北京信息科技大学

毕业设计（论文）

题 目： 特定业务文档规范性检测系统设计与实现

学 院： 计算机学院

专 业： 软件工程

学生姓名： 杨慧文 班级/学号 软工1603/2016011368

指导老师/督导老师： 侯霞

起止时间： 2020年1月6日 至2020年6月14日

摘 要

随着计算机技术的普及，企事业单位每天都产生大量的业务文档。一般来说，特定业务文档的撰写都需要符合规范性。目前检验文档的工作主要依靠人工检测，耗费大量人力资源且效率不高。本系统设计并开发了一套基于B/S模型的特定业务文档规范性检测系统，可以自动化检测被测文档的逻辑结构与样式信息是否符合规范性要求，从而提高业务文档检测的正确性与效率。

系统功能主要包含：

1. 文档模版的规范抽取：对于给定的文档模版，自动抽取其中的文档逻辑结构和格式规范，保存相关规范性要求作为文档检测的依据。
2. 文档规范性检测：给定被测文档，根据文档检测的依据对其文档结构和格式进行自动化地规范性检测，并保存检测结果。
3. 检测结果的可视化：将检测结果以批注的方式加在被测文档中，同时在Web端为用户提供可视化和交互功能。

系统在实现的过程中采用了混合开发的方式：1）为了便于处理Word文档，采用Microsoft提供的API分析检测文档，并将相关功能封装为WebService；2）采用Spring Boot和MVC设计框架，以及前端框架Vue设计实现Web系统的交互和可视化，通过调用WebService完成文档相关检测。

本文介绍了系统的开发背景以及意义，分析系统需求，阐述了系统设计以及系统实现的具体思路和工作。

**关键词：**web开发;文档检测系统;Word文档;文档逻辑检测;Spring Boot;

Abstract

With the popularization of computer technology, enterprises and institutions produce a large number of business documents every day. Generally speaking, the writing of specific business documents needs to conform to the specification. At present, the work of detecting documents mainly depends on manual operation, which consumes a lot of human resources and is inefficient. This system designs and develops a set of normative detection system for specific business documents based on B / S model, which can automatically detect whether the logical structure and style information of the tested documents meet the normative requirements, so as to improve the correctness and efficiency of business document detection.

System functions mainly include:

(1) Specification extraction of document template: for a given document template, the logical structure and format specifications of the document are automatically extracted, and the relevant specification requirements are saved as the basis for document detection.

(2) Document normalization detection: given the tested document, automatically detect the structure and format of the document according to the basis of document detection, and save the detection results.

(3) Visualization of test results: the test results are annotated in the tested documents, and the visualization and interaction functions are provided for users on the web side.

In the process of implementation, the system adopts the mixed development method: 1) in order to facilitate the processing of word documents, the API provided by Microsoft is used to analyze and detect documents, and the relevant functions are encapsulated as WebService; 2) Spring Boot and MVC design framework, as well as the front-end framework Vue are used to design and realize the interaction and visualization of web system, and the relevant document detection is completed by calling WebService.

This paper introduces the development background and significance of the system, analyzes the system requirements, and expounds the specific ideas and work of the system design and implementation.

**Keywords:** Web Development; Document detection system; Word document; Document logic detection; Spring Boot;

目录

[摘 要 I](#_Toc42517051)

[Abstract II](#_Toc42517052)

[第一章 概述 1](#_Toc42517053)

[1.1 项目需求和背景 1](#_Toc42517054)

[1.2 项目研究的意义 1](#_Toc42517055)

[1.3 项目的组成 1](#_Toc42517056)

[1.4 论文组织结构说明 2](#_Toc42517057)

[第二章 开发环境与技术 3](#_Toc42517058)

[2.1 Web端开发——Spring Boot + Maven + MySQL + Shiro + Vue.js 3](#_Toc42517059)

[2.2 WebService端开发——Microsoft.Office.Interop.Word 4](#_Toc42517060)

[2.3 开发工具——Visual Studio + IntelliJ IDEA + Visual Studio Code 4](#_Toc42517061)

[第三章 项目需求分析 6](#_Toc42517062)

[3.1 系统功能分析 6](#_Toc42517063)

[3.2 系统数据处理逻辑 11](#_Toc42517064)

[第四章 特定业务文档规范性检测系统的设计 14](#_Toc42517065)

[4.1 概要设计 14](#_Toc42517066)

[4.1.1 系统功能设计 14](#_Toc42517067)

[4.1.2 系统整体技术方案设计 15](#_Toc42517068)

[4.2 系统处理逻辑 17](#_Toc42517069)

[4.2.1 系统管理员相关功能处理逻辑 17](#_Toc42517070)

[4.2.2 文档管理员相关功能处理 18](#_Toc42517071)

[4.3 数据库设计 20](#_Toc42517072)

[4.4 详细设计 26](#_Toc42517073)

[4.4.1 登录功能设计 26](#_Toc42517074)

[4.4.2 注册功能设计 26](#_Toc42517075)

[4.4.3 用户管理功能设计 27](#_Toc42517076)

[4.4.5 文档规范信息抽取功能设计 28](#_Toc42517077)

[4.4.4 文档检测功能设计 30](#_Toc42517078)

[4.5 接口设计 35](#_Toc42517079)

[4.5 系统结构设计 37](#_Toc42517080)

[第五章 特定业务文档规范性检测系统的实现 40](#_Toc42517081)

[5.1 C# WebService端程序的实现 40](#_Toc42517082)

[5.1.1 文档结构信息与样式信息抽取功能的实现 42](#_Toc42517083)

[5.1.2 文档检测功能的实现 44](#_Toc42517084)

[5.1.3 文档批注添加功能的实现 46](#_Toc42517085)

[5.1.4 文档操作功能的实现 46](#_Toc42517086)

[5.2 Web系统程序的实现 47](#_Toc42517087)

[5.2.1 Web系统程序结构 47](#_Toc42517088)

[5.2.2 用户管理功能的实现 48](#_Toc42517089)

[5.2.3 个人信息修改功能的实现 50](#_Toc42517090)

[5.2.4 查看上传者信息功能的实现 51](#_Toc42517091)

[5.2.5 模板文档上传功能的实现 52](#_Toc42517092)

[5.2.6 模板文档分析功能的实现 53](#_Toc42517093)

[5.2.7 模板文档查看功能的实现 53](#_Toc42517094)

[5.2.8 模板文档删除功能的实现 54](#_Toc42517095)

[5.2.9 模板文档下载功能的实现 55](#_Toc42517096)

[5.2.10 检测文档比较功能的实现 56](#_Toc42517097)

[5.2.11 树形图查看检测结果功能的实现 56](#_Toc42517098)

[5.2.12 检测文档添加批注、下载批注文件功能的实现 58](#_Toc42517099)

[第六章 特定业务文档规范性检测系统的测试 59](#_Toc42517100)

[6.1 用户管理功能的测试 59](#_Toc42517101)

[6.1.1 用户登录功能测试 59](#_Toc42517102)

[6.1.2 用户注册功能测试 59](#_Toc42517103)

[6.1.3 用户后台管理功能测试 60](#_Toc42517104)

[6.2 文档功能的测试 60](#_Toc42517105)

[6.2.1 文档管理功能测试 60](#_Toc42517106)

[6.2.2 文档检测功能测试 61](#_Toc42517107)

[结束语 64](#_Toc42517108)

[致谢 65](#_Toc42517109)

[参考文献 66](#_Toc42517110)

# 概述

随着计算机技术的普及与发展，办公软件，特别是Microsoft公司的Word、Excel等工具被越来越多的公司企业使用，由此产生了大量的业务文档。业务文档因为具有一定的规范性，所以文档的撰写者必须符合管理者要求的格式，为保证规范性，管理者必须通过抽查或其他方式人工打开文档并检查格式信息。因为Word文档的格式选项众多，且有些不易区分（如行间距等），从而造成人工检验的效率低下。

为解决以上问题，本系统构建在线文档规范性检测平台，基于Microsoft官方提供的API对文档信息进行抽取分析，以友好的前端视图以及Word批注方式，满足管理者对文档规范性检验的需求。

## 1.1 项目需求和背景

业务文档规范性的检验，手工的方式需要逐一打开文档，通过管理者观察对比被检验文档与模板文档的不同，最后通过添加批注或书写文档检测报告等形式，将检测信息返回给用户。受限于电脑性能、文档内容复杂程度以及管理者的熟练程度，以上操作会消耗大量的时间与精力，并且由于Word文档一些样式的表现不够显著，如行间距3厘米和5厘米在页面显示中区别不明显，管理者可能无法正确的区分，而逐段逐行检测段落信息将使管理者的检测操作更多，将消耗更多的时间，从而导致人工检测效率过低。

Microsoft官方为软件开发者提供了API，名为Microsoft.Office.Interop.Word，从而可以实现对Word文档的操作。该API使用C#进行编写，提供了诸多方法，通过组合可以实现对文档的读取，对字体、段落、图片以及表格等信息的提取，以及添加批注等。如何合理调用API，以及过滤掉多余信息，保留段落逻辑信息，是本文研究的重点之一。

## 1.2 项目研究的意义

本系统皆在为企业文档管理者提供一个高效率、高准确性以及更加智能的文档检测系统，通过对Word文档段落信息的识别分析，辅以可视化的Web前端技术，使文档管理者更加清晰明了地查看文档检测结果，从而降低了企业的人力资源成本。

在提供文档规范性检测功能的同时，本系统还支持用户角色管理，企业管理者可根据需要设置不同的角色，不同的角色拥有指定的权限，使文档管理更加高效安全。同时，本系统支持文档管理，可以帮助文档管理者通过Web页面管理文档，且文档存储在服务器中，安全的同时，可随时随地处理公司业务。

## 1.3 项目的组成

本系统可分为三部分。分别为文档信息提取与比较部分，形式为WebService，使用C#编写；Web服务后端，基于Spring Boot框架，使用Java编写；Web页面前端，使用Vue.js框架编写。

其中，Web服务后端Spring Boot接受前端请求，调用特定的WebService接口，得到含有文档信息的XML文件，处理后返回Json形式数据给前端，前端经过渲染后呈现给用户。

## 1.4 论文组织结构说明

本文详细介绍了特定业务文档规范性检测系统设计与实现的全过程，共分为六个章节，章节内容的概述如下。

第一章【概述】，此章主要介绍了本项目的项目需求、背景、研究意义以及项目的组成，最后介绍了论文的组织结构。

第二章【开发环境与技术】，此章主要介绍了项目实现中使用的开发工具、Web端所使用的框架以及文档检测功能使用的API工具等。

第三章【项目需求分析】，此章主要介绍了系统的功能结构分析，并绘制了数据流图。

第四章【系统设计】，此章主要介绍了系统的目标，功能结构图、各个功能的时序图以及详细的数据库设计。

第五章【系统实现】，本章主要介绍了每个功能的具体流程与实现，并展示部分系统的运行截图。

第六章【系统测试】，本章主要介绍了系统的测试流程，通过集成测试来验证本系统的有效性。

最后对本次毕业设计及论文的完成情况进行总结，并提出了对系统的完善及拓展的计划，同时向本次毕业设计中所有帮助过我的老师和同学致谢。最后是本文的参考文献列表。

# 开发环境与技术

此章主要介绍了项目实现中使用的开发工具、Web端所使用的框架以及文档检测功能使用的API工具等。

## 2.1 Web端开发——Spring Boot + Maven + MySQL + Shiro + Vue.js

本系统开发框架使用Spring Boot完成。

Spring是目前最受欢迎的企业级Java开发框架，其性能好、易于测试且可重用的特性被数以百万的Java开发人员喜爱。Spring框架是一个开源的Java平台，由Apache公司在2003年发布。

Spring的Web框架是一个设计良好的MVC框架，相对于Structs或其他框架，Spring Web提供更加优秀的开发选择。MVC框架指Model、View和Controller，这三层分别分则逻辑处理、渲染模型数据（即前端）以及处理用户请求，MVC框架使程序的三个方面，即输入逻辑、业务逻辑和UI逻辑分离，从而实现这些逻辑之间的解耦。

Spring Boot是在Spring的基础上的拓展，相对于Spring的复杂配置，Spring Boot消除了大量的配置，使用缺省值来代替用户完成开发配置，从而极有效地提升了开发效率，适合开发中小型软件，本系统使用Spring Boot进行开发。

Maven是一个项目管理工具，从项目的构建、管理依赖到最后的打包与项目的发布，Maven可以贯穿开发的始终。Maven可以使开发项目更加科学、更具维护性，它提供的多种自动化项目管理功能使开发者将更少的精力放在项目配置中，从而提高开发效率。Spring Boot集合了Maven，本项目的依赖管理与发布等均使用Maven操作。

MySQL是Oracle旗下的产品，是目前最流行的关系型数据库管理系统之一。MySQL对多种语言提供API接口，其中Spring Boot可以使用Spring JDBC对MySQL进行操作操作，提供原生SQL语句与JPA SQL等多种操作形式，使开发更加灵巧且便捷。且MySQL分为企业版和社区版，社区版使免费开源的软件，节约了开发成本。本系统使用MySQL作为后端数据库，提供持久化数据存储。

Shiro是Apache的一个Java安全框架，具有执行身份验证、授权、密码和会话管理等功能，Shiro拥有三个核心组件，用户自定义配置完成这三个组件后，即可轻松使用Shiro强大的用户身份管理功能。本系统使用Shiro作为用户管理的框架。

Vue.js使一种渐进式JavaScript框架，它被设计为可以自底向上逐层应用。Vue.js是中国人开发的框架，所以中文文档很全面，使Vue易于上手。且Vue拥有数量庞大的第三方库，使用Vue.js开发前端可以快速构建项目，且成效很快。

本系统中使用的是Vue.js的UI组件——ElementUI，ElementUI是一套基于Vue 2.0的桌面端组件库，它易于使用，提供多种实用组件，且为每个组件提供多种配置属性。本系统的前端页面基于ElementUI，为本系统提供组件，特效以及动态效果。

## 2.2 WebService端开发——Microsoft.Office.Interop.Word

Microsoft.Office.Interop.Word是微软官方提供的API，目的是方便开发人员通过C#操作Word文档。

该API提供了多种操作方法，开发人员通过代码，调用Office.Word的Com组件对文档进行操作。表2.1简略介绍了API提供的方法及其描述。

表2.1 API方法及其描述

|  |  |  |
| --- | --- | --- |
| 方法/属性名 | 参数 | 描述 |
| Application | 无 | 新建一个Word进程 |
| Documents.Open | filePath, ReadOnly, Visible | 打开filePath地址的文档，是否是只读打开，Word窗口是否可见 |
| Document.Close | SaveChanges, OriginalForamt | 关闭文档，默认为不保存，当SaveChanges为wdSaveChanges时，保存文档，OriginalFormat指保存文件类型是否是原格式 |
| Application.close | 无 | 关闭Word进程 |
| Document.Paragraphs | 无 | 该文档的所有段落数组 |
| Paragraph | 无 | 段落，可以使用Paragraph遍历Paragraphs数组 |
| Paragraph.OutlineLevel | 无 | 该段落的大纲等级 |
| Paragraph.Range | 无 | 指该段落范围，可以用此获得该段落的文本信息 |
| Range.Text | 无 | 该范围的文本 |
| Range.Font.Size | 无 | 该范围的字体大小 |
| Paragraph.SpaceBefore | 无 | 段前间距 |
| Paragraph.LineSpaceing | 无 | 行间距 |

## 2.3 开发工具——Visual Studio + IntelliJ IDEA + Visual Studio Code

Visual Studio是微软公司的开发工具，VS是一个完整的开发工具集，支持包括C、C++、C#以及Python等多语言的支持。同时作为一个集成开发环境，VS还提供了极为强大的代码补全、函数联想以及自动化代码整理等工具，使代码的编写更加快捷。也使代码的阅读更加美观。本系统因为需要使用操作Word的API（Microsoft.Office.Interop.Word），语言为C#，故使用Visual Studio作为文档检测功能的开发工具。

IntelliJ IDEA是JetBrains公司的开发工具，IDEA是开发人员最喜爱的Java开发工具之一，虽然不像Eclipse一样开源，但IDEA同样支持多种控件，拥有出色的拓展性。同时，IDEA在智能代码助手以及代码补全方面十分优秀，作为一个集成开发环境，其对Git、Junit等工具的支持与兼容也做得十分优异。本系统使用IntelliJ IDEA 2019作为后端Spring Boot的开发工具。

Visual Studio Code同样是微软公司的开发工具，VS Code是一个编译器而不是集成开发平台，但其出色的拓展性和其跨平台的特性，使其成为使用人数最多的开发工具。VS Code支持37种语言的编写，并且这个数量随着拓展插件的出现而不断增多。本系统前端使用Vue.js编写，VS Code对HTML和JS的支持很完善，具有强大的代码补全以及智能目录选取等功能，增加了开发效率。本系统使用Visual Studio Code作为前端的开发工具。

# 项目需求分析

此章主要介绍了系统的功能结构分析，并绘制了数据流图。通过用例图、字典以及数据流图，可以清晰地分析出本系统的需求，为系统设计提供帮助。

## 3.1 系统功能分析

系统的参与者有三种，分别为系统管理员，文档管理员以及用户，他们的功能如下。

系统管理员具有以下功能。

1. 登录
2. 设置用户权限

文档管理员具有以下功能。

1. 登录
2. 上传规范文档
3. 分析规范文档，系统自动抽取文档的逻辑结构和格式规范，作为后面文档检测的规范
4. 上传待检测文档
5. 查看规范文档
6. 下载规范文档
7. 分析待检测文档，系统自动利用检测规范对其进行检测，并记录检测结果
8. 查看检验结果
9. 下载包含批注的文档

用户有以下功能。

1. 登录
2. 查看规范文档
3. 下载规范文档
4. 上传待检测文档
5. 分析待检测文档
6. 查看检验结果
7. 下载包含批注的文档

根据以上功能描述，系统的用例图如图3.1所示。

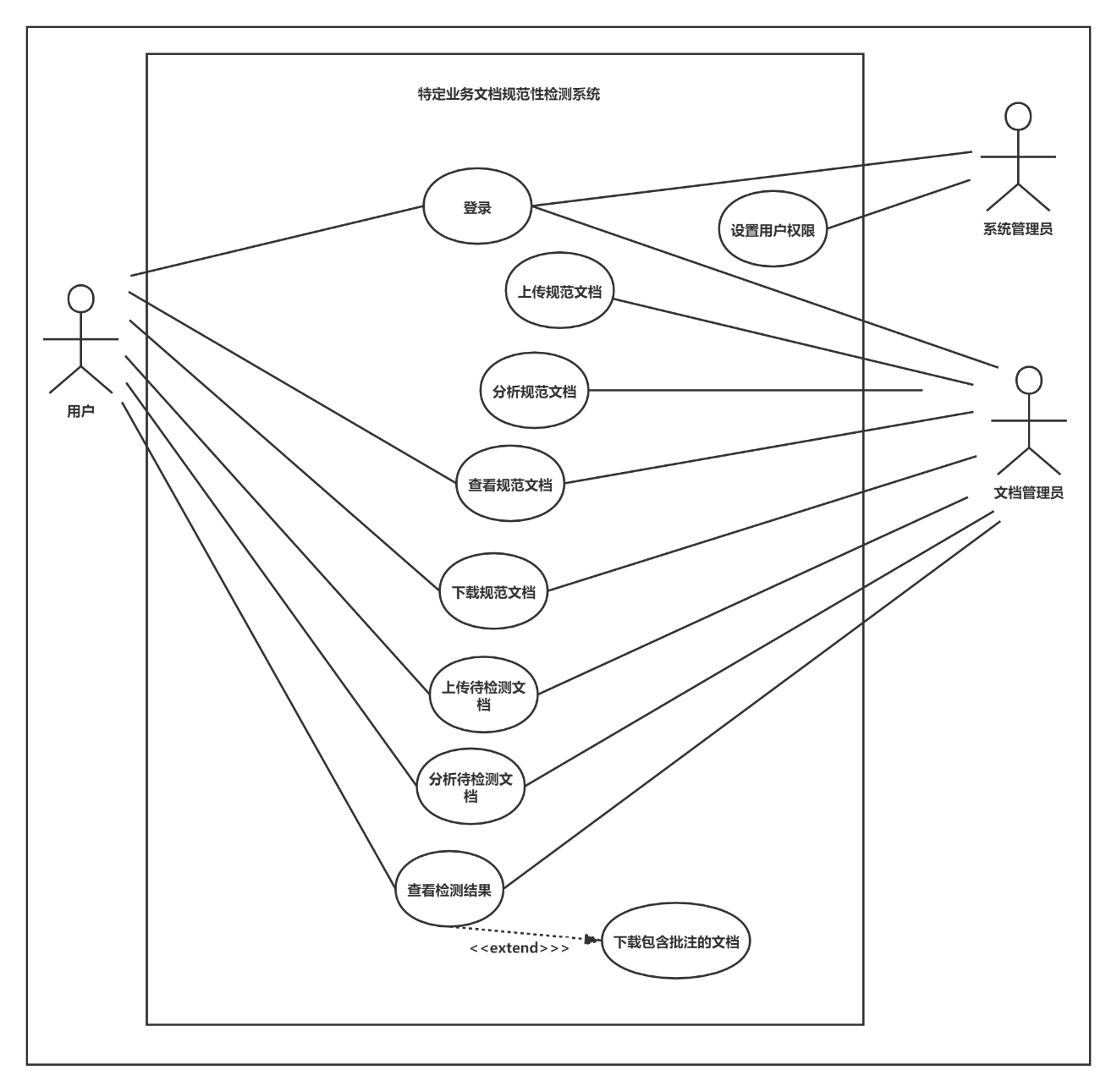


图3.1 系统用例图

用例名称、参与者、目的、前置条件以及基本事件流等用例详情，见表3.1至表3.10.

* “登录”用例：参与者可通过用户名密码进入系统，之后完成相应操作，具体详情见表3.1.

表3.1 登录用例详情表

|  |  |
| --- | --- |
| 用例名称： | 登录 |
| 参与者： | 系统管理员、文档管理员以及用户 |
| 目的： | 文档管理员或者用户登录本系统Web端，进入系统，继而进行其他操作。 |
| 前置条件： | 进入系统登录页面 |
| 基本事件流： | 1. 输入用户名以及密码，点击登录 2. Web服务后端进行登录验证：    1. 用户名以及密码正确，根据登录者身份进入不同的页面。系统管理员进入后台管理界面；文档管理员和用户进入文档检测主页面。    2. 用户名或密码错误，弹出错误提示，返回登录页面。 |

* “设置用户权限”用例：系统管理员可对其他系统使用者设置对应的权限，具体详情见表3.2.

