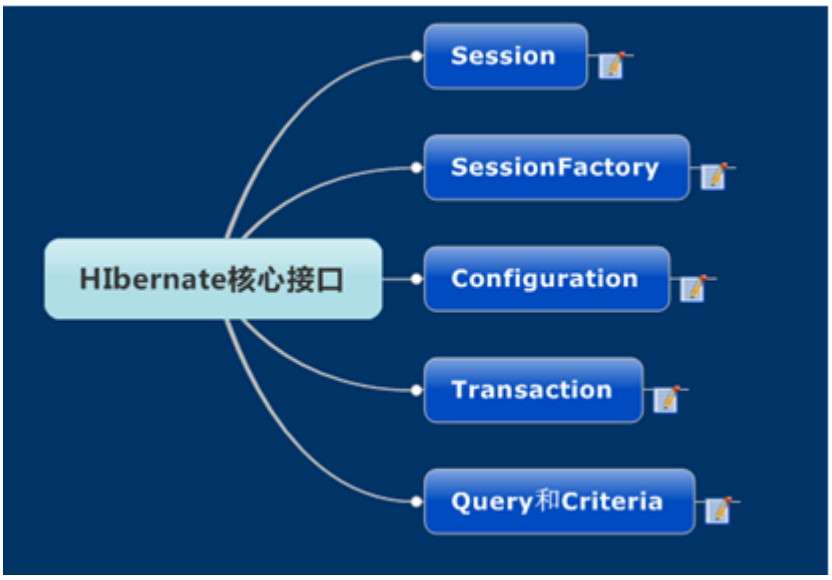

图 2.2 Hibernate 接口结构图

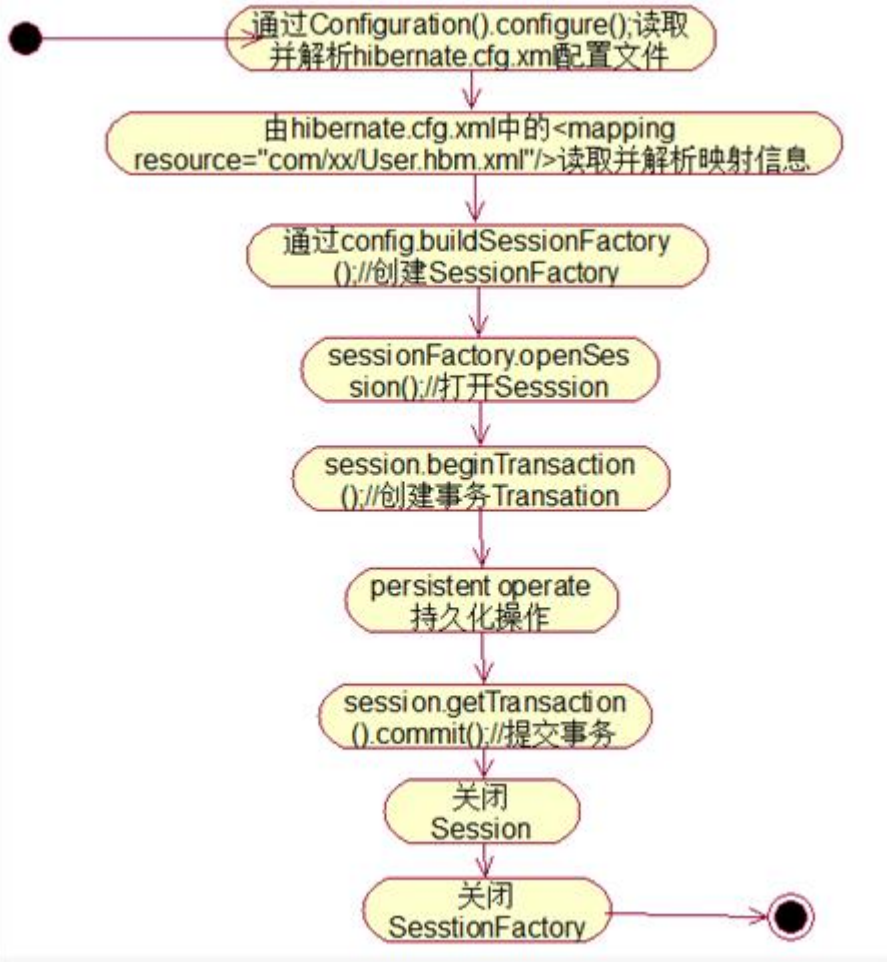

图 2.3 Hibernate 流程图

和用户直接进行交互的表示层、执行 Action 的业务逻辑层、与数据库打交道的 DAO 层和数据持久层，有助于开发人员进行有序清晰的 web 应用开发，其处理客户端请求流程结构如图 2.1 所示。

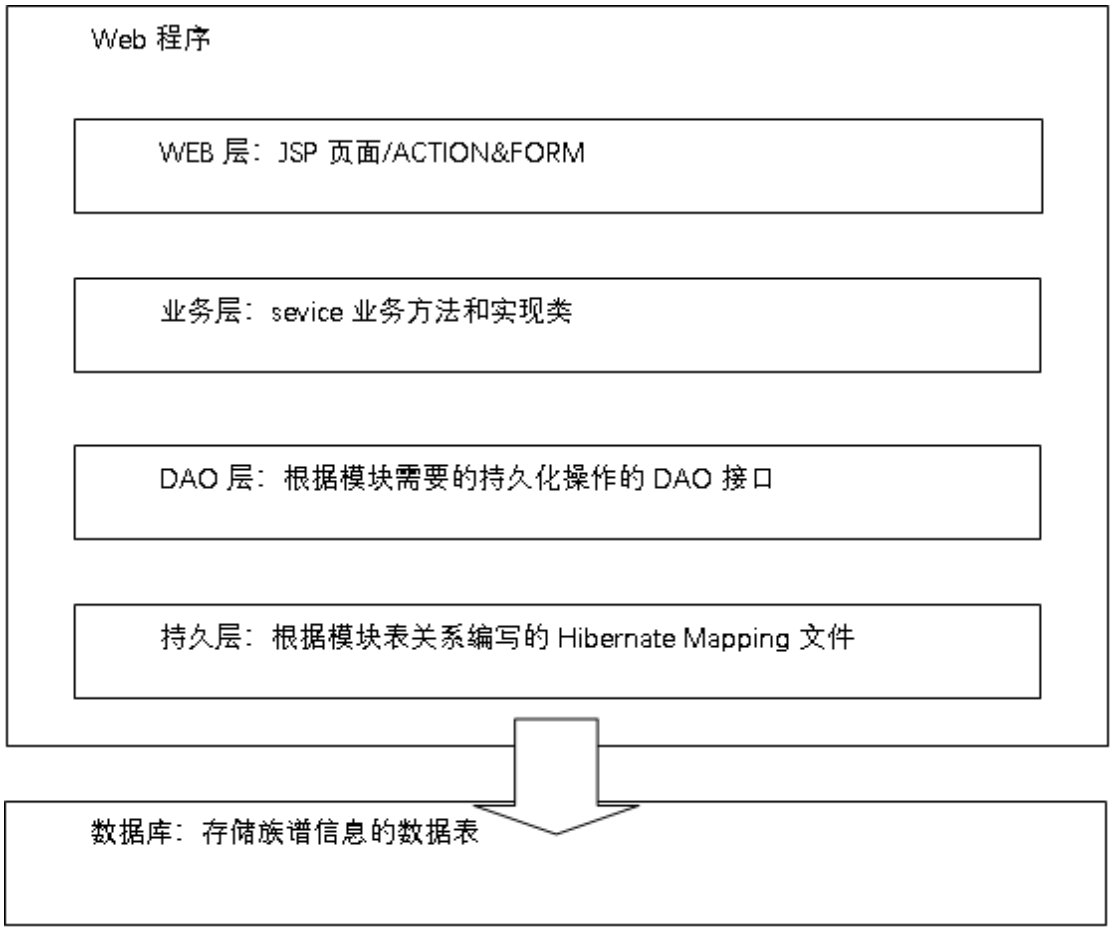

图 2.4 SSH 框架结构图

Hibernate 是一种对象关系映射框架，主要用于 SSH 框架中与底层数据库进行交互，以实现实体化 bean 向符合对象关系映射的框架，将对数据库的操作转化为对 Java 对象的交互，有效降低了不同数据库的差异性带来的开发难度。并能生成执行 SQL 语句。Hibernate 主要包括以下核心接口，如图 2.2 所示。

而 Hibernate 处理事务也主要是由 session 实现的，它负责执行被持久化对象的 CRUD 操作，其流程图如图 2.3 所示。

在现有的族谱录入系统和族谱查询系统分析的基础上可知，二者均使用了 SSH 框架，因而具有类似的程序层次结构，包括 WEB 层，业务层，DAO 层，持久层等等，共同点如图 2.4 所示。

因而，二者的程序运行流程也具有共性，都是服务器识别客户端的请求，再识别 web.xml，解析标签查找，找到配置文件中 action，从而执行后台业务处理函数。其过程用图 2.5 表示：

