**北京邮电大学计算机学院（国家示范性软件学院）**

**多源灾情数据管理服务系统**

**系统测试设计与测试报告(MSHD-TEST-T5)**

**课程名称： 软件工程理论**

**指导教师：**

**日 期： 2023年 12月 19日**

目录

[1.导言 4](#_Toc161584429)

[1.1项目背景介绍 4](#_Toc161584430)

[1.2项目范围 4](#_Toc161584431)

[1.4测试项目需求 5](#_Toc161584432)

[1.1.1.功能性需求 5](#_Toc161584433)

[1.1.2.非功能性需求 6](#_Toc161584434)

[2.测试项目介绍 9](#_Toc161584435)

[2.1测试环境 9](#_Toc161584436)

[2.1.1 网络拓扑结构 9](#_Toc161584437)

[2.2.2服务器端环境 10](#_Toc161584438)

[2.2.3客户端环境 11](#_Toc161584439)

[2.3测试模型 11](#_Toc161584440)

[3.单元测试 11](#_Toc161584441)

[3.1地震单元测试 12](#_Toc161584442)

[4.接口测试 13](#_Toc161584443)

[4.1 查询所有震情 - `queryAll` 13](#_Toc161584444)

[4.2 添加震情 - `addEarthquake` 13](#_Toc161584445)

[4.3 更新震情信息 - `updateEarthquake` 13](#_Toc161584446)

[4.4测试总结 14](#_Toc161584447)

[4.5 改进建议 14](#_Toc161584448)

[4.6 附录 14](#_Toc161584449)

[5. UI测试 15](#_Toc161584450)

[5.1页面加载： 15](#_Toc161584451)

[5.2页面布局和设计： 16](#_Toc161584452)

[5.3导航： 17](#_Toc161584453)

[5.4表单和输入： 18](#_Toc161584454)

[5.5交互元素： 20](#_Toc161584455)

[5.7用户登录： 20](#_Toc161584456)

[5.8错误处理： 20](#_Toc161584457)

[5.9动态内容： 21](#_Toc161584458)

[6. 探索性测试 21](#_Toc161584459)

[6.1登陆测试-用户界面处理请求申请 21](#_Toc161584460)

[6.2通知公告新建 23](#_Toc161584461)

[6.3基本震情的编辑 25](#_Toc161584462)

[6.4文件上传 26](#_Toc161584463)

[5） 错误结果：震情码重复，无法正常添加文件； 26](#_Toc161584464)

[7.测试结论 28](#_Toc161584465)

[7.1测试报告总结： 28](#_Toc161584466)

[7.1.1单元测试： 28](#_Toc161584467)

[7.1.2接口测试： 28](#_Toc161584468)

[7.1.3 UI测试： 28](#_Toc161584469)

[7.1.4探索性测试： 28](#_Toc161584470)

[7.2综合总结： 29](#_Toc161584471)

[7.3建议和改进： 29](#_Toc161584472)

# 1.导言

## 1.1项目背景介绍

随着自然灾害和突发事件频发，对于快速、准确、多源、全面的灾情信息获取与管理需求日益迫切。本项目是国家震情应急管理系统的一部分，将完成一个多源灾情数据管理服务系统，多源社会灾情数据通过接口输入到多源灾情数据管理服务系统平台。对震后灾情获取缓慢且碎片化，针对公众涉灾信息数据异构、多维、数据格式差异大、部分数据维度缺失导致的数据无法得到综合利用的现状，研究基于异构公众涉灾信息，进行一体化编码；用开放接口技术和实时动态管理技术将编码后的数据输入到虚拟化管理系统对灾情信息进行管理，实现必要的可视化。数据来源包括业务报送数据、泛在感知数据、舆情感知数据和承载体基础数据等，将来可以扩展更多数据来源。当外界发出数据请求时，从管理系统中获取目标数据，并通过接口发送给请求方。还需要设计一个通用的架构以适应众多异源异质异构的数据请求。实时处理技术的研究，实现灾情数据统一管理和高效合理利用，构建多源社会灾情数据管理服务系统。

## 1.2项目范围

本文档是针对项目测试和可信评估的报告，主要包括项目的功能和性能测试结果分析，以及相应的可信研究报告。文档预期读者包括：

· 软件评估人员

· 编程人员

· 测试人员

· 用户

## 1.4测试项目需求

### 1.1.1.功能性需求

具体需求描述如下:

#### 1）数据输入

能够从以不同的方式获取灾情数据，灾情数据都是经过一体化编码规范的，能够批量处理从不同格式的文件中读取出灾情数据并解码（例如Json，XML、CVS、txt、sql等数据），目前主要实现Json，Excel格式，附1给出一些数据内容的例子。

#### 2）数据编码（一体化编码）

震情编码采用分层组合编码方法，依次为地理信息编码和时间编码。其中：

第一段地理信息码用于描述基础地理信息，用12位阿拉伯数字标识；

第二段时间码用于记录地震发生的具体时间和具体时刻，用14位阿拉伯数字标识。

数据来源分为业务报送数据、泛在感知数据、其他数据三个大类。采用组合码，分为两段，由3位阿拉伯数字组成。

第一段表示大类代码，由1位阿拉伯数字表示，分别对应三种数据来源；

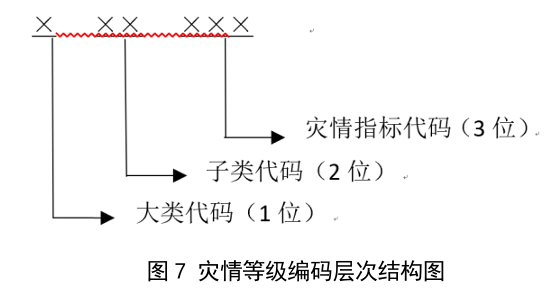
第二段表示子类代码，由2位阿拉伯数字表示，采用系列顺序码，在各类别代码取值范围内对编码对象顺序地赋予代码值。

灾情信息码编码采用组合码，分为3段，编码结构见表 5和图7。

第一段表示灾害大类代码，由1位阿拉伯数字表示。

第二段表示灾害子类代码，由2位阿拉伯数字表示。

第三段表示灾情指标代码，由3位阿拉伯数字表示。



#### 3）数据管理

a)数据存储–数据入库（数据检查）

b)数据可视化-查询、图和列表展示（可以通过GIS服务展示震情数据）

c)数据动态管理-增删改、备份、统计分析

#### 4）数据输出

根据用户请求，抽取合适的灾情数据的数据编码（编码信息或者文字信息）

根据用户请求，形成Json、XML、txt、Cvs等格式并能将文件保存到用户本地。

### 1.1.2.非功能性需求

#### 1）性能与效率

a)时间性能

 • 系统反应时间

多源灾情数据管理服务系统客户端点击相关按钮后的响应时间应不超过3秒。

 • 系统并发数

当多用户同时访问并使用系统时，不会出现服务器宕机的情况。本多源灾情数据管理服务系统应当最低支持50位用户同时并发使用。

b)空间性能

 • 数据库容量应能够存储至少1000条震情数据、1000条人员伤亡及失踪数据、1000条房屋破坏数据等数据记录。

 • 系统有足够大的缓存空间，保障系统运行流畅。

#### 2）可靠性

可靠性是指项目在规定条件下，规定时间内，完成规定功能的能力。软件可靠性可以进一步分解为3个子特性：成熟性、容错性、易恢复性。

本系统面向社会，用户数量非常庞大，可能会因为用户的误操作引起系统的异常。因此本系统要求具有较强的容错能力，能够捕捉由于用户误操作引起的异常，并在可容忍错误程度的情况下，保持稳定运行。本系统的可靠性需求具体体现在系统应能长时间下稳定运行。当用户在本系统内的各种输入不符合要求规范的时候，均不会引起系统的故障，并能提示用户错误操作。选择数据库产品时，要考虑一定的数据负载能力。当设备故障时，系统需要具备一定的恢复能力，数据需要有至少一个备份（可以采用异地容灾备份的方式），从而提高多源灾情数据管理服务系统中数据抵抗外界破坏的能力。系统具有一定的容错和抗干扰能力，在非硬件故障或非通讯故障时，系统能够保证正常运行，并有足够的提示信息帮助用户有效正确地完成任务。

#### 3）易用性

易用性是从用户使用观点来看的，指在指定条件下使用时，软件产品被理解、学习、使用和吸引用户的能力。易用性可以分解为4个子特性：易理解性、易学性、易操作性、吸引性。

a)系统的维护时间应当尽量降低，软件结构应当清晰明了，方便后期维护。系统平均维修时间应不超过12小时。

b)多源灾情数据管理服务系统面向用户为普通群众，因此系统应当具备良好的人机交互界面，符合大众的使用习惯，操作方便简单，任何新用户都能在很短的时间内熟悉系统的使用操作。

c)在系统平台中设置操作引导或操作说明手册，支持不熟悉计算机操作甚至没有任何相关经验的人都能够使用本系统。

d)确保录入信息完整且合理。本系统能够使用相应的检测与提示机制，确保信息能够被完整正确地录入。

e)操作完成时或者操作有误均会有统一规范的提示信息。

#### 4）密安性

a)系统的各级权限应当有明确的规划，系统管理员应当管理好每个权限，例如每个用户之间不能查看或修改其他用户的账号信息；

b)数据库中的信息必须要得到良好保护，系统应该通过设置防火墙，确保数据库存储和传输的安全，不会因为恶意攻击而崩溃。

#### 5）可维护性

可维护性是指软件可被修改和维护的能力，包括易分析性、易修改性、稳定性和易测试性。在运行中，应当容易判断出系统的缺陷和失效原因。代码、设计和文档应当结构清晰，易于修改。同时，保证系统的稳定性，避免多次修改造成代码混乱，文档不清晰。当系统应用于不同区域和场景的环境之下时，应该具备良好的适应性，不需要通过大幅度的接口与内部过程修改，就能使用户进行使用。

#### 6）可移植性

可移植性是指软件产品从一种环境迁移到另外一种环境的能力。包括易安装性、共存性、易普换性。

本系统应当具有一定的跨平台和环境的能力。作为网页端平台，应当能够在不同内核的浏览器中正常运行。系统接口易于调用和改造，可以方便地移植到不同的设备上。

# 2.测试项目介绍

## 2.1测试环境

### 2.1.1 网络拓扑结构

测试环境如图1.1，原始数据库中的数据通过FTP服务器和Web分布式服务器的处理后传递给交换机，ftp服务器上存储系统的输入输出文件，分布式服务器上部署了系统的后端微服务以及前端网页。数据库分为MySQL数据库和MongoDB数据库，分别存储系统的基本数据以及进行舆论烈度分析的数据。PC终端通过Internet和防火墙从交换机中获取数据。

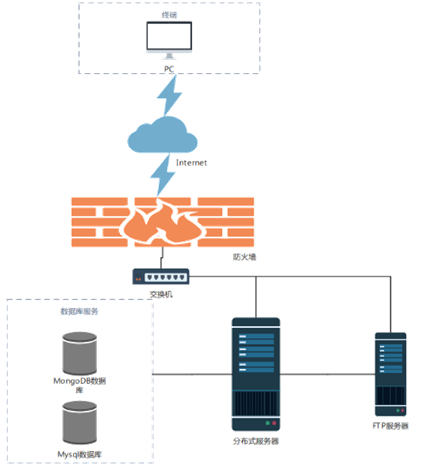


图1.1

### 2.2.2服务器端环境

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | 分布式服务器 | FTP服务器 | 数据库服务 |
| 硬件 | 型号 | 华为云云耀服务器HECS | 阿里云云服务器ECS | 阿里云云数据库RDS |
| CPU | 4vCPUs Intel Xeon Gold 6278C CPU 2.60GHz | Intel Xeon Platinum 8269CY CPU 2.50GHz |  |
| 内存 | 8GB | 2GB | 1GB |
| 硬盘 | 40G | 40GB | 20GB |
| 软件 | 操作系统 | ubuntu-18.04.6-desktop-amd64 | Windows Server 2016 Datacenter |  |
| 软件 | Java 1.8、Nginx 1.14.0、MongoDB 4.4.5、Nodejs 10.16.0 | FTP Server文件服务器 | MySQL 8.0  MongoDB 4.4.5 |