表3.2 设置用户权限用例详情表

|  |  |
| --- | --- |
| 用例名称： | 设置用户权限 |
| 参与者： | 系统管理员 |
| 目的： | 系统管理员对系统使用者，设置其对应的权限。 |
| 前置条件： | 通过系统管理员登录验证，进入后台管理页面 |
| 基本事件流： | 1. 查看所有本系统的使用者信息。 2. 根据需要，对不同的使用者设置权限，可设为系统管理员、文档管理员以及用户。 |

* “上传规范文档”用例：参与者可向系统上传规范文档，系统进行相关验证并保存文档信息，具体详情见表3.3.

表3.3 上传规范文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 上传规范文档 |
| 参与者： | 文档管理员 |
| 目的： | 文档管理员上传规范文档到服务器。 |
| 前置条件： | 通过文档管理员登录验证，进入“模板文件”管理页面 |
| 基本事件流： | 1. 点击“上传模板文档”按钮， 2. 根据提示上传Word文档，支持doc/docx文件。 3. 上传不符合文件类型要求的文件时，弹出错误提示。 4. 上传成功，弹出成功信息，并刷新文件列表。 |

* “分析规范文档”用例：参与者可分析已经上传的规范文档，具体详情见表3.4.

表3.4 分析规范文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 分析规范文档 |
| 参与者： | 文档管理员 |
| 目的： | 文档管理员分析服务器上已经上传的规范文档，从而可以查看其逻辑结构 |
| 前置条件： | 1. 通过文档管理员登录验证，进入“模板文件”管理页面。 2. 待分析的规范文档，已经上传至服务器中。 |
| 基本事件流： | 1. 点击“分析”按钮， 2. Web后端调用WebService服务，对指定Word文档进行分析。 3. 分析成功，弹出成功提示。 4. 分析失败，弹出失败原因。 |

* “查看规范文档”用例：参与者可查看已经上传至系统的规范文档信息及详细内容。具体详情见表3.5.

表3.5 查看规范文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 查看规范文档 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户查看系统内的规范文档。 |
| 前置条件： | 通过文档管理员或用户的登录验证，进入“模板文件”管理页面。 |
| 基本事件流： | 1. 查看所有本系统内已经上传的规范文档。 2. 点击“查看”按钮，跳转到文档逻辑查看界面，查看该规范文档的逻辑结构。 |

* “下载规范文档”用例：参与者可下载已经上传至系统的规范文档。具体详情见表3.6.

表3.6 下载规范文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 下载规范文档 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户下载系统内的规范文档。 |
| 前置条件： | 通过文档管理员或用户的登录验证，进入“模板文件”管理页面。 |
| 基本事件流： | 1. 查看所有本系统内已经上传的规范文档。 2. 点击“下载”按钮，下载该规范文档。 |

* “上传待检测文档”用例：参与者可上传待检测文档至系统服务器。具体详情见表3.7。

表3.7 上传待检测文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 上传待检测文档 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户上传待检测文档到服务器中。 |
| 前置条件： | 通过文档管理员或用户的登录验证，进入“待检测文件”管理页面。 |
| 基本事件流： | 1. 点击“上传待检测文档”按钮。 2. 根据提示上传Word文档，支持doc/docx文件。 3. 上传不符合文件类型要求的文件时，弹出错误提示 4. 上传成功，弹出成功信息，并刷新文件列表。 |

* “分析待检测文档”用例：参与者可分析已经上传至系统的待检测文档。具体详情见表3.8.

表3.8 分析待检测文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 分析待检测文档 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户分析服务器上已经上传的规范文档，从而可以查看其逻辑结构 |
| 前置条件： | 1. 通过文档管理员或用户登录验证，进入“待检测文件”管理页面。 2. 待分析的待检测文档，已经上传至服务器中。 |
| 基本事件流： | 1. 点击“分析”按钮， 2. Web后端调用WebService服务，对指定Word文档进行分析。 3. 分析成功，弹出成功提示。 4. 分析失败，弹出失败原因。 |

* “查看检测结果”用例：参与者可通过系统查看文档检测结果。具体详情见表3.9.

表3.9 查看检测结果用例详情表

|  |  |
| --- | --- |
| 用例名称： | 查看检测结果 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户将待检测文件与规范文档进行比较，从而得到检测结果。 |
| 前置条件： | 1. 通过文档管理员或用户登录验证，进入“待检测文件”管理页面。 2. 该待检测文件的模板文件，已经进行过分析。 3. 该待检测文件，已经进行过分析。 |
| 基本事件流： | 1. 点击“比较”按钮。 2. Web后端调用WebService服务，对指定待检测文档进行比较。 3. 比较成功，弹出成功提示。 4. 比较失败，弹出失败原因。 |

* “下载包含批注的文档”用例：参与者可下载批注文档，具体详情见表3.10.

表3.10 下载包含批注的文档用例详情表

|  |  |
| --- | --- |
| 用例名称： | 载包含批注的文档 |
| 参与者： | 文档管理员、用户 |
| 目的： | 文档管理员或用户下载含有比较结果的批注文档。 |
| 前置条件： | 1. 通过文档管理员或用户登录验证，进入“待检测文件”管理页面。 2. 该待检测文件，已经进行过比较。 |
| 基本事件流： | 1. 点击“比较并下载批注文件”按钮。 2. Web后端调用WebService服务，根据比较结果，对指定待检测文档进行添加批注操作。 3. 添加批注成功，弹出成功提示。 4. 再次点击“比较并下载批注文件”按钮，开始下载批注文件 5. 添加批注失败，弹出失败原因。 |

## 3.2 系统数据处理逻辑

本文通过数据流图详细介绍系统数据处理逻辑。其中，本系统顶层数据流图如图3.2所示，1级数据流图如图3.3所示。

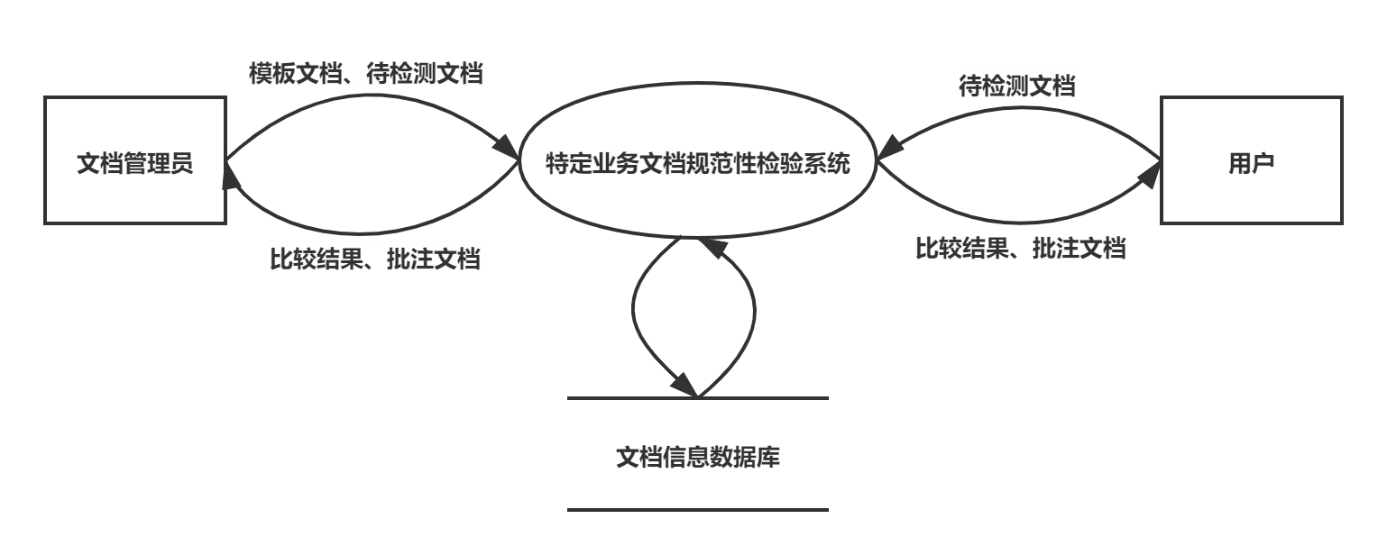


图3.2 顶层数据流图

在本系统中，外界的数据来源有：

* 文档管理员：输入模板文档以及待检测文档。系统应返回比较结果和添加过批注的检测文档。
* 用户：输入待检测文档。系统应返回比较结果以及添加过批注的检测文档。

文档信息数据库中应储存模板文档、待检测文档的服务器中的文件地址，比较后的XML文件地址等。

根据顶层数据流图的分析，可以分析得到1级数据流图，1级数据流图如图3.3所示。

从1级数据流图分析得知系统主要功能有：

1. 登录
2. 修改个人信息
3. 上传并分析模板文档
4. 上传并分析待检测文档
5. 检测文档规范性
6. 添加批注
7. 下载文档

其中，（3）上传并分析模板文档功能仅文档管理员拥有。

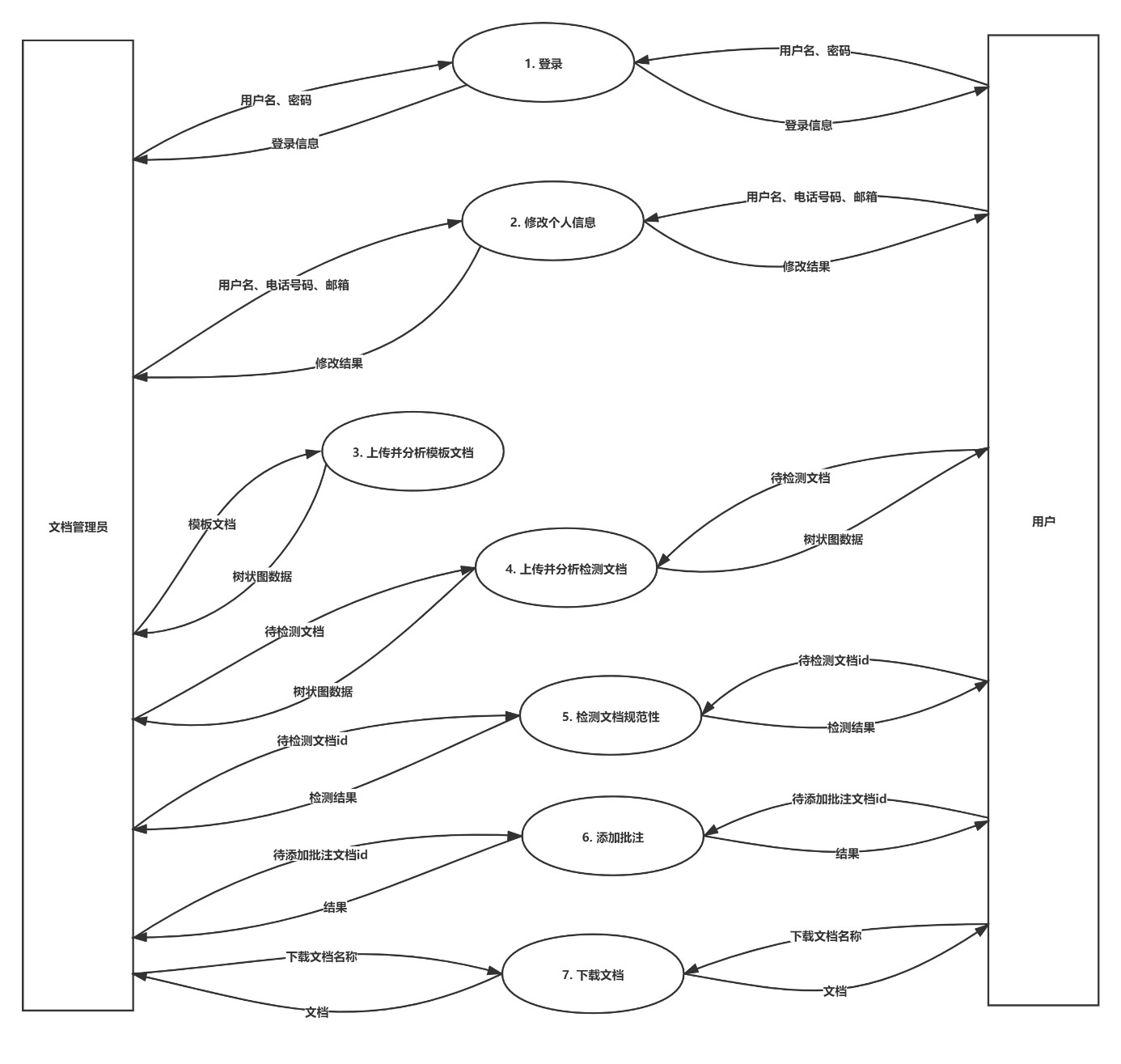


图3.3 1级数据流图

# 特定业务文档规范性检测系统的设计

本章主要说明系统的功能结构、系统活动图以及系统的数据库设计。

## 4.1 概要设计

4.1.1 系统功能设计

特定业务文档规范性检测系统需要完成以下目标：

1. 文档模版的规范抽取：对于给定的文档模版，自动抽取其中的文档逻辑结构和格式规范，保存相关规范性要求作为文档检测的依据。
2. 文档规范性检测：给定被测文档，根据文档检测的依据对其文档结构和格式进行自动化地规范性检测，并保存检测结果。
3. 检测结果的可视化：将检测结果以批注的方式加在被测文档中，同时在Web端为用户提供可视化和交互功能。

特定业务文档规范性检测系统的功能结构如图4.1所示。

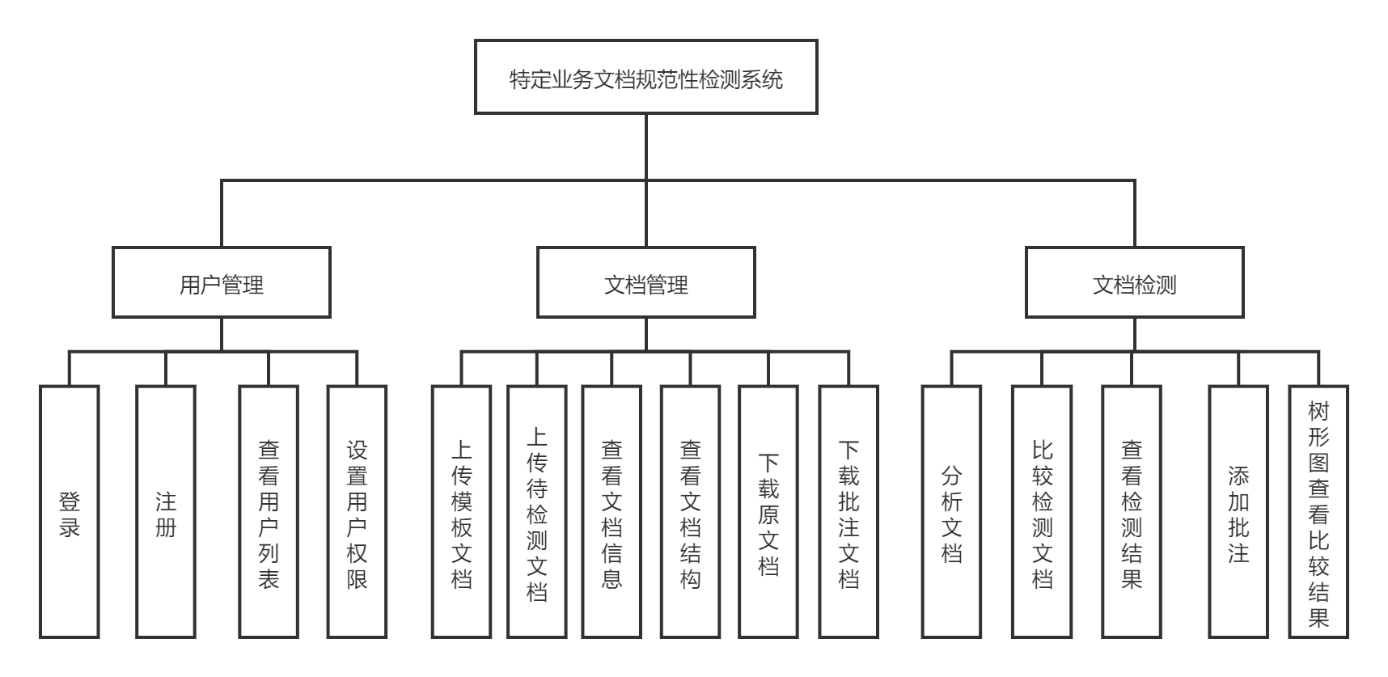


图4.1 特定业务文档规范性检测系统的功能结构图

本系统的功能主要分为三个模块，分别为用户管理、文档管理与文档检测。三个模块下细分为多个功能，具体描述如下。

1. 用户管理：主要负责用户的登录注册，信息与权限管理等
2. 【登录】用户在系统登录界面登录个人账号
3. 【注册】新用户在系统注册界面注册新账号
4. 【查看用户列表】系统管理员通过后台管理界面查看所有用户信息
5. 【设置用户权限】系统管理员通过后台管理界面设置修改用户权限。
6. 文档管理：主要负责文档的上传下载，文档信息的获取等
7. 【上传模板文档】用户上传模板Word文档至服务器。
8. 【上传待检测文档】用户上传待检测Word文档至系统。
9. 【查看文档信息】用户通过文档列表查看文档信息，包括文件名、上传人、上传时间等
10. 【查看文档结构】用户查看系统上Word文档的逻辑结构
11. 【下载原文档】用户通过文档列表下载指定的原文档。
12. 【下载批注文档】用户通过文档列表下载经过比较，添加过批注的文档。
13. 文档检测：主要负责文档逻辑信息与样式信息的提取，文档比较检测等
14. 【分析文档】系统调用服务分析文档，生成XML文件
15. 【比较模板文档与待检测文档】系统调用服务，将模板文档和待检测文档进行比较，比较的结果保存为XML文件
16. 【查看检测结果】用户查看检测结果，以表格的形式展示检测结果。
17. 【添加批注】系统通过比较结果生成的XML，为文档添加批注。
18. 【树形图查看比较结果】系统以树形图的方式向用户展示比较结果。

4.1.2 系统整体技术方案设计

本系统的三个核心功能分别是：

1. 文档模板的规范抽取
2. 文档规范性检测
3. 检测结果的可视化

其中（1）（2）通过C#实现，（3）通过Web端实现，以上三个部分的调用关系示意图如图4.2所示。

模板文档上传者通过页面上传模板文档，通过Web后端向C# WebService发送文档，文档经过规范抽取后，生成模板XML文件，保存至服务器中。

用户通过页面上传待检测文档，通过Web后端向C# WebService发送文档，文档首先经过信息抽取，生成待检测XML文件，随后用户可使用文档规范性检测功能，系统读取模板XML和待检测XML两个文件后，经分析后生成含有批注的Word文档，将该文档的存储信息返回给Web后端后，Web后端可调用下载将文档传输给页面，用户可下载含有批注的Word文档。

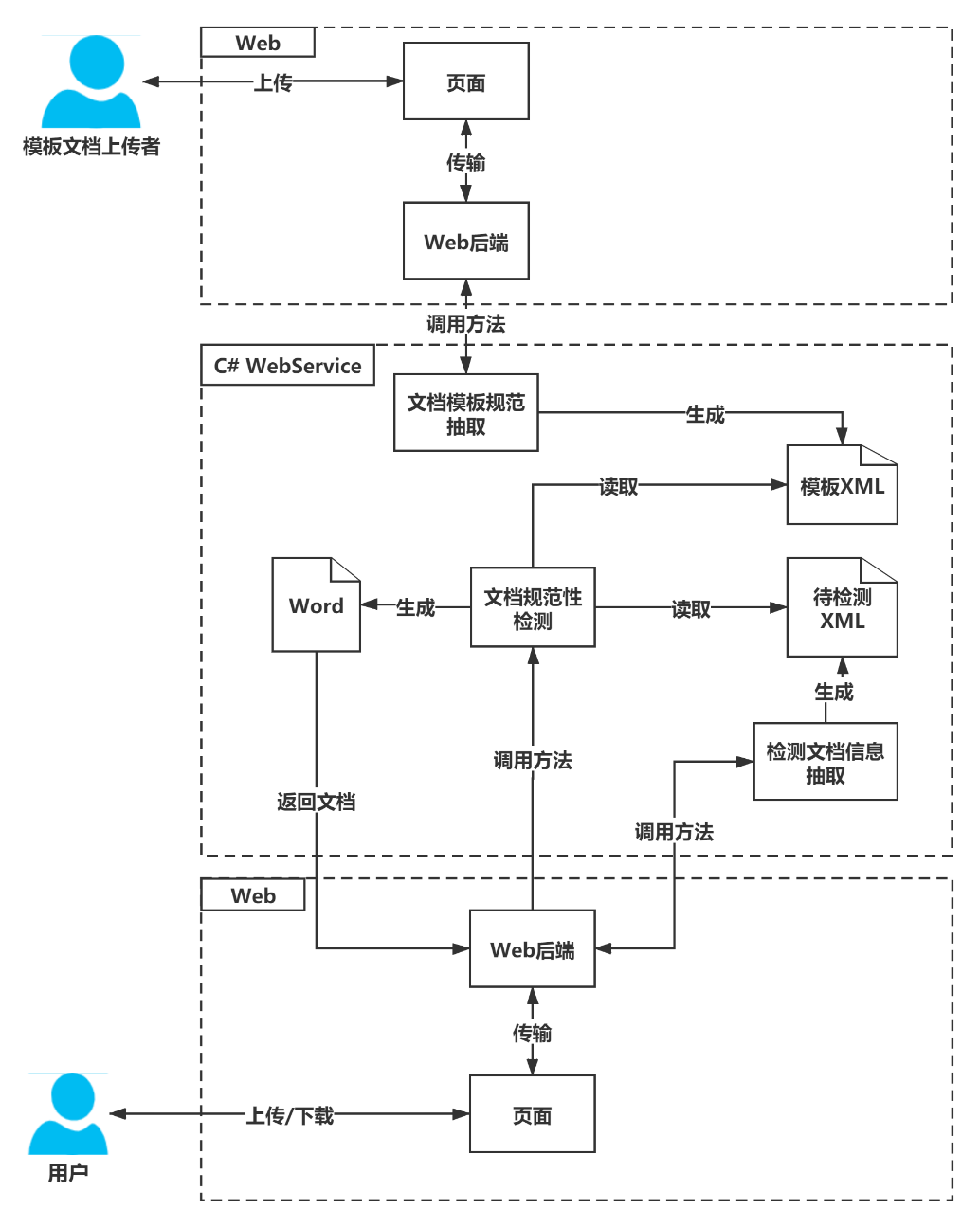


图4.2 核心功能调用关系示意图

以上三个核心功能之间的调用方式，如图4.3所示。

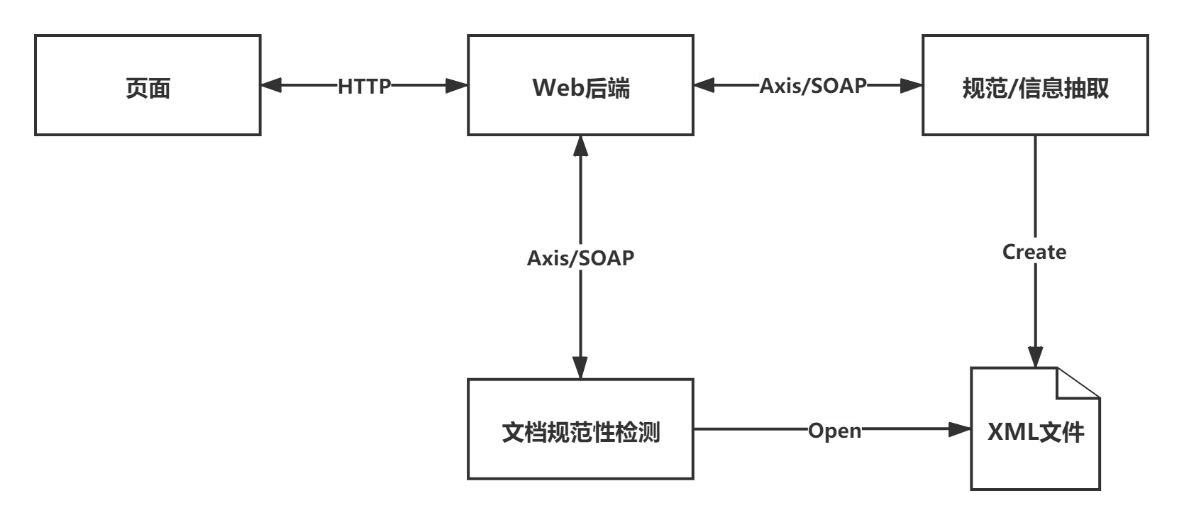


图4.3 核心方法调用方式示意图

页面与Web后端通过HTTP进行交互，具体为HTTP请求。

Web后端与规范/信息抽取功能模块以及文档规范性检测功能模块通过Axis/SOAP进行交互，通过传输WebService设定的参数，如String、BASE64BINARY等格式，WebService返回XML文件，包含状态信息等。