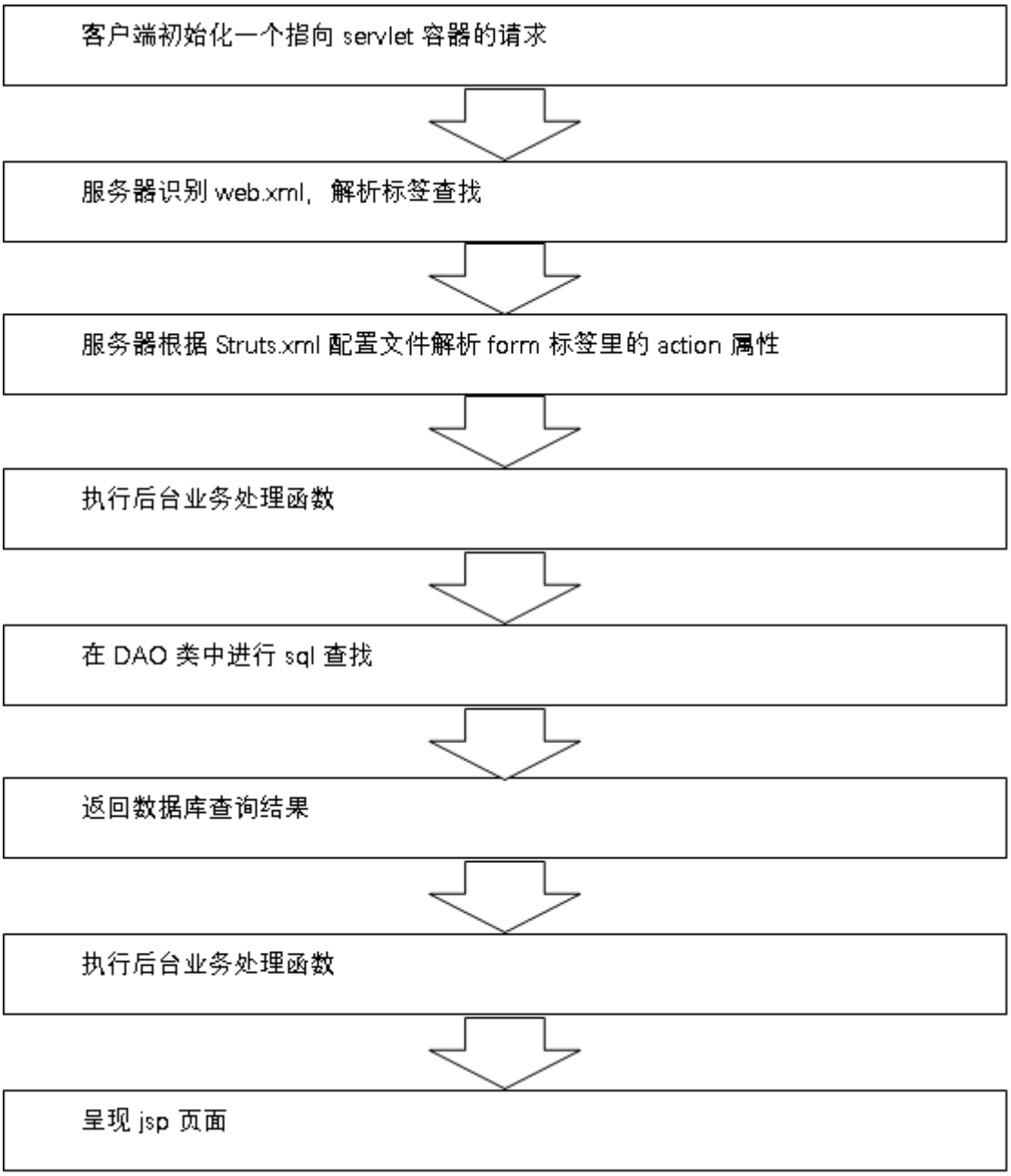

图 2.5 SSH 运行流程图

2.1.2 bootstrap 框架介绍

原有的族谱查询系统使用了 bootstrap 作为前端框架。Bootstrap 框架的许多组件都需要使用 JavaScript 才能运行。具体来说，它们需要 jQuery，Popper.js 和

用户自己的 JavaScript 插件。Bootstrap 支持所有主流浏览器和平台的最新稳定版本，包括 Windows 上的 Internet Explorer 10-11 / Microsoft Edge。它使用了一些重要的专门针对跨浏览器样式的规范化的全局样式和设置。 Bootstrap 不直接或通过平台的 Web 视图 API 使用最新版本的 WebKit，Blink 或 Gecko 的替代浏览器未明确支持。但是，Bootstrap 应该在这些浏览器中正确显示和运行。Bootstrap 使用 Autoprefixer 通过 CSS 前缀处理预期的浏览器支持，通过使用浏览器列表来管理这些浏览器版本。Bootstrap 包含很多个定制的用于 jQuery 的插件，可以单独包含（使用 Bootstrap 的个人 js/dist/*.js），也可以一次性使用 bootstrap.js 或缩小 bootstrap.min.js（不包括两者）。如果使用 bundler（Webpack，Rollup ...），则可以使用/js/dist/*.jsUMD 就绪的文件。插件之间也是可能互相关联的，如果单独包含插件，则需要在文档中检查这些依赖项。几乎所有的 Bootstrap 插件都可以通过 HTML 单独使用数据属性来启用和配置，确保只在一个元素上使用一组数据属性。但是，在某些情况下，可能需要绑定文档命名空间上的所有事件来禁用此功能。Bootstrap 包含丰富的 Web 组件，譬如下拉，弹出窗口和工具提示也取决于 Popper.js。您可以自定义引导组件、更少的变量和 jQuery 插件来获得自己的版本。

2.1.3 postgresql 数据库介绍

PostgreSQL 是一个从 UCB 写的 POSTGRES 软件包发展而来的免费的对象-关系数据库管理系统。在 PostgreSQL 中，即使数据库和操作它们的服务程序在同一台机器上，这些应用程序在访问存储在数据库中的数据的时候，仍然无法直接访问数据，必须通过数据库进程。

2.2 功能模块分析

2.2.1 族谱录入系统

目前现有的族谱录入系统功能主要包括两级菜单：

A．功能导航

功能导航模块目的旨在引导族谱录入人员了解族谱录入及管理流程，包括六个步骤：组建团队-创建族谱-数据录入-生成排版-生成 PDF-校对管理

B．数据管理

数据管理模块是族谱录入系统的核心模块，包括六个二级模块：

1）族谱管理

管理当前已录入族谱的信息，包括族谱名称，录入者/团队，负责人等等信息。可进行增删查改等操作。

2）文档模板管理

本模块主要用于管理族谱录入及展示时相关信息的格式。可以添加或删除分类。

3）数据备份

用于将数据备份到服务器或本地。

4）数据恢复

用于从服务器或本地恢复数据。

5）文档顺序管理

用于新建或删除文档顺序规则。

6）直系与总源关系管理

用于支系族谱管理。

C．系统管理

系统管理用于管理本系统用户的使用权限，主要包括九个二级模块。

1）超级管理员管理

添加，删除，修改超级管理员。

2）代理商管理

添加，删除，审核，修改代理商。

3）谱志管理员管理

修改，删除谱志管理员。

4）族谱编辑管理

修改，删除谱志编辑

5）族谱录入员管理

修改，删除谱志录入员

6）团队管理

删除团队

7）当前用户信息修改

8）修改密码

本模块主要用于管理族谱录入及展示时相关信息的格式。可以添加或删除分类。

9）退出

2.2.2 族谱查询系统

现有族谱查询系统的功能模块划分并不清晰，搜索族谱的功能在多个模块均有入口。经过分析，得出以下主要功能：

A．族谱查询

用户通过输入关键字，或通过优先推荐按钮，或通过省份模糊查找，得出相关族谱的搜索结果。

B．族谱推荐

在页面上展示一部分具有代表性的族谱，包括网络族谱，热门族谱等等。可以通过首字母选择跳转到具体的某一行。通过某一行中特定姓氏的访问可以进入某个姓氏族谱的详细信息页面。

C．索引认证

原有族谱查询系统的用户登录系统。

D．外部链接

原有族谱查询系统的开发公司中根网的官网首页。

2.3 编码分析

2.3.1 族谱录入系统

族谱录入系统的导航页面为 newnav.jsp，通过对用户身份的读入判定不同的权限，从而获取不同级别的网站访问入口。在最高等级的管理员权限下，可以访问 funcNavi.jsp 总体流程页面，MangeGenealogy.jsp 族谱管理页面，DTManager.jsp 文档模板管理页面，backupManager.jsp 数据备份页面，restoreManager.jsp 数据恢复页面，docruleManager.jsp 文档顺序管理页面，branchtotrunk.jsp 支系与总源关系管理页面，userselect.jsp 超级管理员管理页面，dailiman.jsp 代理商管理页面，superpedman.jsp 族谱管理员管理页面，superedit.jsp 族谱编辑管理页面，

superinput.jsp 族谱录入员管理页面，superteamman.jsp 团队管理页面，changeinfo.jsp 当前用户信息修改页面，pwd.jsp 修改密码页面，index.jsp 退出页面。在跳转到特定页面时，网站 title 属性显示该页面所在的分组名，该分组名显示由 function 下的$('#navbar').accordion('select', xxxx');语句实现。整个页面由 head 和 body 两部分组成，head 显示分页面的顶端信息以及左侧导航栏，body 显示正文信息，即特定页面的具体内容。

族谱录入系统的 service 层的 action 文件包括，backupaction.java 用来初始化配置文件，baseaction.java，用来实现基本的 http 通信要求，disposalaction.java 用来对家谱树进行操作，docaction.java 用来对文档进行操作，fileuploadaction.java 用来控制文件上传，genealogyaction.java 用来显示分组，Genwordaction.java 实现对谱志的编辑，pedivolumeaction.java 用来操作族谱分卷，searchaction.java 用来检索人物，Tindividualaction.java 用来更新或添加人物信息，Tphotosction.java 实现对照片的操作，Trelationaction.java 实现对人物树的操作，useraction.java 对录入人员进行管理。

Dao 层及持久层多为 SSH 框架下的原生功能针对项目适当修改后的工程文件，以实现对底层数据库的连接，操作与交互。

2.3.2 族谱查询系统

族谱查询系统的导航栏在 newnav.jsp 中实现，在用户访问时没有 jsp 的访问权限控制。用户认证被放在 login.jsp 即索引认证界面中。index.jsp 是首页，同时包含族谱查询和展示模块。Showthegenealogy.jsp 是网络族谱页面，包括网络族谱和热门族谱，同时是某个姓氏族谱的展示页面 showoneperson.jsp 的上一级入口。Searchthegenealogy.jsp 是搜索族谱页面，可以通过省份搜索或模糊搜索实现对族谱的检索。而在外部链接跳转时，则没有专门的 jsp 文件，通过在标签中加入<a href="http://www.zongen.com/">实现对中根网官网的跳转。

族谱查询系统的 service 层的 action 文件较为简单，包括，backupaction.java 用来初始化配置文件，与族谱录入系统完全相同，genealogyaction.java 完成余下族谱检索，族谱展示的全部主要功能。

Dao 层及持久层编码与族谱录入系统类似，包括 BaseDao.java 等，主要与底层数据库进行交互。

3 整合系统设计

本章的主要内容是对整合后的系统进行一个整体的设计。从系统框架开始分析，到系统的功能划分与整合：保留原本族谱录入系统作为族谱录入模块，合并加入原本族谱查询系统作为族谱查询模块，二者使用同一个用户认证管理系统，减少了原本族谱管理系统的不便与冗余。针对前文所述的系统整合过程中提出的三个问题，提出了一一对应的解决思路。对于网站的前端框架不统一，使用 iframe 进行整合；对于 jsp 网站的访问权限管理，增加了 filter 过滤器；对于网站 service 服务合并的依赖关系保持及冲突处理，使用取二者并集的思路。从而解决问题实现编码。

