|  |  |  |  |
| --- | --- | --- | --- |
| 文 档 编 号 | | 产品版本 | 密级 |
|  | | 1.0 |  |
| 系统名称 | 济青中线智慧高速单元测试报告 | | 共 165 页 |

**济青中线智慧高速**

**单元测试报告**

**修订表**

**更改记录**

| **变更版本** | **日期** | **修改原因及修改情况** | **修订人** | **批准人** |
| --- | --- | --- | --- | --- |
| 1.0 | 2022-11-19 | 创建初版 | 卢强 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[一. 测试情况简述 6](#_Toc4296)

[1.1 编写目的 6](#_Toc20397)

[1.2 项目背景 6](#_Toc10764)

[1.3 参考资料 6](#_Toc16244)

[二. 测试环境 7](#_Toc5282)

[2.1. 技术架构描述 7](#_Toc3098)

[2.2. 测试环境描述 8](#_Toc23320)

[三. 功能测试 10](#_Toc16032)

[3.1. 隧道管理 10](#_Toc4209)

[3.2. 设备管理 11](#_Toc7158)

[3.2.1. 获取设备列表数据 11](#_Toc21486)

[3.2.2. 获取设备详细信息 12](#_Toc17443)

[3.2.3. 设备管理-生成id 13](#_Toc29723)

[3.2.4. 备品备件 14](#_Toc18333)

[3.2.5. 情报板管理发送记录 16](#_Toc13130)

[3.2.6. 情报板模板图片管理 17](#_Toc19543)

[3.2.7. 情报板敏感字管理 19](#_Toc228)

[3.2.8. 情报板模板管理 20](#_Toc26708)

[3.2.9. 获取单条情报板编辑信息 21](#_Toc22065)

[3.2.10. 获取单条情报板运行状态 22](#_Toc17855)

[3.2.11. 获取单条情报板实时信息 23](#_Toc6801)

[3.2.12. 获取单条情报板信息 25](#_Toc1242)

[3.2.13. 发布单条情报板信息 26](#_Toc21043)

[3.3. 机电巡查 27](#_Toc12861)

[3.3.1. 巡查任务 27](#_Toc13261)

[3.3.2. 查询故障清单列表 29](#_Toc7232)

[3.3.3. 新增故障清单数据 32](#_Toc25838)

[3.3.4. 修改故障清单数据 37](#_Toc26757)

[3.3.5. 查询故障检修记录 41](#_Toc22989)

[3.3.6. 新增巡查任务 43](#_Toc4859)

[3.3.7. 修改巡查任务 45](#_Toc31955)

[3.3.8. 查询关联巡检点详细信息 47](#_Toc2514)

[3.3.9. 构建前端所需树结构 49](#_Toc9008)

[3.3.10. 部门隧道节点信息处理 51](#_Toc30264)

[3.3.11. 递归列表 54](#_Toc16173)

[3.3.12. 构建部门隧道树节点数据 56](#_Toc32728)

[3.3.13. 查询任务详情 59](#_Toc1577)

[3.3.14. 查询任务列表 60](#_Toc2415)

[3.3.15. 删除任务列表 60](#_Toc12126)

[3.3.16. 删除故障清单 61](#_Toc5881)

[3.3.17. 查询设备列表 62](#_Toc23007)

[3.3.18. 获取故障列表 64](#_Toc21051)

[3.3.19. 查询班组信息 65](#_Toc23404)

[3.3.20. 查询隧道列表信息 66](#_Toc2235)

[3.3.21. 查询部门列表信息 67](#_Toc1812)

[3.4. 安全预警 68](#_Toc5737)

[3.4.1. 获取安全预警列表数据 68](#_Toc12733)

[3.4.2. 查询事件处理流程 69](#_Toc2682)

[3.4.3. 获取事件详情 71](#_Toc11269)

[3.4.4. 获取当日未处理事件总数 72](#_Toc78)

[3.4.5. 工作台-预警事件 73](#_Toc25631)

[3.4.6. 事件定位隧道分区 74](#_Toc13402)

[3.4.7. 批量删除事件 75](#_Toc14226)

[3.4.8. 事件弹窗当日事件 76](#_Toc23902)

[3.4.9. 应急调度一键恢复 78](#_Toc31756)

[3.4.10. 事件弹窗设备故障 81](#_Toc21632)

[3.5. 应急预案 82](#_Toc16287)

[3.5.1. 获取应急预案列表数据 82](#_Toc11003)

[3.5.2. 获取预案详细信息 85](#_Toc1067)

[3.5.3. 查询隧道分区可执行预案 87](#_Toc642)

[3.5.4. 删除应急预案 88](#_Toc5818)

[3.5.5. 根据预案查询关联策略信息 90](#_Toc25995)

[3.6. 联控策略 91](#_Toc27417)

[3.6.1. 获取联控策略列表数据 91](#_Toc1819)

[3.6.2. 获取联控策略详细信息 94](#_Toc25854)

[3.6.3. 工作台分时控制抽屉策略展示 95](#_Toc19917)

[3.6.4. 分时控制策略开启关闭 96](#_Toc874)

[3.6.5. 执行手动控制策略 97](#_Toc27684)

[3.6.6. 删除策略 98](#_Toc18487)

[3.7. 应急资源 100](#_Toc6253)

[3.7.1. 应急物资 100](#_Toc17643)

[3.7.2. 应急车辆 101](#_Toc18623)

[3.7.3. 应急人员 103](#_Toc16199)

[3.8. 操作日志 106](#_Toc15318)

[3.9. 监控大屏 108](#_Toc13098)

[3.9.1. 大屏-设备运行 108](#_Toc7638)

[3.9.2. 大屏-设备占比 109](#_Toc25800)

[3.9.3. 大屏-近30日隧道预警 111](#_Toc20551)

[3.9.4. 大屏-本月预警事件 112](#_Toc3385)

[3.9.5. 大屏-近12h控制记录 113](#_Toc22212)

[3.9.6. 大屏-今日报警统计 114](#_Toc15602)

[3.9.7. 大屏-本周报警统计 116](#_Toc13865)

[3.9.8. 大屏-本月报警趋势 117](#_Toc10926)

[3.9.9. 大屏-累计报警分析 118](#_Toc16977)

[3.9.10. 大屏-交通事件信息列表 120](#_Toc6847)

[3.9.11. 大屏-主动安全信息列表 121](#_Toc30368)

[3.9.12. 大屏-设备故障信息列表 122](#_Toc10734)

[3.9.13. 大屏-应急物资 124](#_Toc6116)

[3.9.14. 大屏-值班人员 125](#_Toc24654)

[3.9.15. 大屏-预案列表 126](#_Toc4704)

[3.10. 系统管理 128](#_Toc21420)

[3.10.1. 获取用户列表数据 128](#_Toc23796)

[3.10.2. 新增用户信息 129](#_Toc7813)

[3.10.3. 修改用户信息 130](#_Toc12994)

[3.10.4. 删除用户信息 131](#_Toc22444)

[3.10.5. 用户授权角色 132](#_Toc16064)

[3.10.6. 获取用户角色 133](#_Toc22523)

[3.10.7. 获取部门列表数据 134](#_Toc28950)

[3.10.8. 获取部门下拉树列表 135](#_Toc18125)

[3.10.9. 新增部门信息 136](#_Toc13376)

[3.10.10. 修改部门信息 137](#_Toc27413)

[3.10.11. 删除部门信息 139](#_Toc10299)

[3.10.12. 获取角色列表数据 140](#_Toc28230)

[3.10.13. 新增角色信息 141](#_Toc31835)

[3.10.14. 修改角色信息 143](#_Toc26479)

[3.10.15. 删除角色信息 145](#_Toc1134)

[3.10.16. 获取岗位列表数据 146](#_Toc30768)

[3.10.17. 新增岗位信息 148](#_Toc19546)

[3.10.18. 修改岗位信息 149](#_Toc21117)

[3.10.19. 删除岗位信息 151](#_Toc17760)

[3.11. 系统交互 152](#_Toc30021)

[3.11.1. 获取雷达感知数据列表 152](#_Toc17053)

[3.11.2. 获取设备状态数据集合 153](#_Toc8050)

[3.11.3. 获取雷达感知数据 154](#_Toc26251)

[3.11.4. 获取雷达设备运行状态数据 155](#_Toc13287)

[3.11.5. 万集设备运行状态数据发送 156](#_Toc29068)

[3.11.6. 物联中台指令数据模拟发送 157](#_Toc24160)

[3.11.7. 物联中台事件数据模拟发送 158](#_Toc8645)

[3.11.8. 物联中台设备数据模拟发送 159](#_Toc2148)

[3.11.9. 物联中台发送状态数据模拟发送 160](#_Toc5841)

[3.11.10. 接收设备状态数据测试 161](#_Toc26273)

[3.11.11. 接收事件数据测试 162](#_Toc22)