文档规范性检测模块与规范/信息抽取模块通过XML文件交互，读取地址上的XML文件进行规范性分析检测。

## 4.2 系统处理逻辑

动态图又称活动图，适用于阐述业务用例实现的工作流程和处理逻辑。本文使用活动图用于说明系统的处理逻辑，根据使用者的角色划分为系统管理员、文档管理员和用户，其中用户大部分功能与文档管理员相似或重复，在此不再赘述。

4.2.1 系统管理员相关功能处理逻辑

系统管理员具有设置用户信息等功能，特定业务文档规范性检测系统的系统管理员活动图如图4.4所示。

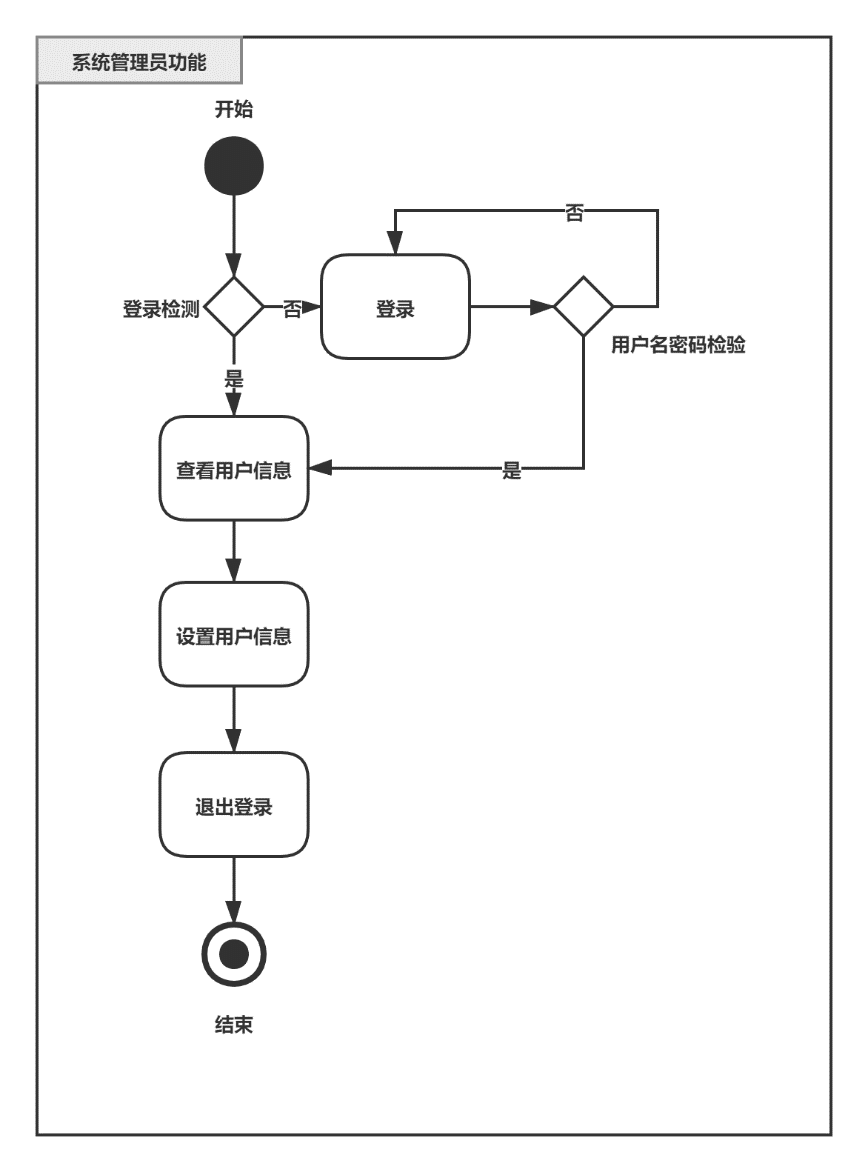


图4.4 特定业务文档规范性检测系统的系统管理员活动图

图4.4描述了系统管理员使用本系统Web端的主要功能调用流程顺序，图中每个活动的含义解释如下。

1. 【登录检测】：系统判定系统管理员是否已经登录，未登录则进入【登录】，否则进入【查看用户信息】。
2. 【登录】：系统的登录功能，接收用户输入的用户名和密码，进入【用户名密码检验】
3. 【用户名密码检验】：系统对用户名以及密码进行检验，若检验通过则进入【查看用户信息】，否则返回至【登录】，重新接收用户名和密码。
4. 【查看用户信息】：系统以列表的形式向系统管理员显示全部用户信息。
5. 【设置用户信息】：系统管理员可以设置用户的相关信息，包括用户的邮箱、联系方式以及权限等。
6. 【退出登录】：系统清除登录信息，返回至登录页面。

4.2.2 文档管理员相关功能处理

特定业务文档规范性检测系统的文档管理员活动图如图4.5所示。

图4.5较为完整的描述了文档管理员以及文档检测功能中各部分的活动流程，图中活动的具体解释如下。

1. 【登陆检测】、【登录】、【用户名密码检验】：与图4.4中解释相同，不再赘述。
2. 【查看模板文档信息】：系统以列表的形式向用户文档管理员显示所有模板文档信息，包括上传人、上传时间等。
3. 【分析模板文档】：Web端调用C# WebService对Word文档进行分析。
4. 【下载模板文档】：系统提供下载按钮，可以下载模板文档。
5. 【提取文档逻辑结构与样式信息】：C# WebService端对文档进行分析，提取其中的逻辑结构与段落样式信息。
6. 【生成XML】：对提取出的逻辑结构与样式信息，以XML文件的格式保存。
7. 【查看模板文档结构】：系统解析XML文件，以树形图显示文档逻辑结构。
8. 【查看待检测文档信息】：系统以列表的形式向用户文档管理员显示所有待检测文档信息，包括上传人、上传时间、该文档对应模板文档的相关信息等。
9. 【下载原文档】：系统提供下载按钮，可以下载未批注的原文档。
10. 【分析待检测文档】：Web端调用C# WebService对Word文档进行分析。
11. 【提取文档逻辑结构与样式信息】：C# WebService端对文档进行分析，提取其中的逻辑结构与段落样式信息。
12. 【生成XML】：对提取出的逻辑结构与样式信息，以XML文件的格式保存。
13. 【查看模板文档结构】：系统解析XML文件，以树形图显示文档逻辑结构。
14. 【检验文档】：Web端调用C# WebService对待检测文档进行检测。
15. 【比较模板文档与待检测文档XML文件】：C# WebService端读取两个XML文件，对两者进行比较分析。
16. 【生成检测结果XML】：对比较分析的结果，以XML文件的形式保存。
17. 【显示检测结果】：系统以树形图的形式显示检测结果，文档中含有逻辑规范性问题的部分给与标注显示。
18. 【添加批注】：Web端调用C# WebService对Word文档进行添加批注。
19. 【根据检测结果XML为Word文档添加批注】：C# WebService端读取检测结果XML，对文档添加批注。
20. 【生成添加批注的Word文档】：C# WebService端将批注文档另存为一个Word文档。
21. 【下载批注文档】：系统提供下载批注文档按钮，用户可下载添加过批注的Word文档。

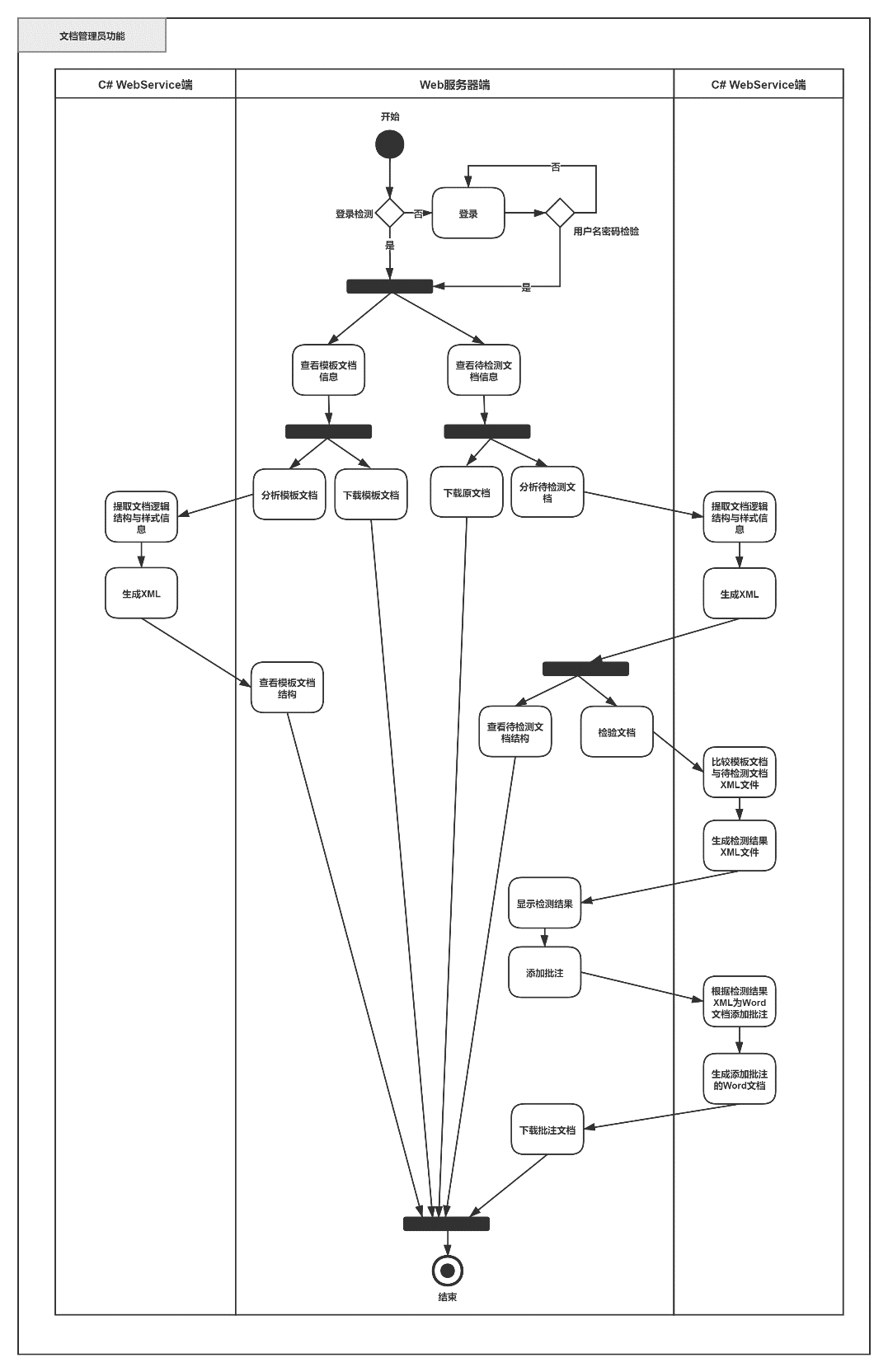


图4.5 特定业务文档规范性检测系统的文档管理员活动图

## 4.3 数据库设计

本系统使用实体关系图描述数据库设计。用户管理的实体关系图如图4.6所示。文档检测的实体关系图如图4.7所示。本系统使用MySQL，可以通过设计E-R图生成数据库表，十分便利。

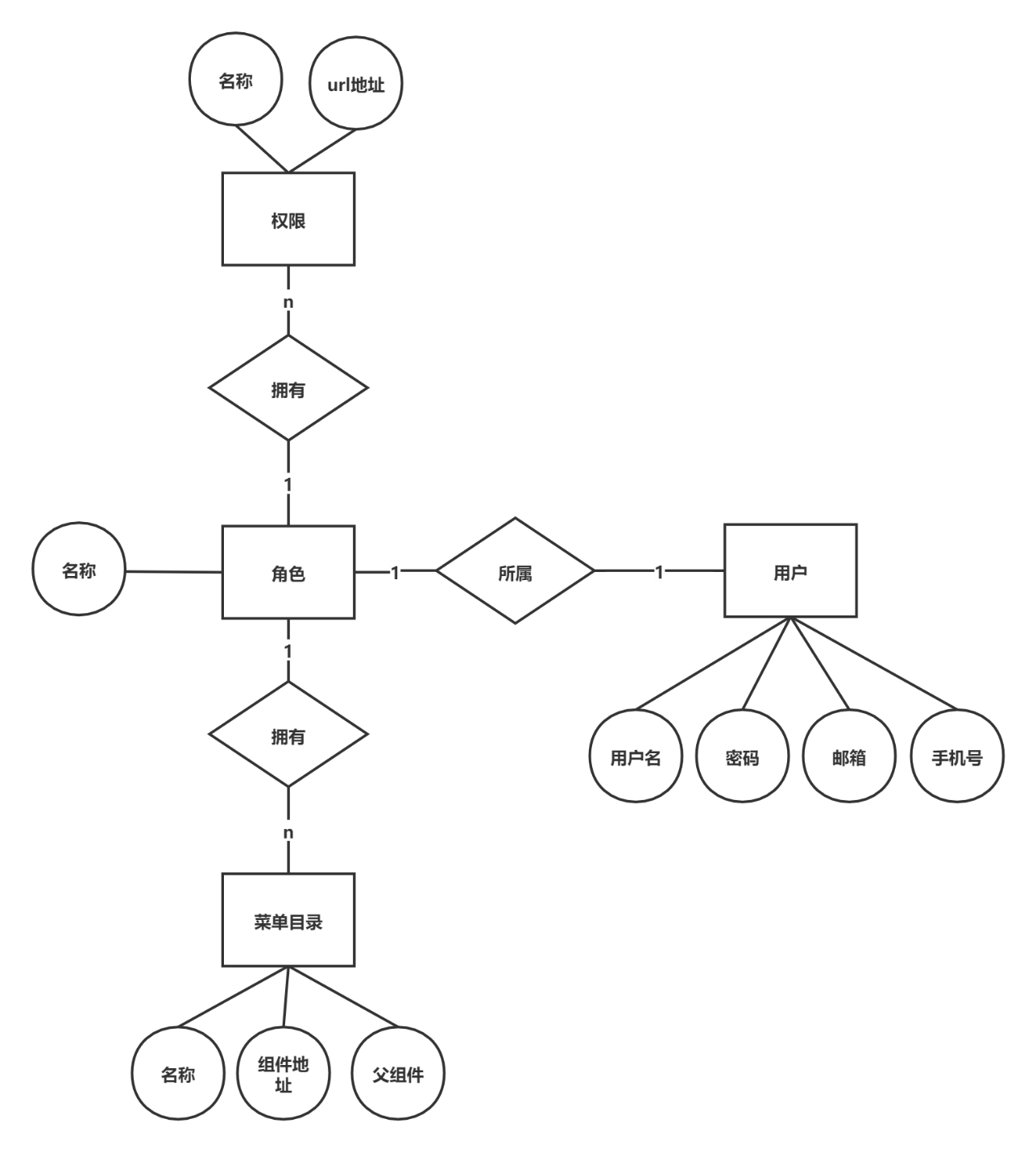


图4.6 用户管理的实体关系图

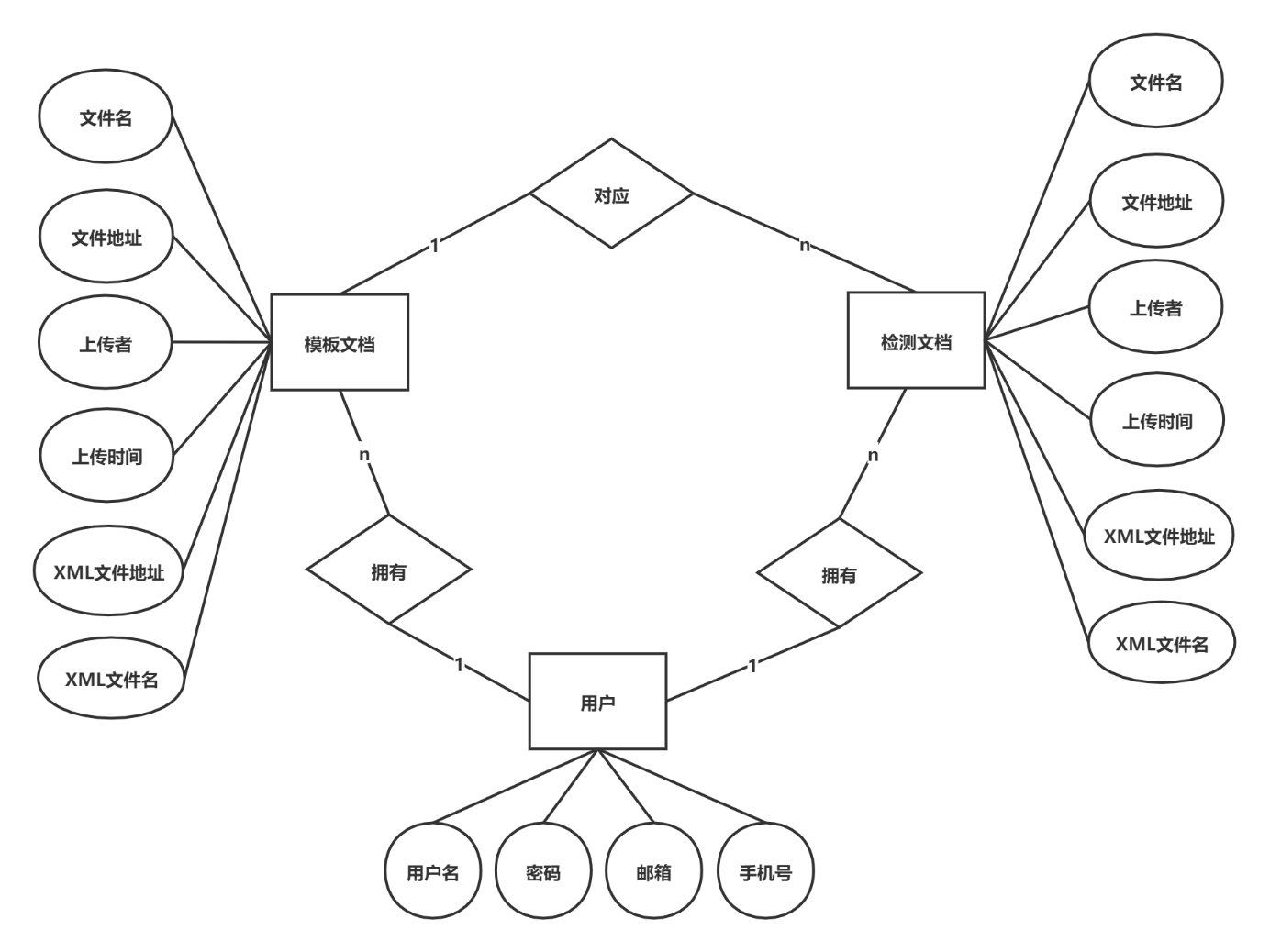


图4.7 文档检测的实体关系图

由图4.6和图4.7，本系统建立数据库wordSyn，并建立数据表user，admin\_user\_role，admin\_role，admin\_role\_permission，admin\_role\_menu，admin\_menu，admin\_permission，modelfile，docfile，comparefile。以上数据表的具体描述如表4.1至表4.10所示。

1. 用户信息表user

用户信息表user主要存储关于用户的基本信息，如用户名、密码、手机号以及邮箱等。具体描述如表4.1所示。

表4.1 用户信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | user | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| id | Y | INT | 主键 | 用户id、自增 |
| username | Y | CHAR |  | 用户名、唯一 |
| password | Y | VARCHAR |  | 经过加密的密码 |
| salt | Y | VARCHAR |  | “盐”，用于密码加密 |
| Name | Y | VARCHAR |  | 真实姓名 |
| Phone | N | VARCHAR |  | 手机号 |
| Emile | N | VARCHAR |  | 邮箱 |
| Enabled | Y | TINYINT |  | 状态，表明是否被禁用 |