3.1 平台环境设计

鉴于之前的两个系统均在相同的环境下可以运行，于是整合后的系统仍然使用 Eclipse+jdk1.8+Tomcat8.5+postgresql10.5 环境。

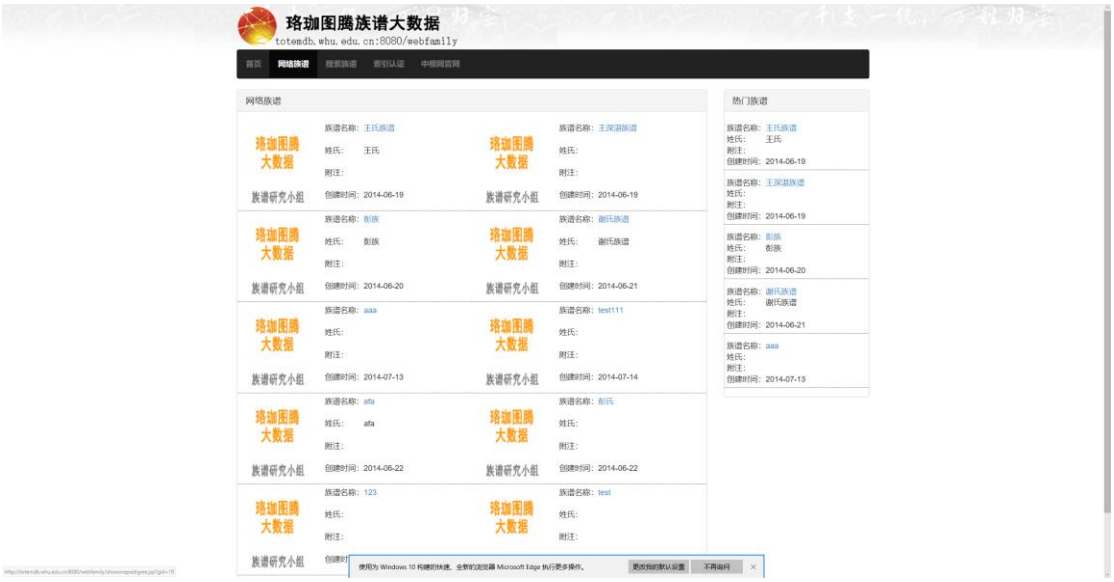

图 3.1 原有族谱查询系统 1

3.2 设计思路分析

如 3.1 所述，两个系统具有类似的程序结构，且在相同环境下均可运行。因

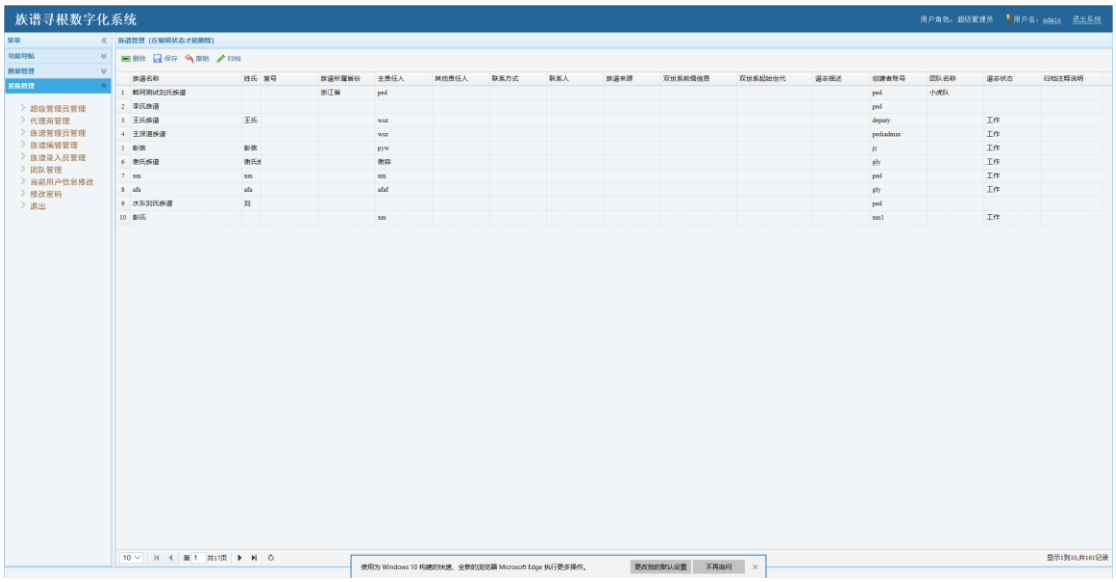

图 3.2 原有族谱录入系统 2

而合并时可行性较高。在分析两个系统的结构之后，划分 web 层，业务层，DAO 层和持久层文件界限。由于族谱录入系统的功能较为复杂，且拥有一套用户信息管理系统，现考虑将族谱查询系统的文件按层次划分后修改合并加入原族谱录入系统中，作为族谱查询模块。业务层及以下尽量保持原有依赖关系。考虑到原有的两个系统界面差异较大，如图 3.1，图 3.2 所示，所以族谱查询模块的 jsp 文件需要重新编写。

3.3 系统整体设计

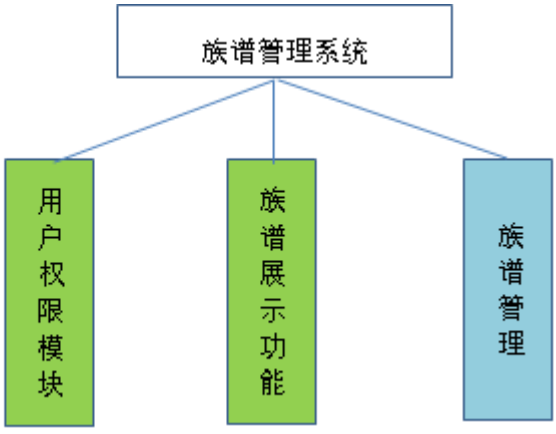

图 3.3 系统功能结构图

3. 3. 1 功能定义

系统功能：添加用户、删除用户、修改用户、指定权限。

业务功能：创建族谱、族谱编辑、族谱录入

族谱展示：查看族谱、申请访问、开放权限

基于以上结果，以及精简管理员结构的需求，调整整合后的族谱管理系统详细结构如图 3.3，图 3.4，图 3.5，图 3.6，图 3.7 所示。族谱管理模块包含族谱录入模块，族谱查询模块，系统管理三个子模块。族谱录入模块包含数据备份等六个子模块，族谱查询模块包括族谱查询等四个子模块，系统管理模块包括超级五个子模块。

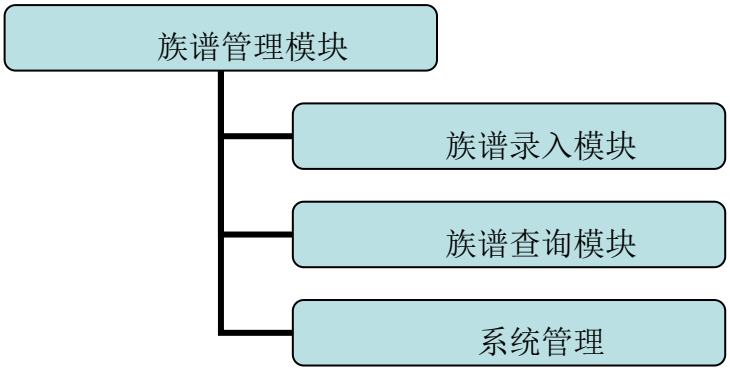

图 3.4 功能结构图第一层族谱管理模块的功能划分

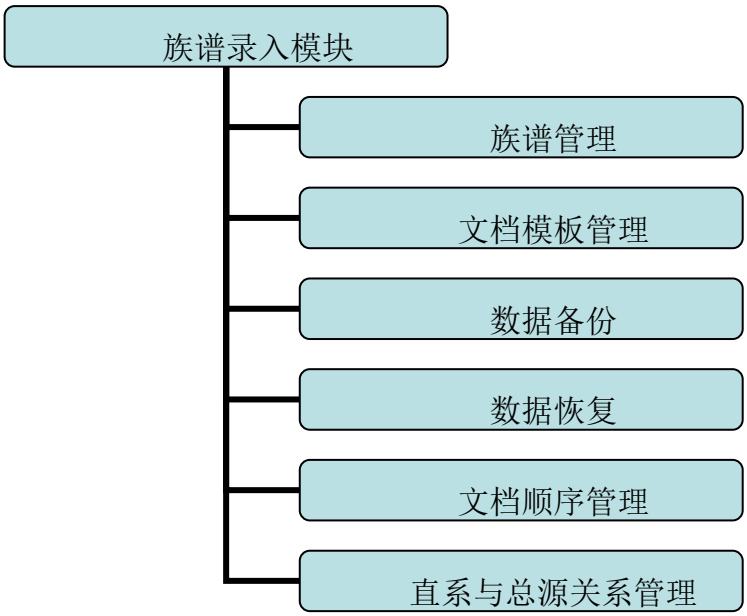

图 3.5 功能结构图族谱录入模块功能划分

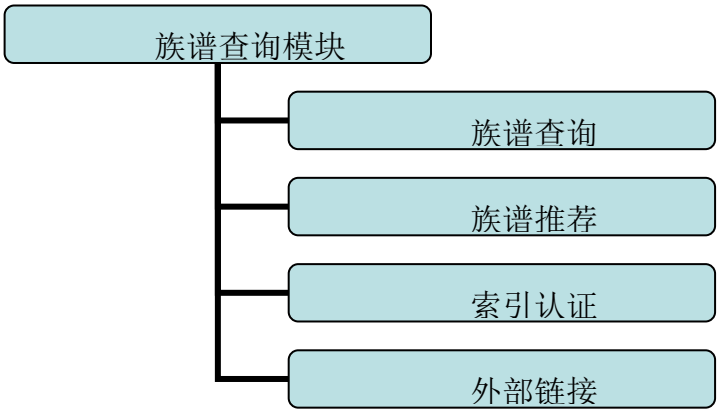

图 3.6 功能结构图族谱查询模块功能划分

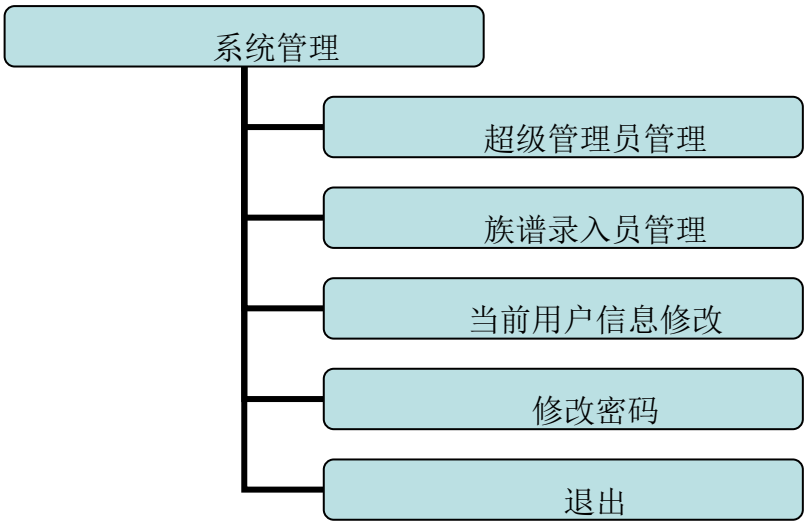

图 3.7 功能结构图系统管理模块功能划分

3.3.2 用户定义

3.3.2.1 管理员

主要实现系统功能，实现对用户的添加、删除、修改。指定用户的权限，当在特定的时期，可以停用系统的业务功能和展示功能。

系统管理员也可以进行普通用户的所有权限。能够批量开放所有族谱的查询、编辑、录入功能。

3.3.2.2 普通用户

普通用户除了具有基本的账户功能（修改个人资料、密码等）外，主要拥有业务方面的功能。

访客只能进行基本信息的简单的访问，要对族谱的内容进行查看，或对族谱的内容进行查找，必须时合法的注册用户。

注册用户能够创建自己的族谱，管理自己的族谱，向其他申请者开放自己族谱的编辑、录入、访问、修改等权限。

注册用户可以申请查看、录入、编辑、管理其他族谱信息。

3.4 系统功能模块设计

3.4.1 系统管理模块

由于此前该系统主要给公司使用，导致超级管理员，代理商，负责人录入员等关系过于冗杂，不利于系统的推广使用，因此有必要对用户管理模块进行精简。

对于整合后的族谱管理系统，用户登录后，应能够根据自身的身份，在后台界面中生成不同的菜单。

A．对于普通用户而言有如下权限菜单：

1）用户自身的管理（资料修改、密码修改等）

2）族谱管理菜单组；此功能模块已经实现，不需要大的修改。直接将链接接入即可。如必要，需要在相关页面中加入身份认证模拟。

3）族谱展示菜单：查看族谱（查看自己创建的或公开的族谱）；申请访问（根据需要，申请查看、申请编辑、申请管理自己感兴趣的族谱）；开放权限（可以开放自己创建族谱的权限、批准申请访问的人员进行相关权限的操作）。