# 测试情况简述

## 编写目的

本文档编写的目的是描述济青中线平台性能测试总结性文件，阐述整个性能测试过程及测试结果，本文档的读者包括但不限于：监理人员、评审专家、业务专家、技术专家、项目建设的相关领导、项目经理、项目组开发成员、测试工程师等。

## 项目背景

“十四五”时期是加强交通强省建设、推动交通运输业高质量发展的重要阶段，随着“数字交通”“交通强国”等交通行业宏观政策指导文件的陆续发布，高速公路运营企业管理模式智慧化转型升级已迫在眉睫。在国家政策大力扶持、科学技术飞速前进的大背景下, 全省交通运输业将逐步向数据赋能的智慧交通方向迈进，其中一个重要方面就是加快智慧高速建设。智慧高速的建设正从小规模实验验证阶段迈向大规模部署阶段，智慧高速的建设已成为未来发展的必然趋势。

济南至潍坊高速公路项目，项目全长162公里；潍坊至青岛连接线项目，项目全长47公里。拟对上述两个项目（合称“济青中线智慧高速项目”）实施智慧高速专项软件项目建设，依托“高速云”，基于京台智慧高速软件和集团已有信息化基础，打造济青中线智慧高速软件管理平台。

济青中线智慧高速作为山东省高速公路网规划布局“九纵五横一环七射多连”的“射三”线，项目自西向东先后连接济南绕城高速二环线东环段、滨莱、长深、潍日高速，通过潍日高速潍坊连接线和潍莱高速，进而连接青银，是连接省会都市圈和胶东都市圈的重要干线，从而也形成济南至青岛的中部通道，项目实施对于提高国家运输主通道的通行能力、加快山东半岛城市群建设、完善高速公路网布局、提升路网抗风险能力、强化济南与青岛和烟台的联系、促进区域经济快速发展和旅游资源开发，为新旧动能转换提供支撑和保障等具有重要作用。

智慧高速公路是为满足日益增长的交通需求，通过智能化的传感、通讯、人工智能研判等手段而大幅提高高速公路通行能力，满足新一代驾驶工具应用要求的全新技术形态。

## 参考资料

GB/T 16260.1 -2006 《软件工程 产品质量 第一部分：质量模型》

GB/T 18905.5 - 2002 《软件工程 产品评价第5部分 评价者用的过程》

GB/T 17544 – 1998 《信息技术 软件质量要求和测试》

《GS-TEST-S-10 GS产品线性能标准（试行）》

GB/T 8567-2006 《计算机软件文档编制规范》

GB/T 9386-2008 《计算机软件测试文档编制规范》

《济青中线智慧高速软件平台建设项目合同》

《智慧高速示范软件管理平台建设项目实施方案》

《济青中线智慧高速软件平台需求规格说明书》

《济青中线智慧高速软件平台详细设计说明书》

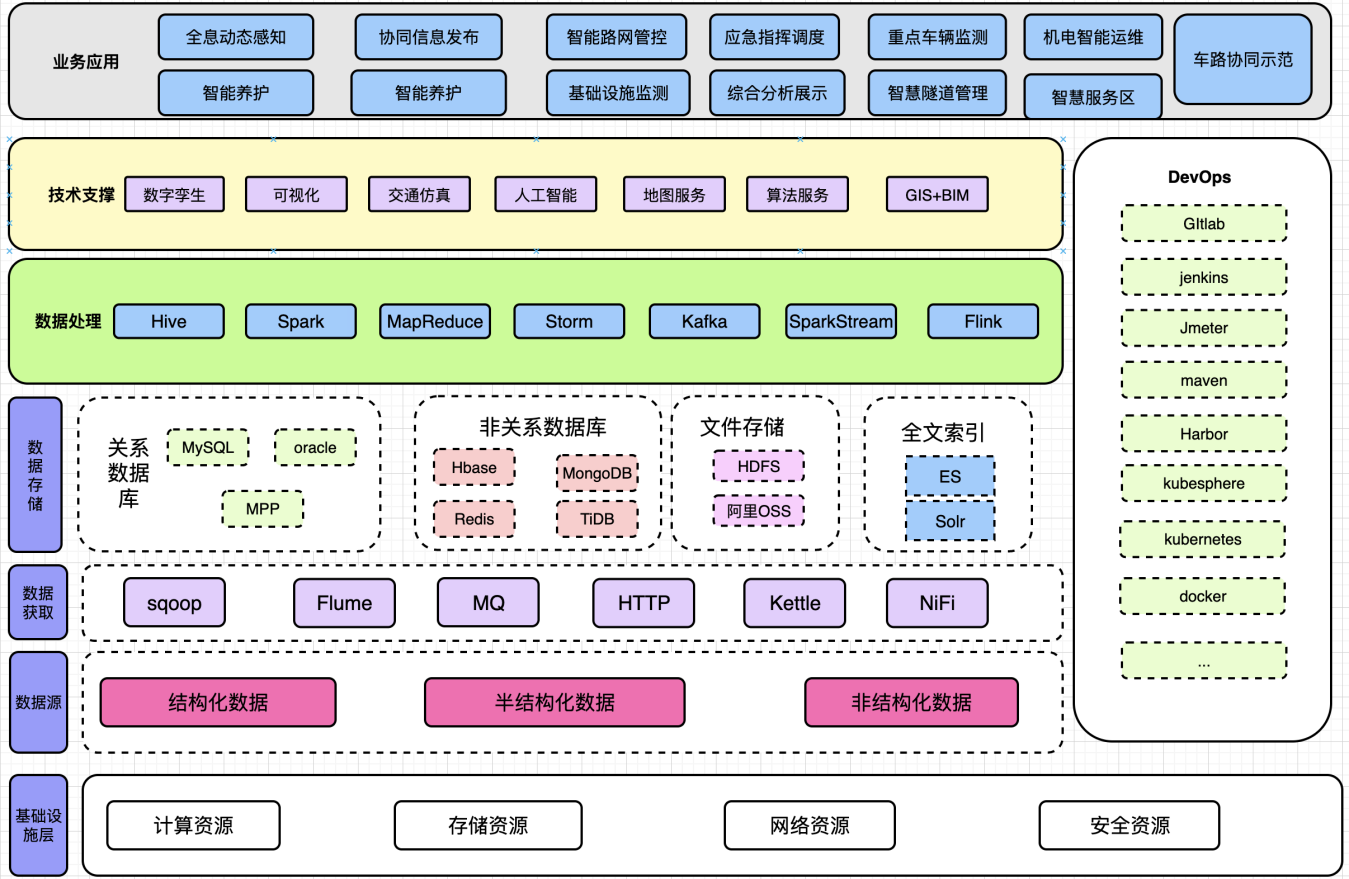
《济青中线智慧高速软件平台概要设计说明书》

《济青中线智慧高速软件平台设计原型》

# 测试环境

## 技术架构描述

济青中线智慧高速整体架构采用云原生架构设计，构建运行平台，基于容器化技术的灵活伸缩能力，利用云原生技术，以容器为基础，进一步简化云端服务的维护，将容器内应用程序和进程作为独立部署单元，从而进一步提高水平资源隔离。同时借助动态管理，通过集中式的编排调度系统来动态的管理和调度服务。整体济青中线智慧高速技术架构由IaaS（基础设施层）、PaaS（平台服务层）、SaaS（应用服务层）、CaaS（容器服务层）组成，技术架构图如下



## 测试环境描述

网络环境：LAN（100M）

硬件、软件环境：

1. 应用服务器

型号：高速云虚拟服务器

CPU：16核

内存：64G

硬盘：500G

操作系统：centos\_7\_06\_x64

软件环境：tomcat9.0 Nginx1.19

1. 数据库服务器

型号：高速云虚拟服务器

CPU：16核

内存：64G

硬盘：100+500G

操作系统：centos\_7\_06\_x64

软件环境：MySQL8.0

1. 测试客户端（PC）

数量：5-8台

型号：HP Laptop

CPU：8核

内存：16G

硬盘：900G

操作系统：Win7以上

软件环境：谷歌浏览器、Jmeter5.0、Fiddler5.0、Office2019

# 功能测试

## 隧道管理

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取隧道列表数据 | | | | 测试用例 | | JQ-TP-TM-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取隧道列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdTunnelsService sdTunnelsService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdTunnels tunnel = new SdTunnels();  tunnel.setTunnelName("凤凰");  tunnel.setPoll(0L);  List<SdTunnels> list = sdTunnelsService.selectSdTunnelsList(tunnel);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 设备管理