1. 用户角色表admin\_user\_role

用户角色表admin\_user\_role主要存储用户与角色的对应关系。具体描述如表4.2所示。

表4.2 用户角色表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_user\_role | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 用户角色表id，自增 |
| Uid | Y | INT | 外键 | 用户id |
| Rid | Y | INT | 外键 | 角色id |

1. 角色信息表admin\_role

角色信息表admin\_role主要存储角色的相关信息，如角色的英文名及中文名等。具体描述如表4.3所示。

表4.3 角色信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_role | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 角色id，自增 |
| Name | Y | VARCHAR |  | 英文名称 |
| Name\_zh | Y | VARCHAR |  | 中文名称 |
| Enabled | Y | TINYINT |  | 状态，该角色是否被禁用 |

1. 角色权限表admin\_role\_permission

角色权限表admin\_role\_permission主要存储角色与权限的对应关系。具体描述如表4.4所示。

表4.4 角色权限表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_role\_permission | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 角色权限id，自增 |
| Rid | Y | INT | 外键 | 角色id |
| Pid | Y | INT | 外键 | 权限id |

1. 菜单信息表admin\_menu

菜单信息表admin\_menu主要存储菜单的信息，如前端地址、中英文名、前端使用的组件及父Id等。具体描述如表4.5所示。

表4.5 菜单信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_menu | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 角色菜单id，自增 |
| Path | Y | VARCHAR |  | 该菜单在前端Vue的路径 |
| Name | Y | VARCHAR |  | 菜单英文名称，与Vue中的name属性对应 |
| Name\_zh | Y | VARCHAR |  | 菜单中文名称 |
| Icon\_cls | N | VARCHAR |  | 菜单的图标 |
| Component | Y | VARCHAR |  | 菜单在Vue中的组件名 |
| Parent\_id | Y | Int |  | 该菜单的父菜单id |

1. 角色菜单表admin\_role\_menu

角色菜单表admin\_role\_mene主要存储了角色与权限的对应关系。具体描述如表4.6所示。

表4.6 角色菜单表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_role\_menu | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 角色菜单id，自增 |
| Rid | Y | INT | 外键 | 角色id |
| Mid | Y | INT | 外键 | 菜单id |

1. 权限信息表admin\_permission

权限信息表admin\_permission主要存储了权限的相关信息，如中英文名以及前端url地址等。具体描述如表4.7所示。

表4.7 权限信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | admin\_permission | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 权限id |
| Name | Y | VARCHAR |  | 英文名称 |
| Desc | Y | VARCHAR |  | 中文名称 |
| url | Y | VARCHAR |  | 该权限可以访问的Api地址 |

1. 模板文档信息表modelfile

模板文档信息表modelfile主要存储了模板文档的基本信息，如文档地址、上传时间、上传人、分析后的XML文件地址等。具体描述如表4.8所示。

表4.8 模板文档信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | modelfile | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 模板文档id |
| File\_location | Y | VARCHAR |  | 文件在服务器的地址 |
| File\_name | Y | VARCHAR |  | 文件名 |
| Update\_time | Y | VARCHAR |  | 上传时间 |
| Uploader | Y | VARCHAR |  | 上传人 |
| Xml\_file\_location | N | VARCHAR |  | 分析后的xml文件地址 |
| Xml\_file\_name | N | VARCHAR |  | Xml文件名 |
| Uploader\_id | Y | INT | 外键 | 上传人id，外键接user表id |

1. 检测文档信息表docfile

检测文档信息表docfile主要存储了检测文档的基本信息，如文档地址、文档名、文件保存地址、上传人等信息。具体描述如表4.9所示。

表4.9 检测文档信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | docfile | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Id | Y | INT | 主键 | 检测文档id |
| File\_location | Y | VARCHAR |  | 文件在服务器的地址 |
| File\_name | Y | VARCHAR |  | 文件名 |
| Update\_time | Y | VARCHAR |  | 上传时间 |
| Uploader | Y | VARCHAR |  | 上传人 |
| Xml\_file\_location | N | VARCHAR |  | 分析后的xml文件地址 |
| Xml\_file\_name | N | VARCHAR |  | Xml文件名 |
| Uploader\_id | Y | INT | 外键 | 上传人id，外键接user表id |

1. 检测关系表comparefile

检测关系表comparefile主要存储了模板文档与检测文档的对应关系，同时存储了比较后的XML文件以及批注文件的相关信息，如检测结果XML文件地址、批注文件地址等。具体描述见表4.10.

表4.10 检测关系表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | Comparefile | | | |
| 字段名 | Not Null | 数据类型 | 约束类型 | 备注 |
| Compare\_id | Y | INT | 主键 | 检测id |
| Compare\_file\_location | N | VARCHAR |  | 比较结果的xml文件地址 |
| Compare\_file\_name | N | VARCHAR |  | Xml文件名 |
| Doc\_file\_id | Y | INT | 外键 | 检测文档id |
| Model\_file\_id | Y | INT | 外键 | 模板文档id |
| Compare\_doc\_word\_file\_location | N | VARCHAR |  | 添加过批注的Word文档地址 |

本系统使用MySQL作为数据库，MySQL中表结构如图4.8所示。

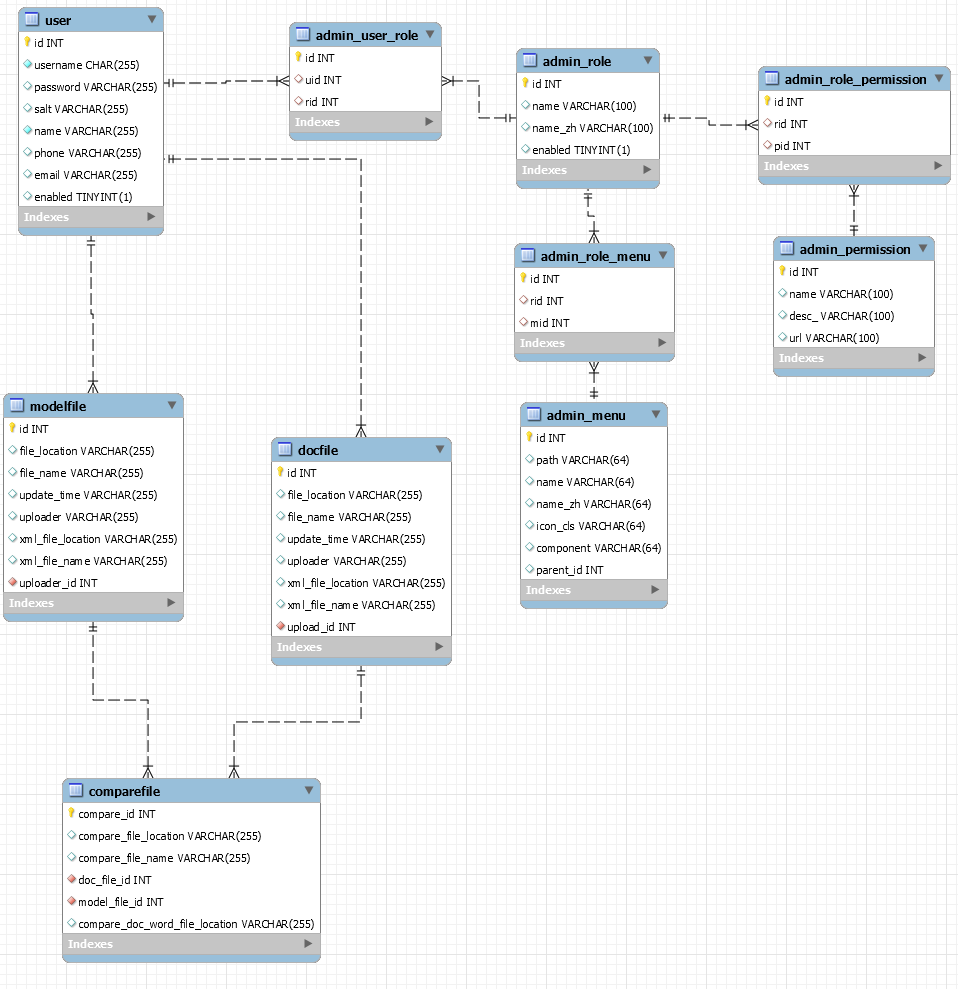


图4.8 MySQL表结构图

## 4.4 详细设计

4.4.1 登录功能设计

用户（系统管理员、文档管理员和用户）登录本系统的时序图如图4.9所示。

用户发送login请求到后端，loginController接收用户名密码，调用loginService层对用户名密码进行验证，将用户名密码发送给Dao层，调用Selcet语句对数据库进行查询，得到其id以及身份信息，返回给Service层，继而返回给Controller层，Controller对数据进行包装，返回Result，含有请求返回值以及用户信息数据，登录页根据返回数据中用户id，即roleId判定用户身份，根据身份跳转至不同页面，系统管理员进入/admin，文档管理员和用户进入/modelFile。

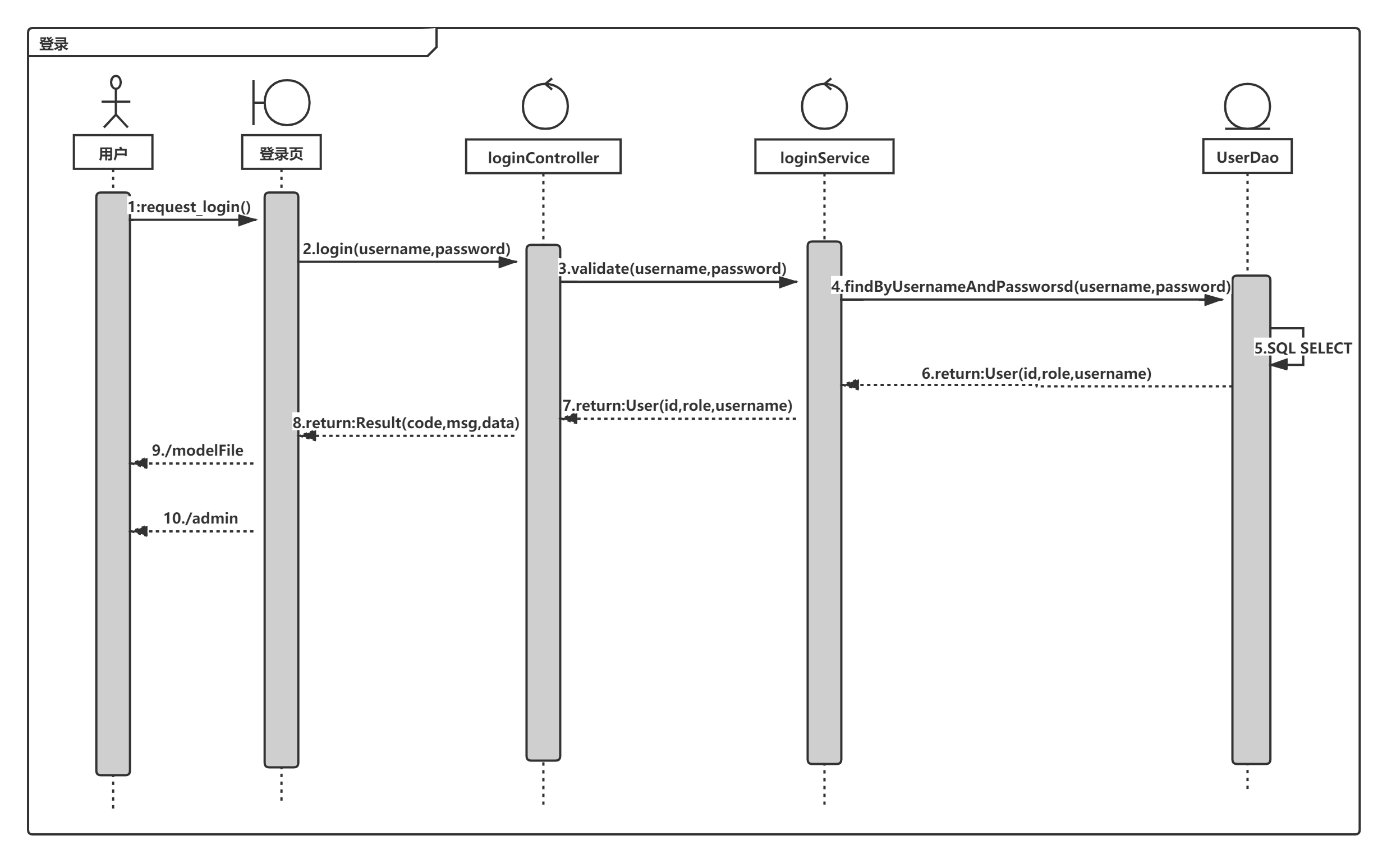


图4.9 登录功能的时序图

4.4.2 注册功能设计

用户注册功能的时序图如图4.10所示。

用户输入用户名和密码，向loginController发送注册请求，Controller将用户名和密码发送给loginService，首先验证该用户是否已存在，存在则直接返回错误信息，不存在则调用save方法将用户信息添加进数据库表中，返回添加用户是否成功的信息，Controller层将信息包装为Result，注册页判断Result的返回值，成功则跳转至/login，不成功则弹出错误信息。

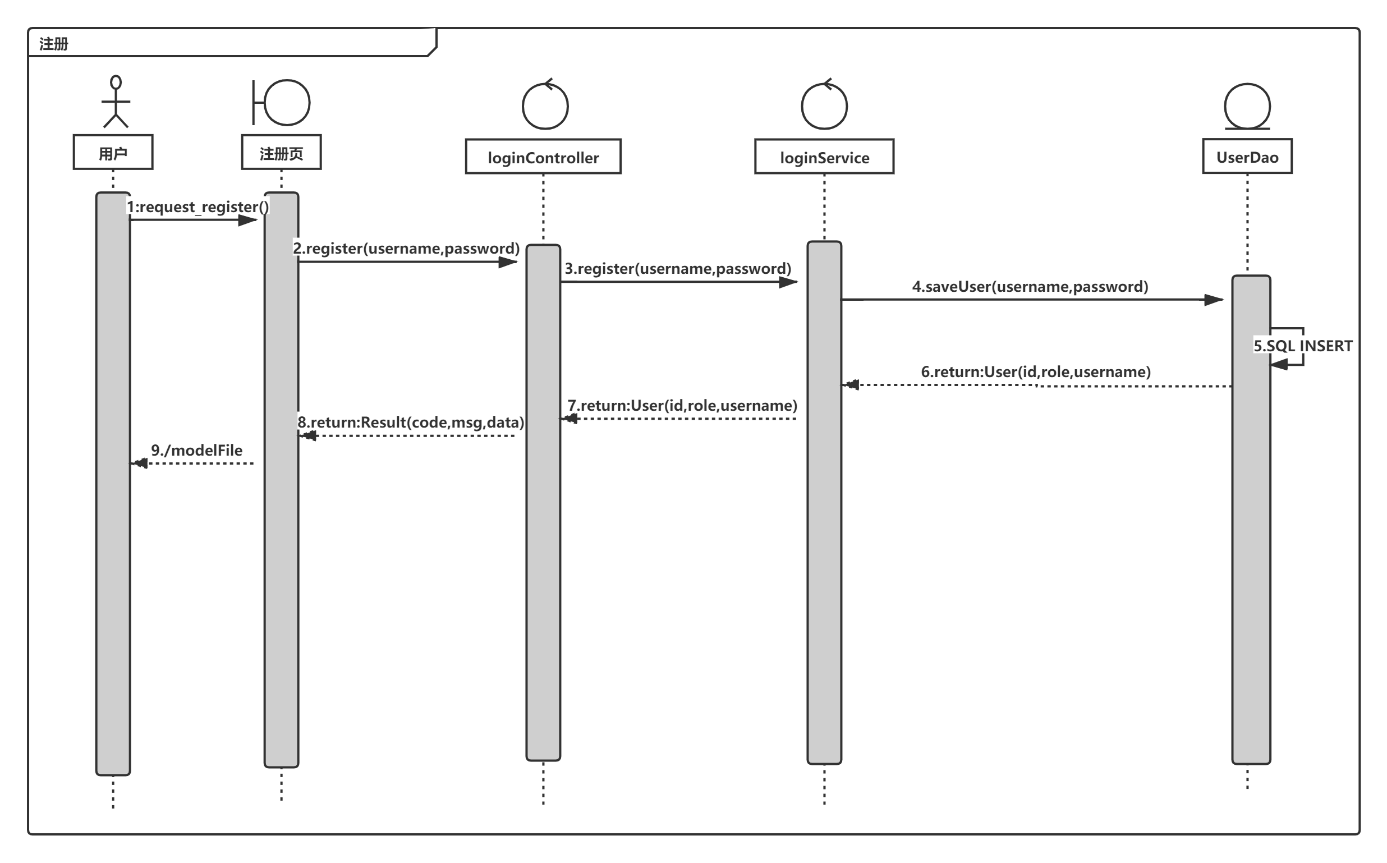


图4.10 注册功能的时序图

4.4.3 用户管理功能设计

用户管理功能的时序图如图4.11所示。

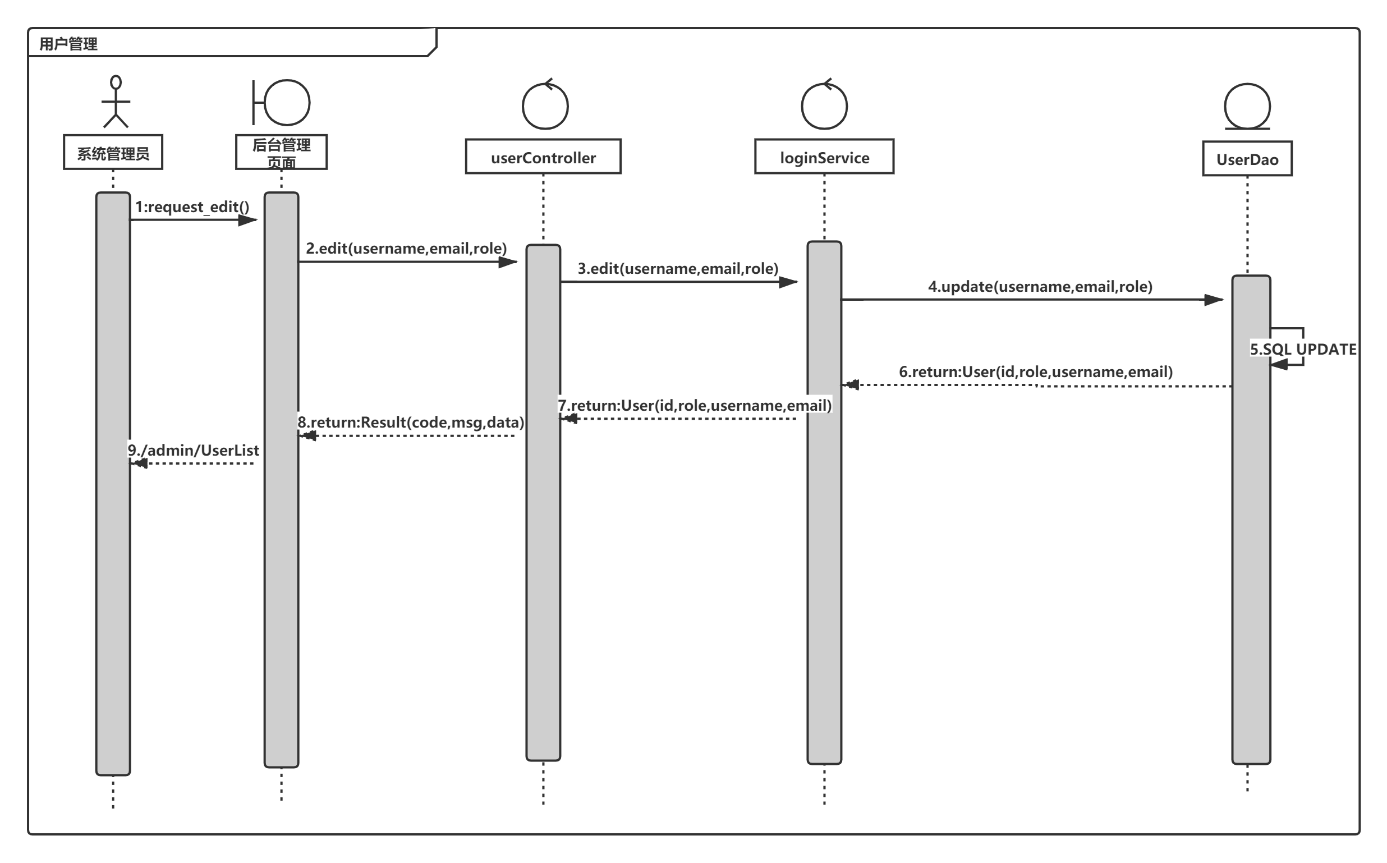


图4.11 用户管理功能的时序图

系统管理员编辑用户信息，向userController发送修改请求，Controller将用户名，修改后的email以及角色信息发送给Service，Service调用update方法修改更新数据库中用户信息，返回更新后的User实例，Controller将用户信息包装为Result返回给后台管理页面，管理页面刷新列表返回给系统管理员。

4.4.5 文档规范信息抽取功能设计

文档规范信息抽取功能为本系统核心功能之一，主要流程为通过Microsoft提供的API进行逻辑信息与样式信息的提取，最终生成XML文件供文档检测功能调用。

文档的格式信息有很多种，经过分析比对后，本系统使用八种信息进行文档段落格式的确定，具体八种信息及其描述如表4.11所示。

表4.11 抽取段落信息名称及描述表

|  |  |
| --- | --- |
| 抽取的段落信息名称 | 描述 |
| title | 标题信息 |
| fontName | 字体名称 |
| fontSize | 字号 |
| fontBold | 是否加粗 |
| fontItalic | 是否斜体 |
| lineBefore | 段前间距 |
| lineAfter | 段后间距 |
| lineHeight | 行间距 |

本系统使用XML文档进行文档规范信息的存储方式。XML文档自身是一种树形结构，用来存储具有层次信息的数据较为合适，同时XML文档的元素与属性的组合，可以便捷直观的存储多种样式信息。除此之外，XML文档可以通过XML验证器的方式，为XML文档制定规范（如XML Schema），以此产生符合规范的XML文档，从而使不同的开发人员可以共享XML文档信息，以此提高开发效率。