### 2.2.3客户端环境

操作系统：Windows 11

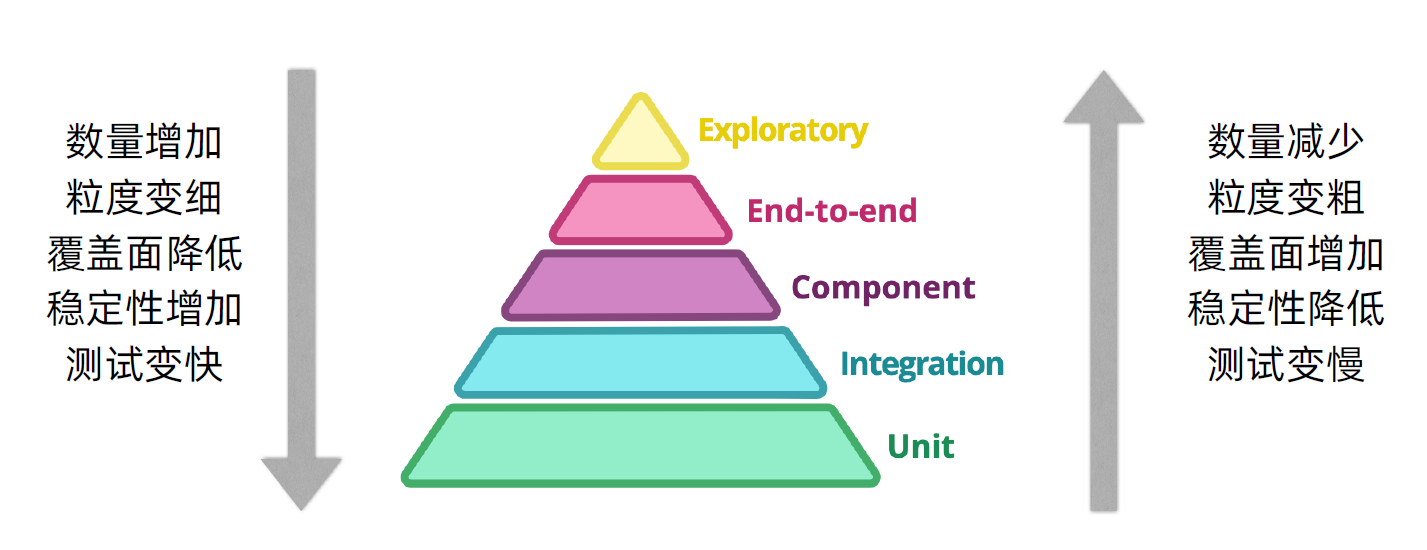
浏览器：Firefox/Google

处理器：i5/i7

Java环境：jdk1.8.0

## 2.3测试模型

本项目测试过程中采用了敏捷测试金字塔模型，如下图所示，我们完成了单元测试、接口测试、UI测试和探索性测试。



# 3.单元测试

单元测试(unit testing)，是指对软件中的最小可测试单元进行检查和验证。对于单元测试中单元的含义，一般来说，要根据实际情况去判定其具体含义，如 C 语言中单元指一个函数，Java 里单元指一个类，图形化的软件中可以指一个窗口或一个菜单等。总的来说，单元就是人为规定的最小的被测功能模块。单元测试是在软件开发过程中进行的白盒测试活动。 敏捷强调 “测试先行”。本项目基于 TDD 的开发模式，即单元测试驱动开发、同时根据单元测试不断完善代码。使用 Junit 工具实现单元测试，在测试过程中遵守 TDD的“从红到绿“的测试过程，因此，单元测试与开发过程同步完成，实现了所有单元的测试，下面给出部分单元测试过程。

## 3.1地震单元测试

本测试是对次生灾害类进行单元测试，测试用例代码如图3-1所示，测试结果如图3-2所示，测试通过。



图3-1 地震单元测试类

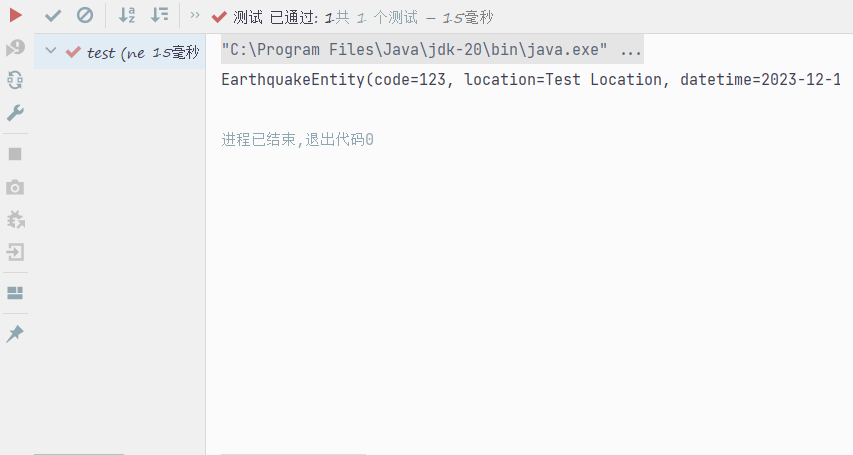


图3-2 第一次执行测试，测试通过

# 4.接口测试

## 4.1 查询所有震情 - `queryAll`

**请求**

接口路径：`/earthquake/queryAll`

请求方法：GET

**预期结果**

返回所有震情的列表

**实际结果**

成功，返回震情列表

4.2 添加震情 - `addEarthquake`

**请求**

接口路径：`/earthquake/add`

请求方法：POST

请求参数：（提供一些合理的参数，包括震情码、地点、时间等）

**预期结果**

返回成功添加震情的消息

**实际结果**

成功，返回添加成功的消息

## 4.3 更新震情信息 - `updateEarthquake`

**请求**

接口路径：`/earthquake/update`

请求方法：POST

请求参数：（提供一些合理的参数，包括需要更新的震情的ID和更新的信息）

**预期结果**

返回成功更新震情信息的消息

**实际结果**

成功，返回更新成功的消息

## 4.4测试总结

成功的测试用例数量：3

失败的测试用例数量：0

遇到的问题：无

## 4.5 改进建议

接口功能正常，未发现明显问题。

建议增加更多复杂场景的测试用例，如异常输入、大数据量等。

## 4.6 附录

`code`: 震情码，36位数字

`location`: 发生的地点

`datetime`: 发生的时间

`source`: 数据来源大类

`subsource`: 数据来源子类

`carrier`: 载体

`disaster`: 灾情大类

`subdisaster`: 灾情子类

`degree`: 灾情指标

`deletedFlag`: 是否删除，0否、1是

`remark`: 备注

# UI测试

## 5.1页面加载：

1）页面是否在合理的时间内加载完成？

2）页面中的图像、样式和脚本是否正确加载？



图5.1

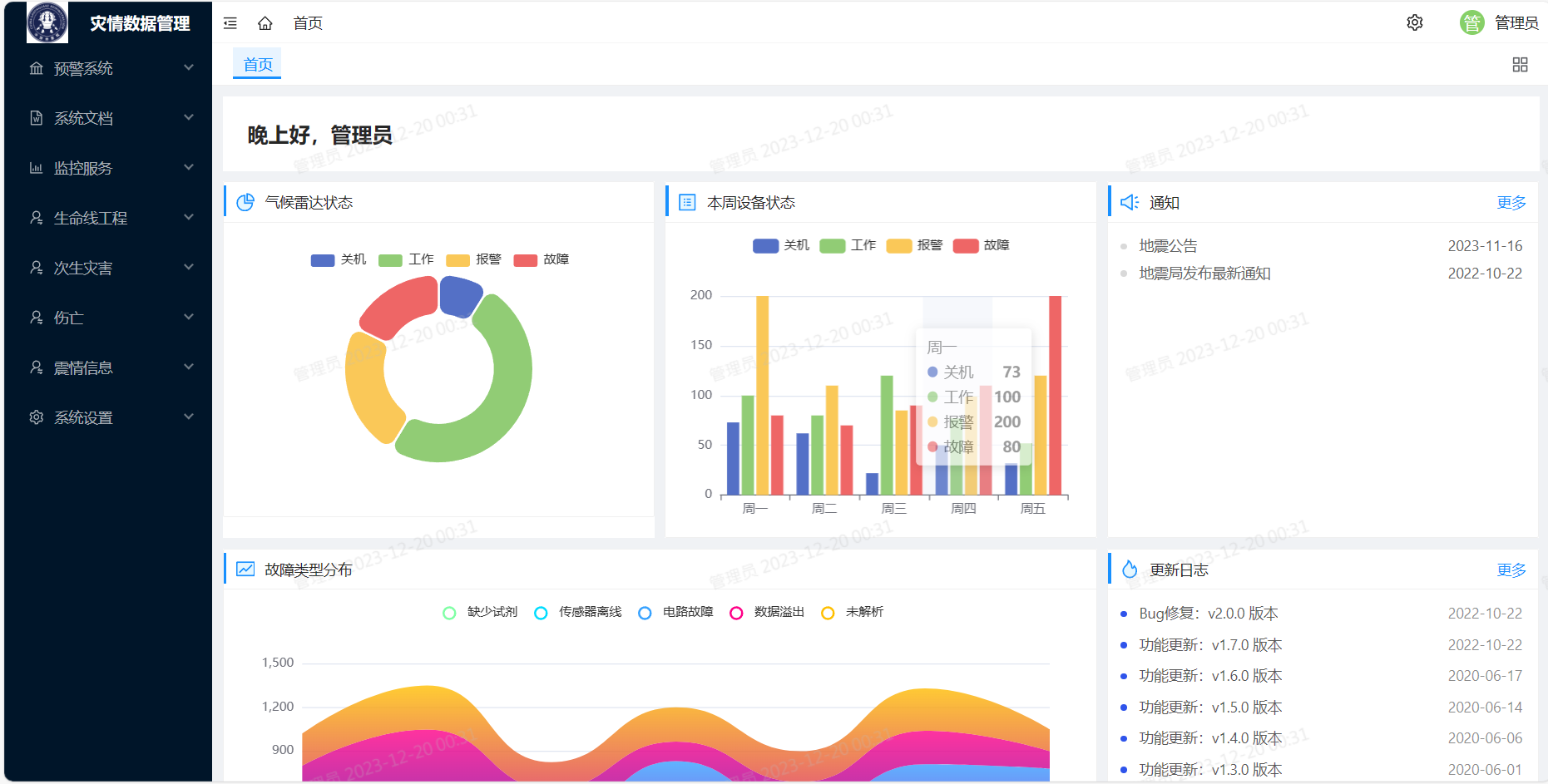


图5.2

由图5.1、图5.2可以发现，网站登录页面和主页面打开正常，相对应的图像和样式也已经正常加载，测试内容1）、2）均已经满足相关要求；页面加载时间是用户体验的重要组成部分。页面在平均2-3秒内加载完成，符合用户的期望。所有图像、样式和脚本都成功加载，没有404或其他加载错误。