### 获取设备列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备列表数据 | | | | 测试用例 | | JQ-TP-EM-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdDevicesService sdDevicesService;  @Test  public void testEquipment() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdDevices sdDevices = new SdDevices();  sdDevices.setDeptId(100L);  List<SdDevices> devicesList = sdDevicesService.selectSdDevicesList(sdDevices);  String jsonStr = JSONUtil.toJsonStr(devicesList);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取设备详细信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备详细信息数据 | | | | 测试用例 | | JQ-TP-EM-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备详细信息数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdDevicesService sdDevicesService;  /\*\*  \* 获取设备详细信息  \*/  @Test  public void getInfo() {  String eqId = "WLJD-JiNan-YanJiuYuan-FHS-COVI-001";  SdDevices sd = sdDevicesService.selectSdDevicesById(eqId);  String jsonStr = JSONObject.toJSONString(sd);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 设备管理-生成id

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 生成id | | | | 测试用例 | | JQ-TP-EM-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 生成id | | | | | | | |
| 测试用例 | @Autowired  private ISdDevicesService sdDevicesService;  /\*\*  \* 生成id  \* @param  \* @return  \*/  @Test  public void autoId(){  String tunnelId = "WLJD-JiNan-YanJiuYuan-FHS";  Integer typeId = 3;  String tunnelId1 = sdDevicesService.autoId(tunnelId, typeId);  System.out.println(tunnelId1);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 备品备件

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取备品备件列表数据 | | | | 测试用例 | | JQ-TP-EM-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取备品备件列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdSparePartsWarehouseService sdSparePartsWarehouseService;  @Test  public void testSpareParts() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);    SdSparePartsWarehouse sdSparePartsWarehouse = new SdSparePartsWarehouse();  sdSparePartsWarehouse.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdSparePartsWarehouse.setPartName("灭火器");    List<SdSparePartsWarehouse> list = sdSparePartsWarehouseService.selectSdSparePartsWarehouseList(sdSparePartsWarehouse);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 情报板管理发送记录

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取发送记录数据 | | | | 测试用例 | | JQ-TP-EM-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取发送记录数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdReleaseRecordService sdReleaseRecordService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdReleaseRecord sdReleaseRecord = new SdReleaseRecord();  List<SdReleaseRecord> list = sdReleaseRecordService.selectSdReleaseRecordList(sdReleaseRecord);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 情报板模板图片管理

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取情报板模板图片数据 | | | | 测试用例 | | JQ-TP-EM-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取情报板模板图片数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdPictureUploadService sdPictureUploadService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdPictureUpload sdPictureUpload = new SdPictureUpload();  List<SdPictureUpload> list = sdPictureUploadService.selectSdPictureUploadList(sdPictureUpload);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 情报板敏感字管理

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取情报板敏感字数据 | | | | 测试用例 | | JQ-TP-EM-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取情报板敏感字数据 | | | | | | | |
| 测试用例 | @Autowired  private IIotBoardVocabularyService iotBoardVocabularyService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  IotBoardVocabulary sdAuditVocabulary = new IotBoardVocabulary();  List<IotBoardVocabulary> list = iotBoardVocabularyService.selectIotBoardVocabularyList(sdAuditVocabulary);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 情报板模板管理

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取情报板模板管理数据 | | | | 测试用例 | | JQ-TP-EM-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取情报板模板管理数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdVmsTemplateService sdVmsTemplateService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdVmsTemplate sdVmsTemplate = new SdVmsTemplate();  List<SdVmsTemplate> list = sdVmsTemplateService.selectSdVmsTemplateList(sdVmsTemplate);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取单条情报板编辑信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取单条情报板编辑信息 | | | | 测试用例 | | JQ-TP-EM-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取单条情报板编辑信息 | | | | | | | |
| 测试用例 | @GetMapping("/getNewBoardEditInfo")  public AjaxResult getNewBoardEditInfo(String deviceId) {  AjaxResult ajaxResult = new AjaxResult();  List<String> paramsList = new ArrayList<String>();  SdDevices sd = getDevicesInfo(deviceId);  try {  // 1.获取设备状态  Map<String, String> displayList = DeviceManagerFactory.getInstance().getDeviceDisplayListByDeviceId(deviceId);  String result = displayList.get("result");  String protocolType = displayList.get("vender");  String jsonContent = DataUtils.itemContentToJson(result, protocolType);  releaseContentMap.put(deviceId.toString(), result);  JSONArray contentObject = JSONArray.parseArray(jsonContent);  JSONObject items = new JSONObject();  items.put("content", contentObject);  items.put("support", DataUtils.getSupport(String.valueOf(deviceId), protocolType));  paramsList.add(items.toString());  ajaxResult = new AjaxResult(HttpStatus.SUCCESS, "返回成功", paramsList);  } catch (Exception e) {  return AjaxResult.error(-1, e.getMessage());  }  return ajaxResult;  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取单条情报板运行状态

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取单条情报板运行状态 | | | | 测试用例 | | JQ-TP-EM-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取单条情报板运行状态 | | | | | | | |
| 测试用例 | @Test  public void getIcyData() {    List<SdIotDevice> list = sdIotDeviceService.selectIotDeviceList(new SdIotDevice());  for(int i=0;i<list.size();i++){  SdIotDevice iotDevice = list.get(i);  try {  DeviceManagerFactory.getInstance().getDeviceDisplayListByDeviceId(String.valueOf(iotDevice.getDeviceId()));  iotDevice.setDeviceStatus("0");  sdIotDeviceService.updateIotDevice(iotDevice);  } catch (Exception e) {  iotDevice.setDeviceStatus("1");  sdIotDeviceService.updateIotDevice(iotDevice);  }  }    } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取单条情报板实时信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取单条情报板实时信息 | | | | 测试用例 | | JQ-TP-EM-011 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取单条情报板实时信息 | | | | | | | |
| 测试用例 | @GetMapping("/getBoardContent")  public AjaxResult loadRealtimeInf(Long deviceId) {  List<String> paramsList = new ArrayList<String>();  AjaxResult ajaxResult;  try {  Map<String, String> displayList = DeviceManagerFactory.getInstance().getDeviceDisplayListByDeviceId(String.valueOf(deviceId));  String result = displayList.get("result");  String protocolType = displayList.get("vender");  String jsonResult = DataUtils.itemContentToJson(result, protocolType);  JSONObject items = new JSONObject();  JSONArray resultObj = JSONArray.parseArray(jsonResult);  items.put("content", resultObj);  paramsList.add(items.toString());  ajaxResult = new AjaxResult(HttpStatus.SUCCESS, "返回成功", paramsList);  } catch (Exception e) {  return AjaxResult.error(-1, e.getMessage());  }  return ajaxResult;  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取单条情报板信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取单条情报板信息 | | | | 测试用例 | | JQ-TP-EM-012 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取单条情报板信息 | | | | | | | |
| 测试用例 | @GetMapping("/getBoardInfo")  public AjaxResult getBoardInfo(Long deviceId){  SdIotDevice iotDevice = sdIotDeviceService.selectIotDeviceById(deviceId);  Map map = sdIotDeviceService.selectIotDeviceAccessById(deviceId);  Map<String,Object> returnMap = new HashMap<String,Object>();  returnMap.put("deviceId", iotDevice.getDeviceId());  returnMap.put("deviceName", iotDevice.getDeviceName());  returnMap.put("manageAgencyId", iotDevice.getManageAgencyId());  returnMap.put("pileNumber", iotDevice.getPileNumber());  returnMap.put("route", iotDevice.getRouteId());  returnMap.put("firmId", iotDevice.getFirmId());  returnMap.put("ip", map.get("device\_ip"));  returnMap.put("port", map.get("device\_port"));  returnMap.put("pixel", map.get("device\_pixel"));  return AjaxResult.success(returnMap);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 发布单条情报板信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 发布单条情报板信息 | | | | 测试用例 | | JQ-TP-EM-013 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 发布单条情报板信息 | | | | | | | |
| 测试用例 | @GetMapping("/uploadBoardEditInfo")  public AjaxResult uploadBoardEditInfo(String deviceId, String protocolType,String parameters) {  AjaxResult ajaxResult = new AjaxResult();  List<String> paramsList = new ArrayList<String>();  try {  parameters = URLDecoder.decode(parameters, "UTF-8");  parameters = parameters.replaceAll("—", "-");  String commands = DataUtils.contentToGb2312\_CG(deviceId, parameters, protocolType);  Boolean result = DeviceManagerFactory.getInstance().controlDeviceByDeviceId(deviceId, protocolType, commands);  if (result) {  if (protocolType.startsWith(IDeviceProtocol.XIANKE)) {  String XKcommands = "02 32 32 30 30 30 30 30 2E 78 6B 6C 7A 93 03";  result = DeviceManagerFactory.getInstance().controlDeviceByDeviceId(deviceId, protocolType, XKcommands);  if (result) {  ajaxResult = new AjaxResult(HttpStatus.SUCCESS, "修改成功");  } else {  ajaxResult = new AjaxResult(HttpStatus.ERROR, "修改失败");  }  } else {  ajaxResult = new AjaxResult(HttpStatus.SUCCESS, "修改成功");  }  } else {  ajaxResult = new AjaxResult(HttpStatus.ERROR, "修改失败");  }  } catch (Exception e) {  ajaxResult = new AjaxResult(HttpStatus.ERROR, "系统异常");  }  return ajaxResult;  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 机电巡查