B．对于管理员，除了能够具有普通用户的所有功能外，还要能够有如下的菜单：

1）拥有对除管理员自身的，所有用户的管理、删除、修改。管理员可以冻结用户、指定某些用户不能访问等。

2）能批量开放族谱的展示、查询、编辑、管理等权限。即网站在特殊的时刻为了安全考虑，可以不开放族谱的管理功能。

3.4.2 族谱录入模块

对于原系统中本身具有的族谱管理功能模块，其功能可以不做大的修改，直接和修改后的用户模块进行对接即可。

但是要求能够使用兼容式的页面，适应多种平台和浏览器。能够实现方便快

捷的录入和编辑。

3.4.3 族谱查询模块

在展示页面，主要是将原姓氏网站的功能纳入到其中，并修改其中的一些功能。 修改原姓氏网站中，通过索引号访问族谱内容的方式。直接改为如下原则：

访客只能查看族谱的基本信息（名称、统计信息、录入人员、时间、世系代数等），不能查看具体的内容。 登录用户只能查看自己管理和参与的族谱的所有内容，也能够查看面向所有用户公开的内容。当登录用户需要查看其他不公开的族谱时，需要向该族谱创建者提出申请。 登录用户自己也可以对自己创建的族谱选择公开的权限范围，或者对申请者的申请进行同意和拒绝。管理员登录族谱展示页面时，可以不受限制的访问和管理任何一个族谱的详细内容。但不可以对申请者进行批准或拒绝。 原姓氏网站的内容的展示要求能够使用兼容式的页面，适应多种平台和浏览器。

原族谱查询系统中实现外部链接的跳转采用的是标签打开外部链接而非网页内的 jsp 文件，集成后为保持 ui 风格的统一应重新撰写一个简单的 jsp 界面实现跳转。同时，跳转的目标也不应为中根网官网，而是改成武汉大学珞珈图腾实验室首页。

余下基本的网络族谱热门族谱展示，及族谱的查询包括模糊查找和地图查询，应尽量保持原有功能不变。

3.5 难点问题解决

在上文 1.4 中，我们提出了本项目设计的三个问题，分别为不同前端框架下网页的整合，网站 jsp 可能存在越权访问，以及网站 service 服务合并的依赖关系保持及冲突处理。原有的两个 Java web 系统，使用了不同的前端框架，原族谱录入系统的界面使用了原生 jsp，而族谱查询系统的界面则使用了 bootstrap 框架，于是在编写 jsp 时出现了问题：如果不使用 bootstrap 框架，则导入后的页面无法保持原有的界面格式。如果使用 bootstrap 框架，则原有的 header 和导航栏会发生 UI 风格的冲突。当前的族谱管理系统具有严格的用户权限管理机制，不同权限的用户仅仅可以访问有限的内容。而在网页的访问过程中，可能会存在一定的安全隐患。比如某些攻击者可能会获取 jsp 的命名方式，从而越过登录校验直接

越权访问相关链接。两个系统的 hibernate 文件有所差异，且由于历代的修改，maven 的依赖关系十分复杂，不利于调试。

3.5.1 不同前端框架下网页的整合

由于原族谱录入系统和族谱查询系统使用了不同的前端框架，在合并时为保证具有相同的 UI 风格，应采取一定的措施保证前端框架的兼容。

3.5.1.1 问题分析

在原有的 searchthegenealogy.jsp 文件中使用了 bootstrap 的 3.2.0 版本下的 bootstrap.min.css 文件定义网页 UI 规范。在该 css 文件中，定义了包括 navbar, button 等等控件的 UI 风格以及盒式排版的规则包括位置排布等等。于是第一种解决思路希望通过注释掉一部分 css 代码实现对原有功能的剪裁。

但是在实际应用中发现并不可行。该 bootstrap.min.css 的内容过于复杂，对于各控件的定义十分精细，可读性较差，由于定义的变量过多，在注释时注释掉哪一部分代码成为了棘手的问题。更何况这种方法也并非一直可行，特殊情况下注释掉的代码段仍然可以被浏览器识别，尤其是在使用 Chrome 浏览器测试时发现注释后的 css 文件仍然可以被识别，网页仍然没有发生任何改变。

在这种情况下，iframe 内嵌框架就可以很好地解决在同一个页面中实现两种页面 css 风格的目的。IFRAME 是一种 HTML 标签，通俗解释来说就是页面中的页面，或者可以浮动的框架。所有的主流浏览器都支持<iframe>标签。你可以把提示的文字放到 <iframe> 和 </iframe>里面，这样不支持 <iframe>的浏览器就会出现提示的文字。Iframe 元素会创建包含另外一个前端页面文档的内联框架或行内框架。Iframe 具有一系列优点：重载页面时减少数据传输，减少网页加载时间；同时，它技术简单，使用方便，主要应用于不需要搜索引擎来搜索的页面；方便开发，减少代码的重复率，比如页面的 header，footer。与此同时也具有一些缺点，比如说 iframe 会产生很多的页面，不易于管理，不易打印；同时，多框架的页面会增加服务器的 http 请求，也可以导致浏览器的后退按钮无效等。

总结来说，iframe 可以改变网页内部代码的作用域，使得 bootstrap 框架仅仅作用于 iframe 内部的页面部分，而外部依然保持 navbar 的原有框架。从而解决了前端框架冲突的问题。

3.5.1.2 解决方案

在原有的族谱查询系统中，只使用了 showthegenealogy.jsp。初步拟定根据其复写的新的 jsp 命名为 zuputuijian.jsp，在其由单独 jsp 页面实现的基础上，拆分成为两个单独的 jsp 文件：zuputuijian.jsp 和 putinpagetuijian.jsp。

在 zuputuijian.jsp 中，保持原有的 navbar 框架，不导入 bootstrap 框架下的 css 文件。保留原有的 navbar 风格和排版。将原有 jsp 中 body 的部分单独提出来作为 putinpagetuijian.jsp，在该 jsp 文件中引入 min.css，这样该 css 的作用域就仅仅只能作用于 zuputuijian.jsp 了。

其原理如图 3.8 所示：

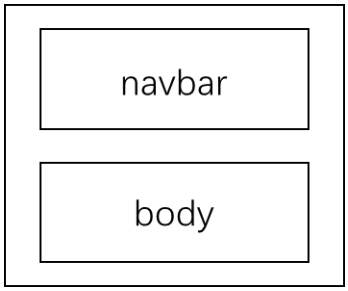

图 3.8 原有族谱录入网站前端结构图

原有的族谱录入网站结构如图 3.8 所示，未使用 bootstrap 框架，body 的排版格式无法保持原有族谱查询系统的 UI 框架。

族谱查询网站，加入了 bootstrap 框架后的的网站结构如图 3.9 所示，body 的排版格式保持原有族谱查询系统的 UI 框架，navbar 的格式发生错乱，其 UI 风格与其他同级 jsp 不符。

根据解决方案，在族谱录入网站基础上加入 iframe 优化后的结构如图 3.10 所示。

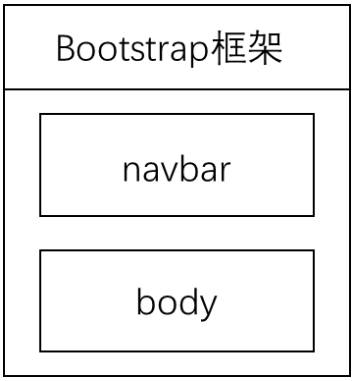

图 3.9 使用了 bootstrap 框架的族谱查询网站前端结构图

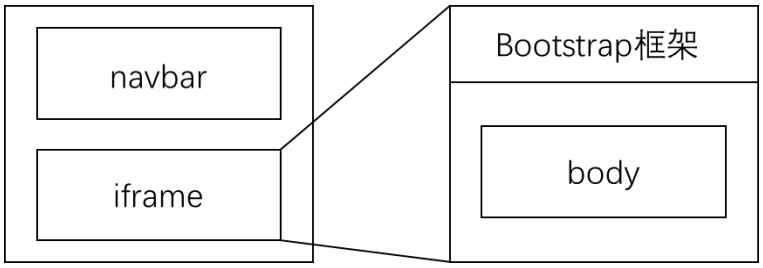

图 3.10 加入 iframe 后的集成网站前端结构图

该结构保持了原网页的 navbar 仍然保持原有封顶基础上，加入了 iframe，在独立的 jsp 文件中使用 bootstrap 框架，其作用域仅仅相当于原有网站的 body 部分，从而实现了 navbar 和 body 使用不同 css 框架的需求。

3.5.2 网站 jsp 访问的安全机制控制

为了防止不怀好意的访客对页面 jsp 进行越权访问，在编写系统时必须对 jsp 的访客身份进行判定。

3.5.2.1 问题分析

越权访问就像是不具备房屋所有权的客人，获得了主人的权限，进入了主人的家，再把屋内翻得一塌糊涂。越权访问无论是对数据库还是后台都可能存在潜在的威胁。因而，防止 jsp 越权访问是十分必要的。这个越权访问的解决方案，在 Java web 项目中有两种主要的解决方法，二者首先都是用 request 获取之前的页面路径，再使用 session 进行判定。

第一种是在 jsp 中编写 session 的判断代码，如果存在该用户的 session，才能执行 jsp 页面代码，否则跳转到登录页面。但是这种方法具有欠缺，如果有很多个 jsp，那么就要写多个判断，这使得逻辑不够清晰，修改起来也较为复杂。所以我们并不能采用这种方法。

另一种解决方案是构造过滤器，在这种情况下，所有的符合过滤器过滤对象的页面被访问时都需要进行过滤验证，如果不存在该用户 session，则跳转到登录页面，存在用户 session 时才能执行页内代码，保存 session 后访问其它页面。这种方法的优点是访问所有 jsp 文件时都可以自动进行校验。具有逻辑简单，修改便捷的特点。