### 5.2页面布局和设计：

1）页面布局是否符合设计和用户体验？

2）页面是否在不同屏幕尺寸和浏览器中正确呈现？



图5.3

放大图5.2的中间部分， 页面布局经过仔细设计，确保在不同屏幕尺寸和浏览器中都能正确呈现。我们使用了响应式设计和媒体查询，以适应移动设备和桌面浏览器，确保用户在任何设备上都能获得一致的用户体验。

## 5.3导航：

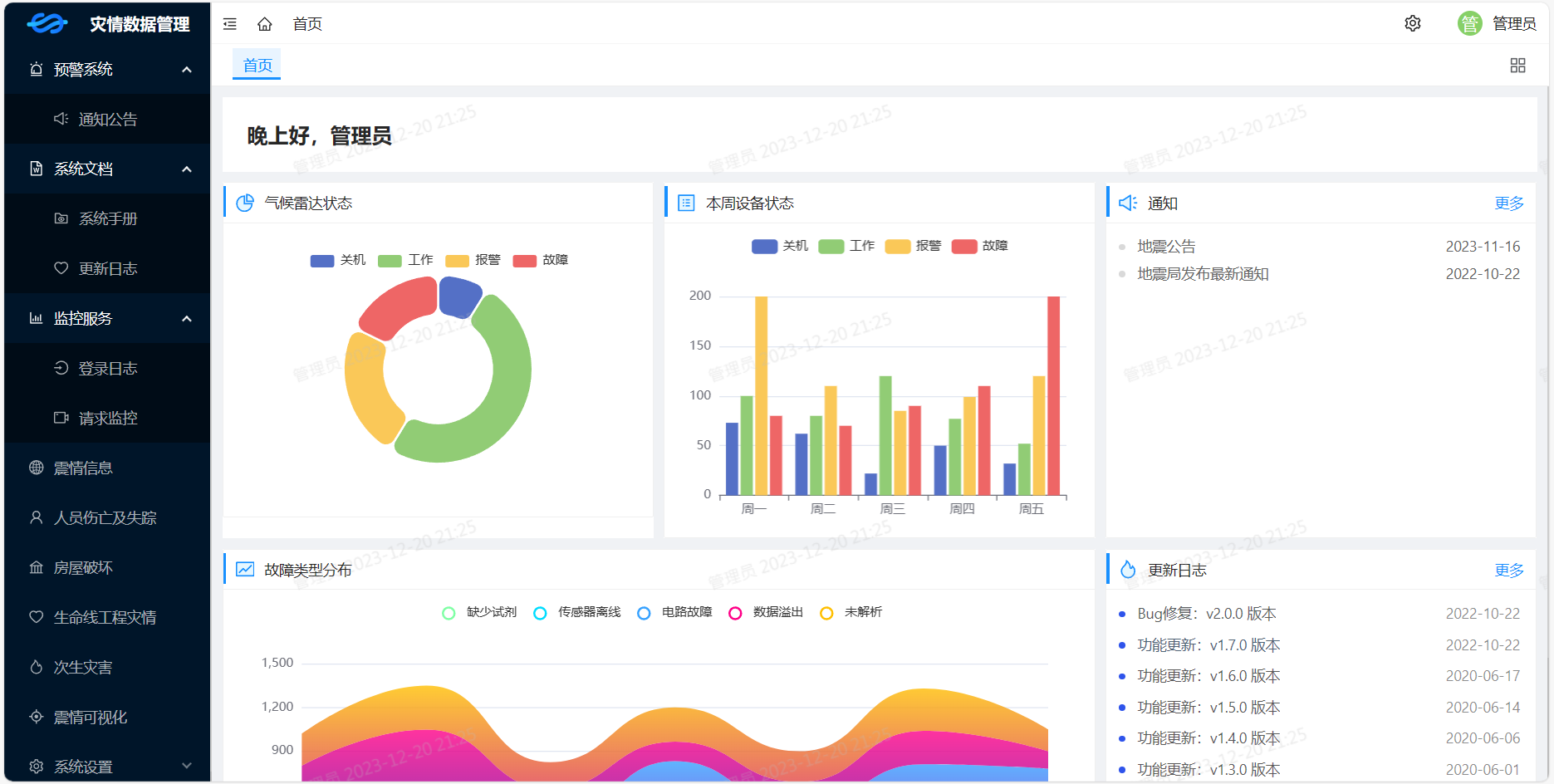


图5.4

1）导航链接是否正确跳转到目标页面？

2）导航菜单是否显示正确？

参考图5.4，可以发现所有导航链接都经过测试，确保正确跳转到目标页面。导航菜单在不同浏览器和平台上都显示正确，且链接的语境相关性被优化，用户可以轻松找到所需的信息。