### 巡查任务

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取巡查任务列表数据 | | | | 测试用例 | | JQ-TP-EI-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取巡查任务列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdTaskListService sdTaskListService;  @Test  public void testSdTask() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdTaskList sdTaskList = new SdTaskList();  sdTaskList.setZzjgId("fhs");  List<SdTaskList> list = sdTaskListService.selectSdTaskListList(sdTaskList);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询故障清单列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取故障清单数据 | | | | 测试用例 | | JQ-TP-EI-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @Override  public SdFaultList selectSdFaultListById(String id)  {  SdFaultList sdFaultList = sdFaultListMapper.selectSdFaultListById(id);  if(sdFaultList!=null){ if(sdFaultList.getImgFileId()!=null&&!"".equals(sdFaultList.getImgFileId())){  SdTrafficImage sdTrafficImage = new SdTrafficImage();  sdTrafficImage.setBusinessId(sdFaultList.getImgFileId());  sdFaultList.setiFileList(sdTrafficImageMapper.selectFaultImgFileList(sdTrafficImage));  }  }  return sdFaultList;  }  @Override public List<SdFaultList> selectSdFaultListList(SdFaultList sdFaultList) {  List<SdFaultList> list = sdFaultListMapper.selectSdFaultListList(sdFaultList);  if(list!=null&&list.size()>0){  for(int i = 0;i<list.size();i++){  if(list.get(i).getFalltRemoveStatue()!=null){  int removeflag = Integer.valueOf(list.get(i).getFalltRemoveStatue());  if(1==removeflag){//未消除 根据当前时间与故障发现时间计算时间差，单位：天、小时  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");  String fxtime = sdf.format(list.get(0).getFaultFxtime());  long time = sdf.parse(fxtime, new ParsePosition(0)).getTime();  long nd = 1000 \* 24 \* 60 \* 60;  long nh = 1000 \* 60 \* 60;  long nm = 1000 \* 60;  long ns = 1000;  // 获得两个时间的毫秒时间差异  long diff = System.currentTimeMillis() - time + 1000;  // 计算差多少天  long day = diff / nd;  // 计算差多少小时  long hour = diff % nd / nh;  // 计算差多少分钟  long min = diff % nd % nh / nm;  // 计算差多少秒//输出结果  long sec = diff % nd % nh % nm / ns;  System.out.println(day + "天" + hour + "小时" + min + "分钟" + sec + "秒");  list.get(i).setFaultCxtime(day + "天" + hour + "小时");  }  }   }   }  return list; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增故障清单数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增故障清单数据 | | | | 测试用例 | | JQ-TP-EI-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 新增故障清单 | | | | | | | |
| 测试用例 | @Override public int insertSdFaultList(MultipartFile[] file, SdFaultList sdFaultList) {  int result = -1;  List<SdTrafficImage> list = new ArrayList<SdTrafficImage>();  try {  sdFaultList.setCreateTime(DateUtils.*getNowDate*());*// 创建时间* sdFaultList.setCreateBy(SecurityUtils.*getUsername*());*// 设置当前创建人* if (file.length > 0) {  String guid = UUIDUtil.*getRandom32BeginTimePK*();*// 生成guid* sdFaultList.setImgFileId(guid);*// 文件关联ID* for (int i = 0; i < file.length; i++) {  *// 图片Base64* String imageBaseStr = null;  try {  String contentType = file[i].getContentType();  if (!contentType.contains("image")) {  throw new RuntimeException("文件类型不正确!");  }  byte[] imageBytes = file[i].getBytes();  BASE64Encoder base64Encoder = new BASE64Encoder();  imageBaseStr = "data:" + contentType + ";base64," + base64Encoder.encode(imageBytes);  imageBaseStr = imageBaseStr.replaceAll("[\\s\*\t\n\r]", "");  } catch (IOException e) {  throw new RuntimeException("图片转换base64异常");  }  *// 从缓存中获取文件存储路径* String fileServerPath = RuoYiConfig.*getUploadPath*();  *// 原图文件名* String filename = file[i].getOriginalFilename();  *// 原图扩展名* String extendName = filename.substring(filename.lastIndexOf("\\") + 1);  *// 新的全名* String fileName = extendName;  *// 加路径全名* File dir = new File(fileServerPath + "/faultIcon/" + fileName);  File filepath = new File(fileServerPath + "/faultIcon");   SdTrafficImage iconFile = new SdTrafficImage();  iconFile.setBusinessId(guid);  iconFile.setImgUrl(imageBaseStr);  iconFile.setImgName(fileName);  iconFile.setCreateBy(SecurityUtils.*getUsername*());  iconFile.setCreateTime(DateUtils.*getNowDate*());  list.add(iconFile);   if (!filepath.exists()) {  filepath.mkdirs();  } else {  }  file[i].transferTo(dir);  }  result = sdTrafficImageMapper.brachInsertFaultIconFile(list);  if (result > -1) {  result = sdFaultListMapper.insertSdFaultList(sdFaultList);  }   } else {  sdFaultList.setImgFileId(null);*// 图标文件ID* result = sdFaultListMapper.insertSdFaultList(sdFaultList);  }   } catch (IOException e) {  e.printStackTrace();  return 0;  }  return result; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改故障清单数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改故障清单数据 | | | | 测试用例 | | JQ-TP-EI-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 修改故障清单  \*   \* @param sdFaultList 故障清单  \* @return 结果  \*/* @Override public int updateSdFaultList(MultipartFile[] file,SdFaultList sdFaultList,Long[] removeIds) {  int result = 0;  List<SdTrafficImage> list = new ArrayList<SdTrafficImage>();  try {  sdFaultList.setUpdateTime(DateUtils.*getNowDate*());*// 创建时间* sdFaultList.setUpdateBy(SecurityUtils.*getUsername*());*// 设置当前创建人* String guid = sdFaultList.getImgFileId();*// 关联ID--guid* if (guid == null || guid.equals("null") || guid.equals("")) {  guid = UUIDUtil.*getRandom32BeginTimePK*();*// 生成guid* }  sdFaultList.setImgFileId(guid);*// 文件关联ID* if (file != null && file.length > 0) {  for (int i = 0; i < file.length; i++) {  *// 图片Base64* String imageBaseStr = null;  try {  String contentType = file[i].getContentType();  if (!contentType.contains("image")) {  throw new RuntimeException("文件类型不正确!");  }  byte[] imageBytes = file[i].getBytes();  BASE64Encoder base64Encoder = new BASE64Encoder();  imageBaseStr = "data:" + contentType + ";base64," + base64Encoder.encode(imageBytes);  imageBaseStr = imageBaseStr.replaceAll("[\\s\*\t\n\r]", "");  } catch (IOException e) {  throw new RuntimeException("图片转换base64异常");  }  *// 从缓存中获取文件存储路径* String fileServerPath = RuoYiConfig.*getUploadPath*();  *// 原图文件名* String filename = file[i].getOriginalFilename();  *// 原图扩展名* String extendName = filename.substring(filename.lastIndexOf("\\") + 1);  *// 新的全名* String fileName = extendName;  *// 加路径全名* File dir = new File(fileServerPath + "/faultIcon/" + fileName);  File filepath = new File(fileServerPath + "/faultIcon");   SdTrafficImage iconFile = new SdTrafficImage();  iconFile.setBusinessId(guid);  iconFile.setImgUrl(imageBaseStr);  iconFile.setImgName(fileName);  iconFile.setCreateBy(SecurityUtils.*getUsername*());  iconFile.setCreateTime(DateUtils.*getNowDate*());  list.add(iconFile);   if (!filepath.exists()) {  filepath.mkdirs();  } else {  }  file[i].transferTo(dir);  }  result = sdTrafficImageMapper.brachInsertFaultIconFile(list);  }  if (removeIds.length > 0) {  result = sdTrafficImageMapper.deleteFaultIconFileByIds(removeIds);*//ids 为要删除的sd\_traffic\_image img\_id数组* }  if (result >= 0) {  result = sdFaultListMapper.updateSdFaultList(sdFaultList);  }  } catch (IOException e) {  e.printStackTrace();  return 0;  }  return result;} | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询故障检修记录