理论上说，理想的 jsp 访问控制，在用户未登录时，只可以访问登录界面或是一部分开放的页面。而用户登录之后，也只能访问该用户权限范围内有限的界面。

3.5.2.2 过滤器原理

过滤器是 JavaWeb 的三大组件之一，可以独立于 Servlet 对请求或者回应进行修改。对于 Web 应用程序来说，过滤器是一个 Web 组件，它驻留在服务器端，可以对从客户端向服务器端发送的请求进行过滤，也可以对服务器端返回的响应进行处理。其原理如图所示，filter 对客户端向 servlet 端发出的请求进行预处理，也可以对 servlet 端向客户端的响应进行处理，而其本身不是 servlet，不能回应请求。其结构如图 3.11 所示。

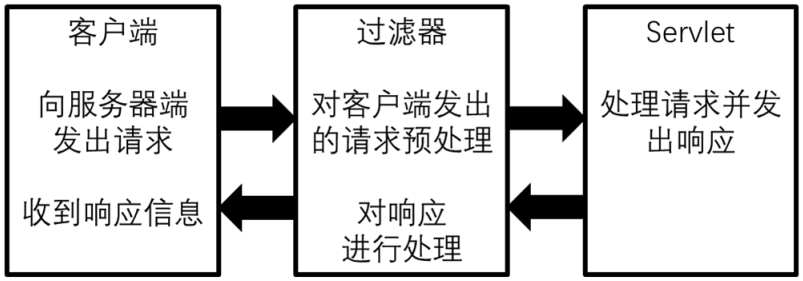

图 3.11 过滤器原理图

当客户端发生请求后，在 HttpServletRequest 到达 Servlet 之前，过滤器拦截客户的 HttpServletRequest。根据需要检查 HttpServletRequest，也可以修改 HttpServletRequest 头和数据。 在过滤器中调用 doFilter 方法，对请求放行。请求到达 Servlet 后，对请求进行处理并产生 HttpServletResponse 发送给客户端。在 HttpServletResponse 到达客户端之前，过滤器拦截 HttpServletResponse。根据需要检查 HttpServletResponse，可以修改 HttpServletResponse 头和数据。最后，HttpServletResponse 到达客户端。

在当前的族谱综合管理系统中，在 web.xml 文件下定义了所有的页面访问均需要通过过滤器经过身份信息的验证。而与此同时的过滤器 Java 文件则可以定义何种网站访问需要过滤，过滤后的去向等等。同时，即使是新编写的 jsp 文件，在访问时依然需要进行身份验证，而不需要在每个 jsp 文件中重新定义访问权限。

从而实现 jsp 的防越权跳转机制。

3.5.3 网站 service 服务合并的依赖关系保持及冲突处理

原有的族谱录入系统和族谱查询系统使用了相同的 service 框架，然而它们的后台方法并不完全相同。因此，在合并时需要对原有两个子系统的后台代码进行处理。

3.5.3.1 一般分析

在 SSH 框架下，客户端的请求通常通过前端 jsp 中的 action 处理，而 action 的具体实现方法则在业务层的 Java 文件中进行处理。而业务层的 Java 文件与 web 层用户界面 jsp 文件通常不是一一对应的关系，一个 jsp 文件中的 action 方法可能分别在多个业务层的 Java 文件中实现，而一个业务层的 Java 文件同样可能实现多个 jsp 中的 action 方法。这给我们整理逻辑关系带来了一些困难。

除此之外，业务层的 Java 文件并不完全是并列的，单个 action 仅仅通过一个业务层的 Java 文件就可以处理完毕，或再向更底层的 DAO 层传递。Maven 之后的项目会分为源码，资源等等文件夹进行分类存储，这使得原本资源-源码的依赖关系被打乱，而与此同时，原本的业务层文件之间同样可能存在依赖。因此，对单个业务层 Java 文件进行处理，都可能导致不确定的后果。为了尽可能节约时间，快速开发，对原有的 Java 文件依赖关系应该尽量保护原状不要打破，修改时尽可能在原有命名规范的基础上修改。

3.5.3.2 具体分析

原有族谱查询系统的业务层较为简单。Pedi/action 路径下仅包含两个 Java 文件，baseaction.java 以及 genealogyaction.java。

baseaction.java 主要使对网站运行时的客户端服务器间的 servlet 请求，HTTP 传输进行规范和功能的实现。同时，族谱录入系统和族谱查询系统由于都使用了 ssh 框架，因而具有相同的 baseaction.java 文件，在集成时不需过多操作。

而 genealogyaction.java 则是对族谱查询的具体功能进行实现。在原有的族谱录入系统中，同样包含 genealogyaction.java 文件。二者功能现有功能部分重叠，主要思路就是取二者的并集，集成到新的系统当中。主要分为以下几个方面。

头文件方面，需要引入存在于族谱查询系统但不存在与族谱录入系统的 genealogy.java 的头文件，比如 pageModel.Allthegenealogy 等等。方法方面，对于

此前已经共同存在的同名方法，比较其功能，实现并集，比如 public GenealogyServiceI getGenealogyService()方法等等，对于原本不存在的方法，需要按照对应位置集成，比如 public void showallgenealogy()等等。对于成员变量同样需要取交集，先检查是否有同名变量，所实现的功能是否有差异。再加入原本不存在的变量。

在完成上述修改后，应在前端中分别进行各项功能的测试，观察是否能够实现原有的功能。因为不报错可能只是同名方法的巧合，需要进行单元测试，功能测试等方法进行验证。

4 编码设计思路

本章选取具有代表性的几个编码点对整合族谱管理系统的过程进行了介绍。前端的处理较为多种多样，根据具体情况给出了具体的解决方案，除了最基本的嵌套入 navbar 的设计外，其他功能包括 iframe 实现，外部链接跳转，页面内跳转到特定行数等等功能也都得到了实现。业务层及以下更底层的部分则采用了统一的取并集的原则，力求保证原有功能的维持和合并后功能的正常使用。对于访问控制则采用了过滤器的方法，由 LoginFilter.java 实现，同时需要在 web.xml 中配置过滤器信息，以确保网站访问到任意 jsp 页面都会发生跳转。

4.1 web 层

4.1.1 jsp 设计

根据原族谱录入系统中的 jsp 文件（位于\pedigree\src\main\webapp 路径下），以二级目录为例，online.jsp 是原有族谱管理系统中的高层目录，核心代码如图 4.1 所示。

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
String path = request.getContextPath();
%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
  <meta http-equiv="pragma" content="no-cache" />
    <title>功能导航</title>

  <jsp:include page="inc.jsp"></jsp:include>

  </head>
   <body class="easyui-layout" data-options="fit:true" style="width:1200px;height:800px;" >
    <div data-options="region:'north'" style="height:50px;overflow:hidden;">
        <div align=right>
        <span>
            <jsp:include page="./header.jsp"></jsp:include>
        </span>
        </div>
    </div>
    <div data-options="region:'south'" style="height:20px"></div>
    <div data-options="region:'west',title:'功能导航'"
        style="width:200px">

        <jsp:include page="./newnav.jsp"></jsp:include>

    </div>
    <div data-options="region:'center',overflow:'hidden',iconCls:'icon-ok'"></div>
  </bodv>
```

图 4.1 online.jsp 核心代码

因此，在编写族谱查询系统时，应用相同格式编写名为“族谱查询”的一级目录。而二级目录以 MangeGenealogy.jsp 为例，应以类似格式，结合原族谱查询系统的对应功能的 jsp 文件进行整合编写。编写时注意 Action 命名是否冲突或重名，在没有上述隐患的前提下尽量保证命名规则不改变。

4.2.2 框架冲突时的 iframe 实现 jsp

在原有的 zuputuijian.jsp 中的 body 部分插入如图 4.2 所示代码，作为 iframe 的入口。

这里定义了 iframe 的名称和 id，同时定义了 iframe 的指向路径。指向独立的 putinpagetuijian.jsp。这个 jsp 主要是对原 jspbody 部分的复写。

```
<body>
<iframe name="myiframe" id="myrame" src="putinpagetuijian.jsp"
frameborder="0" width=100% height=100% scrolling="auto">

    </iframe>
</body>
```

图 4.2 iframe 实现入口代码

4.2.3 外部跳转模块的 jsp 实现

重新编写的 waibulianjie.jsp 核心代码如图 4.3 所示。

```
<div class="panel-heading">
    <h3 class="panel-title">
        <strong>武汉大学珞珈图腾实验室</strong>
    </h3>
</div>
<div class="panel-body">
    <button style="width:100px"><span>
    <a href="http://bdi.whu.edu.cn/" target="_blank">
    跳转</a></span></button>
</div>
```

图 4.3 waibulianjie.jsp 核心代码

通过绑定 button 事件实现对外部链接的跳转，target=”_blank”语句实现在外部页面跳转。

4.2.4 族谱推荐列表跳转到特定行数的 jsp 实现

该操作在<table>标签中实现，核心代码如图 4.4 所示。

其关键部分在于<a href="putinpagechaxun.jsp#X">，指向该 jsp 自身。

4.3 业务层