## 5.4表单和输入：

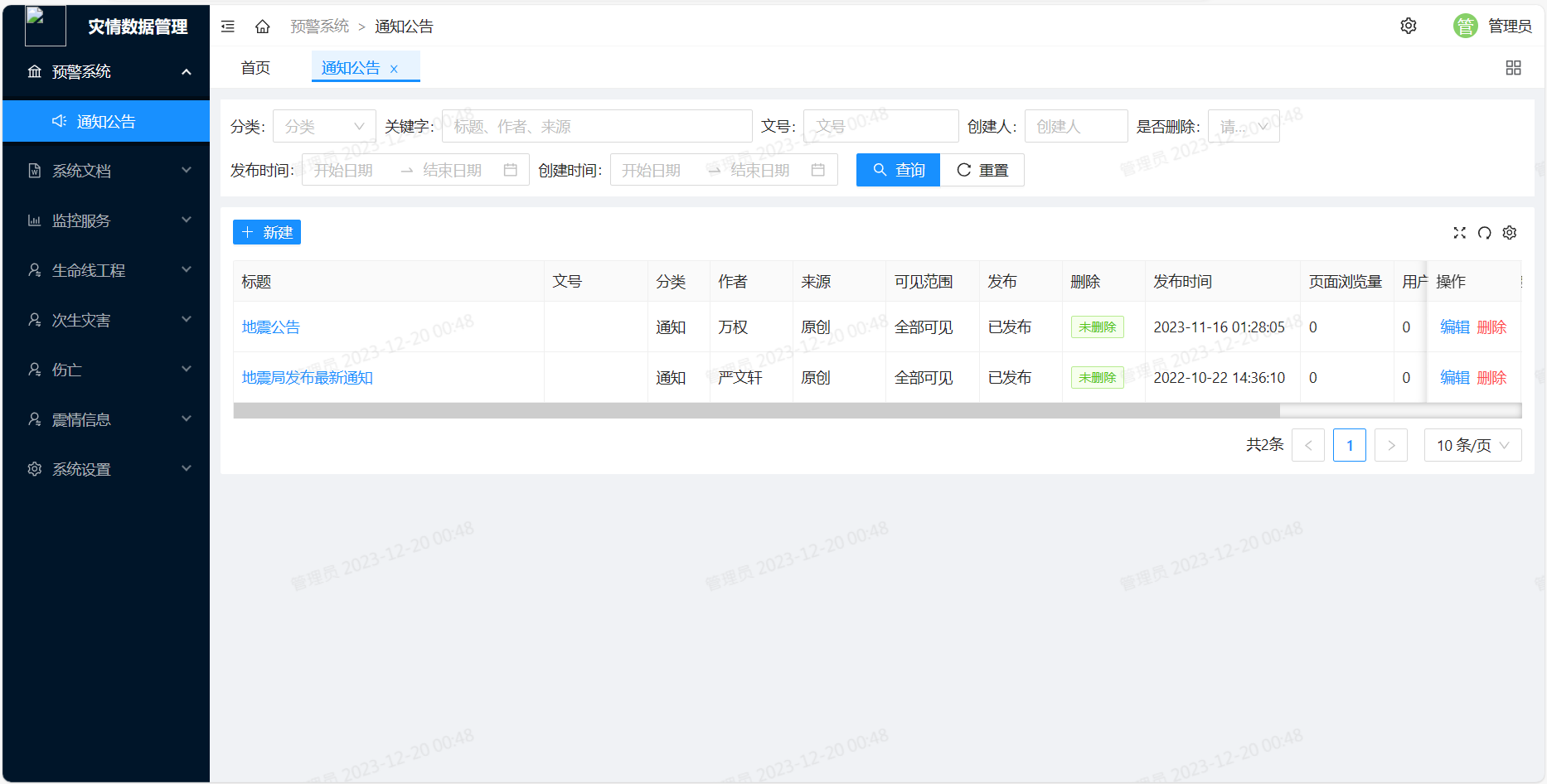


图5.5



图5.6

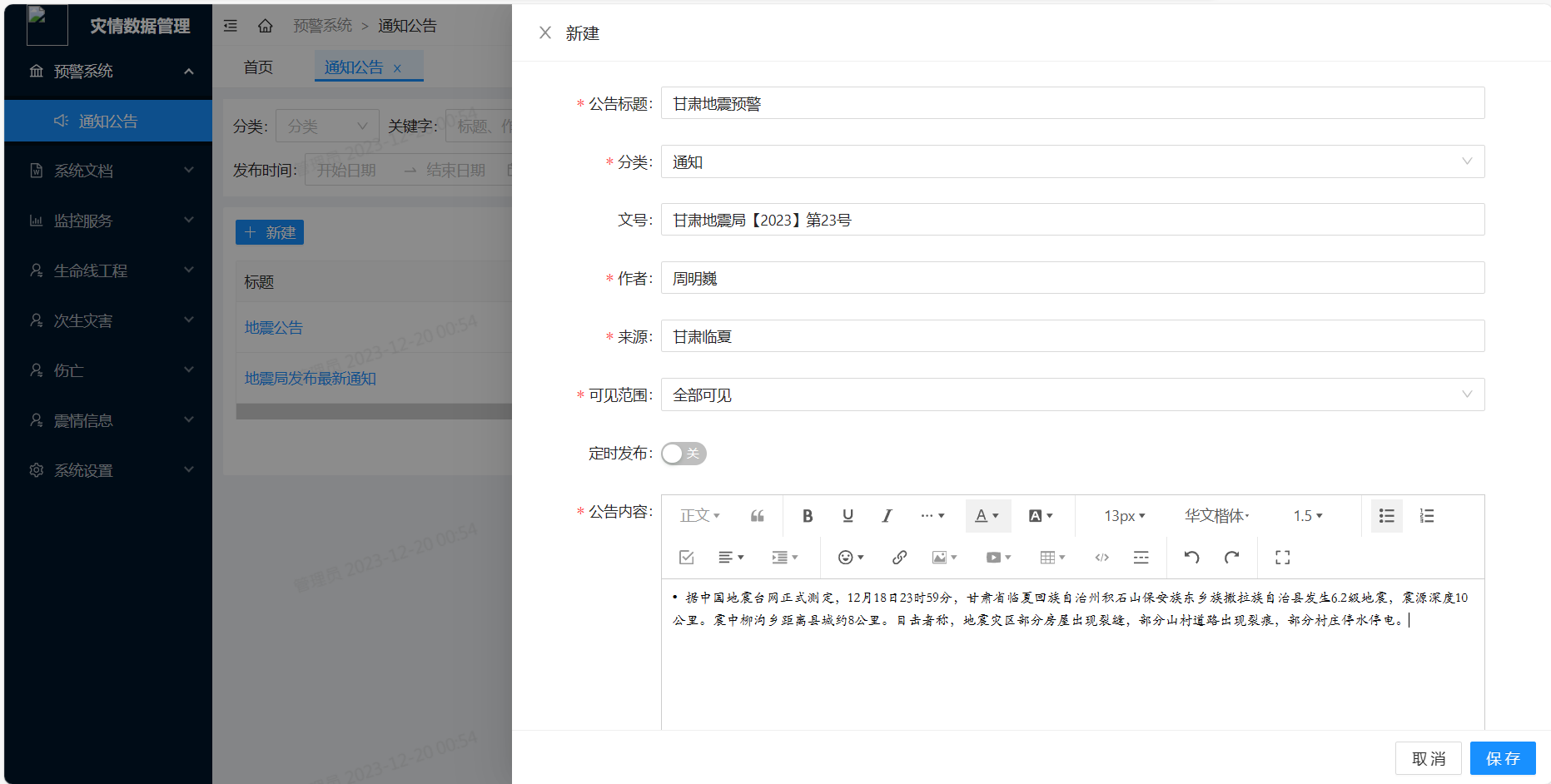


图5.7

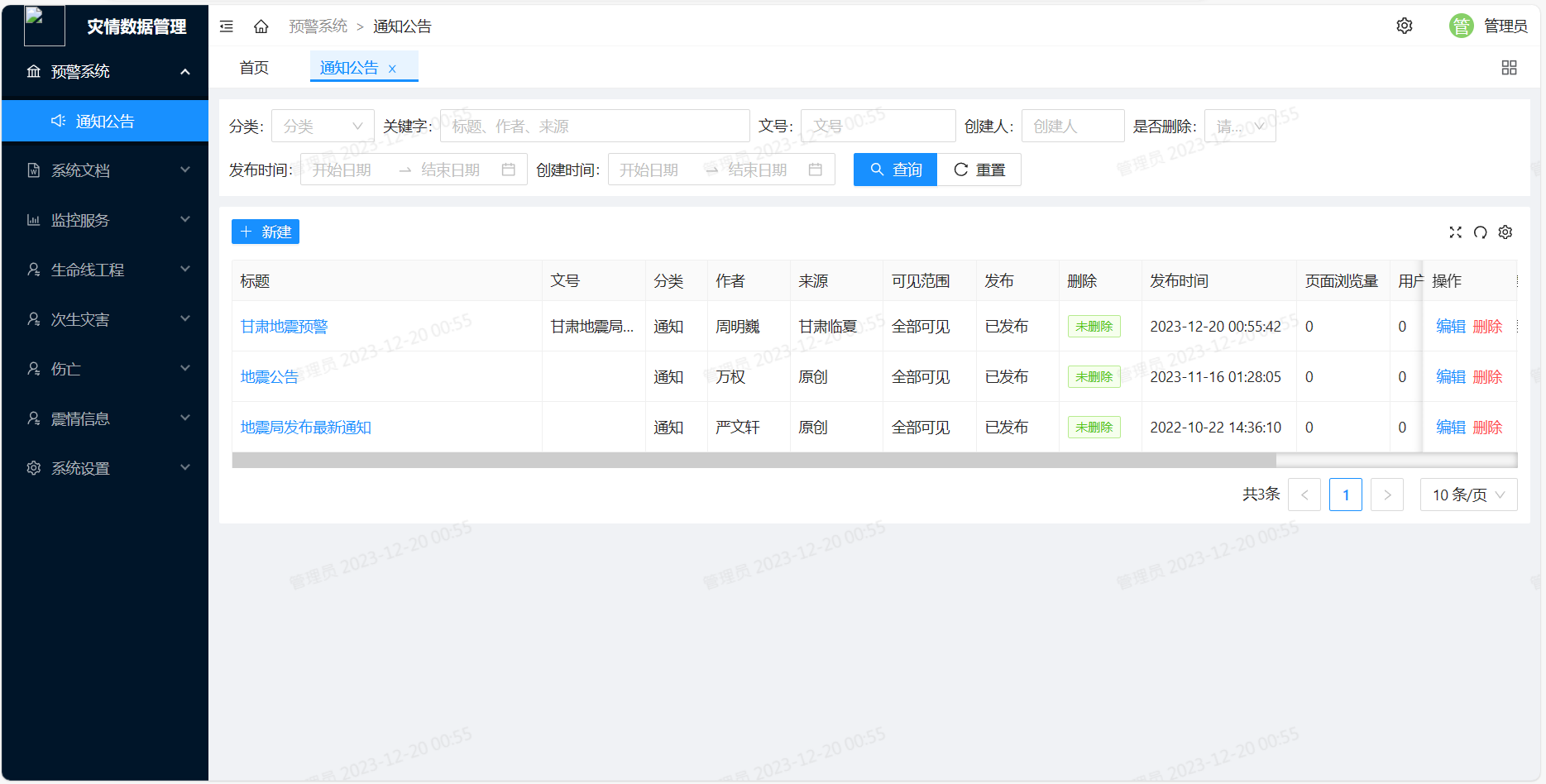


图5.8

1）表单是否能够正确提交？

2）输入框是否接受正确的数据类型？

3）表单字段是否有必填项，是否做了适当的验证？

参考图5.5、5.6、5.7、5.8，我们通过测试可以发现所有表单都能够正确提交，输入框接受正确的数据类型，并进行了适当的验证。有必填项的字段都被明确定义，错误的输入会得到清晰的错误提示，帮助用户更正输入。

## 5.5交互元素：

1）页面中的按钮、链接、下拉框等交互元素是否可点击？

2）是否有隐藏的交互元素，用户可以通过特定操作显示它们？

此处可以参考图5.6、图5.7， 页面中的按钮、链接、下拉框等交互元素都可点击，没有隐藏的交互元素。用户可以通过标准操作发现和使用这些元素，提高了页面的可用性。

## 5.7用户登录：

1）用户登录功能是否正常工作？

2）登录后是否能正确显示用户信息？

此处参考图5.1、图5.2， 用户登录功能正常工作，用户登录后能正确显示其个人信息。我们通过安全的身份验证机制确保了用户的登录安全性。

## 5.8错误处理：

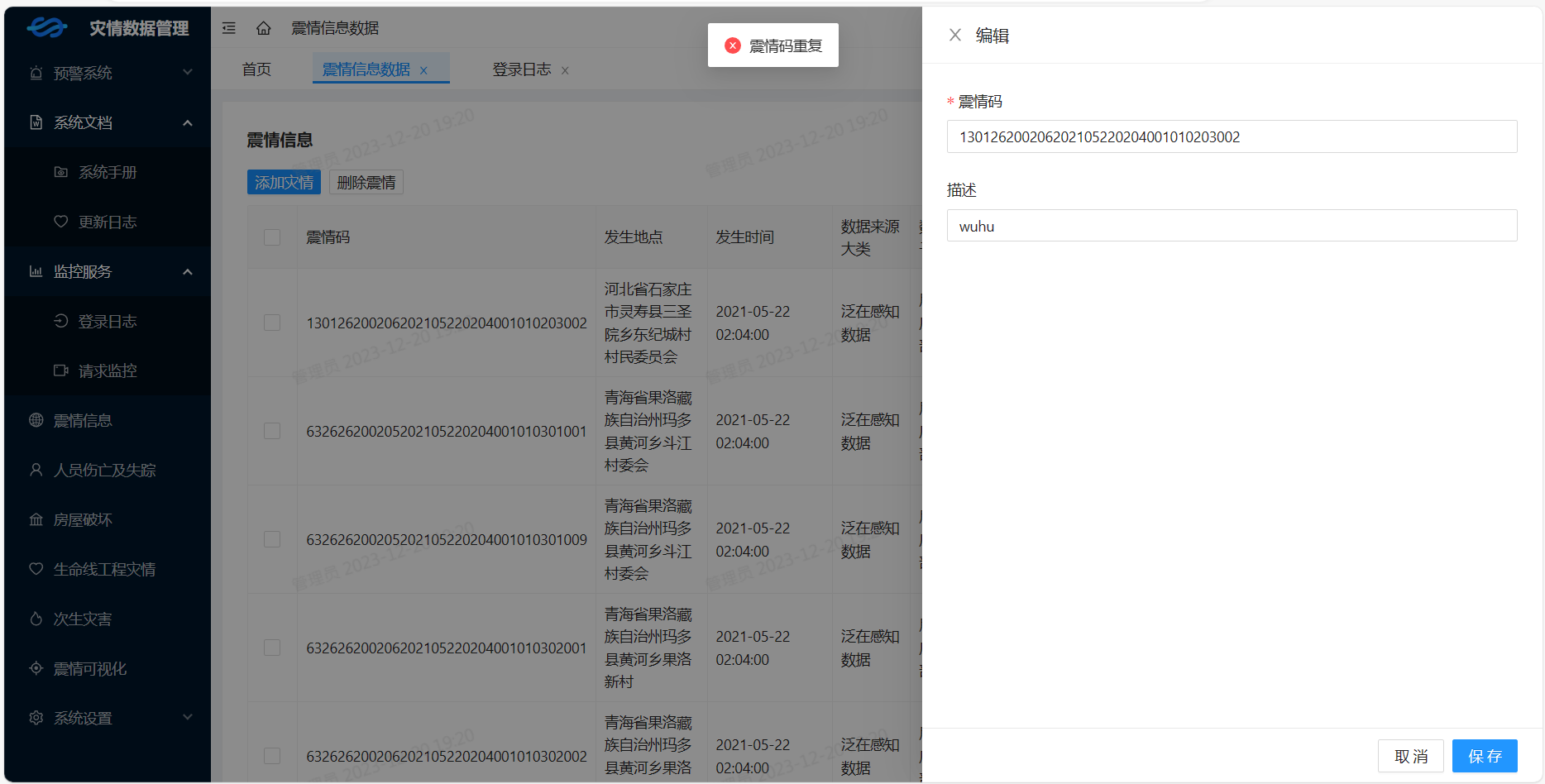


图5.9

1）是否能够正确处理用户输入错误？

2）错误页面是否友好、清晰，并提供帮助信息？

参考图6.9，可以发现当输入的灾情码重复时，可以发现系统能够正确处理用户输入错误，提供了清晰友好的错误页面，并包含帮助信息以指导用户纠正问题。安全性方面，输入都经过了有效的验证和过滤，防范了潜在的安全威胁。

## 5.9动态内容：

1）是否能够正确加载和显示动态生成的内容？

2）是否实时更新数据，例如聊天、通知等？

参考图5.4、图5.8，系统中包含了需要实时更新的数据，例如最新的新闻、实时报价、或者其他用户生成的动态内容。这些信息会在后台通过定时任务或实时触发机制进行更新，以确保用户能够看到最新的数据。如果系统中有与用户相关的事件提醒，比如提醒用户完成任务、定时提醒等，系统通过合适的方式（例如弹窗、通知栏消息）向用户展示这些提醒。系统中包含数据图表，这些图表的数据将根据用户选择的时间范围实时更新，以确保用户能够获取到准确的统计信息。 在用户与动态内容进行交互时，系统提供了友好的用户界面，确保用户能够轻松理解和参与到相关的活动中。

# 探索性测试

## 6.1登陆测试-用户界面处理请求申请

1. 测试流程： 输入网址--输入用户名和密码（数据操作员账号：admin 密码：246747hw）--输入验证码--点击登录--进入系统；
2. 页面正常显示，参考图6.1、图6.2、图6.3；
3. 预测结果：页面正常显示；
4. 实测结果：页面正常显示；
5. 错误结果：页面无法正常打开，参考图6.4；