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询故障检修记录 | | | | 测试用例 | | JQ-TP-EI-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 查询故障检修记录  \* @param faultId  \* @return  \*/* @Override public List<SdPatrolList> getFaultRepairInfo(String faultId) {  List<SdPatrolList> patrolList = sdFaultListMapper.getFaultRepairInfo(faultId);  if(patrolList!=null&&patrolList.size()>0){  String fileId = patrolList.get(0).getImgFileId();  if (fileId != null && !"".equals(fileId) && !"null".equals(fileId)) {  SdTrafficImage sdTrafficImage = new SdTrafficImage();  sdTrafficImage.setBusinessId(fileId);  patrolList.get(0).setiFileList(sdTrafficImageMapper.selectFaultImgFileList(sdTrafficImage));  }  }   return patrolList; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增巡查任务

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增巡查任务 | | | | 测试用例 | | JQ-TP-EI-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 新增巡查任务  \*  \* @param sdTaskList 巡查任务  \* @return 结果  \*/* @Override public int insertSdTaskList(SdTaskList sdTaskList, List<SdPatrolList>sdPatrolList) {  int flag = -1;  sdTaskList.setId(UUIDUtil.*getRandom32BeginTimePK*());  sdTaskList.setZzjgId(String.*valueOf*(SecurityUtils.*getDeptId*()));  sdTaskList.setCreateTime(DateUtils.*getNowDate*());  flag = sdTaskListMapper.insertSdTaskList(sdTaskList);*//添加巡查任务* if(flag==1){*//添加巡查点* flag = sdPatrolListMapper.batchInsertPatrolList(sdPatrolList,sdTaskList.getId());  }  return flag; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改巡查任务

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改巡查任务 | | | | 测试用例 | | JQ-TP-EI-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 修改巡查任务  \*  \* @param sdTaskList 巡查任务  \* @return 结果  \*/* @Override public int updateSdTaskList(SdTaskList sdTaskList,List<SdPatrolList>sdPatrolList) {  int result = 0;  sdTaskList.setUpdateTime(DateUtils.*getNowDate*());  result = sdTaskListMapper.updateSdTaskList(sdTaskList);  if(result>0){  result = sdPatrolListMapper.batchDeletePatrolListByTaskId(sdTaskList.getId());  }  if(result>0){  result = sdPatrolListMapper.batchInsertPatrolList(sdPatrolList,sdTaskList.getId());  }  return result; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询关联巡检点详细信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询关联巡检点详细信息 | | | | 测试用例 | | JQ-TP-EI-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 根据任务id查询关联巡检点详细信息  \* @param task\_id  \* @return  \*/* @Override public List<SdPatrolList> getPatrolListsInfo(String task\_id) {   List<SdPatrolList> sdPatrolList = sdPatrolListMapper.getPatrolListsInfo(task\_id);  if(sdPatrolList!=null && sdPatrolList.size()>=0){  for(int i =0;i<sdPatrolList.size();i++){  String fileId = sdPatrolList.get(i).getImgFileId();  if (fileId != null && !"".equals(fileId) && !"null".equals(fileId)) {  SdTrafficImage sdTrafficImage = new SdTrafficImage();  sdTrafficImage.setBusinessId(fileId);  sdPatrolList.get(i).setiFileList(sdTrafficImageMapper.selectFaultImgFileList(sdTrafficImage));  }  }  }  return sdPatrolList; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 构建前端所需树结构

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 构建前端隧道树结构 | | | | 测试用例 | | JQ-TP-EI-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 构建前端所需要树结构  \*  \* @param deptTunnels 隧道部门列表  \* @return 树结构列表  \*/* public List<SysDeptTunnel> buildDeptTree(List<SysDeptTunnel> deptTunnels) {  List<SysDeptTunnel> returnList = new ArrayList<SysDeptTunnel>();  List<String> tempList = new ArrayList();  for (SysDeptTunnel dept : deptTunnels)  {  tempList.add(dept.getDeptId());  }  for (Iterator<SysDeptTunnel> iterator = deptTunnels.iterator(); iterator.hasNext();)  {  SysDeptTunnel dept = (SysDeptTunnel) iterator.next();  *// 如果是顶级节点, 遍历该父节点的所有子节点* if (!tempList.contains(dept.getParentId()))  {  recursionFn(deptTunnels, dept);  returnList.add(dept);  }  }  if (returnList.isEmpty())  {  returnList = deptTunnels;  }  return returnList; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 部门隧道节点信息处理

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 部门隧道节点信息处理 | | | | 测试用例 | | JQ-TP-EI-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 构建前端所需要树结构  \*  \* @param deptTunnels 隧道部门列表  \* @return 树结构列表  \*/* public List<SysDeptTunnel> buildDeptTree(List<SysDeptTunnel> deptTunnels) {  List<SysDeptTunnel> returnList = new ArrayList<SysDeptTunnel>();  List<String> tempList = new ArrayList();  for (SysDeptTunnel dept : deptTunnels)  {  tempList.add(dept.getDeptId());  }  for (Iterator<SysDeptTunnel> iterator = deptTunnels.iterator(); iterator.hasNext();)  {  SysDeptTunnel dept = (SysDeptTunnel) iterator.next();  *// 如果是顶级节点, 遍历该父节点的所有子节点* if (!tempList.contains(dept.getParentId()))  {  recursionFn(deptTunnels, dept);  returnList.add(dept);  }  }  if (returnList.isEmpty())  {  returnList = deptTunnels;  }  return returnList; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 递归列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 递归节点列表 | | | | 测试用例 | | JQ-TP-EI-011 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 递归列表  \*/* private void recursionFn(List<SysDeptTunnel> list, SysDeptTunnel t) {  *// 得到子节点列表* List<SysDeptTunnel> childList = getChildList(list, t);  t.setChildren(childList);  for (SysDeptTunnel tChild : childList)  {  if (hasChild(list, tChild))  {  recursionFn(list, tChild);  }  } }   */\*\*  \* 得到子节点列表  \*/* private List<SysDeptTunnel> getChildList(List<SysDeptTunnel> list, SysDeptTunnel t) {  List<SysDeptTunnel> tlist = new ArrayList<SysDeptTunnel>();  Iterator<SysDeptTunnel> it = list.iterator();  while (it.hasNext())  {  SysDeptTunnel n = (SysDeptTunnel) it.next();  if (StringUtils.*isNotNull*(n.getParentId()) && n.getParentId().equals(t.getDeptId()) )  {  tlist.add(n);  }  }  return tlist; } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 构建部门隧道树节点数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 构建部门隧道树节点数据 | | | | 测试用例 | | JQ-TP-EI-012 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @GetMapping("/treeselect") @ApiOperation("获取隧道树形结构") public Result treeselect() {  String deptId = String.*valueOf*(SecurityUtils.*getDeptId*());  if (deptId == null) {  throw new RuntimeException("当前账号没有配置所属部门，请联系管理员进行配置！");  }  *//List<SysDept> depts = deptService.selectTunnelDeptList(deptId);* List<SysDeptTunnel>deptTunnels = new ArrayList<>();   List<SysDept> depts = deptService.selectTunnelDeptList(deptId);  List<SdTunnels> tunnels = tunnelsService.selectTunnelList(deptId);  if(depts!=null&&depts.size()>0){  for(int i = 0;i<depts.size();i++){  SysDeptTunnel deptTunnel = new SysDeptTunnel();  deptTunnel.setDeptId((depts.get(i).getDeptId()).toString());  deptTunnel.setDeptName(depts.get(i).getDeptName());  deptTunnel.setParentId((depts.get(i).getParentId()).toString());  deptTunnel.setParentName(depts.get(i).getParentName());  deptTunnels.add(deptTunnel);  }  }  if(tunnels!=null&&tunnels.size()>0){  for(int i = 0;i<tunnels.size();i++){  SysDeptTunnel deptTunnel = new SysDeptTunnel();  deptTunnel.setDeptId(tunnels.get(i).getTunnelId());  deptTunnel.setDeptName(tunnels.get(i).getTunnelName());  deptTunnel.setParentId((tunnels.get(i).getDeptId()).toString());  deptTunnel.setParentName(tunnels.get(i).getTunnelStationName());  deptTunnels.add(deptTunnel);  }  }  sdTaskListService.buildDeptTunnelTreeSelect(deptTunnels);  return Result.*success*(sdTaskListService.buildDeptTunnelTreeSelect(deptTunnels)); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询任务详情

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询任务详情 | | | | 测试用例 | | JQ-TP-EI-013 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @PostMapping("/getTaskInfoList") public Result getTaskInfoList(@RequestBody String taskId){  List<SdTaskList> taskList = sdTaskListService.getTaskInfoList(taskId);  List<SdPatrolList> patrolLists = sdTaskListService.getPatrolListsInfo(taskId);  Map<String, Object> map=new HashMap<String, Object>();  map.put("task",taskList);  map.put("patrol", patrolLists);  return Result.*success*(map); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询任务列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询任务列表 | | | | 测试用例 | | JQ-TP-EI-014 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @GetMapping("/list") public TableDataInfo list(SdTaskList sdTaskList) {   startPage();  List<SdTaskList> list = sdTaskListService.selectSdTaskListList(sdTaskList);  return getDataTable(list); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除任务列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除任务列表 | | | | 测试用例 | | JQ-TP-EI-015 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @Log(title = "巡查任务", businessType = BusinessType.*DELETE*) @DeleteMapping("/{ids}")  public AjaxResult remove(@PathVariable String[] ids)  {  return toAjax(sdTaskListService.deleteSdTaskListByIds(ids));  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除故障清单