```
<table style="font-size:19px;"><tr id="alp">
<td><a href="putinpagechaxun.jsp#A">A</a></a>     </td>
<td><a href="putinpagechaxun.jsp#B">B</a>     </td>
<td><a href="putinpagechaxun.jsp#C">C</a>     </td>
<td><a href="putinpagechaxun.jsp#D">D</a>     </td>
<td><a href="putinpagechaxun.jsp#E">E</a>     </td>
<td><a href="putinpagechaxun.jsp#F">F</a>     </td>
<td><a href="putinpagechaxun.jsp#G">G</a>     </td>
 <td><a href="putinpagechaxun.jsp#H">H</a>     </td>
 <td><a href="putinpagechaxun.jsp#I">I</a>     </td>
 <td><a href="putinpagechaxun.jsp#J">J</a>     </td>
 <td><a href="putinpagechaxun.jsp#K">K</a>     </td>
 <td><a href="putinpagechaxun.jsp#L">L</a>     </td>
 <td><a href="putinpagechaxun.jsp#M">M</a>     </td>
 <td><a href="putinpagechaxun.jsp#N">N</a>     </td>
 <td><a href="putinpagechaxun.jsp#O">O</a>     </td>
 <td><a href="putinpagechaxun.jsp#P">P</a>     </td>
 <td><a href="putinpagechaxun.jsp#Q">Q</a>     </td>
 <td><a href="putinpagechaxun.jsp#R">R</a>     </td>
 <td><a href="putinpagechaxun.jsp#S">S</a>     </td>
 <td><a href="putinpagechaxun.jsp#T">T</a>     </td>
 <td><a href="putinpagechaxun.jsp#U">U</a>     </td>
 <td><a href="putinpagechaxun.jsp#V">V</a>     </td>
 <td><a href="putinpagechaxun.jsp#W">W</a>     </td>
 <td><a href="putinpagechaxun.jsp#X">X</a>     </td>
 <td><a href="putinpagechaxun.jsp#Y">Y</a>     </td>
 <td><a href="putinpagechaxun.jsp#Z">Z</a></td>
</tr></table>
```

图 4-4 族谱推荐列表跳转到特定行数的 jsp 实现

4.3.1 action 设计

原有族谱录入网站和族谱查询网站具有相同的 baseaction.java。因此，在合并时，该文件无需改动。而 genealogyaction.java 文件内部，二者存在相同点也存在不同点，需要着重讨论。

现分析对比两个 genealogyaction.java 内部方法，结果如下文所述：

只属于族谱录入网站 genealogyaction.java 的方法：

addthebranchtotrunk: 增加关联人物

addthestartperson: 增加分组人物

choosethemother: 获取母亲列表

createthegroup: 新建一个分组

creatthenewPedigree: 新建族谱

deleteTheGroup: 删除族谱分组

deletethetask:删除任务

editthegroup：编辑分组

getModel：获取模型

getPersonNodeByGrpid：通过分组 ID 和父亲 ID 获取儿子节点

getRootPersons：根据前台 pediId 查询出族谱号为 pediId 的世代数最小的或没有父亲节点的人物

initgroupTree：初始化分组树

initPediTree：初始化族谱树

intisvail：判断是否合理有效

loadtheinputsort：每次进入录入页面的时候去读取每个用户设定的录入习惯顺序

loadtheteamdetail：加载团队信息

reloadthedoc：按照规则去给文档排序

reloadthedocbystyle：按照 style 规则去给文档排序

removethebranchtotrunk：删除该条支系与总源族谱的关联关系

savethedocsort：保存对应的文档顺序

savethegenealogy：保存族谱

savethePersonAlert：保存人物信息

savetherules：保存前端对规则名字和描述的更新

savethetask：保存任务

searchthedoc：搜索特定文件名的文档

selectdocbystyle：获取文档顺序 style 规则列表

selectrule：获取文档顺序规则列表

setPediVolService：初始化族谱数据服务

showpname：展示人物名称

showthebranchtotrunk：展示支系与总源之间的关系

showthedisposal：按照录入人员和编辑权限来获取族谱列表

showthedisposalbyeditor：按照编辑权限获取族谱列表

showthedocsort：载入对应 ruleid 的 datagrid

showthegenealogy：族谱编辑，超级管理员获得其族谱列表

showtheinputman：获取录入人员列表

showthePbyVol：获取人物数据

showthepedigree：展示族谱

showtherule：展示所有的文档顺序规则

showthestartperson：显示本分组人物

showtheuserrole：获取可分配任务

showtheVolume：获取族谱分卷列表

writeTreeToPage：写家族树

以上方法不需要修改，保持原样即可。

只属于族谱查询网站 genealogyaction.java 的方法：

Addthetext：返回图片的文本信息

getPersonTree()：根据 id 来请求人物树

getPersonTree(int, int)：获得树的深度

getRemortIP：获取远程请求 IP

gettheancestorline：一键寻祖

isIdValid：判断是否合理有效

leavethemessage：插入一条新的留言

login：索引号认证登录

logoff：注销 SESSION

onkeyfindthef：一键寻亲

searchperson：根据姓氏，谱名，父亲谱名，性别，生死状况来搜索人物，如果是搜索分组的话就得到分组始祖的那个人的人物界面

searchpersonno：根据姓氏，谱名，父亲谱名，性别，生死状况来搜索人物，搜索的结果不用分页

searchthegenealogy：搜索对应的族谱

setcity：用于网络族谱搜索页面，返回地图中省市；里面族谱总数以及每个族谱的名字

setsessionurl：增加挑战的路径和页面的参数在 session 里面

setsessionurlgrp：增加挑战的路径和页面的参数(只是分组的信息)在 session 里面

showallgenealogy：分页去展示所有的族谱列表

showhotgenealogy：查找五个热门族谱

showonegenealogy：利用 GID 获取某一个族谱信息

showonegroup：利用 grpid 获取某一个分组信息

showoneperson：在每个人物的展示页面中获取相关信息

showsomegenealogy：按照录入人员和编辑权限来获取族谱列表

showthegroups：展示所有的分组（世系图）

showthemark：获得每个省的族谱相关信息

showthemessages：展示所有的留言

showthestatic：获得目前数据库的统计信息

以上方法直接集成进入原族谱录入网站的 genealogyaction.java 中即可。

同时属于两个网站 genealogyaction.java 的方法：

getGenealogyService：获取族谱管理服务

setGenealogyService：初始化族谱服务

经过对比发现两个方法在两个 genealogyaction.java 文件中完全相同，保留任意均可。

4.3.2 过滤器设计

具体的解决主要由 LoginFilter.java 实现，该文件位于 pedi/util 文件夹下，该类继承了 HttpServlet 大类，是对 Filter 接口的具体实现。在头文件中引入了诸多与 servlet 相关的头文件。下面对该 Java 文件进行解释：

```
package pedi.util;

import java.io.IOException;

import javax.servlet.Filter;
import javax.servlet.FilterChain;
import javax.servlet.FilterConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletRequest;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

图 4.1 LoginFilter.java 加载头文件

首先如图 4.1 所示，加载了 servlet.Filter 等头文件，实现了对 HTTPfilter 相关头文件下方法的继承。

接着如图 4.2 所示，获取当前访问的 HttpServletRequest，HttpServletResponse，HttpSession。下一个判定规范 URL 的格式必然以“/”开头。

```
public class LoginFilter extends HttpServlet implements Filter {

    private static final long serialVersionUID = 1L;
    public void init(FilterConfig arg0) throws ServletException {
        //do something
    }
    public void doFilter(ServletRequest sRequest, ServletResponse sResponse,
        FilterChain filterChain) throws IOException, ServletException{
        HttpServletRequest request = (HttpServletRequest) sRequest;
        HttpServletResponse response = (HttpServletResponse) sResponse;
        HttpSession session = request.getSession();
        String url=request.getServletPath();
        String contextPath=request.getContextPath();
        if(url.equals(""))url+="/";
```

图 4.2 LoginFilter.java 获取访问 request

```
        if((url.startsWith("/")&&(url.endsWith("action")
                ||url.endsWith(".jsp"))&&!url.contains("login")
                &&!url.contains("index"))&&!url.contains("fUploadAction"))
        {//若访问后台资源 过滤到login
             String user=(String)session.getAttribute("usercode");
             if(user==null){//转入管理员登陆页面
                  response.sendRedirect(contextPath+"/index.jsp");
                  return;
             }
        }
          filterChain.doFilter(sRequest, sResponse);
    }

}
```

图 4.3 LoginFilter.java 判断访问者是否访问后台资源

之后图 4.3 的判定为整个 filter 的核心部分，判断访问者是否在访问后台资源，若是，则对用户的 session 进行判定，若不是管理员，则强行跳转到 login 界面。

Filter 的实现如上文 3.5.2 所述。与此同时，需要在 web.xml 中进行修改，以保证网站在服务器上运行时，所有 jsp 文件的访问都不能绕过校验。其核心相关代码如图 4.4 所示。

该代码段的 filter-name 标签定义了 filter 的名称，filter-class 标签定义了 filter 的路径，param-name 和 param-value 定义了 filter 的生命周期一直存在。

filter-mapping 标签下的 url-pattern 的内容“/*”定义了对访问任何以“/”开头的 url 路径资源访问均需要经过校验。

```
<!--登陆配置 -->
  <filter>
    <filter-name>loginFilter</filter-name>
    <filter-class>pedi.util.LoginFilter</filter-class>
    <init-param>
          <param-name>targetFilterLifecycle</param-name>
          <param-value>true</param-value>
    </init-param>
</filter>
 <filter-mapping>
    <filter-name>loginFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>
```

图 4.4 filter 核心代码

5 测试分析

本章介绍对集成后的族谱管理系统进行测试的目的和意义是为了保证功能的完善和集成后的鲁棒性。通过不同的测试方法和测试用例，分别实现单元测试和系统测试。从测试中我们找到了可以改进的点，比如 iframe 框架的自适应大小，也发现了一些目前暂时尚未解决的问题，比如对于具备 cookies 登录信息的浏览器可能会导致 jsp 访问不安全等等。这些问题将在后续的工作中，逐步得以解决。

5.1 测试目的

原有的族谱查询和族谱录入为两个分离的网站，其 action 与 service 直至更底层部分的依赖关系复杂，在集成过程中可能存在一系列未知的错误与问题，由于该平台存在大量族谱信息，未知的漏洞甚至可能关系到网站的安全性能。因此，网站的测试不仅仅是校验网站的功能实现情况，更是对网站安全性能的检验和确认。

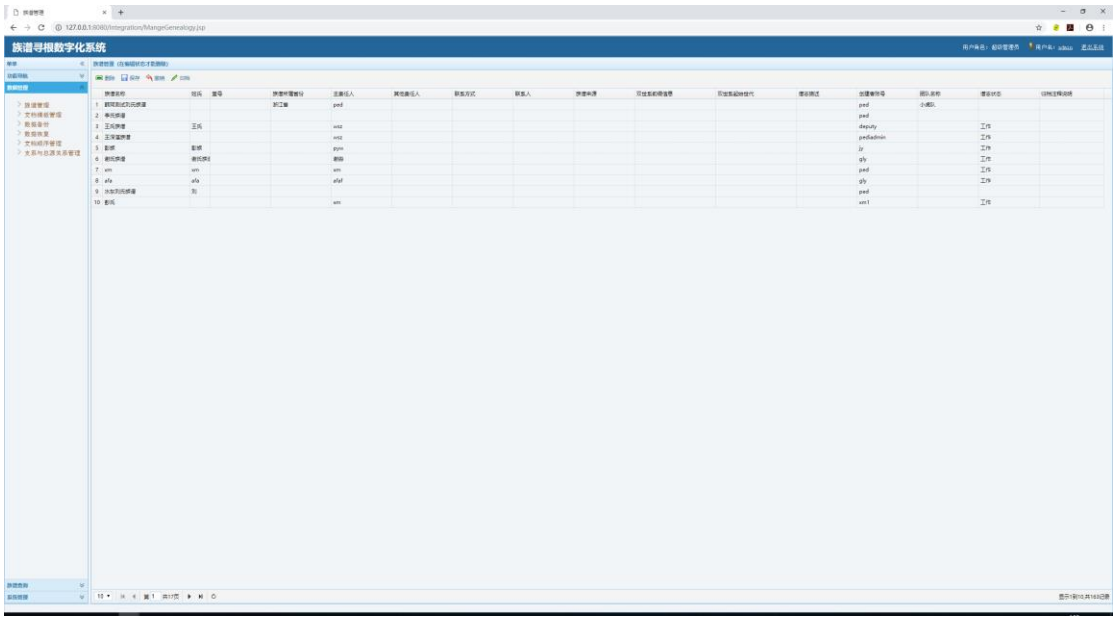

图 5.1 族谱录入模块测试：可以正常使用原有族谱录入网站的功能

5.2 单元测试

5.2.1 族谱录入模块

对族谱录入模块中的部分功能进行测试，原有的功能完全可以正常使用。这里以族谱管理模块为例，可以对后台数据库进行正常的增删查改等等操作。证明原有系统在集成后的功能的保持程度还是相当高的。