图6.1



图6.2

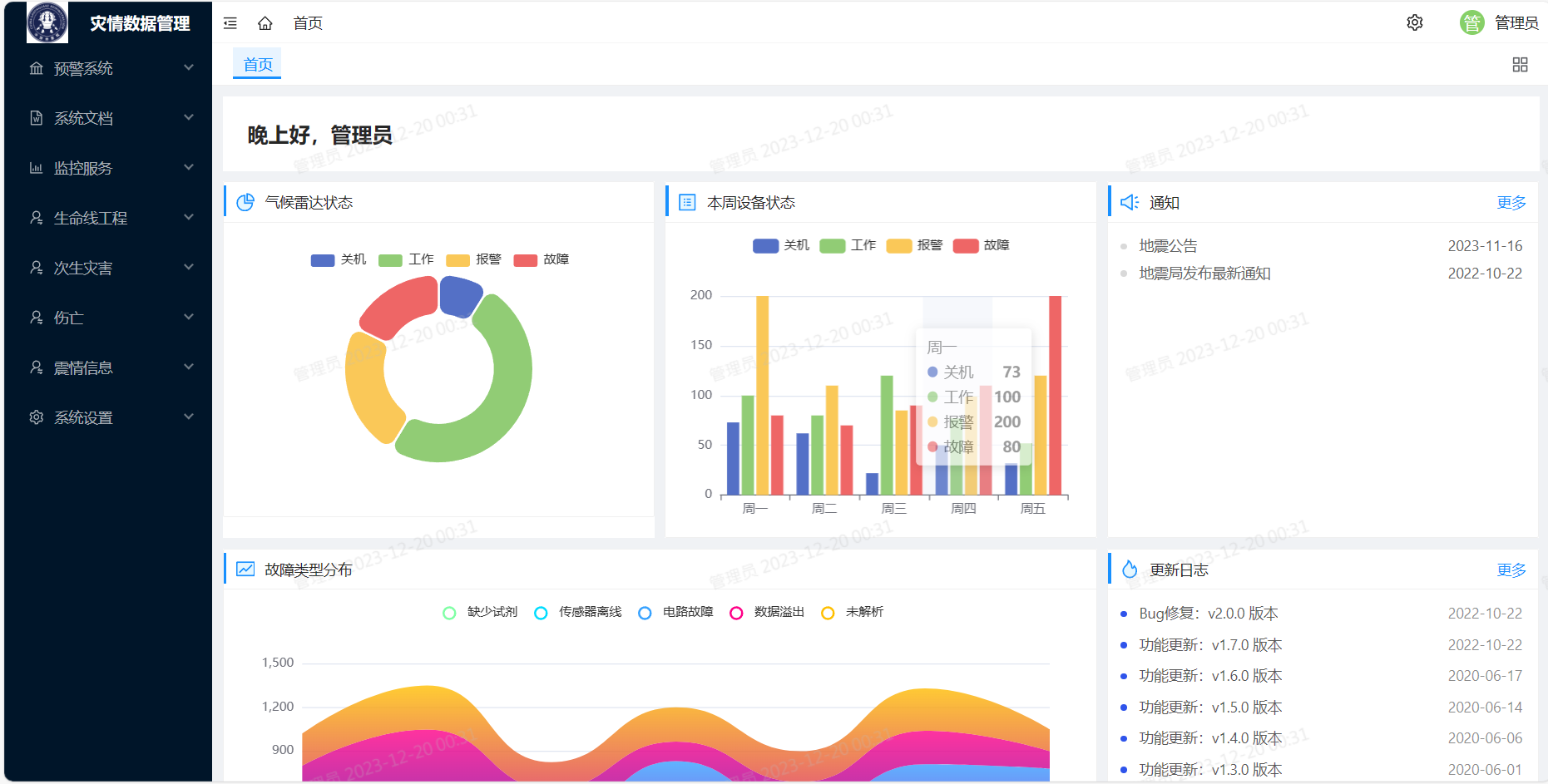


图6.3

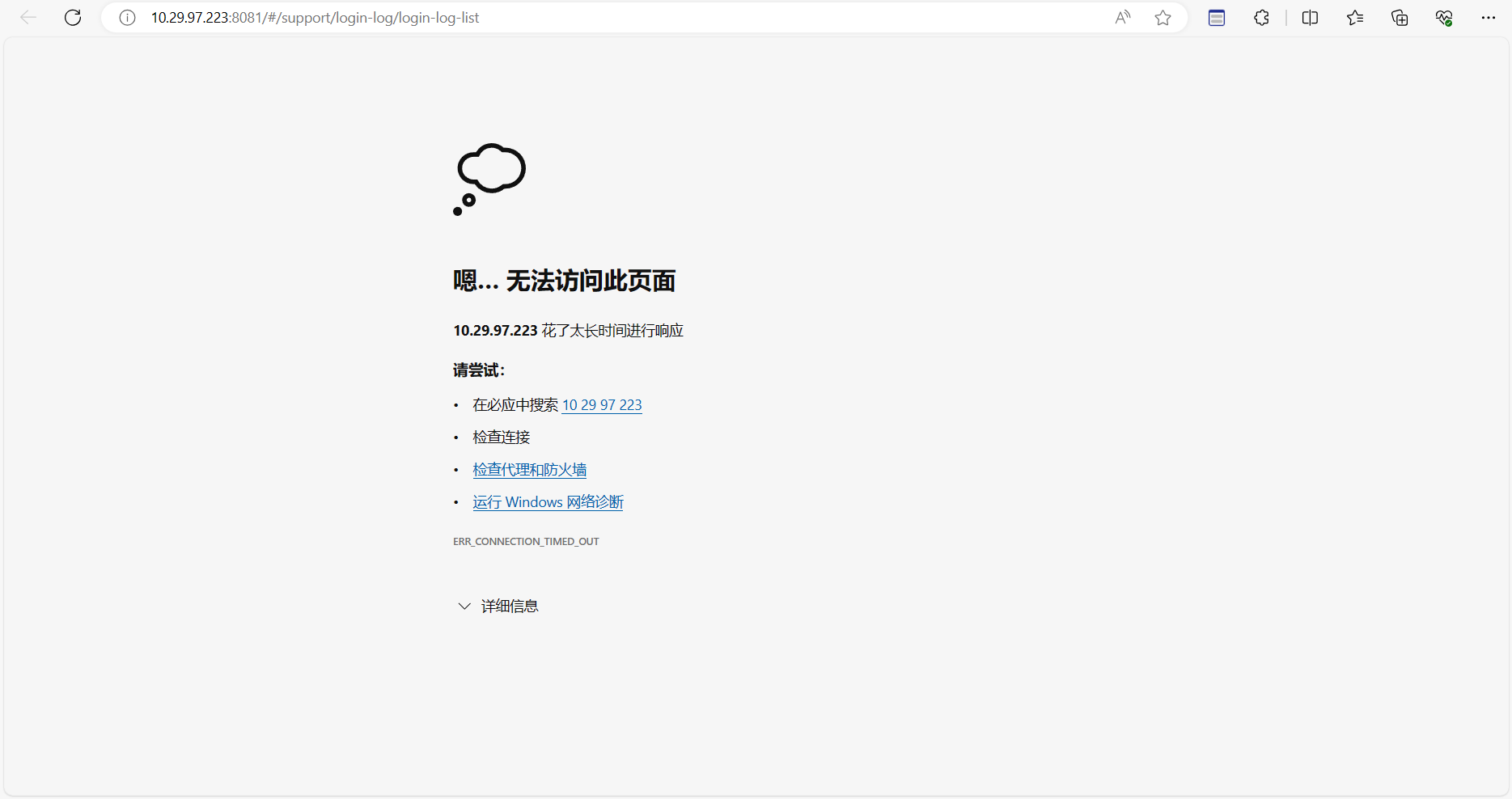


图6.4

## 6.2通知公告新建

1. 操作流程：在侧边框打开预警系统下拉框 -- 进入通知公告 -- 点击新建 -- 添加相关信息 -- 点击保存；
2. 页面正常显示，通告可以正常添加；
3. 预测结果：页面正常显示，通告正常的添加；
4. 实测结果：页面正常显示，通告正常的添加；
5. 错误结果：页面无法正常打开