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除故障清单 | | | | 测试用例 | | JQ-TP-EI-016 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @Log(title = "故障清单", businessType = BusinessType.*DELETE*) @DeleteMapping("/{ids}")  public AjaxResult remove(@PathVariable String[] ids)  {  return toAjax(sdFaultListService.deleteSdFaultListByIds(ids));  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询设备列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询设备列表 | | | | 测试用例 | | JQ-TP-EI-017 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 查询设备列表  \* @param tunnelId  \* @return  \*/* @GetMapping("/getDevicesList") public TableDataInfo getDevicesList(String tunnelId, String deviceType){  startPage();  List<SdDevices> devices = devicesService.getDevicesList(tunnelId,deviceType);  return getDataTable(devices); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取故障列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 根据隧道故障等级获取故障列表 | | | | 测试用例 | | JQ-TP-EI-018 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 根据隧道、故障类型获取故障列表  \* @param tunnelId  \* @param faultLevel  \* @return  \*/* @GetMapping("/getFaultList") public Result getFaultList(String tunnelId,String faultLevel){  List<SdFaultList> faultList = sdFaultListService.getFaultList(tunnelId,faultLevel);  return Result.*success*(faultList); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询班组信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询班组信息 | | | | 测试用例 | | JQ-TP-EI-019 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 查询班组列表  \*/* @ApiOperation("查询班组列表") @GetMapping("/getListBz") public TableDataInfo<List<SysDept>> list() {  startPage();  Long deptId = SecurityUtils.*getDeptId*();  List<SysDept> list = sdTaskListService.selectTableBzDataInfo(deptId);  return getDataTable(list); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询隧道列表信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询隧道列表信息 | | | | 测试用例 | | JQ-TP-EI-020 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | */\*\*  \* 查询隧道列表  \* @param deptId  \* @return  \*/* @Override public List<SdTunnels> selectTunnelList(String deptId) {  return sdTunnelsMapper.selectTunnelList(deptId); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询部门列表信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询部门列表信息 | | | | 测试用例 | | JQ-TP-EI-021 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
|  | 测试用例 | @Override public List<SysDept> selectTunnelDeptList(String deptId) {  return deptMapper.selectTunnelDeptList(deptId); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 田江伟 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 安全预警

### 获取安全预警列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取安全预警列表数据 | | | | 测试用例 | | JQ-TP-SP-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取安全预警列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdEventMapper sdEventMapper;  @Test  public void testSdEvent() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdEvent sdEvent = new SdEvent();  sdEvent.setEventTypeId(11L);  sdEvent.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdEvent.setEventState("0");  List<SdEvent> list = sdEventMapper.selectSdEventList(sdEvent);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-15 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询事件处理流程

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询事件处理流程 | | | | 测试用例 | | JQ-TP-SP-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 查询事件处理流程 | | | | | | | |
| 测试用例 | @Autowired private ISdEventFlowService sdEventFlowService;  @Test public void testSdEventFlow() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdEventFlow sdEventFlow = new SdEventFlow();  sdEventFlow.setEventId("204876");  List<SdEventFlow> list = sdEventFlowService.selectSdEventFlowList(sdEventFlow);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取事件详情

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取事件详情 | | | | 测试用例 | | JQ-TP-SP-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取事件详情 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  Long id = 1116009L;  SdEvent event = sdEventService.selectSdEventById(id);  String jsonStr = JSONUtil.toJsonStr(event);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取当日未处理事件总数

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 当日未处理事件总数 | | | | 测试用例 | | JQ-TP-SP-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 当日未处理事件总数 | | | | | | | |
| 测试用例 | @Resource private SdEventMapper sdEventMapper;  @Test public void testSdEvent() {  Integer num = sdEventMapper.getEventUntreatedNum();  System.out.println(num); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 工作台-预警事件

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 预警事件列表数据 | | | | 测试用例 | | JQ-TP-SP-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 工作台预警事件列表数据 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  SdEvent event = new SdEvent();  event.setEventState("3");  List<SdEvent> list = sdEventService.getEvent(event);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 事件定位隧道分区

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 事件定位隧道分区 | | | | 测试用例 | | JQ-TP-SP-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 事件定位隧道分区 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  String tunnelId = "WLJD-JiNan-YanJiuYuan-FHS";  String stakeNum = "K10+989";  String direction = "0";  Long directId = sdEventService.getSubareaByStakeNum(tunnelId,stakeNum,direction);  //String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(directId); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 批量删除事件

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 批量删除事件 | | | | 测试用例 | | JQ-TP-SP-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 批量删除事件 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  Long[] ids = {166367L,166368L};  int upRows = sdEventService.deleteSdEventByIds(ids);  System.out.println(upRows); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 事件弹窗当日事件

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 事件弹窗当日事件 | | | | 测试用例 | | JQ-TP-SP-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 事件弹窗当日事件 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  Integer allNum = SpringUtils.getBean(SdEventMapper.class).getEventUntreatedNum();  String subIndex = "0,1";  Map<String,Object> map = new HashMap<>();  map.put("total",allNum);  map.put("data",sdEventService.eventPopAll(subIndex));  String jsonStr = JSONUtil.toJsonStr(map);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 应急调度一键恢复

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 应急调度一键恢复 | | | | 测试用例 | | JQ-TP-SP-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 应急调度一键恢复 | | | | | | | |
| 测试用例 | @Test public void testSdEvent() {  String eventId = "204852";  List<SdOperationLog> logData = SpringUtils.getBean(SdOperationLogMapper.class).getEventOperationLog(eventId);  if(logData.isEmpty()){  System.out.println("处理失败，未获取到操作记录");  }  Map<String,Object> map = new HashMap<>();  try {  //默认值：诱导灯、疏散标志亮度为50，频率为60，疏散标志地址标号为255  logData.forEach(data->{  map.put("devId",data.getEqId());  map.put("state",data.getBeforeState());  map.put("controlType","4");  map.put("eventId",eventId);  //疏散标志默认值  if(data.getEqTypeId().equals("30")){  map.put("brightness","50");  map.put("frequency","60");  }  //诱导灯默认值  if(data.getEqTypeId().equals("31")){  map.put("brightness","50");  map.put("frequency","60");  map.put("fireMark","255");  }  try {  map.put("operIp", InetAddress.getLocalHost().getHostAddress());  } catch (UnknownHostException e) {  e.printStackTrace();  }  SpringUtils.getBean(SdDeviceControlService.class).controlDevices(map);  map.clear();  });  //保存事件处理记录  SdEventFlow flow = new SdEventFlow();  flow.setFlowDescription("执行一键恢复操作");  flow.setEventId(eventId);  flow.setFlowTime(DateUtils.getNowDate());  flow.setFlowHandler(SecurityUtils.getUsername());  JSONObject json = new JSONObject();  json.put("eventFlow",flow);  WebSocketService.broadcast("eventFlow",json);  SpringUtils.getBean(SdEventFlowMapper.class).insertSdEventFlow(flow);  } catch (Exception e) {  System.out.println("操作失败，数据处理异常");  }  System.out.println("操作成功"); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 事件弹窗设备故障

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 事件弹窗设备故障 | | | | 测试用例 | | JQ-TP-SP-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 事件弹窗设备故障 | | | | | | | |
| 测试用例 | @Autowired private ISdEventService sdEventService;  @Test public void testSdEvent() {  String subIndex = "0,1";  SdEventMapper mapper = SpringUtils.getBean(SdEventMapper.class);  Integer allNum = mapper.eventPopFaultCount();  Map<String,Object> map = new HashMap<>();  map.put("total",allNum);  map.put("data",mapper.eventPopFault(subIndex));  String jsonStr = JSONUtil.toJsonStr(map);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 应急预案