5.2.2 族谱查询模块

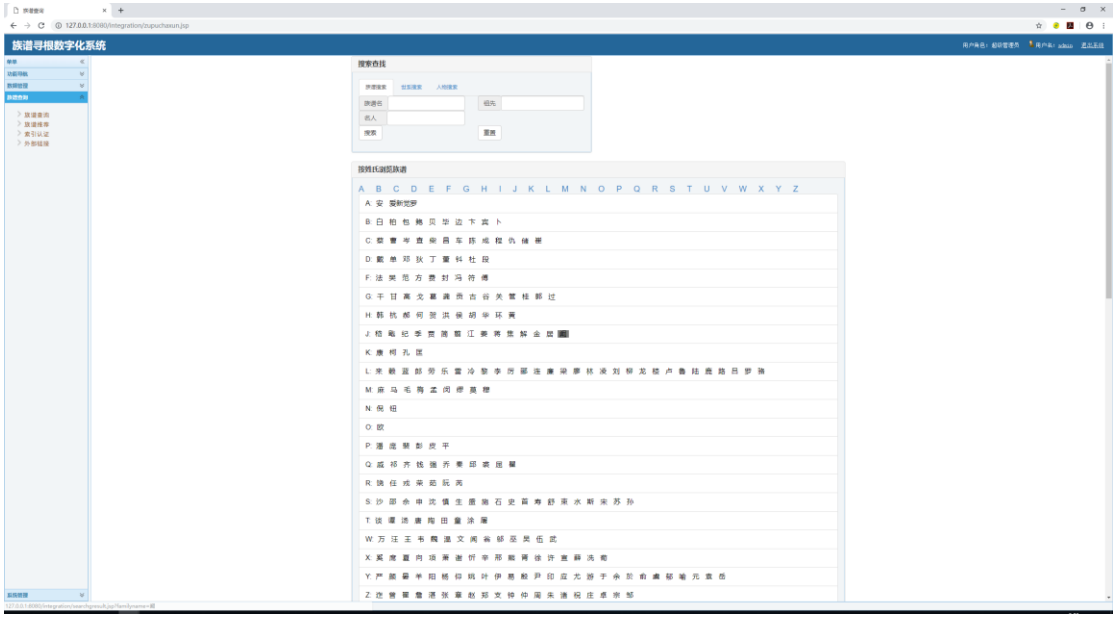

图 5.2 族谱查询模块测试 1：上方有正常的跳转索引

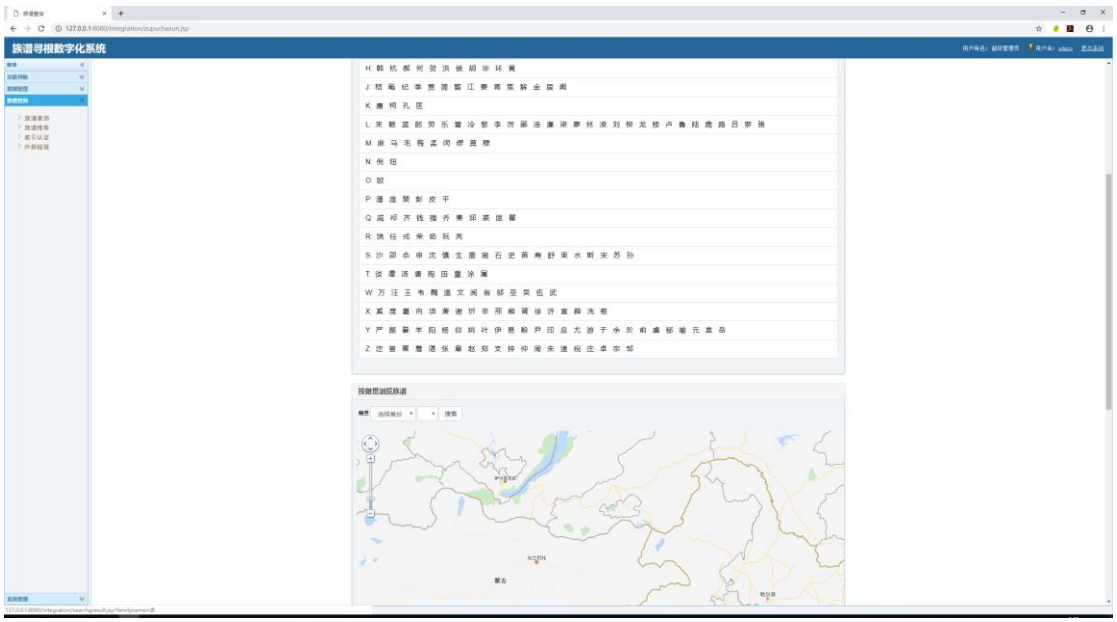

图 5.3 族谱查询模块测试 2：点击跳转索引可以正常实现跳转

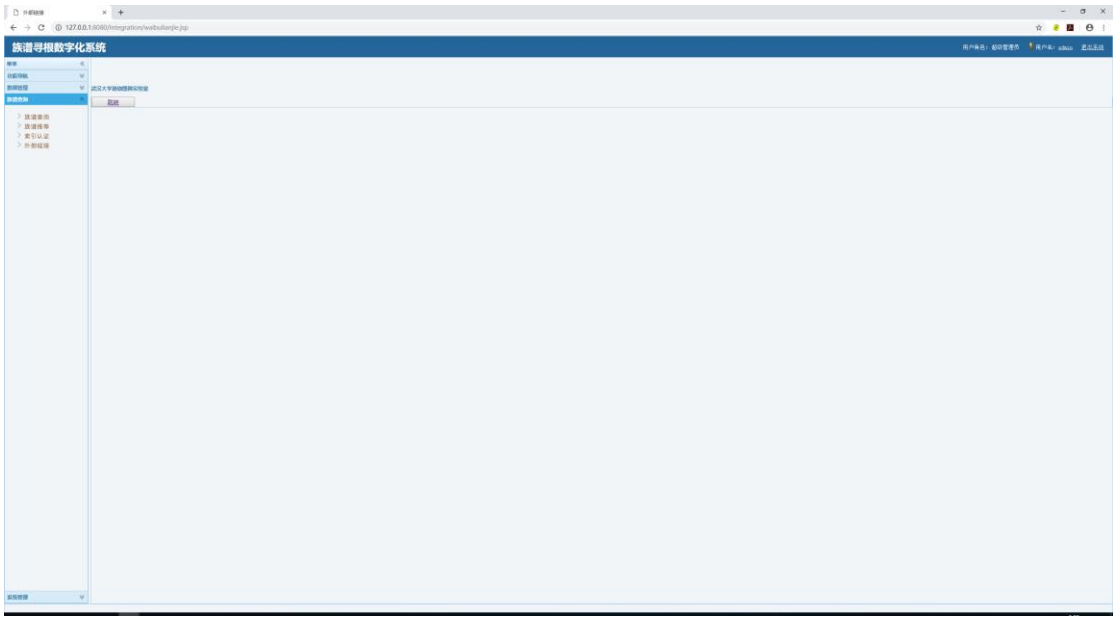

图 5.4 族谱查询模块测试 3：跳转前页面，嵌在原有的导航栏界面中

首先检查族谱查询功能。在这个网页内，原有网页根据字母跳转到推荐族谱特定的行。我们测试发现，集成后的族谱查询模块同样可以实现。

以 H 为例进行测试，结果如图 5.2，图 5.3 所示。

可以证明其跳转功能可以正常使用。另一方面，外部链接跳转方式也发生了改变，跳转目标由中根网改为武汉大学珞珈图腾实验室。如图 5-4 所示。

5.3 集成测试

图 5.5 系统测试 1：登录界面

集成测试同样主要包含两个方面，一是通过统一的族谱录入系统的登录入口使用管理员账号登录可否获取全部权限。如图 5.5 所示。

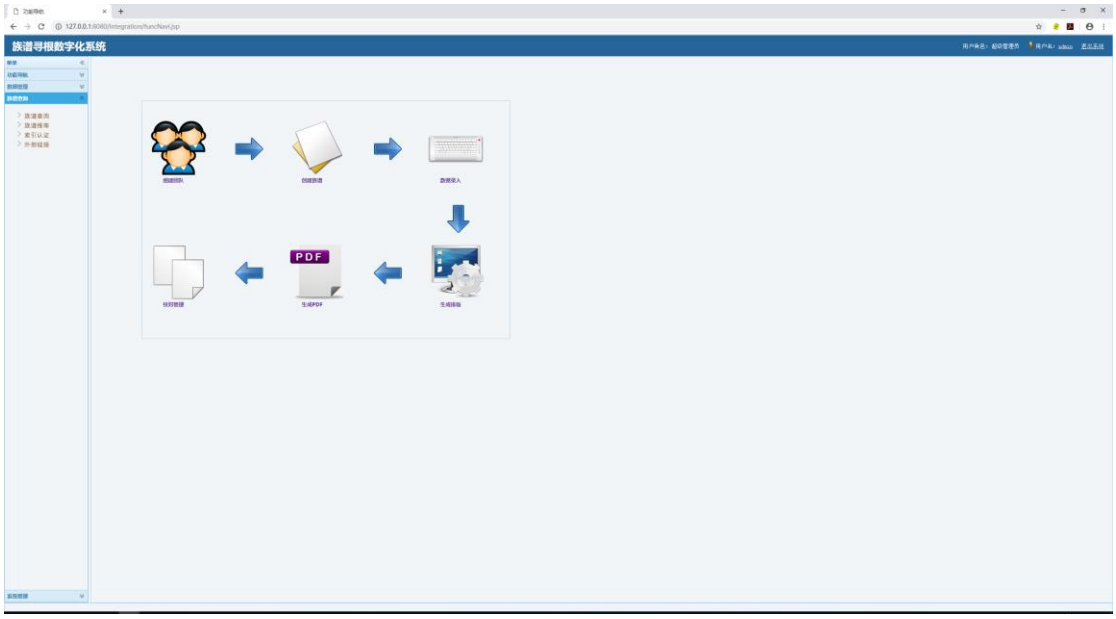

图 5.6 系统测试 2：登录后直接进入导航页面

这里首先使用管理员账号登录进入网站，在这种情况下，访问者获得了最高权限。如图 5.6 所示，可以看到，族谱录入模块和族谱查询模块的功能，都是正常获取了的。

另一方面，我们需要检查 jsp 的安全性，也就是说有没有不怀好意的来访者

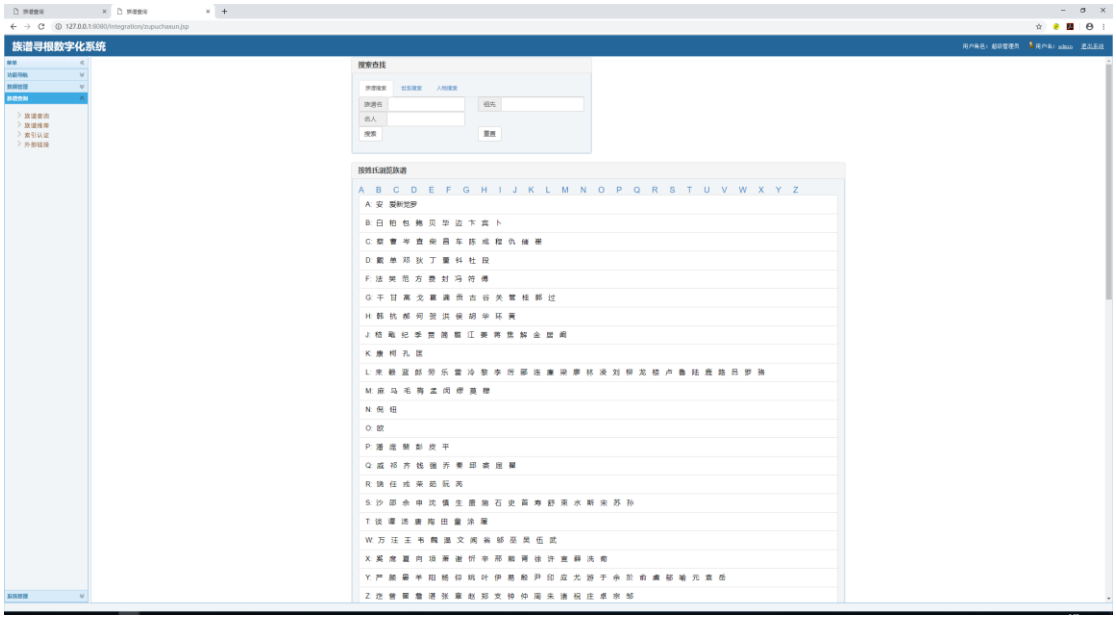

图 5.7 系统测试 3：在 Chrome 浏览器中可以越过登录直接 jsp 跳转

可能可以越权访问 jsp。这里使用两个不同的浏览器，Chrome 和 Edge，在地址栏直接输入 http://127.0.0.1:8080/integration/zupuchaxun.jsp 进行测试。

可以看到在 chrome 中是直接登录了的，然而在 edge 等其他浏览器中，会跳转到初始的登录页面。经过分析可知，这是因为此前用管理员账号登录了 chrome，并且浏览器的 cookies 保留了登录信息。而对于从来没有登录过的浏览器而言，我们的 jsp 防越权身份检查机制，是行之有效的。访问者在使用 jsp 地址强行跳转之前，必然会经过登录页面认证其身份信息。

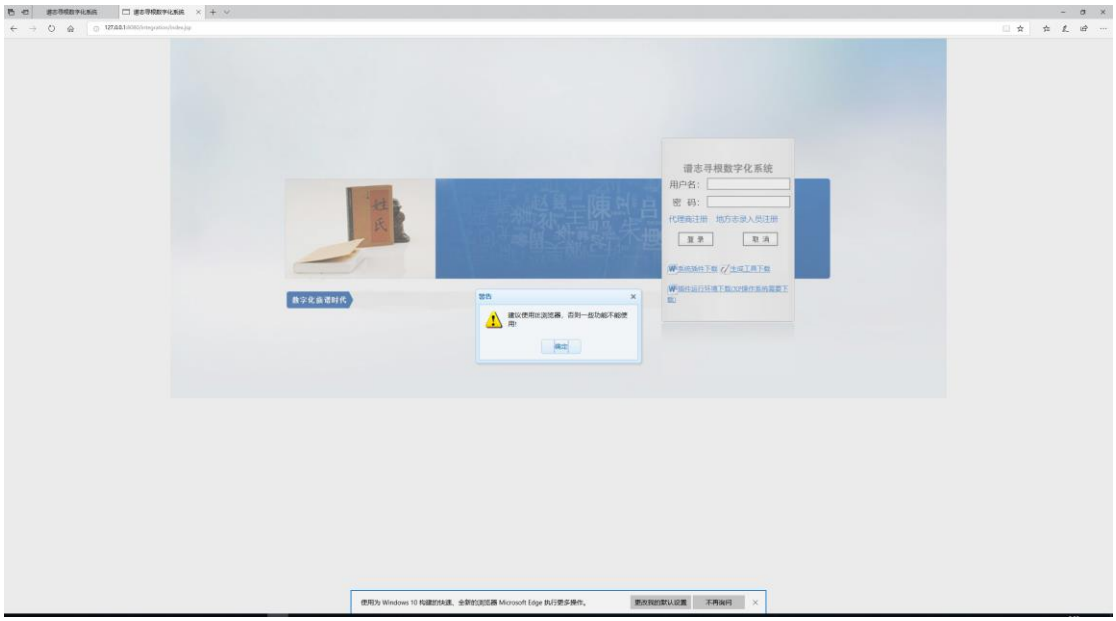

图 5.8 系统测试 4：在 edge 浏览器中可以实现对越权 jsp 访问的限制

5.4 测试效果

5.4.1 前端

按照最开始写成的 jsp 文件运行，即插入未声明长宽的 iframe 时，系统自动生成了一个长宽固定的页面。无法实现对 iframe 内容的自适应。对于这个问题，需要在初始化 iframe 的时候加入 frameborder="0" width=100% height=100%就可以了，其代码如下所示：

<iframe name="myiframe" id="myrame" src="putinpagechaxun.jsp" frameborder="0" width=100% height=100% scrolling="auto">

5.4.2 jsp 访问安全性

Filter 过滤器可以有效防止未登录人员越权访问加密资源，然而在实际的测

试中仍然存在一定问题。在用 Chrome 浏览器测试时，如果之前已经使用账号密码登录过，则登录记录会自动保存在 Chrome 的 cookies 中，下次使用 URL 越权访问时，浏览器自动使用 cookies 进行登录，默认访问者为上一次登录的身份，从而可能发生越权访问。该漏洞目前还没有得到有效解决，计划在未来的工程升级迭代维护时对此进行改进弥补[1]。

6 展望总结

在武汉大学珞珈图腾实验室负责的族谱管理系统中，我有幸参与了一部分族谱录入系统与族谱查询系统整合的相关工作。对整个大的项目来说，有助于实现整个项目功能的升级完善，早日实现项目的上线使用，为整个族谱电子化的推广起到一定的积极作用。对于我个人来说，这是我第一次使用 Java 进行大型项目开发，在这个过程中我学习到 Java web 的许多知识，包括 servlet 的原理，网站处理 http 请求的流程等等，以及 ssh 框架的原理和应用。我相信这次的项目开发会对我整个职业生涯具有开创性和启蒙性的作用。

另一方面，目前做出的网站只是一个满足基本要求的成品，在上文陈述的测试部分中可知，尚有许多可以完善升级的部分，包括网站的 jsp 访问不能解决 cookies 访问的问题等等。日后可能在当前网站上开发出更多功能，譬如离线的族谱查询系统等等。这些工作我相信会由一代代珞珈图腾人继续下去，为整个民族的族谱电子化，为文化纽带的传承尽最大的努力。

参考文献

[1] 基于Neo4j图数据库的课程体系知识图谱系统设计与实现[J]. 肖庆都,屈亮亮,侯霞. 电脑知识与技术. 2017(36)

[2] Spring MVC学习指南 Paul Deck. 中国工信出版集团. 2017.5