基于以上描述，本系统首先设计XML Schema验证器，为不同的信息设定的存储格式，具体存储格式如表4.12所示。

表4.12表示，一级大纲名为<chapter1>，个数限制为至少一个，内部含有三个部分，分别为标题title，类型为string；第二部分为样式style，类型为attribute，以属性的形式存储各种样式信息，如字体字号等；第三部分为该一级标题的子标题，即二级标题<chapter2>，个数限制为大于0个，即该部分可以为无，内部同样含有三个部分，与一级标题相似，但子标题部分为<chapter3>，以此类推，得到XML的验证器，编写为XSD文件。

表4.12 验证器设计表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 一级大纲 | 段落信息 | 信息名称 | 信息类型 | 描述 |
| chapter1 [1..\*] | title | - | xs:string | 一级标题 |
| style | font\_type | xs:attribute | 字体 |
| font\_size | 字号 |
| bold | 是否加粗 |
| line-height | 段前间距 |
| line-before | 段后间距 |
| line-after | 行间距 |
| italic | 是否斜体 |
| chapter2 [0..\*] | title | xs:String | 二级标题 |
| style | xs:arrtibute | 二级样式信息 |
| chapter3 [0..\*] | - | 三级标题 |

通过Visual Studio的架构管理视图可以更为清晰地查看XSD文件结构，架构管理视图如图4.12所示。

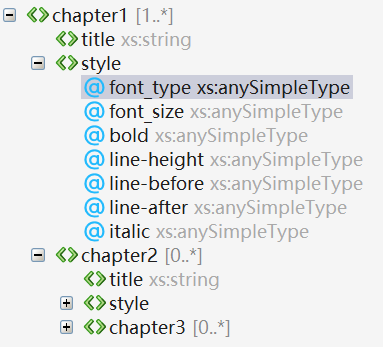


图4.12 XSD文件架构管理视图

以此XSD文档作为验证器，对于图4.13所示的Word文档，调用Microsoft提供的API，可以得到如图4.14的XML文档。可以看出，通过XML文档可以描述Word文档的逻辑信息与样式信息。

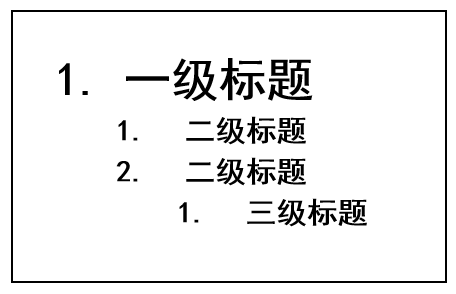


图4.13 Word文档示例

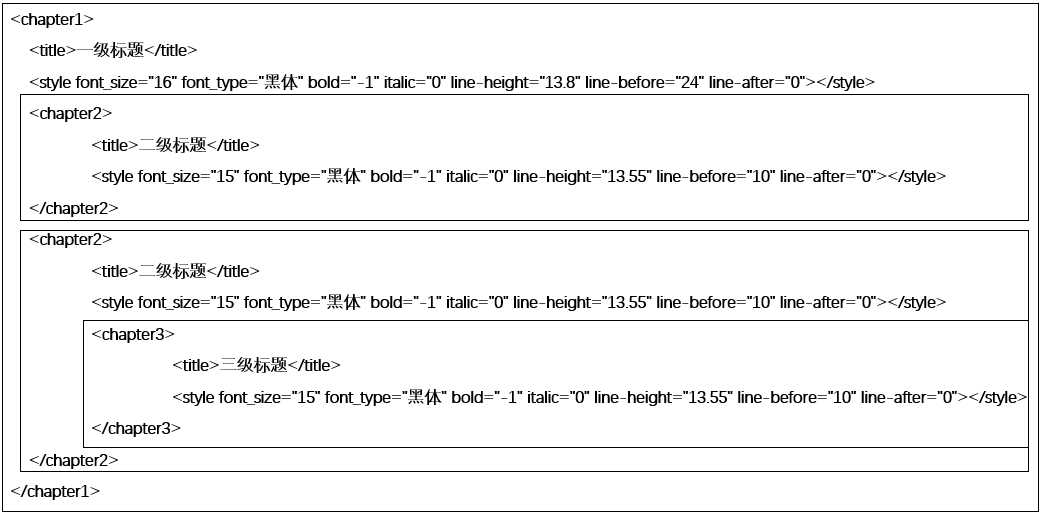


图4.14 XML的存储形式示意图

对于文档的另一种形式的段落，即无大纲等级的正文段落，本系统使用如图4.15的XML存储形式进行信息存储。使用<paragraph>元素为正文段落，该元素在XML文档中的个数限制为大于0个。

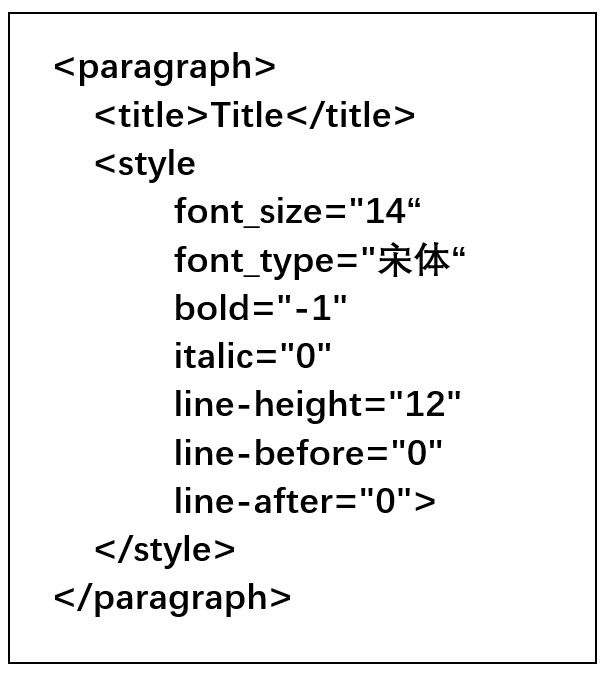


图4.15 正文在XML中的存储形式示意图

4.4.4 文档检测功能设计

文档检测功能为本系统的核心功能之一，具体流程为读取两个XML文件，分别为模板文档XML和待检测文档XML，比较检测后生成检测结果XML，供前端以及文档批注功能调用。

文档检测功能的时序图如图4.16所示。

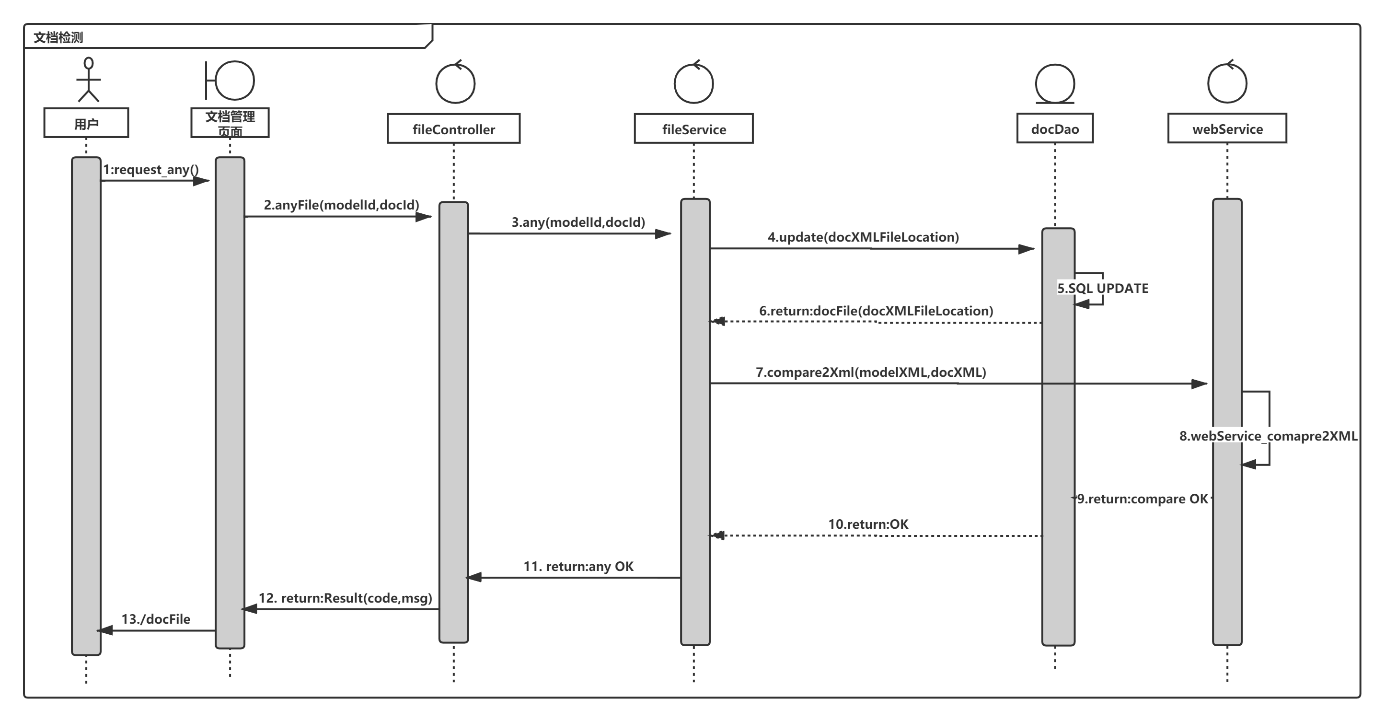


图4.16 文档检测功能的时序图

用户（文档管理员或用户）发送检测请求，fileController接收到请求的模板文档id和检测文档id，将以上两个id发送给fileService，Service首先调用update方法，将分析后的文件地址添加进检测表中，Dao层返回添加成功后的信息，随后，fileService调用WebService方法，对模板文档和检测文档进行分析检测，返回检测结果，包装为Result返回给文档管理页面， 文档管理页面得到Result，将检测结果表格呈现给用户。

其中，WebService端的具体检测流程图如图4.17所示，对于每个段落的检测逻辑流程图如图4.18所示。

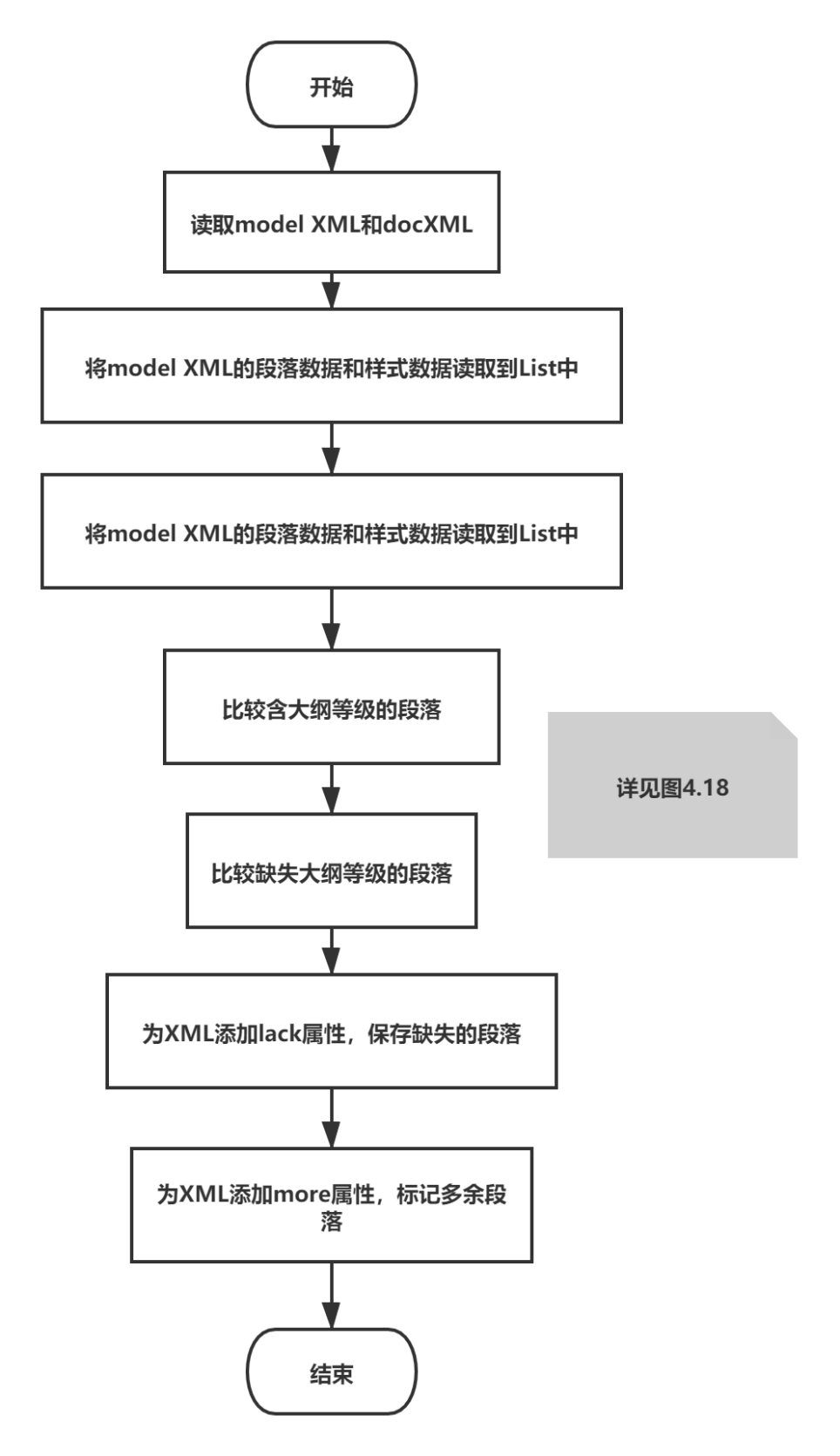


图4.17 文档检测逻辑流程图

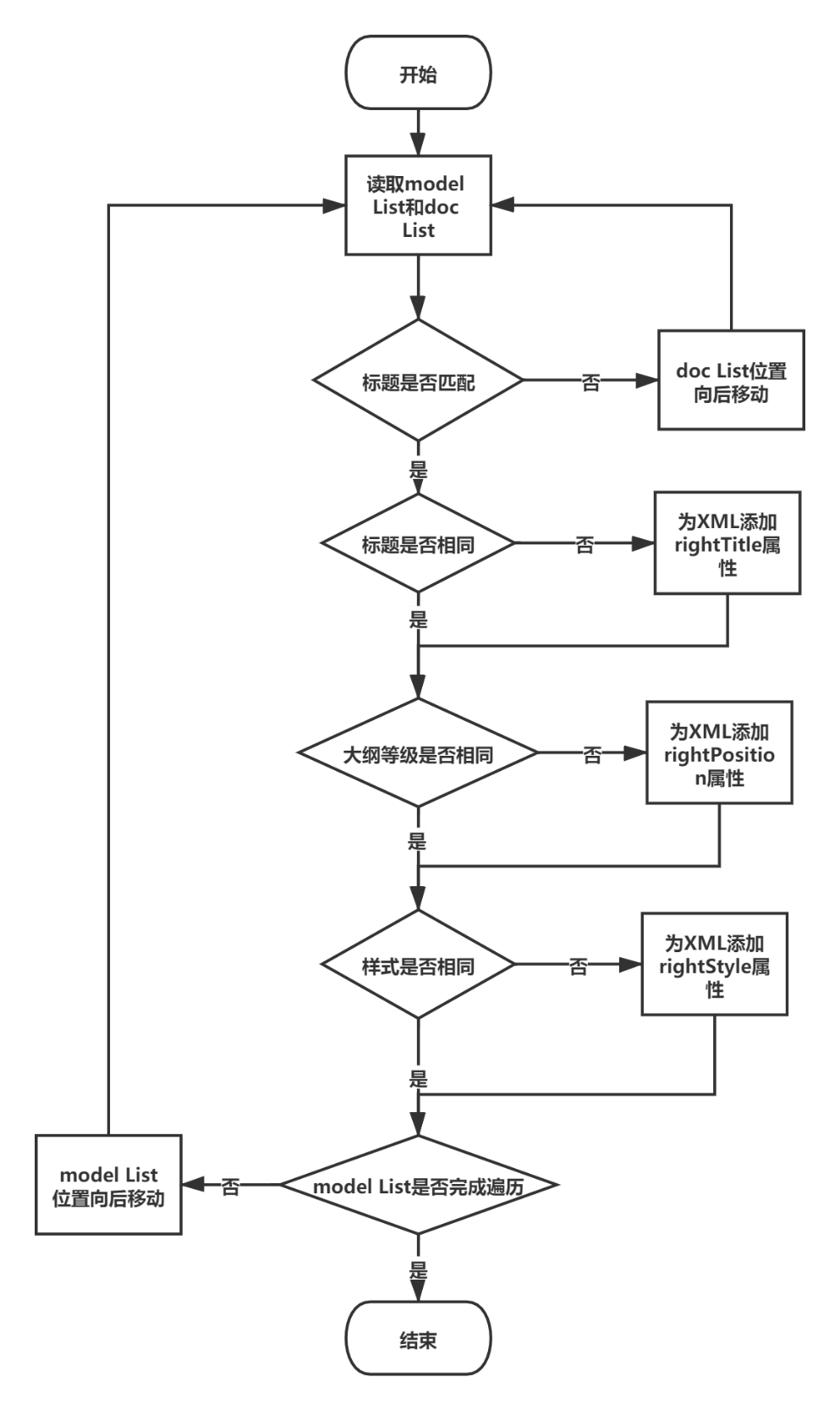


图4.18 段落检测逻辑流程图

对于缺失大纲的段落，本系统通过向量空间模型VSM为每个段落建立了“身份角色”，通过比较余弦相似度为每个缺失大纲的段落确立“身份”[1]。算法流程图如图4.19所示。

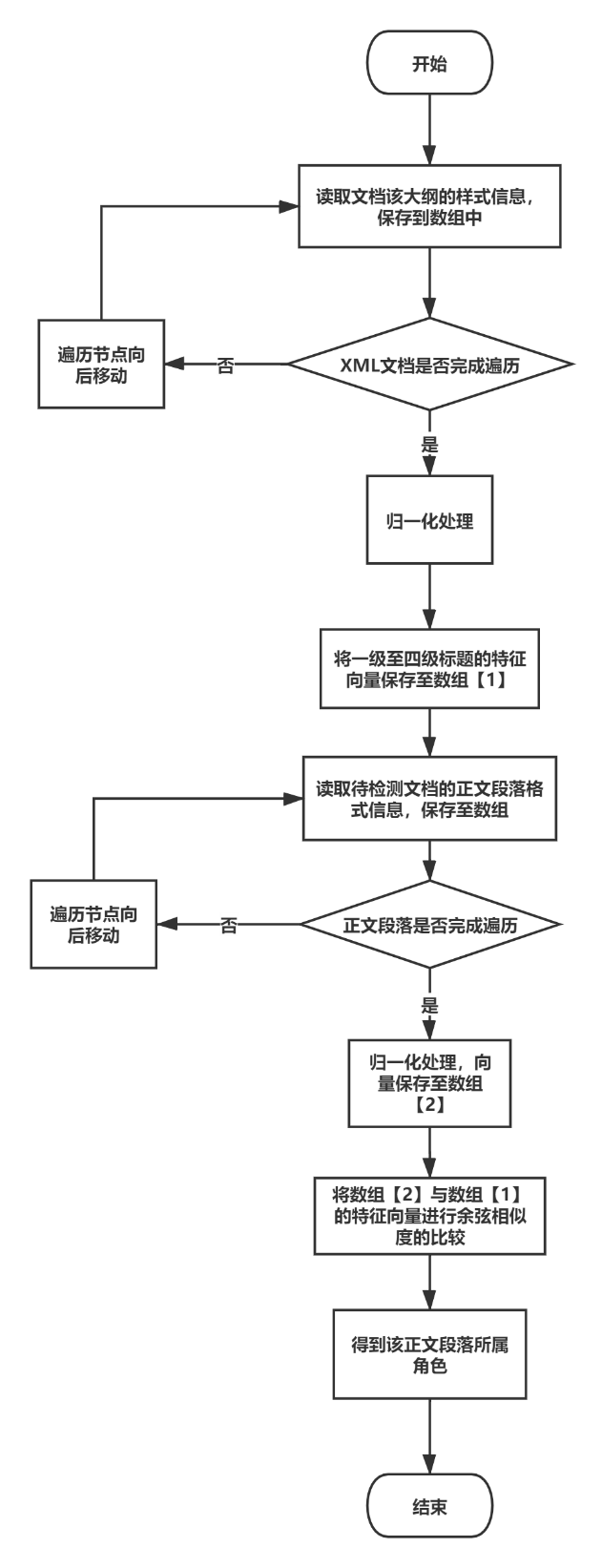


图4.19 VSM算法流程图

具体实现过程描述如下。

1. 在遍历模板文档的同时，将字体、字号、加粗、斜体、行间距、段前间距以及段后间距保存到数组中。
2. 遍历数组，读取每个属性的最大值与最小值，通过公式
3. 对每个属性进行归一化处理，其中代表当前段落的各个属性值，代表最小值，代表最大值。并提取其中的一级标题到四级标题的特征向量。
4. 遍历待检测文档的缺失大纲的段落，即所有<paragraph>节点的信息，通过公式
5. 计算每个段落与角色相似度，从而得到该缺失大纲的段落的应属大纲等级，保存到数组中。
6. 将数组与模板文档的段落进行对比，从而得到缺失大纲等级的段落的检测结果。

最终生成的检测结果XML包含待检测文档的逻辑信息与样式信息，同时每个段落均具有含有比较结果的属性，如“rightPosition”、“diff-bold”等，生成的比较结果如图4.20所示。