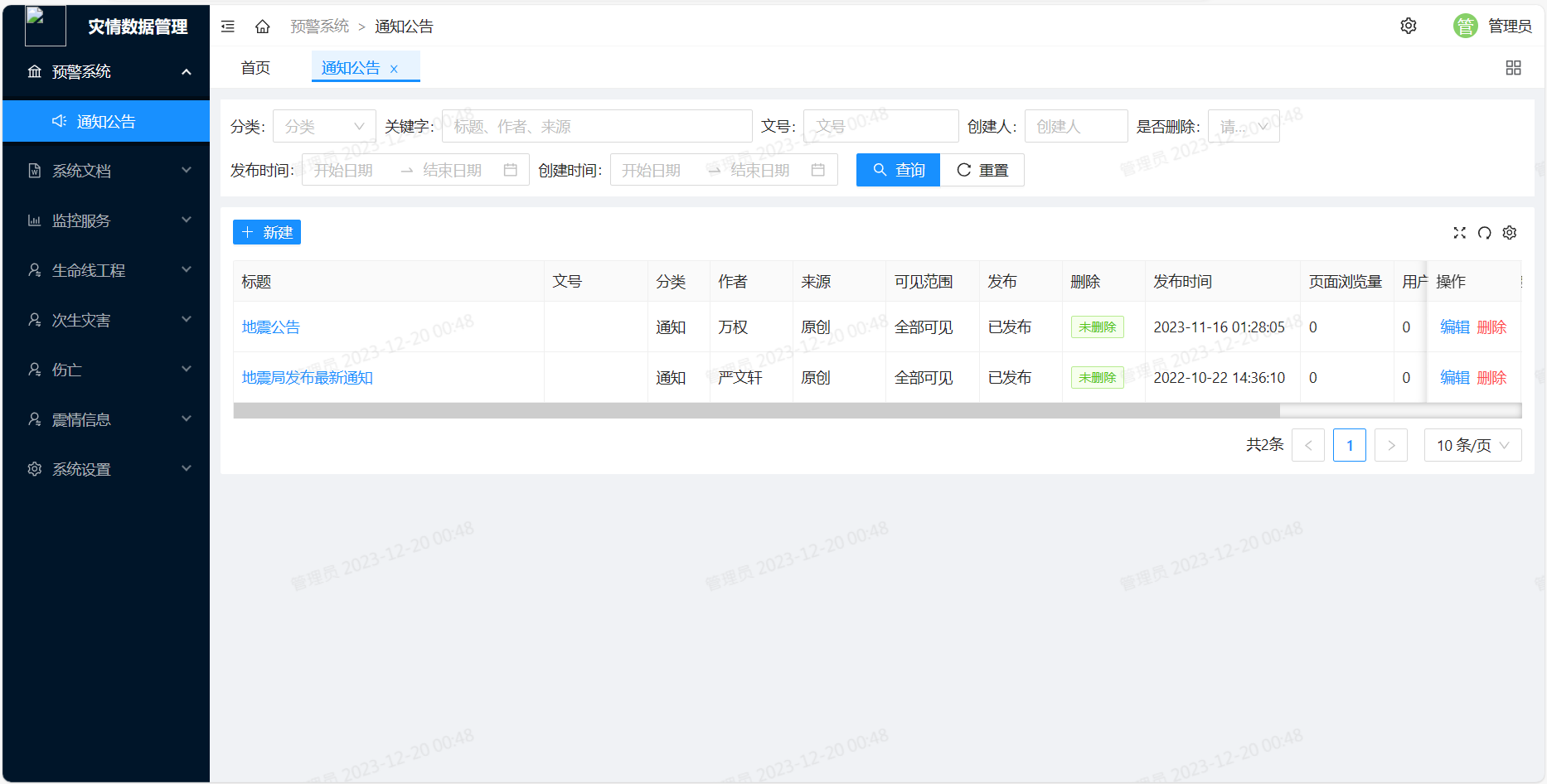


图6.5



图6.6

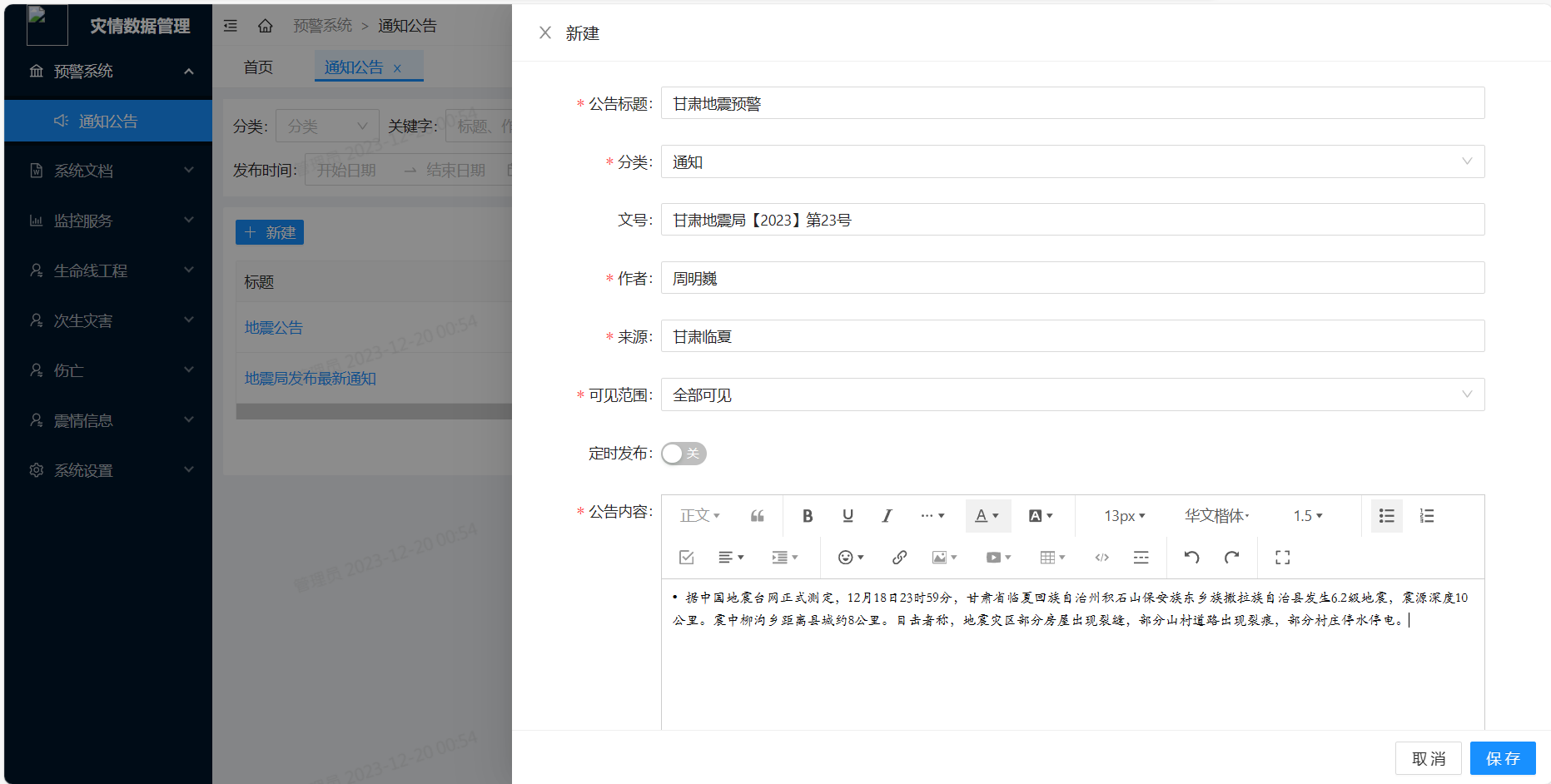


图6.7

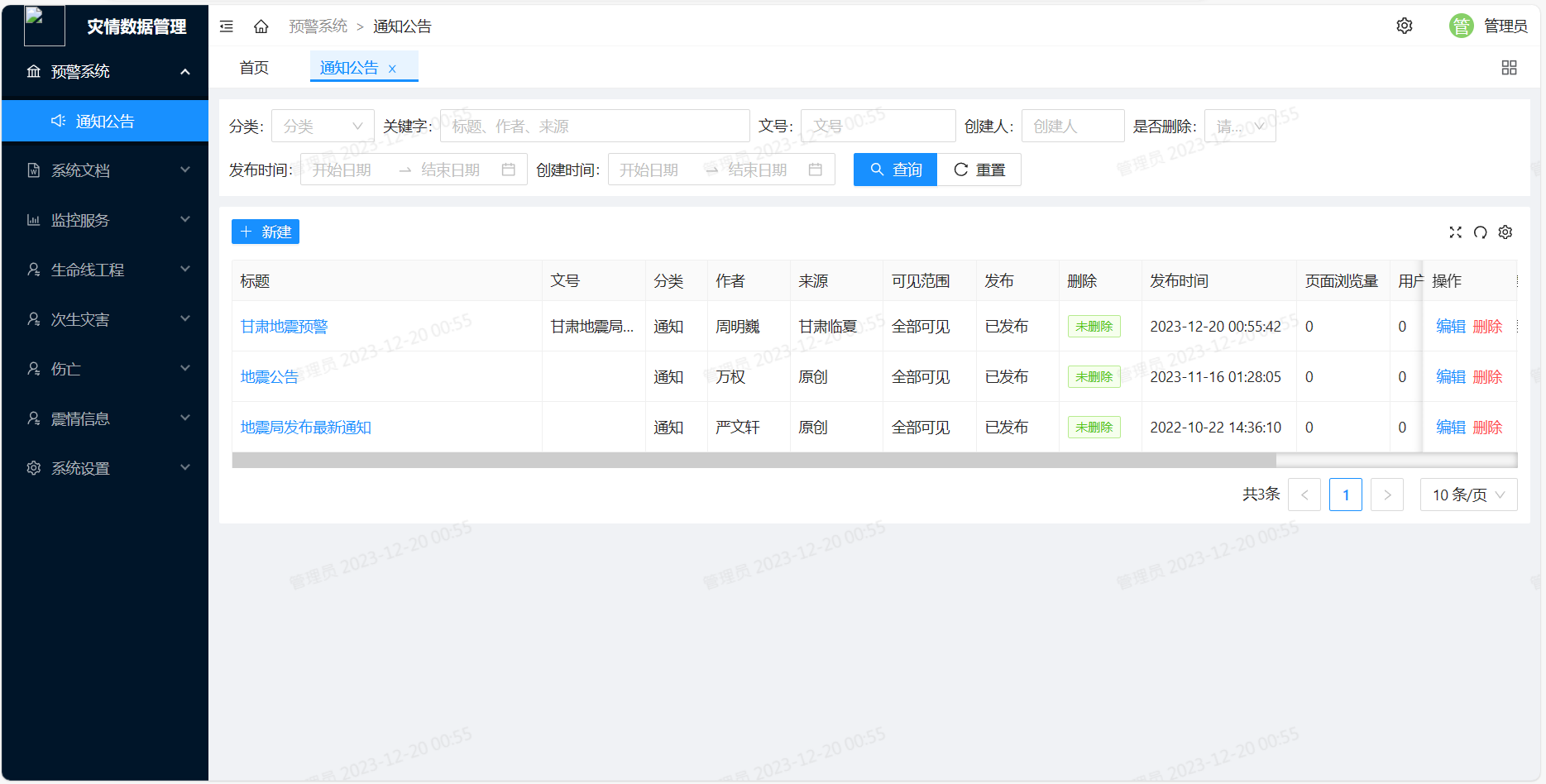


图6.8

可以发现已经添加进入了通知公告；

## 6.3基本震情的编辑

1. 操作流程：在侧边框打开震情信息 -- 进入震情信息管理 -- 点击添加震情-- 输入相关震情码，添加描述 -- 点击保存；
2. 页面正常显示，震情可以正常添加，参考图6.9、图6.10
3. 预测结果：页面正常显示，震情正常的添加；
4. 实测结果：震情码重复，无法正常添加参考图6.11
5. 错误结果：震情码重复，无法正常添加参考；