[3]递归查询在农村土地承包经营权流转统计中的应用[J]. 于鹏翔,温婉丽. 河北建筑工程学院学报. 2017(02)

[4]基于表分区和内存数据库的族谱生成系统优化[J]. 徐明民,彭中华,王黎维. 计算机与数字工程. 2017(02)

[5]基于异构数据库的海量数据查询系统设计[J]. 王言通,魏青. 信息通信. 2016(04)

[6]基于图形数据库Neo4J的合著网络研究与实践[J]. 路莹,罗荣庆,王青春,牛晓芳,穆晓倩,祝茜. 中华医学图书情报杂志. 2016(04)

[7] 为了家的记忆——湖南图书馆家谱文献收集、研究、保存与服务概略[J]. 雷树德. 数字与缩微影像. 2018(03)

[8] 面向内容整理的家谱数字化系统分析与设计[J]. 杨志丹,蔡跃进. 泉州师范学院学报. 2018(04)

[9] 当代青年对族谱的传承现状研究——以潮汕地区为例[J]. 黄小奇,吴浩,邱珊珊. 信阳农林学院学报. 2018(02)

[10] Genealogy registry system Kent W. Huff 2000.03.15

[11] A Study of an Event Oriented Data Management

Method for Displaying Genealogy: Widespread

Hands to InTErconnect BASic Elements Seiji Sugiyamal Atsushi Ikuta Miyuki Shibata and Tohru Matsuura 2011

[12] Genealogy system for interfacing with social networks Ilya NikolayevAndrew Merkatz 2008.06.24

[13] Providing alternatives within a family tree systems and methods Bennett Cookson, Jr.Ken BoyerJames Mark HamiltonKendall J.

JeffersonDaren ThayneMichael J. Wolfgramm 2003.12.29

[14] Method and system for building a family tree Richard C. NotargiacomoDavid L. PattonFrank Pincelli 2001.07.03

[15] Electronic family tree generation and display system Klaus Peters 2010.08.27

[16] Data management system for building a database with multi-dimensional search tree nodes Frederick A. PowersStanley R. Zanarotti 1990.03.16

[17] Mending the family tree a reconciliation of the linearization and levels schools of AGE modelling Thomas W.HertelaJ.Mark HorridgebK.R.Pearsonc 1992.8

[18] Anabaptist genealogy database Richa Agarwala Leslie G. Biesecker Alejandro A. Schäffer 2003.06.17

[19] System and method for displaying and manipulating hierarchically linked data in a genealogy database using a graphical interface John H. SherwoodMichael C. Miller 2007.05.16

[20] Multigenerational Information: The Example of the Icelandic Genealogy Database Hrafn Tulinius 2009.09.23

致谢

在四年的大学生活即将结束之时，我借此机会对给我毕业论文带来巨大帮助的师长和四年来给我支持照顾的同仁一并致以感谢。

感谢李蓉蓉老师对我毕业设计论文的引导，在毕业论文的编写过程中给予我十分耐心的指点，一次次指出我的不足并提出修改意见。感谢江欢学长对我毕业设计项目提供的技术支持，对我遇到的问题一次次提出解决思路。感谢珞珈图腾实验室提供的这样一个接触大型实操项目的机会，也感谢在我之前的一代代开发者的工作为我奠定了基础。

同时感谢四年来执教我的老师们，为我的理论知识奠定了基础，感谢我的同学，在我四年的求学之路上提供帮助。

感谢武汉大学，给我提供了一个求知的环境，感谢珞珈山上的一草一木，在我身心疲惫的时候给予我心灵上的慰藉和鼓励。

我希望我的一点工作能够为族谱电子化，为珞珈图腾实验室，为后来的开发者们，带来一点点的帮助，如有所得，不胜荣幸。