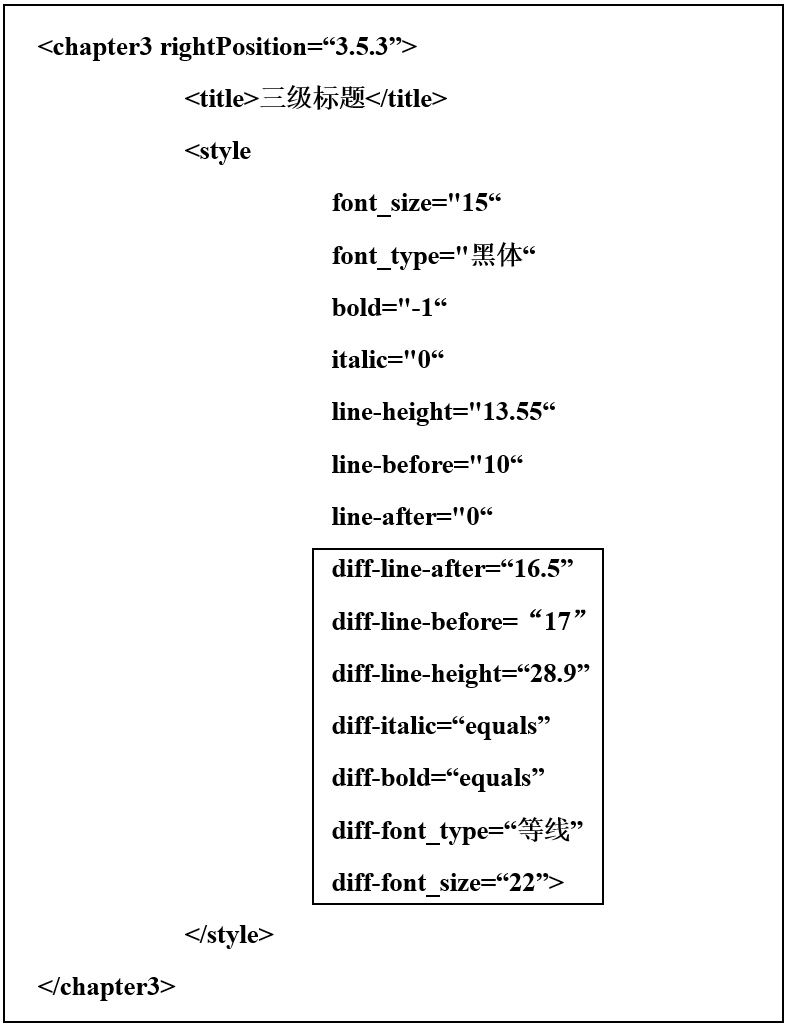


图4.20 检测结果XML文档示意图

## 4.5 接口设计

本系统Web前端与Web后端通过HTTP请求交互，Web后端与Web Service通过Axis/SOAP进行交互，Web前端无法直接访问Web Service。以上交互的接口设计如表4.13和表4.14所示。其中，表4.13为Web前端与Web后端的接口设定，表4.14为Web后端与Web Service的接口设定。

表4.13所有接口均省略了统一接口地址”/api”。返回值均封装为Result进行传输。

表4.13 Web前端与Web后端的接口设定表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 接口 | 方法 | 参数 | 返回值 | 描述 |
| /login | POST | User | Token, roleId | 登录接口 |
| /register | POST | User | String | 注册接口，返回注册信息 |
| /logout | GET | 空 | String | 登出接口 |
| /admin/get | GET | username | User | 返回该用户的基本信息 |
| /admin/user/password | PUT | User | String | 重置密码接口 |
| /admin/user | PUT | User | String | 修改用户信息接口 |
| /admin/deleteUser | POST | Uid | String | 删除用户接口 |
| /wordsyn/file/modelFile/all | GET | 空 | List<ModelFile> | 返回所有模板文档信息 |
| /wordsyn/file/docFile/all | GET | 空 | List<DocList> | 返回所有检测文档信息 |
| /wordsyn/file/uploadFile/modelFile | POST | MutipartFile, uploaderName | String | 上传文档接口 |
| /wordsyn/file/deleteFile | POST | Classification, id | String | 删除文档接口 |
| /wordsyn/file/downloadFile | GET | Path | 空 | 下载文档接口，会打开Http下载流 |
| /wordsyn/file/addComment | GET | docFileId | String | 添加批注接口 |
| /wordsyn/xml/any | GET | Classification, id | String | 分析文档接口 |
| /wordsyn/compare/comapre2Xml | POST | docFileId, modelFileId | Vector | 检测接口，返回检测结果数据 |

表4.14 Web后端与Web Service的接口设定表

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名 | 参数 | 返回值 | 描述 |
| CreateFile | fileName | SOAP\_BOOLEAN | 在服务器创建文件 |
| Append | fileName, buffer | SOAP\_BOOLEAN | 通过buffer上传文件 |
| DownloadFile | strFilePath | SOAP\_BASE64BINARY | 下载服务器文件 |
| Word2Xml\_model | modelFilePath | SOAP\_STRING | 分析模板文档文件 |
| Word2Xml\_doc | docFilePath | SOAP\_STRING | 分析待检测文档文件 |
| Compare2Xml | modelFilePath, docFilePath | SOAP\_VECTOR | 比较检测文档 |
| DeleteFile | strFilePaht | SOAP\_STRING | 删除指定文档 |
| WriteComment | docFilePath | SOAP\_STRING | 为指定文档添加批注 |

## 4.5 系统结构设计

系统总体基于MVC框架，其中三层结构及其子结构如图4.21所示。

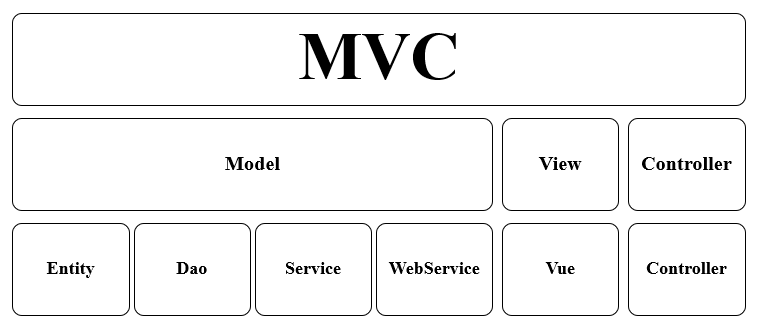


图4.21 系统结构图

系统共分为两大部分，分别为Web端以及C# WebService端。两者涉及到的文件及其功能说明表4.15以及表4.16所示。

表4.15 Web端文件及其功能说明表

|  |  |  |
| --- | --- | --- |
| 层次 | 文件名 | 功能说明 |
| 配置文件 | Pom.xml | 项目依赖管理 |
| Application.properties | 项目配置文件，包含Spring Boot端口、MySQL配置等 |
| 启动类 | WordSynApplication.java | Spring Boot启动类 |
| Controller | LoginController.java | 登录控制器，响应登录页面的请求 |
| UserController.java | 用户信息控制器，响应用户信息操作请求 |
| RoleController.java | 角色信息控制器，响应角色信息操作请求 |
| MenuController.java | 菜单控制器，响应对菜单信息的请求 |
| FileController.java | 文档信息控制器，响应文档信息页面的请求 |
| XmlController.java | XML文件控制器，响应对XML文档的操作的请求 |
| CompareController.java | 比较检测控制器，响应对XML文档比较检测的请求 |
| Service | UserService.java | 用户信息逻辑处理 |
| AdminUserRoleService.java | 用户角色逻辑处理 |
| AdminRoleService.java | 用户角色操作的实现 |
| AdminRolePermissionService.java | 角色权限逻辑处理 |
| AdminRoleMenuService.java | 角色目录的实现 |
| AdminPermissionService.java | 权限逻辑处理 |
| AdminMenuService.java | 菜单逻辑处理 |
| ModelFileService.java | 模板文档相关实现 |
| DocFileService.java | 检测文档相关实现 |
| CompareService.java | 比较文档相关实现 |
| Dao | UserDAO.java | 对用户信息表进行持久化操作的接口类 |
| AdminUserRoleDAO.java | 对用户角色表进行持久化操作的接口类 |
| AdminRoleDAO.java | 对角色表进行持久化操作的接口类 |
| AdminRolePermissionDAO.java | 对角色权限表进行持久化操作的接口类 |
| AdminRoleMenuDAO.java | 对角色菜单表进行持久化操作的接口类 |
| AdminPermissionDAO.java | 对权限表进行持久化操作的接口类 |
| AdminMenuDAO.java | 对菜单表进行持久化操作的接口类 |
| ModelFileDAO.java | 对模板文档表进行持久化操作的接口类 |
| DocFileDAO.java | 对检测文档表进行持久化操作的接口类 |
| CompareDAO.java | 对检测关系表进行持久化操作的接口类 |
| Entity | User.java | 用户信息的实体类 |
| AdminUserRole.java | 用户角色的实体类 |
| AdminRole.java | 角色表的实体类 |
| AdminRolePermission.java | 角色权限的实体类 |
| AdminRoleMenu.java | 角色菜单的实体类 |
| AdminPermission.java | 权限的实体类 |
| AdminMenu.java | 菜单的实体类 |
| ModelFile.java | 模板文档的实体类 |
| DocFile.java | 检测文档的实体类 |
| Compare.java | 检测关系的实体类 |
| Config | MyWebConfigurer.java | 跨域请求配置 |
| ShiroConfiguration.java | Shiro配置文件 |
| Filter | URLPathMatchingFilter.java | 用户权限拦截器 |
| Error | ErrorConfig.java | 错误页面配置文件 |
| Exception | DefaultExceptionHandler.java | 默认异常处理配置文件 |
| realm | MyRealm.java | Shiro域自定义配置文件 |
| Result | Result.java | 前端返回数据格式 |
| ResultCode.java | 返回值枚举类 |
| ResultFactory.java | 返回数据工厂类 |
| VueResult | DocList.java | 检测文件信息返回数据 |
| LoginResult.java | 登录信息返回数据 |
| util | MyDate.java | 自定义日期类 |
| StringUtils.java | 字符串处理工具类 |
| XmlReader.java | Xml文档转换类 |
| WebService | WebService.java | C# WebService的操作类 |

表4.16 C# WebService端文件及其功能说明

|  |  |  |
| --- | --- | --- |
| 层次 | 文件名 | 功能说明 |
| WordSyn | Program.cs | 文档操作入口 |
| MyWord.cs | 对word文档的操作类，包括提取信息和添加批注等 |
| XmlCreater.cs | 对XML文件的操作类，包括创建XML文件以及添加节点、元素或属性等 |
| Compare.cs | 文档检测类 |
| NodeAndData.cs | XML文档数据类 |
| Attribute.cs | 段落样式属性类 |
| WebService | WebService1.asmx | WebService对外API接口 |
| Web.config | WebService配置文件 |

# 特定业务文档规范性检测系统的实现

本系统的技术组成有：C#、Java、Vue.js，数据管理系统采用MySQL。系统采用了Spring Boot以及MVC设计框架，通过前端界面Vue向Spring Boot发送请求，调用相关逻辑处理方法，按需对数据库进行相关操作，同时调用C#提供的WebService方法，对Word文档进行相关处理，最后返回数据给前端解析，展示给用户。

本系统前端使用了Element-UI设计组件，具体使用了其中按钮、表单、标签以及上传组件等，前端与后端通过Http请求交互。

后端使用Spring Boot及Spring MVC，通过DispatcherServlet分析Http请求，并使用HandlerMapping等寻找定位控制器，调用Controller及Service等处理相应的业务逻辑，将结果封装为Result返回给前端。

同时，后端还包含WebService，WebService与Spring Boot通过SOAP交互，通过传递相应的参数调用WebService接口，WebService部分功能会生成XML文件，并提供对文件的下载、读取以及删除功能。

## 5.1 C# WebService端程序的实现

本系统C# WebService端的代码结构如图5.1所示。WebService端分为两个部分，分别为ASMX文件和CS文件。ASMX文件提供HTTP或SOAP接口，供外界调用；CS文件为文档检测功能的核心代码文件，该部分的文件功能具体描述如下。

* MyWord.cs用于调用Microsoft官方API，对Word文档进行操作，负责逻辑信息与样式信息的提取，XML文件的生成，以及对文档添加批注等功能。
* XmlCreater.cs是XML文档的工具类，包装了部分常用的XML文档操作方法，如创建元素，添加属性等。
* Compare.cs是文档检测的核心，主要功能为读取两个XML文件进行比较检测，包括逻辑信息与样式信息的比较，生成检测结果XML，将检测结果以属性的方式添加进XML文件。
* NodeAndData.cs和attribute.cs是XML元素和样式信息的存储类。

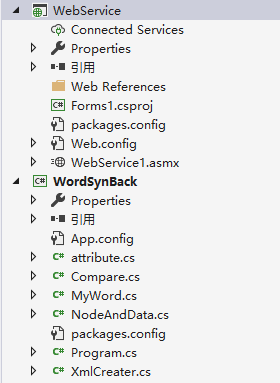


图5.1 C# WebService端的代码结构图

C# WebService端的运行截图如图5.2所示。



图5.2 C# WebService端的运行截图

5.1.1 文档结构信息与样式信息抽取功能的实现

文档结构信息与样式信息抽取功能的接口调用界面如图5.3所示。

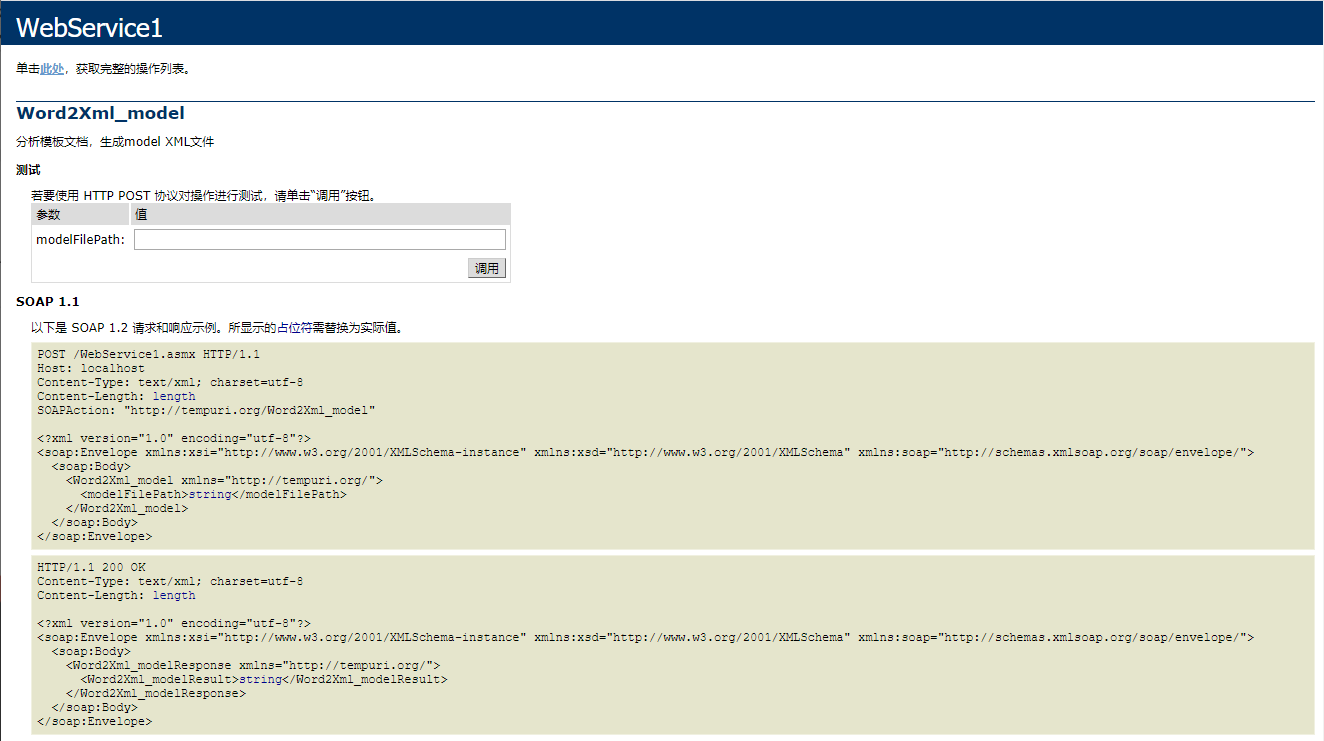


图5.3 文档结构信息与样式信息抽取功能的接口调用界面

WebService：接收一个含有Word文档地址的字符串参数，调用Microsoft.Office.Interop.Word.Application.Documents.Open方法，打开该Word文档，同时为该Word文档创建一个保存段落信息的XML文件。根据Microsoft.Office.Interop.Word的MSDN（Microsoft Developer Network，微软开发者网络），文档的段落可以通过遍历document.Paragraphs中的Paragraph获得，其中，每个段落的内容、大纲等级以及样式信息等可通过相应的方法或属性获得。

Microsoft.Office.Interop.Word提供了多种API，具体如表5.1所示。其中p代表Paragraph。

表5.1 MSDN提供的方法或属性及描述表

|  |  |
| --- | --- |
| 方法/属性 | 描述 |
| p.OutlineLevel | 段落的大纲等级 |
| p.Range.Text | 段落的内容 |
| p.Range.Font.Name | 段落的字体名称 |
| p.Range.Font.Size | 段落的字体字号 |
| p.Range.Font.Bold | 段落是否加粗，-1为加粗，0为未加粗 |
| p.Range.Font.Italic | 段落是否斜体，-1为斜体，0为未斜体 |
| p.SpaceBefore | 段前间距 |
| p.SpaceAfter | 段后间距 |
| p.LineSpacing | 行间距 |

其中，p.OutlineLevel的值有多种情况，具体如表5.2所示。

表5.2 p.OutlineLevel的值及其描述表

|  |  |
| --- | --- |
| p.OutlineLevel的值 | 描述 |
| wdOutlineLevelBodyText | 正文 |
| wdOutlineLevelX（X为1，2，3，4……） | 该段落所处大纲等级 |

经过多次实验处理后，发现OutlineLevel无法直接提供类似1.3.1的大纲等级信息，而是类似{1,2,2,3,2,1,2,……}这种数组，以上表示的逻辑结构如图5.4所示，为了方便储存其逻辑信息，希望将其处理为以下数据结构。



图5.4 示例数组所对应的逻辑结构

本系统使用以下步骤进行数据的处理：

1. 定义大纲等级变量level1,level2,level3,level4，均置为0，本系统支持检测到第四大纲等级。
2. 顺序遍历整篇文档，判断是否为正文，若为是，则跳转至下一个段落；若为否，则进入下一步。
3. 通过字符串剪切其“wdOutlineLevel”部分，仅保留大纲等级，记录为X，通过int.TryParse函数，将其转换为整型。
4. 判断X的值若为1则level1自增1，其他置为0；若为2则level2自增1，level3和level4置为0；若为3则level3自增1，level4置为0；若为4则level4自增1。
5. 最终，由level1至level4拼接而成的字符串即为所求数据。

以上处理流程的部分核心C#代码如图5.5所示。

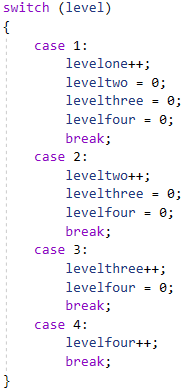


图5.5 大纲等级数据处理的C#代码示意图

循环遍历整个文档的段落，从而生成文档的逻辑信息与样式信息，保存为XML文件，供文档检测功能调用该XML文件进行比较检测。

5.1.2 文档检测功能的实现

文档检测功能的接口调用界面如图5.6所示。

WebService：接收两个字符串参数，分别为模板文档的XML文件地址与待检测文档的XML文件地址，随后进入文档信息处理流程。

首先使用C#的工具System.Xml打开并读取两个XML文档，分别对应模板文档XML和待检测文档XML，使用XML.SelectSingleNode(String str)方法，读取Root元素，即<doc>。

第二步，分别调用getListFromRoot\_model和getListFromRoot\_doc两个方法，按照树的先根遍历的方法，将XML文档中各个段落的信息保存为数组，数组的类型为List<NodeAndData>，其中NodeAndData包含如该段落对应XML节点位置、大纲等级等信息。

第三步，分别调用normalization\_model以及normalization\_doc两个方法，提取模板文档的段落信息，归一化后，为每个大纲建立角色向量。提取待检测文档的每个无大纲等级的段落信息，归一化后求得每个无大纲等级段落的向量

第四步，调用nodeInParagraphWithNoLevel方法，将无大纲等级向量与大纲角色向量进行余弦相似度的比较，以此得到无大纲等级段落应属的大纲等级，添加进数组paragraphWithNoLevel中。

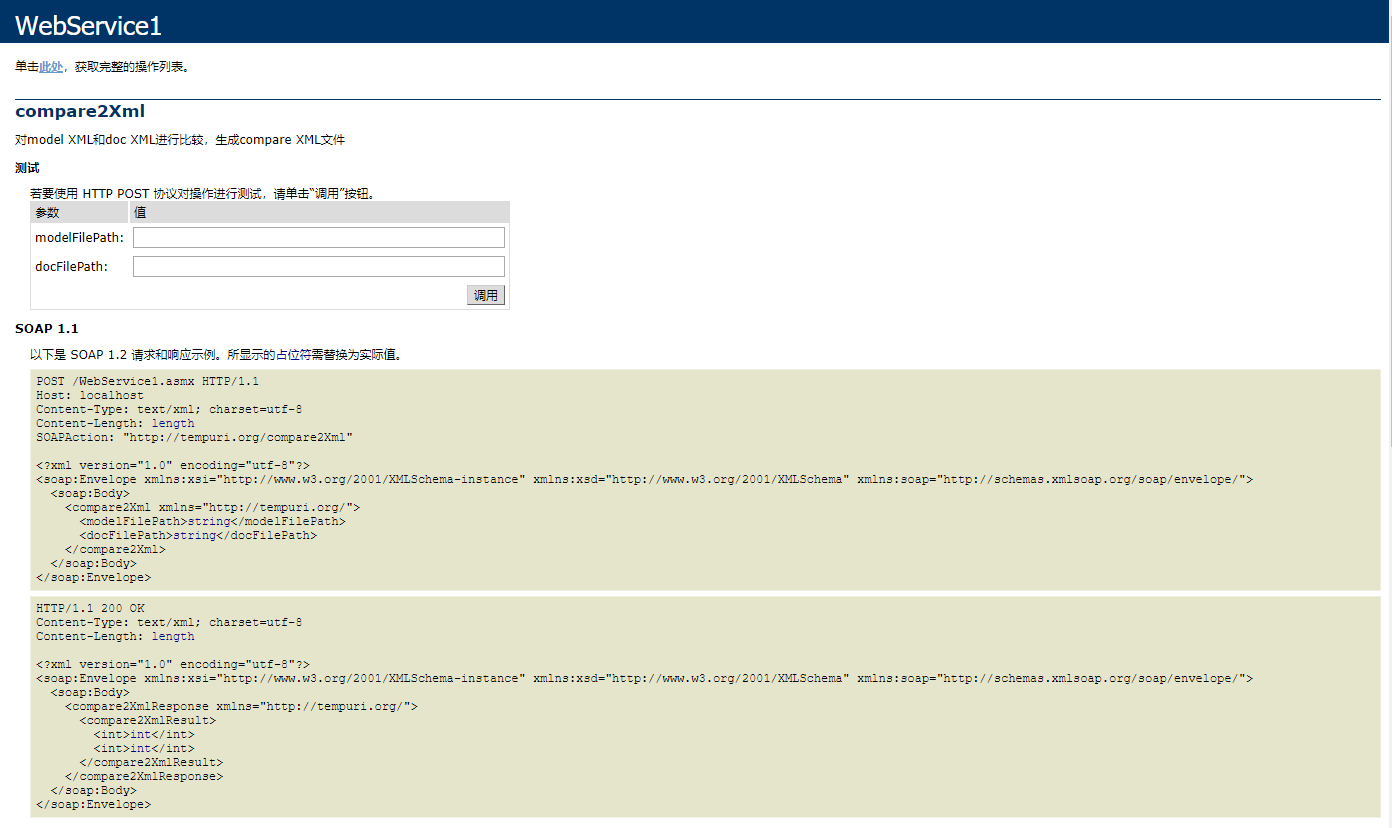


图5.6 文档检测功能的接口

至此，数据已经存储在三个数组中，分别包含模板段落信息、待检测文档段落信息、未标注大纲等级的待检测文档段落信息。数据准备完毕，进入检测流程。

因为List的移除方法Remove的特性，需要从而往前进行遍历。具体的C#遍历代码如图5.7所示。其中，matching函数为核心方法。

Matching方法首先比较标题是否相同，若相同则顺序检测大纲、样式是否相同，若大纲或样式信息不同则添加对应的XML属性，并返回true。如若标题不同，则判断是否标题匹配，即待检测文档标题是否包含模板文档，若匹配则顺序检测大纲、样式信息，与上述相同，返回true。如若标题不匹配，则认为该段落不是对应的待检测段落，则matching返回false。