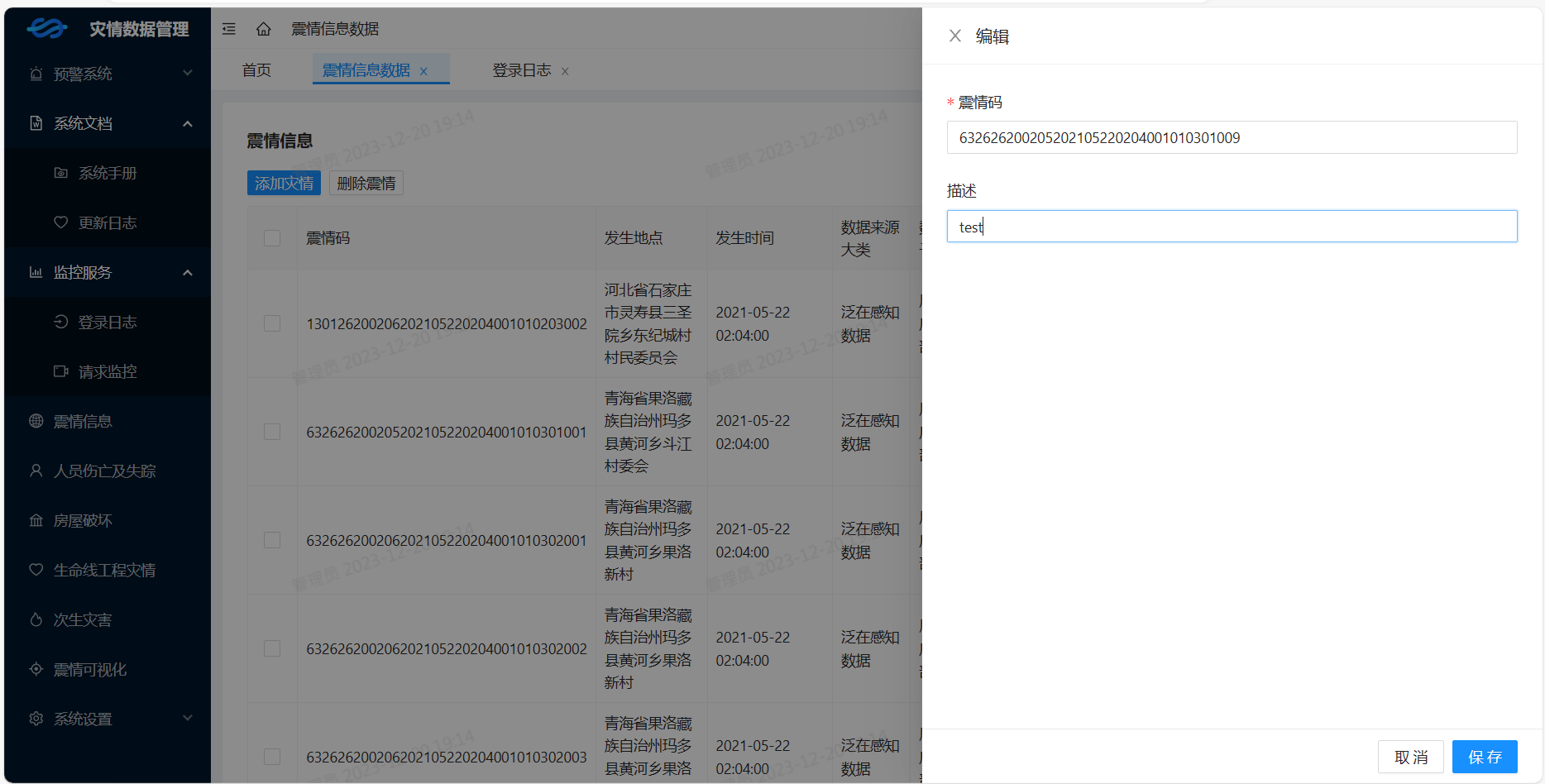


图6.9

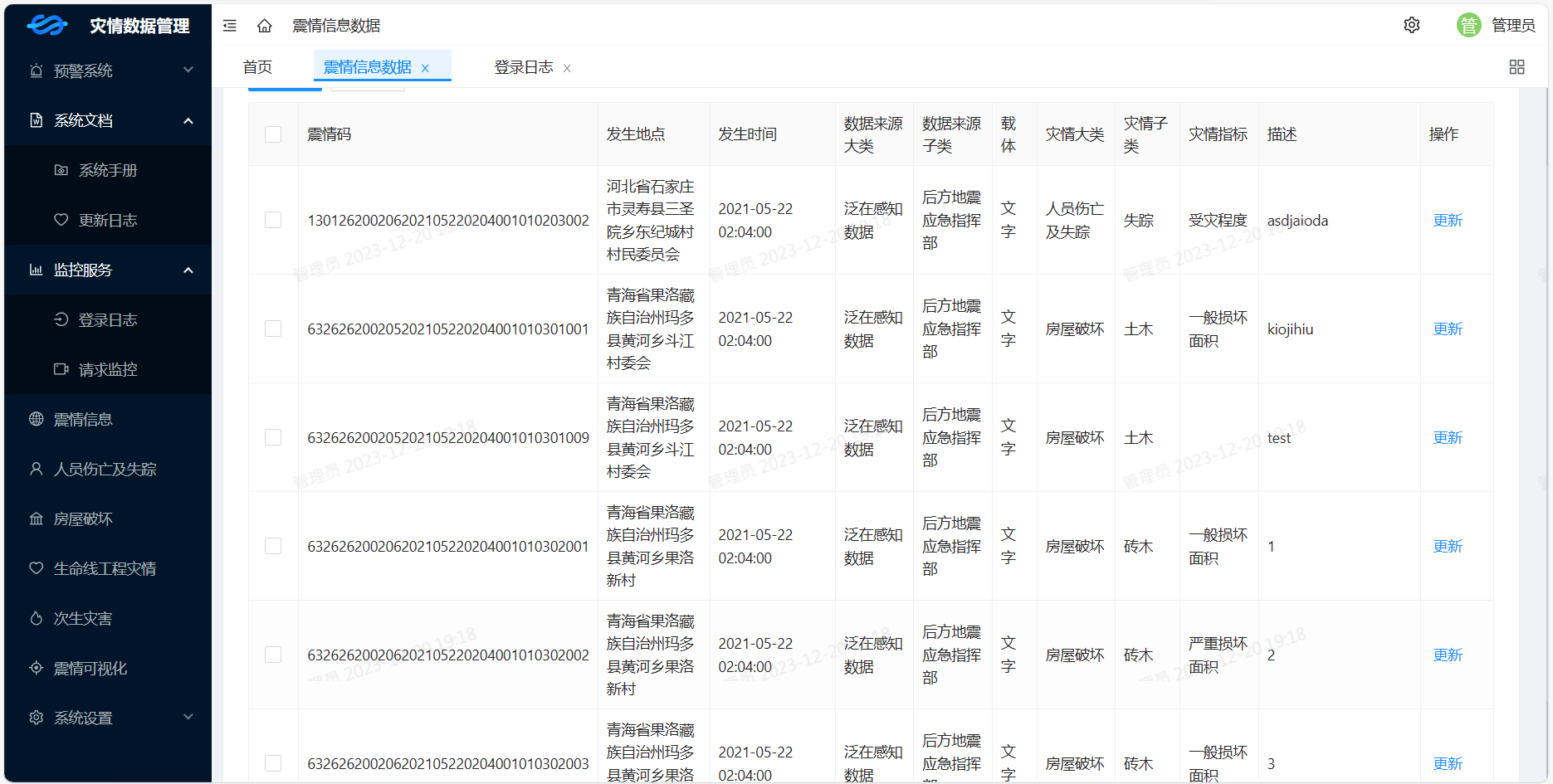


图6.10

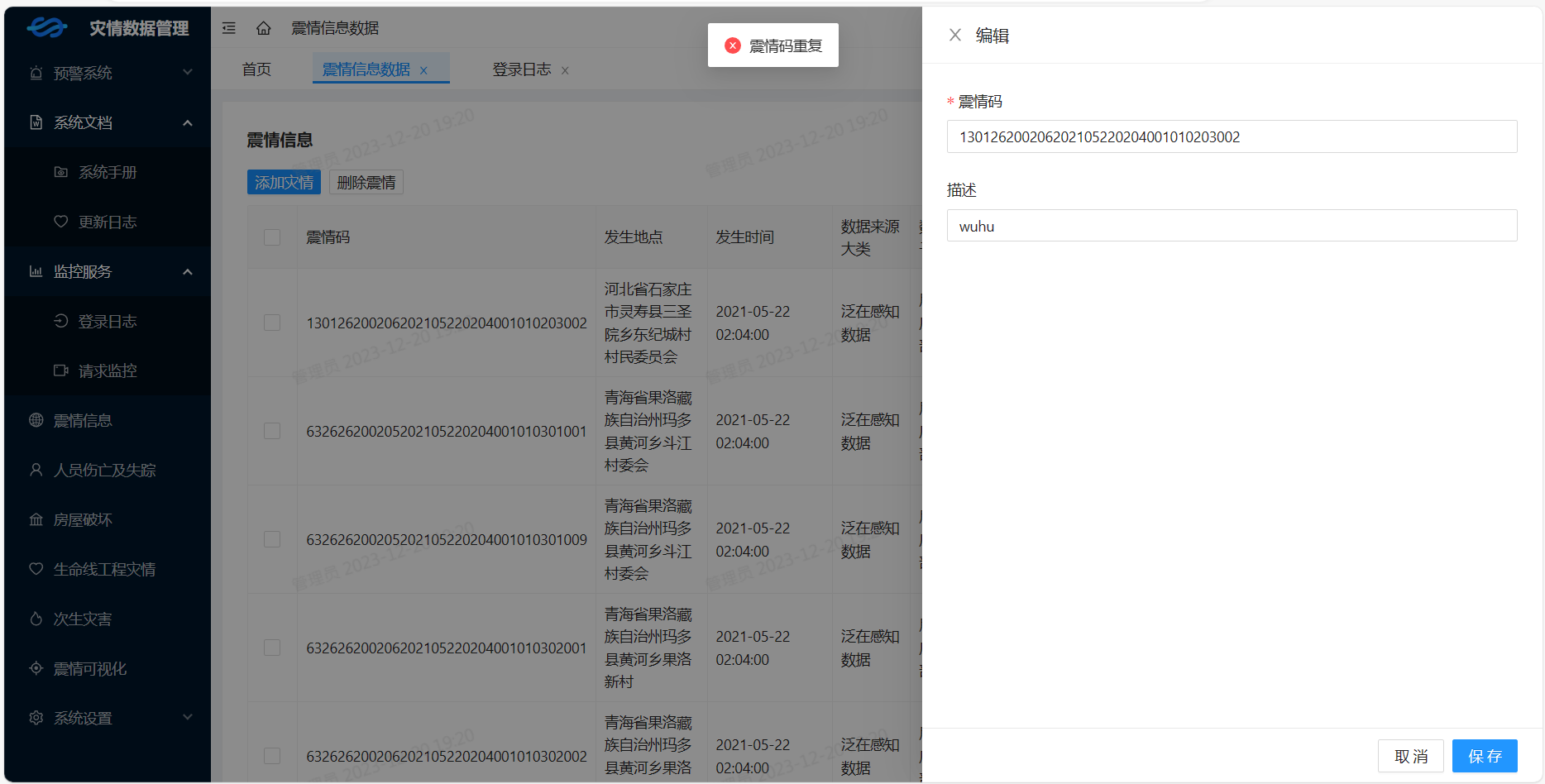


图6.11

## 6.4文件上传

1. 操作流程：在侧边框打开系统设置 -- 进入震情实况上传 -- 点击上传文件-- -- 选择需要上传的文件 -- 点击打开；
2. 页面正常显示，文件可以正常添加；
3. 预测结果：页面正常显示，文件正常的添加；
4. 实测结果：后端出现问题，文件无法正常上传，详见图6.12、图6.13、图6.14；
5. 错误结果：震情码重复，无法正常添加文件；