### 获取应急预案列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取应急预案列表数据 | | | | 测试用例 | | JQ-TP-CP-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取应急预案列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdReservePlanMapper sdReservePlanMapper;  @Autowired  private SdTunnelSubareaMapper sdTunnelSubareaMapper;  @Test  public void testSdReservePlan() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdReservePlan sdReservePlan = new SdReservePlan();  sdReservePlan.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdReservePlan.setCategory("1");  sdReservePlan.setPlanTypeId(10L);  sdReservePlan.getParams().put("deptId", 100L);  List<SdReservePlan> list = sdReservePlanMapper.selectSdReservePlanList(sdReservePlan);  StringBuffer buffer = new StringBuffer();  for (int i = 0; i < list.size(); i++) {  List<String> strategyNames = new ArrayList<>();  Long subareaId = list.get(i).getSubareaId();  SdTunnelSubarea sdTunnelSubarea = sdTunnelSubareaMapper.selectSdTunnelSubareaBySId(subareaId);  if (sdTunnelSubarea == null) {  continue;  }  list.get(i).setSdTunnelSubarea(sdTunnelSubarea);  list.get(i).setTunnelId(sdTunnelSubarea.getTunnelId());  String strategyNamesStr = list.get(i).getStrategy().getStrategyName();  if (StrUtil.isNotBlank(strategyNamesStr)) {  strategyNames = Arrays.asList(strategyNamesStr.split(","));  for (int j = 0; j < strategyNames.size(); j++) {  int num = j + 1;  strategyNames.set(j, buffer.append(num).append("、").append(strategyNames.get(j)).toString());  buffer.setLength(0);  }  }  list.get(i).setStrategyNames(strategyNames);  }  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取预案详细信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取预案详细信息 | | | | 测试用例 | | JQ-TP-CP-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取预案详细信息 | | | | | | | |
| 测试用例 | @Autowired private ISdReservePlanService sdReservePlanService; @Autowired private ISdReservePlanFileService sdReservePlanFileService; @Autowired private ISdEquipmentFileService sdEquipmentFileService;  @Test public void testSdEvent() {  Long planId = 13L;  SdReservePlan planInfo = sdReservePlanService.selectSdReservePlanById(planId);  String jsonStr = JSONUtil.toJsonStr(planInfo);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 查询隧道分区可执行预案

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 查询隧道分区可执行预案 | | | | 测试用例 | | JQ-TP-CP-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 查询隧道分区可执行预案 | | | | | | | |
| 测试用例 | @Autowired private ISdReservePlanService sdReservePlanService; @Autowired private ISdReservePlanFileService sdReservePlanFileService; @Autowired private ISdEquipmentFileService sdEquipmentFileService;  @Test public void testSdEvent() {  SdReservePlan plan = new SdReservePlan();  plan.setSubareaId(1L);  List<SdReservePlan> list = sdReservePlanService.selectSdReservePlanBySubareaId(plan);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除应急预案

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除应急预案 | | | | 测试用例 | | JQ-TP-CP-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 删除应急预案 | | | | | | | |
| 测试用例 | @Autowired private ISdReservePlanService sdReservePlanService; @Autowired private ISdReservePlanFileService sdReservePlanFileService; @Autowired private ISdEquipmentFileService sdEquipmentFileService;  @Test public void testSdEvent() {  Long planId = 13L;  int upRows = sdReservePlanService.deleteSdReservePlanById(planId);  System.out.println(upRows); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 根据预案查询关联策略信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 根据预案查询关联策略信息 | | | | 测试用例 | | JQ-TP-CP-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 根据预案查询关联策略信息 | | | | | | | |
| 测试用例 | @Autowired private ISdReservePlanService sdReservePlanService; @Autowired private ISdReservePlanFileService sdReservePlanFileService; @Autowired private ISdEquipmentFileService sdEquipmentFileService;  @Test public void testSdEvent() {  Long planId = 13L;  List<SdStrategy> list = sdReservePlanService.selectStrategyListByPlanId(planId);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 联控策略

### 获取联控策略列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取联控策略列表数据 | | | | 测试用例 | | JQ-TP-CS-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取联控策略列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdStrategyMapper sdStrategyMapper;  @Autowired  private SdStrategyRlMapper sdStrategyRlMapper;  @Autowired  private SdEquipmentStateMapper sdEquipmentStateMapper;  @Autowired  private SdEquipmentTypeMapper sdEquipmentTypeMapper;  @Test  public void testSdStrategy() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdStrategy sdStrategy = new SdStrategy();  sdStrategy.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdStrategy.setStrategyName("博山方向一车道通行");  sdStrategy.setStrategyType("0");  sdStrategy.getParams().put("deptId", 100L);  List<SdStrategy> list = sdStrategyMapper.selectSdStrategyList(sdStrategy);  for (int i = 0; i < list.size(); i++) {  List<String> sList = new ArrayList<String>();  SdStrategyRl rl = new SdStrategyRl();  rl.setStrategyId(list.get(i).getId());  String strategyType = list.get(i).getStrategyType();  if ("1".equals(strategyType)) {  sList.add(list.get(i).getStrategyInfo());  }  List<SdStrategyRl> rlList = sdStrategyRlMapper.selectSdStrategyRlList(rl);  //策略关联表信息  for (int j = 0; j < rlList.size(); j++) {  SdEquipmentType typeObject = sdEquipmentTypeMapper.selectSdEquipmentTypeById(Long.parseLong(rlList.get(j).getEqTypeId()));  String typeName = typeObject.getTypeName();  SdEquipmentState state = new SdEquipmentState();  state.setStateTypeId(Long.parseLong(rlList.get(j).getEqTypeId()));  state.setDeviceState(rlList.get(j).getState());  List<SdEquipmentState> stateObject = sdEquipmentStateMapper.selectDropSdEquipmentStateList(state);  if(stateObject.size()<1){  continue;  }  String stateName = stateObject.get(0).getStateName();  sList.add(typeName + "控制执行：" + stateName + "；");  }  list.get(i).setSlist(sList);  }  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  }  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取联控策略详细信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取联控策略详细信息 | | | | 测试用例 | | JQ-TP-CS-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取联控策略详细 | | | | | | | |
| 测试用例 | @Autowired private ISdStrategyService sdStrategyService; @Test public void testSdEvent() {  Long strategyId = 129L;  SdStrategy strategy= sdStrategyService.selectSdStrategyById(strategyId);  System.out.println(strategy); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 工作台分时控制抽屉策略展示

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 工作台分时控制抽屉 | | | | 测试用例 | | JQ-TP-CS-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 分时策略信息 | | | | | | | |
| 测试用例 | @Autowired private ISdStrategyService sdStrategyService; @Test public void testSdEvent() {  String tunnelId = "WLJD-JiNan-YanJiuYuan-FHS";  List<Map> strategy= sdStrategyService.getTimeSharingInfo(tunnelId);  String jsonStr = JSONUtil.toJsonStr(strategy);  System.out.println(jsonStr); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 分时控制策略开启关闭

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 分时控制开启关闭 | | | | 测试用例 | | JQ-TP-CS-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 分时控制策略开关 | | | | | | | |
| 测试用例 | @Autowired private ISdStrategyService sdStrategyService; @Test public void testSdEvent() {  Long strategyId = 122L;  //0开启1关闭  String od = "0";  int upRows = sdStrategyService.strategySwitch(strategyId,od);  System.out.println(upRows); } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 执行手动控制策略

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 执行手动控制策略 | | | | 测试用例 | | JQ-TP-CS-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 执行手动控制策略 | | | | | | | |
| 测试用例 | @Autowired private ISdStrategyService sdStrategyService; @Test public void testSdEvent() {  Long strategyId = 122L;  try {  sdStrategyService.handleStrategy(strategyId);  System.out.println("执行成功");  } catch (Exception e) {  e.printStackTrace();  } } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除策略

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除策略 | | | | 测试用例 | | JQ-TP-CS-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 删除策略 | | | | | | | |
| 测试用例 | @Autowired private ISdStrategyService sdStrategyService; @Test public void testSdEvent() {  Long strategyId = 122L;  try {  int upRows = sdStrategyService.deleteSdStrategyById(strategyId);  System.out.println(upRows);  } catch (Exception e) {  e.printStackTrace();  } } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 应急资源

### 应急物资

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取应急物资列表数据 | | | | 测试用例 | | JQ-TP-ER-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取应急物资列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdMaterialMapper sdMaterialMapper;  @Test  public void testSdMaterial() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdMaterial sdMaterial = new SdMaterial();  sdMaterial.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdMaterial.setMaterialType("1");  sdMaterial.setStation("K105+185");  sdMaterial.setEndStation("K105+485");  sdMaterial.setDeptId(100L);  List<SdMaterial> list = sdMaterialMapper.selectSdMaterialList(sdMaterial);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 应急车辆

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取应急车辆列表数据 | | | | 测试用例 | | JQ-TP-ER-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取应急车辆列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdEmergencyVehicleService sdEmergencyVehicleService;  @Test  public void testSdMa3terial() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdEmergencyVehicle sdEmergencyVehicle = new SdEmergencyVehicle();  sdEmergencyVehicle.setvType("2");  sdEmergencyVehicle.setUseStatus("0");  List<SdEmergencyVehicle> list = sdEmergencyVehicleService.selectSdEmergencyVehicleList(sdEmergencyVehicle);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 应急人员