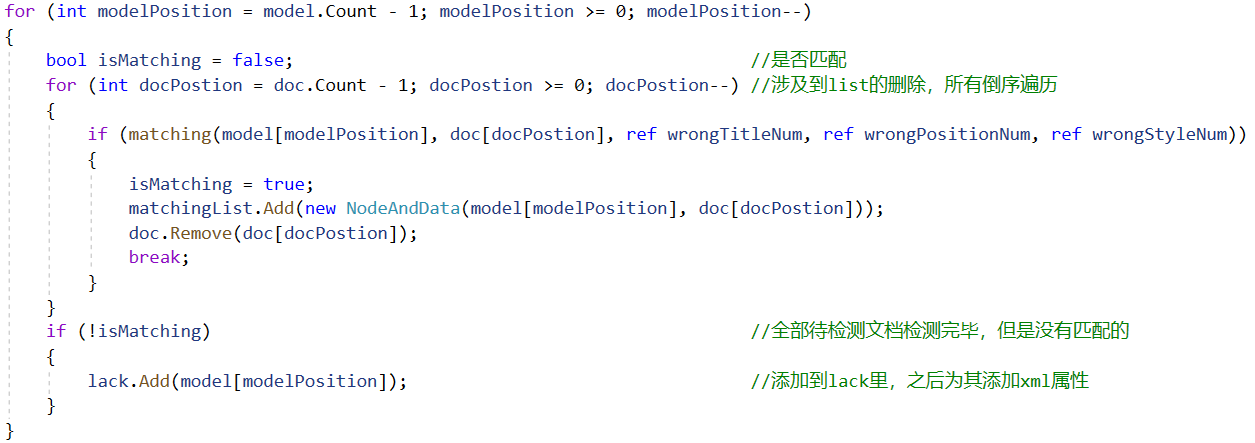


图5.7 文档检测的遍历代码

顺序遍历，直到模板文档遍历完成。接下来进入非大纲段落、多余段落和缺失段落的判定流程。

首先为遍历缺失段落，尝试非大纲段落的匹配，匹配仍使用matching方法，不再赘述。

以上匹配结束后，lack数组保存的便是缺失段落，doc节点列表储存的便是多余段落。分别为其添加缺失段落属性”lackParagraph”和多余段落属性”more”。

将上述结果保存为检测结果XML文件，同时将检测结果的信息，如缺失段落数目，样式错误数目，大纲等级错误数目等，封装为Tuple，返回给Web后端。

5.1.3 文档批注添加功能的实现

文档批注添加功能的接口调用界面如图5.8所示。

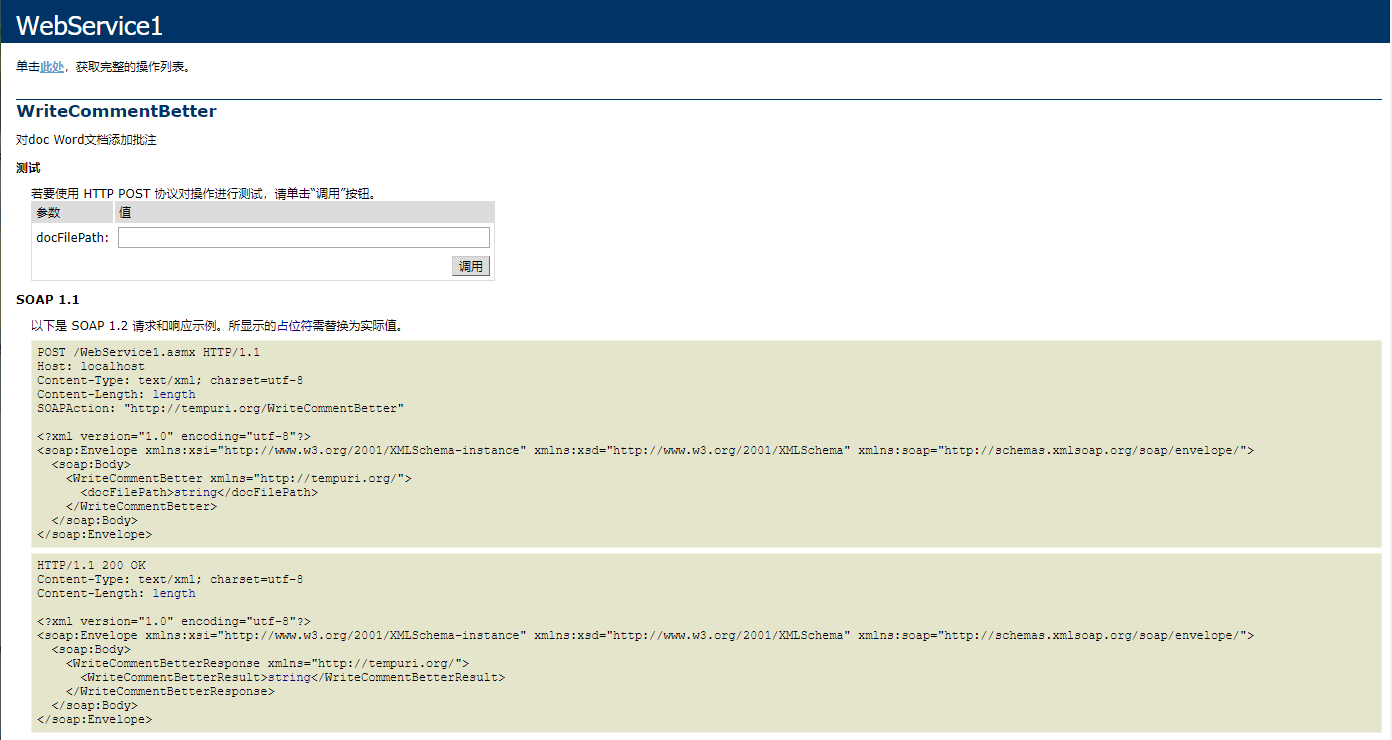


图5.8 文档批注添加功能的接口调用界面

WebService：接收一个包含待添加批注的Word文档的地址的字符串。首先会遍历整个文档，以哈希表的形式存储每个段落的标题及其Range（即在Word文档的位置信息）

随后会遍历检测结果XML文档，查找指定的属性为Word文档添加批注，属性由“文档检测功能”添加，具体属性及其含义如表5.3所示。

最后会根据属性为不规范的段落添加批注，另存为Word文档供用户下载。

表5.3 检测结果XML文档中属性及其含义表

|  |  |
| --- | --- |
| XML属性 | 含义 |
| More | 该段落为多余段落 |
| LackParagraph | 该大纲等级下缺失标题，具体缺失段落为lackParagraph的值 |
| RightPosition | 该段落的大纲等级位置不正确，正确位置为rightPosition的值 |
| Style/diff-font\_size | 该段落的字体大小不正确，正确大小为diff-font\_size的值 |
| Style/diff-font\_type | 该段落的字体类型不正确，正确类型为diff-font\_type的值 |
| Style/diff-bold | 该段落的加粗设置不正确，正确设置为diff-bold的值 |
| Style/diff-italic | 该段落的斜体设置不正确，正确设置为diff-italic的值 |
| Style/diff-line-height | 该段落的行间距不正确，正确的值为diff-line-height的值 |
| Style/diff-line-before | 该段落的段前间距不正确，正确的值为diff-line-before的值 |
| Style/diff-line-after | 该段落的段后间距不正确，正确的值为diff-line-after的值 |

5.1.4 文档操作功能的实现

文档操作功能，包含创建、上传、下载以及验证四个功能，用于操作与WebService相关的Word文档以及XML文档。以上四个接口如图5.9所示。



图5.9 文档操作功能接口的视图

## 5.2 Web系统程序的实现

5.2.1 Web系统程序结构

Web系统为用户提供各种功能及其交互，并调用Webservice完成相关的文档处理功能，分为前端与后端，前端使用Vue.js框架，后端使用Spring Boot框架。前后端的代码结构分别如图5.10和图5.11所示。

本系统Web系统包含较多与文档相关功能，且文档管理员与用户功能较为相似，故本节仅详细描述关键功能的实现。

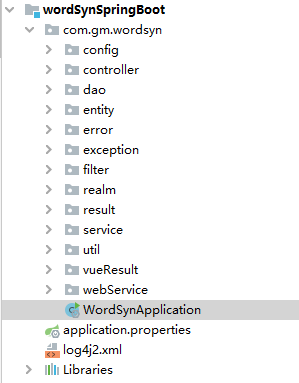


图5.10 后端代码结构图

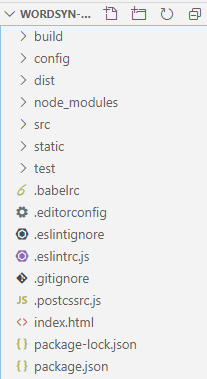


图5.11 前端代码结构图

5.2.2 用户管理功能的实现

用户管理功能的前端界面、用户信息修改界面如图5.12和5.13所示。

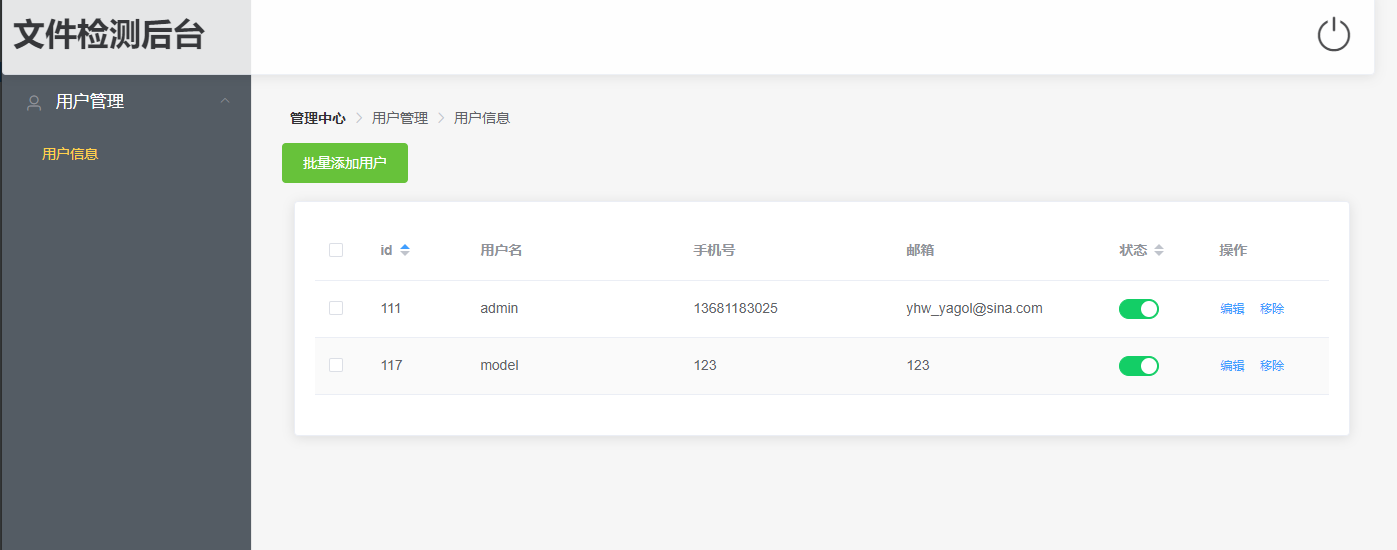


图5.12 用户管理的前端界面



图5.13 用户信息修改的界面

* View层UserProfile.vue：在页面加载时，调用listUser方法，向后端发送Get请求，根据返回值加载所有用户信息，通过表格显示用户信息。
  + 点击“编辑”按钮，显示弹出层，修改完用户信息后点击“确定”按钮，向后端发送Put请求，根据返回值显示修改成功或错误信息。
  + 点击“移除”按钮，弹出确认框，点击“确认”按钮，向后端发送Post请求，参数为用户id，根据返回值显示是否移除成功或错误信息，并重新刷新用户信息列表。
  + 修改用户状态，像后端发送Put请求，根据返回值显示禁用用户信息。
* Controller层UserController：根据Api的不同以及传输方法不同，调用不同的方法，具体调用如表5.4所示。

表5.4 用户管理功能Controller层的描述

|  |  |  |
| --- | --- | --- |
| API URL | Method | Controller方法描述 |
| /api/admin/user | GET | 调用UserService层的list方法，将所有用户信息包装为Result返回给前端 |
| /api/admin/user/status | PUT | 调用UserService层的updateUserStatus方法，修改用户的状态，将修改状态信息返回给前端 |
| /api/admin/user/password | PUT | 调用UserService层的resetPassword方法，重置用户密码，将重置信息返回给前端 |
| /api/admin/user | PUT | 调用UserService层的editUser方法，修改用户信息，将修改后是否成功的信息返回给前端 |
| /api/admin/deleteUser | POST | 调用UserService层的deleteDataByUserId方法，删除用户信息，将删除是否成功的信息返回给前端 |

* Service层UserService：调用Dao层的相关接口，对用户信息进行操作，具体操作如表5.5所示。

表5.5 用户管理功能Service层的描述

|  |  |  |
| --- | --- | --- |
| 方法名 | 参数 | UserSerivce方法描述 |
| list | 空 | 调用Dao层的list方法，得到所有用户信息 |
| isExist | String username | 调用Dao层的findByUsername方法，检索是否该用户名已存在，返回布尔信息表示是否存在。 |
| register | User user | 检查用户名密码是否为空，检查用户是否存在，调用Dao层的Save方法添加用户信息，并为该用户添加角色信息，默认为“用户” |
| updateUserStauts | User user | 调用Dao层的setEnabled方法，改变用户状态。 |
| resetPassword | User user | 生成“盐”，重置密码为“123”，调用Dao层的Save方法修改用户密码。 |
| editUser | User user | 调用Dao层的Save方法，修改用户信息 |
| deleteDataByUserId | Int id | 调用Dao层的deleteDataByUserId方法，根据id删除用户信息 |

* Dao层UserDAO：提供对用户信息表的增删改查。

5.2.3 个人信息修改功能的实现

用户登录成功后，可在系统内修改个人信息，例如联系方式或电子邮箱。个人信息修改功能的前端如图5.14和图5.15所示。

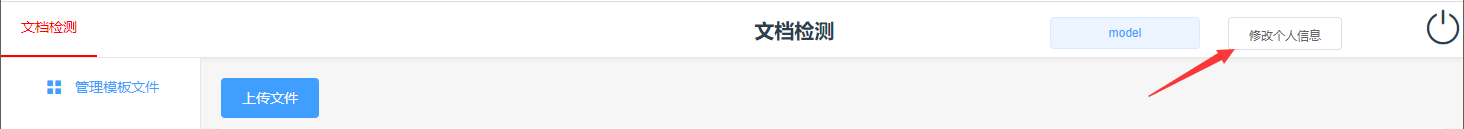


图5.14 个人信息修改按钮的视图



图5.15 个人信息修改界面的视图

* View层UserProfile.vue：点击“修改个人信息”按钮，将修改个人信息的窗口设为可见。该窗口为一个form表单，其中，电话号码和电子邮箱可进行修改。在点击“修改”按钮时，首先会对电话号码和电子邮箱进行正则表达式检查，检测是否符合规范。具体的正则表达式如表5.6所示。若匹配成功，则向后端发送Put请求，参数为用户名、手机号以及电子邮箱，根据返回值返回修改信息，并刷新用户信息；若匹配不成功，则弹出提示框提示错误信息。

表5.6 正则表达式及其解释表

|  |  |  |
| --- | --- | --- |
| 正则表达式 | 解释 | 示例 |
| /^1[3456789]\d{9}$/ | 首位为1，第二位为[3456789]中的一位，其余为9位数字。共计11位，是手机号码的正则表达式 | 13681183698 |
| /^[A-Za-z0-9]+@[a-zA-Z0-9]+(\.[a-zA-Z]+)+$/ | 在’@’的左侧，可为大小写字母或数字，加号表示可重复出现，在’@’右侧，首先是出现任意位数大小写字母和数字，然后为一个’.’，随后是任意字符的大小写字母 | Yhw\_yagol@sina.com |

* Controller层UserController：接收Put请求，接收前端传来的参数用户名、手机号以及电子邮箱，调用userService中的editUser方法，对用户信息进行修改，根据方法返回值，将修改是否成功的信息包装为Result，返回个前端。
* Serivce层UserService：首先调用Dao层的findByUsername方法，将用户信息保存到User类中，之后通过User.setPhone以及User.setEmail方法，修改User中的手机号和电子邮箱。调用Dao层的Save方法，修改个人信息。
* Dao层UserDAO：提供Save方法，修改个人信息。

5.2.4 查看上传者信息功能的实现

用户可查看上传文档者的相关信息，以此查看联系方式。查看上传者信息功能的前端视图如图5.16以及5.17所示。



图5.16 查看文档信息



图5.17 查看上传者详细信息

* View层modelFile.vue/docFile.vue：在任意列表中（模板文档、检测文档） ，轻碰文件信息标签，会弹出该文档的信息，如文件名称以及上传者。点击上传者的名字，首先向后端发送Get请求，参数为上传者的名字，根据返回值，提取电话号码和电子邮箱，弹出窗口，显示具体的上传者的信息。
* Controller层UserController：接收Get请求，参数为用户名username， 调用UserService的findByUsername方法，将用户信息包装为Result返回给前端。
* Serivce层UserService：调用UserDao层的findByUsername方法得到指定用户的信息。
* Dao层UserDAO：JpaRepository提供按照方法名的匹配查找方法，因此不用提供具体的SQL语句，只需按照格式编写方法名即可，此处使用findByUsername进行指定用户名的查找。

5.2.5 模板文档上传功能的实现

模板文档上传功能的前端界面如图5.18和5.19所示。



图5.18 模板上传按钮的视图



图5.19 模板上传窗口的视图

* View层modelFile.vue：点击“上传文档”按钮，将上传模板文件的窗口设为可见。前端上传文件组件使用element-UI的el-upload，点击“上传模板文件”按钮后，根据提示上传文件，检查文件类型是否为Word文档，类型不匹配则弹出提示框提示，匹配则向后端发送Post请求，参数为该文件以及上传人姓名。
* Controller层FileController：接收Post请求，参数类型为MultipartFile，调用WebService层的uploadFile方法，将文件保存至服务器。调用Service层将文件信息保存至数据库。
* Serivce层ModelFileService：调用ModelFileDAO层的Save方法添加文件信息。
* Dao层ModelFileDAO：提供SQL insert方法，将文件信息添加至数据库。
* WebService层：使用Axis调用WebService提供的方法，调用createFile，Append两个方法将文件上传至服务器。

5.2.6 模板文档分析功能的实现

模板文档分析功能的前端界面如图5.20所示。

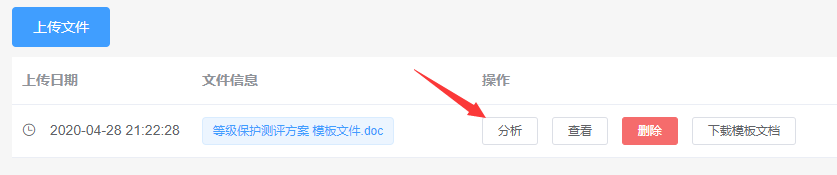


图5.20 模板文档分析功能的前端界面

* View层modelFile.vue：向后端发送请求，参数为classification和文档id，其中classification表明该文档为模板文档，分析成功后弹出提示信息。
* Conteoller层XmlController：接收前端请求，调用ModelFileService层anyModelFileById方法，将分析信息返回给前端。
* Service层ModelFileService：调用WebService层的anyModelFile，分析Word文档，并调用Dao层的update方法，将分析信息添加到模板文档信息表中。
* WebService层：使用Axis调用WebService提供的接口，调用Word2Xml\_model分析模板文档。

5.2.7 模板文档查看功能的实现

模板文档的查看功能的前端视图如图5.21和图5.22所示。



图5.21 查看功能按钮的视图



图5.22 模板文档查看的前端视图

* View层wordShow.vue：点击“查看”按钮，带参跳转至/wordShow页面，参数为该文档id，/wordShow页面向后端发送get请求，参数为该文档的id，根据返回值处理XML数据，通过x2js控件将XML解析为JSON数据，通过push()方法处理JSON数据，调用el-tree将数据在前端以树形图的形式呈现。
* Controller层XmlController：接收前端的参数，调用ModelFileService层的modelFileOnXml，得到指定文档的分析后的XML字符串，将字符串包装为Result返回给前端。
* Service层ModelFileService：根据id，调用Dao层的相关方法得到XML文档在服务器中的地址，调用WebService层的方法下载XML文档，下载完的文档是字节数组的形式，经过工具类XmlReader中相关方法的转换，将字节数组转换为String，返回给XmlController。
* Dao层ModelFileDAO：通过SQL Selcet方法，通过模板文档id读取该文档分析完成的XML文档地址。
* WebService层：调用DownloadFile方法，参数为文档地址，将服务器文档下载至后端，得到的文件形式为字节数组。