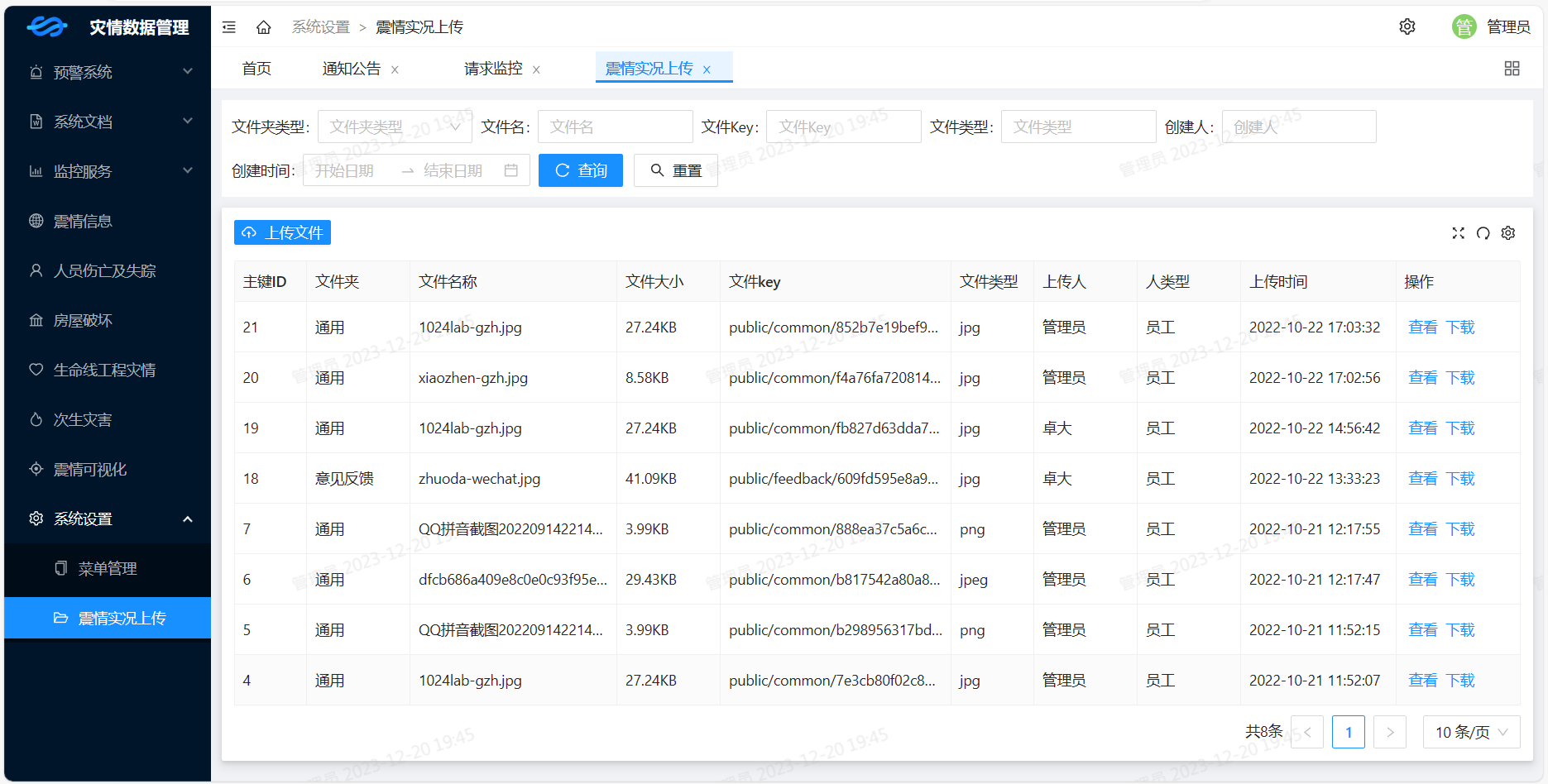


图6.12

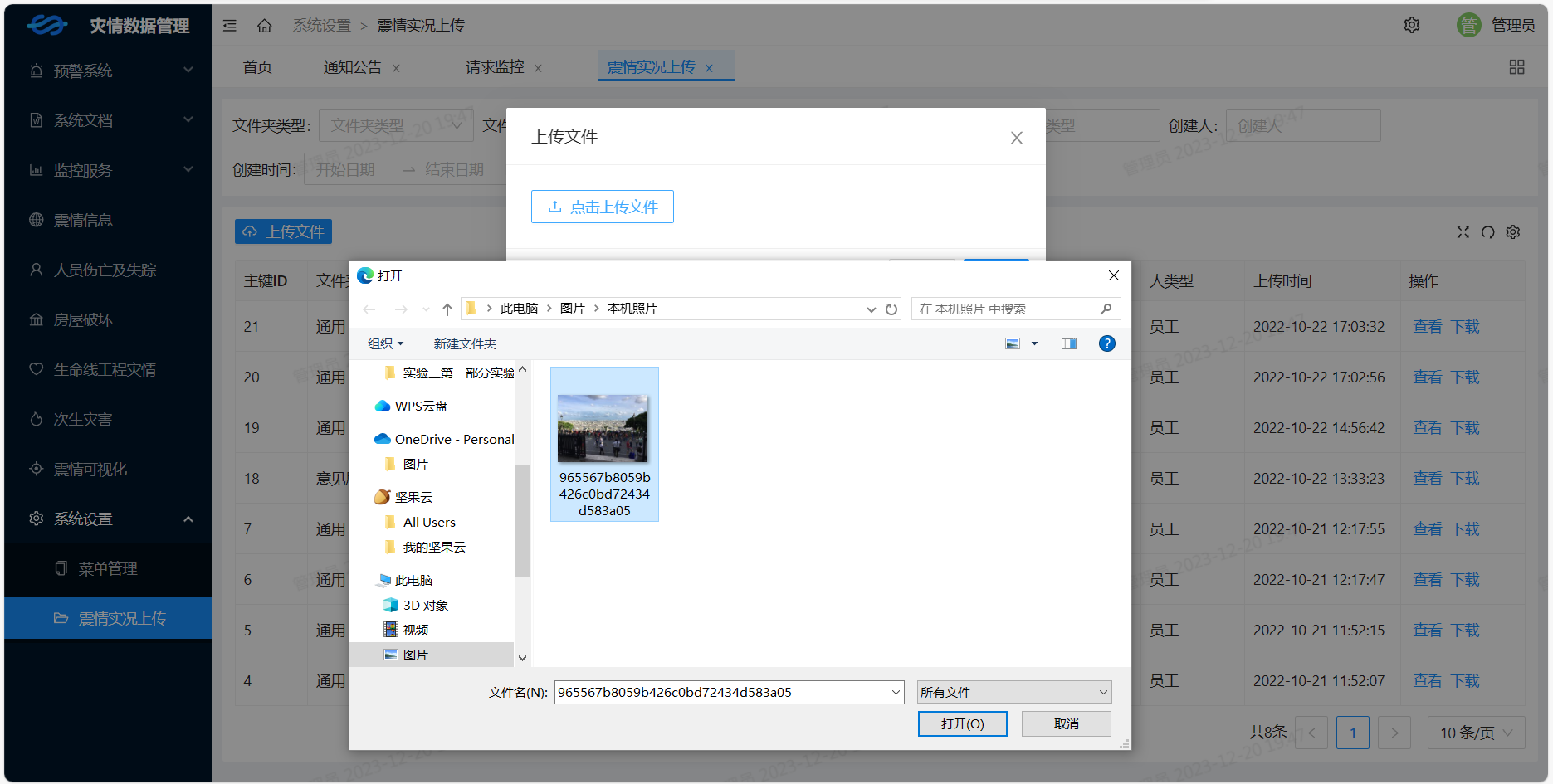


图6.13

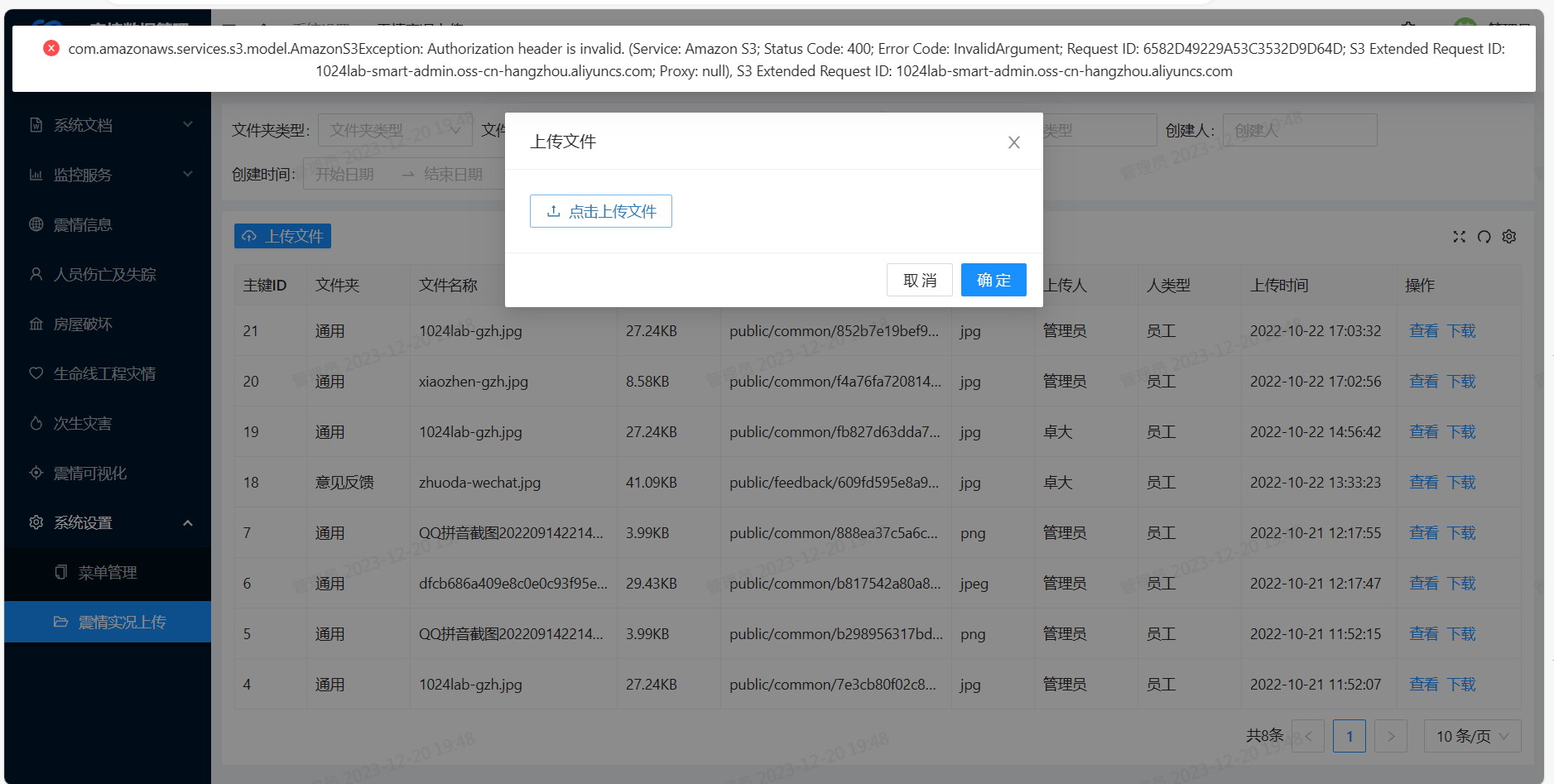


图6.14

# 7.测试结论

## 7.1测试报告总结：

本次测试旨在确保Web项目在不同层面具有高质量的可用性、性能和安全性。以下是对各个测试阶段的总结：

### 7.1.1单元测试：

目标： 通过对项目中各个独立单元的测试，验证其功能的正确性。

总结： 单元测试覆盖了项目中的关键组件和模块，所有测试用例均通过，说明项目的基础功能单元具有稳健性和正确性。

### 7.1.2接口测试：

目标： 验证系统内部不同组件之间的交互和数据传递的准确性。

总结： 接口测试覆盖了系统内各个关键接口，通过了所有测试用例。这确保了系统各个部分之间的协同工作和数据传递是可靠的。

### 7.1.3 UI测试：

目标： 确保用户界面的友好性、一致性和符合设计要求。

总结： UI测试涉及了窗口、导航、表单、交互元素等方面。所有测试用例通过，项目在各个屏幕尺寸和浏览器中都能正确呈现，用户体验良好。

### 7.1.4探索性测试：

目标： 发现潜在的用户体验问题、性能瓶颈、安全漏洞等。

总结： 探索性测试围绕窗口、导航、表单、动态内容等方面展开。通过对不同层面的测试，发现并解决了一些潜在的问题，提升了系统的整体质量。

## 7.2综合总结：

通过对单元测试、接口测试、UI测试和探索性测试的全面覆盖，我们验证了项目的各个方面的可靠性和稳健性。所有测试用例均通过，项目在功能、交互和性能方面表现出色。对发现的一些问题进行了及时修复和优化，确保了项目的高质量上线。

## 7.3建议和改进：

在将来的版本中，可以考虑引入更多的自动化测试，以加速测试流程。

对于用户体验方面，可以持续关注用户反馈，进一步优化界面和交互设计。

定期进行安全性审查，确保项目在面临潜在威胁时能够有效应对。

总体而言，项目通过了全面的测试，为用户提供了高品质的使用体验，各个方面都处于可控和可维护状态。