5.2.8 模板文档删除功能的实现

模板文档删除功能的前端视图如图5.23所示。

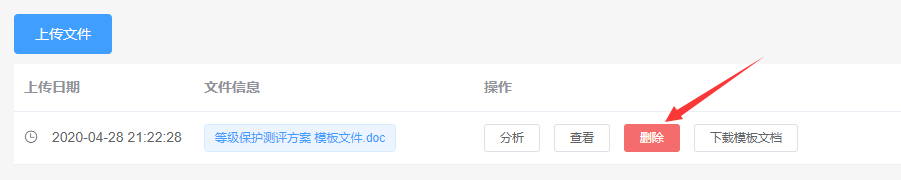


图5.23 模板文档删除功能的前端视图

* View层modelFile.vue：点击“删除”按钮，向后端发送Post请求，参数为classification和该文档id，其中classification标识删除的文件类型为模板文档。根据返回信息，提示删除成功信息或错误信息。
* Controller层FileController：接收前端传来的参数，调用ModelFileService的deleteModelFile方法，根据方法的返回值，将删除信息包装为Result返回给前端。
* Service层ModelFileService：首先根据模板文档的id得到该文档的地址以及分析后生成的XML文档的地址，同时根据该模板文档id，将与该文档相关的检测文档的相关地址也查找出来，调用WebService层的deleteFile方法，在服务器上根据上面四个地址删除文件，之后调用Dao层的delete方法，将模板文档信息表、检测文档信息表和检测表三个表中的相关信息删除，保证数据库的完整性。
* Dao层modelFileDAO：三种删除SQL delete语句，分别根据id删除模板文档信息表、检测文档信息表以及检查表中的信息。
* WebService层：使用Axis，调用DeleteFile接口，根据文件地址删除文件。

5.2.9 模板文档下载功能的实现

模板文档下载功能的前端视图如图5.24所示。

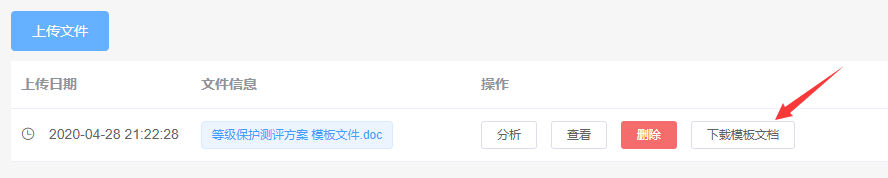


图5.24 模板文档下载功能的前端视图

* View层modelFile.vue：点击“下载模板文档”按钮，动态生成<a>标签，同时设置其”href”为一串URL，参数为Path，即该模板文档的地址，scrip调用click方法，模拟跳转至该地址。
* Controller层FileController：接收前端发送的Get请求，参数为需要下载的文件地址，使用HttpServletRequest，取得输出流，调用Service层方法，将字节数组形式的文件传给前端。
* Service层CompareService：根据文件地址，调用WebService层的downloadFile方法，传回字节数组形式的文件数据。
* WebService层：根据文件地址，调用DownloadFile方法，下载服务器中的文件。

5.2.10 检测文档比较功能的实现

检测文档比较功能的前端视图如图5.25所示。



图5.25 检测文档比较功能的前端试图

* View层docFile.vue：点击“比较”按钮，向后端发送Post请求，参数为该文档的id以及对应的模板文档的id。根据返回值，输出检测结果，如图5.26所示，检测结果中包括标题、位置、样式、缺少大纲等级、缺少段落以及多余段落的个数统计。

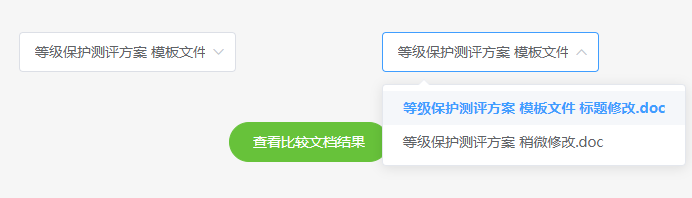


图5.26 检测结果前端视图

* Controller层CompareController：接收前端传来的模板文档id和检测文档id，调用CompareService层的方法，将检测结果包装为Result返回给前端。
* Service层CompareService：根据模板文档id和检测文档id，取得两者的分析后的XML文档的地址，判断地址是否为空，为空则返回错误信息，不为空则调用WebService层的方法对两个XML文档进行分析，将分析结果返回给Controller层。
* Dao层CompareDAO：根据模板文档id和检测文档id，取得两个分析后生成的XML文档的地址。
* WebService层：调用compare2Xml方法，参数模板文档和检测文档的XML文件，生成含有比较结果的XML文档，并将检测结果以数组的形式返回。

5.2.11 树形图查看检测结果功能的实现

树形图查看检测结果功能的前端视图如图5.27和5.28所示。



5.27 选择需要查看结果的检测文档

* View层CompareFile.vue：进入检测页面时，首先加载两个列表，分别为模板文档列表以及经过检测后的检测文档列表，选择完模板文档和检测文档后，点击“查看比较结果”，向后端发送Get请求，请求两个XML文档，分别为模板文档分析后的XML文档和生成的检测文档结果的XML文档，调用x2js将XML文档解析为Json格式，并经过循环处理为el-tree需要的数据形式。同时设置点击相应函数，读取XML文档中rightTitile、rightPosition等信息，将其写入el-input中，并根据值是否为空决定是否可见。
* Controller层XmlController：接收前端传来的Get请求，根据检测文档id调用DocFileService层方法得到含检测结果的XML文档的字符串，封装为Result返回给前端。根据模板文档id调用ModelFileService层，得到经过分析后的模板文档的XML文档的字符串，封装为Result返回给前端。
* Service层DocFileService：根据id得到XML文档所在的地址，调用WebService层的downloadFile方法下载文件为字节数组，使用工具类XmlReader中的方法将字节数组转换为字符串，返回给Controller层。
* Dao层DocFileDAO/CompareFileDAO：使用SQL Select语句，根据id得到XML文档的存储地址。
* WebService层：根据地址下载服务器中的文件。



图5.28 树形图查看检测结果

5.2.12 检测文档添加批注、下载批注文件功能的实现

添加批注、下载批注文件功能的前端视图如图5.29所示。



图5.29 添加批注、下载批注文件功能的前端视图

* View层docFile.vue：点击“添加批注并下载”按钮，首先会检测该文档是否已经添加过批注，即检测批注文档地址是否存在，存在则直接调用下载，新建<a>标签，并将批注文档地址作为参数，自动打开调用下载Api进行下载。若不存在则向后端发送Get请求，将文档id作为参数发送给后端，根据返回值提示添加批注是否成功，刷新列表，同时提示再次点击该按钮开始下载。
* Controller层FileController：
  + 接收前端传来的id，调用DocFileService为文档添加批注，根据返回值返回添加批注是否成功的信息。
  + 使用HttpServletRequest，开启输出流，调用WebService根据地址下载文件，将字节数组写入输出流，完成下载。
* Service层DocFileService：根据检测文档id，在检测表中查询对应的检测结果XML文档地址，调用WebService层的方法对文档添加批注，将批注信息填写进检测表中，返回添加批注是否成功的信息。
* WebService层：调用WriteCommentBetter方法，对检测文档添加批注，返回添加批注是否成功的信息。

# 特定业务文档规范性检测系统的测试

本节主要对系统进行集成测试，设置合适的测试用例对系统进行测试，测试的重点主要在文件检测上。

## 6.1 用户管理功能的测试

用户管理功能包括登录、注册以及后台管理。主要对用户名、密码、手机号、邮箱以及角色等信息的处理逻辑进行测试。

6.1.1 用户登录功能测试

用户登录测试主要针对用户名以及密码的格式以及验证逻辑。用户登录功能测试的具体描述如表6.1所示。

表6.1 用户登录功能测试描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试用例 | 预期结果 | 测试结果 |
| 1 | 用户名密码为空 | 弹出用户名密码不能为空的提示 | Pass |
| 2 | 用户名为admin，密码为空 | 弹出密码不能为空的提示 | Pass |
| 3 | 用户名为空，密码为123 | 弹出用户名不能为空的提示 | Pass |
| 4 | 输入错误的密码，用户名为admin，密码为321 | 弹出密码错误的提示 | Pass |
| 5 | 输入正确的密码，用户名为admin，密码为123 | 验证成功，跳转至后台管理页面 | Pass |
| 6 | 输入正确的密码，用户名为model，密码为123 | 验证成功，以文档管理员身份进入文档检测主页 | Pass |
| 7 | 输入正确的密码吗，用户名为doc，密码为123 | 验证成功，以用户的身份进入文档检测主页 | Pass |

6.1.2 用户注册功能测试

用户注册测试主要针对用户名重复、手机号以及邮箱的规范性等。用户注册功能测试的具体描述如表6.2所示。

表6.2 用户注册功能测试描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试用例 | 预期结果 | 测试结果 |
| 1 | 账号密码为空，其他为123 | 弹出用户名和密码不能为空的提示 | Pass |
| 2 | 账号为test，密码为空，其他为123 | 弹出密码不能为空的提示 | Pass |
| 3 | 账号为空，其他为123 | 弹出账号不能为空的提示 | Pass |
| 4 | 账号为admin，密码为123，其他为321 | 弹出用户已存在的提示 | Pass |
| 5 | 账号为test，密码为123，其他为321 | 弹出注册成功的提示，跳转至登录界面 | Pass |
| 6 | 账号为test2，密码为123，其他为空 | 弹出注册成功的提示，跳转至登录界面 | Pass |
| 7 | 在test用户注册成功的情况下，用户名输入test，密码为123 | 以用户的身份跳转进文档检测主页面 | Pass |
| 8 | 账号为test3，密码为123，邮箱为123@com，手机号码为13681136589 | 弹出“邮箱格式不正确”的提示，注册失败。 | Pass |
| 9 | 账号为test4，密码为123，邮箱为123@sina.com，手机号码为123456 | 弹出“手机不正确”的提示，注册失败 | Pass |

6.1.3 用户后台管理功能测试

用户后台管理功能测试主要针对用户角色的改变，及信息是否能及时更新等。用户后台管理功能测试的具体描述如表6.3所示。

表6.3 用户后台管理功能测试描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试用例 | 预期结果 | 测试结果 |
| 1 | 修改用户test的状态设为禁用 | 使用test登录时，提示该用户已被禁用 | Pass |
| 2 | 修改test的状态为启用 | 使用test登陆时，正常跳转至文档检测页面 | Pass |
| 3 | 修改test的用户信息，将身份修改为文档管理员 | 使用test登陆时，以文档管理员身份进入文档检测页面 | Pass |
| 4 | 修改test2的用户信息，添加手机号为13681235025 | Test2的用户信息手机号显示为13681235025 | Pass |
| 5 | 修改test2的身份为系统管理员 | 使用test2登录，跳转至后台管理界面 | Pass |
| 6 | 移除test2用户 | 使用test2登录时，提示无此用户 | Pass |
| 7 | 重置test的密码 | 使用test登录时，原本密码321失效，使用123登陆时成功进入文档检测页面 | Pass |

## 6.2 文档功能的测试

文档功能检测包括文档管理与文档检测两部分，主要涉及对文档的分析、查看、比较检测以及下载等处理逻辑的检测。

6.2.1 文档管理功能测试

文档管理功能测试主要针对角色的操作权限、能否正常分析与下载等。文档管理功能测试的具体描述如表6.4所示。

表6.4 文档管理功能测试描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试用例 | 预期结果 | 测试结果 |
| 1 | 登录身份为model，可以查看model1用户的模板文件，但不能进行分析和删除操作 | 分析和删除按钮均为灰色，无法点击 | Pass |
| 2 | 登录身份为model，对与model（自己）上传的模板文档，可以分析与删除 | 分析和删除按钮可以点击，且成功响应 | Pass |
| 3 | 登录身份为doc，可以查看模板文档，无法上传，无法分析与删除 | 页面无“上传文件”按钮，分析和删除按钮为灰色，无法点击。查看按钮可点击，树形图显示正确 | Pass |
| 4 | 登录身份为doc，可以查看doc1用户的检测文档，但无法分析或删除 | 分析或删除按钮为灰色，无法点击。查看按钮可点击，树形图显示正确 | Pass |
| 5 | 登录身份为doc，可以对doc（自己）上传的文档进行分析与删除操作 | 分析与删除按钮可以点击，且响应正确 | Pass |
| 6 | 登录身份model，可以下载模板文档和检测文档的所有文件 | 所有文件的下载按钮均可以点击，且可以下载 | Pass |
| 7 | 登录身份为doc，可以下载模板文档和检测文档的所有文件 | 所有文件的下载按钮均可以点击，且可以下载 | Pass |

6.2.2 文档检测功能测试

文档检测功能作为本系统的核心，测试用例的选择与测试的实施需要更加细致，具体文档检测功能测试的描述如表6.5所示。表中所述的“模板文档1”的大纲视图如图6.1所示，“检测文档1”和“检测文档2”的大纲视图如图6.2和6.3所示，以上三个文档的描述如表6.6所示。

表6.5 文档检测功能测试描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试用例 | 预期结果 | 测试结果 |
| 1 | 对检测文档1进行检测，使用树形图查看检测结果 | “参考文献”、“数据流图”以及“附录”显示位置错误，“外文翻译”显示为多余段落 | Pass |
| 2 | 对检测文档1进行添加批注操作，下载并查看批注文档 | “参考文献”、“数据流图”以及“附录”显示位置错误，并显示正确段落位置，“外文翻译”显示为多余段落，“开发环境与技术”及其下属二级标题的字号错误，并显示正确字号 | Pass |
| 3 | 对检测文档2进行检测，使用树形图查看检测结果 | “IDEA”和“Visual Studio”显示缺少大纲等级，并显示正确的标题位置 | Pass |
| 4 | 对检测文档2进行添加批注操作，下载并查看批注文档 | “IDEA”和“Visual Studio”显示为缺少大纲等级，并显示正确的大纲等级，同时显示样式信息的差异。 | Pass |

表6.6 测试文档描述表

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 测试文档id | 测试文档描述 | 大纲视图编号 |
| 1 | 模板文档1 | 模板文档，包含11个一级标题，12个二级标题，5个三级标题以及2个四级标题 | 图6.1 |
| 2 | 检测文档1 | 检测文档，对应模板文档为“模板文档1”。其中，“参考文献”位置错误，“开发环境与技术”及其下属二级标题的字号错误，“数据流图”、“附录”位置错误，“外文翻译”为多余段落。 | 图6.2 |
| 3 | 检测文档2 | 检测文档，“开发环境与技术”的两个二级标题取消了大纲等级，即“IDEA”和“Visual Studio”为正文等级。 | 图6.3 |

图6.1为模板文档1的大纲视图，包含多个不同等级的标题。

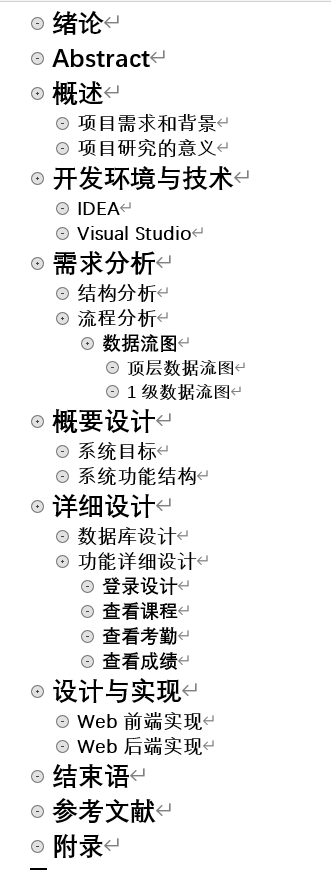


图6.1 模板文档1的大纲视图

图6.2为检测文档1的大纲视图，相较于模板文档1，“参考文献”、“开发环境与技术”等存在逻辑或样式错误。

图6.3为检测文档2的大纲视图，相对与模板文档1，部分标题没有设置为大纲等级。

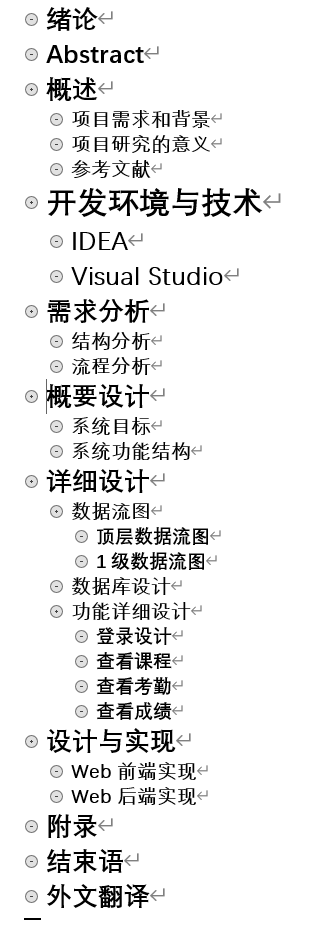


图6.2 检测文档1的大纲视图

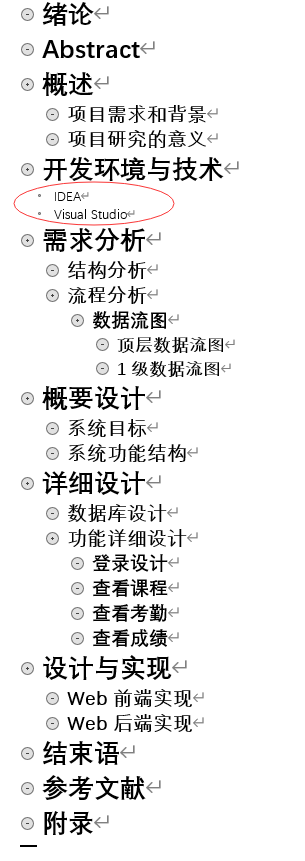


图6.3 检测文档2的大纲视图

结束语

本系统的开发总共分为两个部分，分别是文档检测程序、Web端用户管理与文档管理程序。

Web端使用的Spring Boot是我较为熟悉的框架，Web MVC设计模式是目前开发中经常使用的开发模式，通过开发本系统使我对其的理解更深一步。

在本系统中，我尝试使用另一种技术作为用户管理的框架——Shiro，Shiro相对于其他安全框架，更加容易上手，通过配置三个组件，即可使用Shiro作为系统的用户管理工具。虽说如此，但三个配置文件选项较多，且需要对数据库具有较高的理解。在实现过程中，我参考了较多文档，最终实现了登录、注册以及后台的基于用户身份的安全管理，并通过在数据库中存储前端信息，实现了前端菜单的动态加载，提升了安全性。但是，文件管理部分仍使用简单的加载判定，使部分使用者可以通过某些手段（如修改Store的信息），从而绕过拦截进入非法页面，这是本系统未来需要完善的部分。

在文档管理程序部分，在设计本系统前，C#是没有了解过的编程语言，在学习实践过程中，我发现其与Java等面向对象变成语言相似，基于对其他编程语言的了解，我较快上手了C#，从而完成了本次系统设计。系统设计的一个难点在于如何设计好WebService，从而使别的系统（如Spring）能够轻松调用。C#支持创建发布WebService，支持使用SOAP或Http等多种调用方式，同时使用Visual Studio可以轻松地将文档检测程序与WebService进行链接，提升了开发效率。

随着技术的发展，我们应该尝试使用自己未曾使用过的新技术。可能刚开始有很多困难，但当成果做出来时，将同时获得项目完成以及自身进步的双重收获。在学习新技术时，先尝试使用，做出成果，再深入了解其内部，使理解加深，我认为这才是真正的研究学习，从而使我们在科研的道路前进。

致谢

在本次毕业设计中，我要感谢所有帮助过我的同学和老师，他们无一都对我以及本系统的完成提供了大量的帮助，同时也激励我完成更加优秀的毕设成功，使我在各个方面取得了进步。

首先，我要感谢我的毕设指导老师，侯霞老师。在本系统的整个开发过程中，侯老师从毕设的选题、解析题目到中期的设计优化，结题时的论文辅导，都提出了非常宝贵的意见。在侯老师的指导帮助下，本系统进行过多次完善与优化，尝试了很多新技术，从而使得我的技术水平提高，并且最终的毕设成果达到了预期效果。本系统与毕设的完成，要感谢侯老师对我以及本系统开发工作的支持与付出。

其次，我还要感谢所有帮助过我，给予过我建议的同学。在本次毕设的设计过程中，我曾向他们咨询过业务方面的问题以及我的疑惑，这些同学均提出了宝贵的意见，为我在设计中提供了参考，在开发实现过程中遇到技术难题，愿意与我一同探索解决方法。在此十分感谢他们。

参考文献

1. 赵雪, 侯霞. 一种流式文档逻辑结构重建方法[J]. 北京信息科技大学学报（自然科学版）, 2017(5).
2. 王帅群, 夏斌, 孔薇. 基于.NET的论文格式自动检查系统[C]// 全国计算机技术与应用. 2010.
3. 侯伟波. 软件项目文档格式审查系统的设计与实现[D]. 西安电子科技大学,2015.
4. 彭欣, 李宁. 面向段落逻辑标签判断的VSM改进算法[J]. 北京信息科技大学学报：自然科学版, 2014(29):24.
5. 赵林, 李宁, 彭欣. 基于有向图的流式文档逻辑结构重构方法[J]. 计算机工程与设计, 2016, 37(5):1239-1244.
6. 陈国胜, 何宗明. 基于XML技术的Word文档录入及格式检测系统设计[J]. 计算机时代, 2009(4):38-40.
7. 朴勇. 基于XML的文本结构信息抽取与聚类研究[D]. 大连理工大学, 2014.
8. 张立科. C#开发经验技术宝典[M]. 北京：人民邮电出版社.2007.11
9. Salton G . A Vector space model for automatic indexing[J]. Communications of the ACM, 1975, 18(11):613-620.
10. Andreas Dengel. Document Structure Analysis Based on Layout and Textual Features[C]// Proceedings DAS2000, Int'l Workshop on Document Analysis Systems, 2000.
11. Langer H , Lüngen, Harald, Bayerl P S . Text type structure and logical document structure[C]// Acl Workshop on Discourse Annotation. Association for Computational Linguistics, 2004.