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取应急人员列表数据 | | | | 测试用例 | | JQ-TP-ER-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取应急人员列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdEmergencyPerMapper sdEmergencyPerMapper;  @Autowired  private SysDictDataMapper sysDictDataMapper;  @Test  public void testSdEmergencyPer() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdEmergencyPer sdEmergencyPer = new SdEmergencyPer();  sdEmergencyPer.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdEmergencyPer.getParams().put("deptId", 100L);  List<SdEmergencyPer> pers = sdEmergencyPerMapper.selectSdEmergencyPerList(sdEmergencyPer);  List<SysDictData> dictData = sysDictDataMapper.selectDictDataByType("sd\_emergency\_post");  for (int p = 0;p < pers.size();p++) {  SdEmergencyPer per = pers.get(p);  for (int i = 0;i < dictData.size();i++) {  SysDictData sysDictData = dictData.get(i);  if (sysDictData != null && sysDictData.getDictValue() != null && !sysDictData.getDictValue().equals("")) {  if (per != null && per.getGroupName().equals(sysDictData.getDictValue())) {  per.setGroupName(sysDictData.getDictLabel());  break;  }  }  }  }  String jsonStr = JSONUtil.toJsonStr(pers);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-15 | 测试地点 | 智能交通产业园 | | 测试人员 | | 王畅 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 操作日志

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取操作日志列表数据 | | | | 测试用例 | | JQ-TP-LM-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取操作日志列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdOperationLogMapper sdOperationLogMapper;  @Test  public void testSdOperationLog() {  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdOperationLog sdOperationLog = new SdOperationLog();  sdOperationLog.setTunnelId("WLJD-JiNan-YanJiuYuan-FHS");  sdOperationLog.setEqTypeId(31L);  sdOperationLog.getParams().put("deptId", 100L);  List<SdOperationLog> list = sdOperationLogMapper.selectSdOperationLogList(sdOperationLog);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 监控大屏

### 大屏-设备运行

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备运行统计数据 | | | | 测试用例 | | JQ-TP-MS-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备运行统计数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdProportionOfEquipmentServcie iSdProportionOfEquipmentServcie;  /\*\*  \* 获取设备运行状态  \* @return  \*/  @Test  public void getEquipmentOperationStatus(){  String tunnelId = "";  List<SdProportionOfEquipment> equipmentOperationStatus = iSdProportionOfEquipmentServcie.getEquipmentOperationStatus(tunnelId);  String jsonStr = JSONObject.toJSONString(equipmentOperationStatus);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-设备占比

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备占比统计数据 | | | | 测试用例 | | JQ-TP-MS-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备占比统计数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdProportionOfEquipmentServcie iSdProportionOfEquipmentServcie;  /\*\*  \* 查询设备占比  \* @param  \* @return  \*/  @Test  public void getProportionOfEquipment(){  Integer pageNum = 1;  Integer pageSize = 5;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  String tunnelId = "";  List<Map<String, Object>> proportionOfEquipment = iSdProportionOfEquipmentServcie.getProportionOfEquipment(tunnelId);  String jsonStr = JSONObject.toJSONString(proportionOfEquipment);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-近30日隧道预警

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取近30日隧道预警列表数据 | | | | 测试用例 | | JQ-TP-MS-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取近30日隧道预警列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdTunnelWarningService iSdTunnelWarningService;  /\*\*  \* 查询近30日隧道预警  \* @return  \*/  @Test  public void getTunnelWarningNumber(){  List<Long> tunnelWarningNumber = iSdTunnelWarningService.getTunnelWarningNumber();  String jsonStr = JSONObject.toJSONString(tunnelWarningNumber);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-本月预警事件

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取本月预警事件列表数据 | | | | 测试用例 | | JQ-TP-MS-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取本月预警事件列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询本月预警事件  \* @return  \*/  @Test  public void getEventWarning(){  String tunnelId = "";  Map<String, Object> eventWarning = sdSmartBigScreenService.getEventWarning(tunnelId);  String jsonStr = JSONObject.toJSONString(eventWarning);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-15 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-近12h控制记录

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取近12h控制记录列表数据 | | | | 测试用例 | | JQ-TP-MS-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取近12h控制记录列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISdControlRecordService iSdControlRecordService;  /\*\*  \* 近12h控制记录  \* @param  \* @return  \*/  @Test  public void getRecentControlRecordMsg(){  String tunnelId = "";  List<Map<String, Object>> recentControlRecordMsg = iSdControlRecordService.getRecentControlRecordMsg(tunnelId);  String jsonStr = JSONObject.toJSONString(recentControlRecordMsg);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-今日报警统计

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取今日报警统计列表数据 | | | | 测试用例 | | JQ-TP-MS-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取今日报警统计列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询当天报警事件  \* @param  \* @return  \*/  @Test  public void getToDayEventWarning(){  String tunnelId = "";  AjaxResult toDayEventWarning = sdSmartBigScreenService.getToDayEventWarning(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)toDayEventWarning.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-本周报警统计

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取本周报警统计列表数据 | | | | 测试用例 | | JQ-TP-MS-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取本周报警统计列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询本周报警事件  \* @param  \* @return  \*/  @Test  public void getToWeekEventWarning(){  String tunnelId = "";  AjaxResult toWeekEventWarning = sdSmartBigScreenService.getToWeekEventWarning(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)toWeekEventWarning.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);;  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-本月报警趋势

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取本月报警趋势列表数据 | | | | 测试用例 | | JQ-TP-MS-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取本月报警趋势列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询本月报警事件  \* @param  \* @return  \*/  @Test  public void getSameMonthEventWarning(){  String tunnelId = "";  AjaxResult sameMonthEventWarning = sdSmartBigScreenService.getSameMonthEventWarning(tunnelId);  Map<String, Object> object = (Map<String, Object>)sameMonthEventWarning.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-16 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-累计报警分析

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取累计报警分析列表数据 | | | | 测试用例 | | JQ-TP-MS-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取累计报警分析列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 累计预警分析  \* @param  \* @return  \*/  @Test  public void getCumulativeAlarm(){  String tunnelId = "";  AjaxResult cumulativeAlarm = sdSmartBigScreenService.getCumulativeAlarm(tunnelId);  Map<String, Object> object = (Map<String, Object>)cumulativeAlarm.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-15 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-交通事件信息列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取交通事件信息列表数据 | | | | 测试用例 | | JQ-TP-MS-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取交通事件信息列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询交通事件信息列表  \* @param  \* @return  \*/  @Test  public void getTrafficIncident(){  String tunnelId = "";  AjaxResult trafficIncident = sdSmartBigScreenService.getTrafficIncident(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)trafficIncident.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-主动安全信息列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取主动安全信息列表数据 | | | | 测试用例 | | JQ-TP-MS-011 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取主动安全信息列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询主动安全信息列表  \* @param  \* @return  \*/  @Test  public void getActiveSafety(){  String tunnelId = "";  AjaxResult activeSafety = sdSmartBigScreenService.getActiveSafety(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)activeSafety.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-设备故障信息列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备故障信息列表数据 | | | | 测试用例 | | JQ-TP-MS-012 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备故障信息列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询设备故障信息列表  \* @param  \* @return  \*/  @Test  public void getequipmentFailure(){  String tunnelId = "";  AjaxResult equipmentFailure = sdSmartBigScreenService.getequipmentFailure(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)equipmentFailure.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-应急物资

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取应急物资列表数据 | | | | 测试用例 | | JQ-TP-MS-013 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取应急物资列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询应急资源  \* @param  \* @return  \*/  @Test  public void getEmergencyMaterials(){  String tunnelId = "";  AjaxResult emergencyMaterials = sdSmartBigScreenService.getEmergencyMaterials(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)emergencyMaterials.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-值班人员

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取值班人员列表数据 | | | | 测试用例 | | JQ-TP-MS-014 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取值班人员列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询值班人员  \* @param  \* @return  \*/  @Test  public void getEmergencyPer(){  String tunnelId = "";  AjaxResult emergencyPer = sdSmartBigScreenService.getEmergencyPer(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)emergencyPer.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 大屏-预案列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取预案列表列表数据 | | | | 测试用例 | | JQ-TP-MS-015 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取预案列表列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SdSmartBigScreenService sdSmartBigScreenService;  /\*\*  \* 查询预案列表  \* @param  \* @return  \*/  @Test  public void getReservePlan(){  String tunnelId = "";  AjaxResult reservePlan = sdSmartBigScreenService.getReservePlan(tunnelId);  List<Map<String, Object>> object = (List<Map<String, Object>>)reservePlan.get("data");  String jsonStr = JSONObject.toJSONString(object);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-17 | 测试地点 | 智能交通产业园 | | 测试人员 | | 翟昌宏 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 系统管理

### 获取用户列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取用户列表数据 | | | | 测试用例 | | JQ-TP-SM-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取用户列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SysUserMapper userMapper;  @Test  public void testUserList() {  SysUser user = new SysUser();  user.setUserName("张三");  user.setStatus("0");  List<SysUser> list = userMapper.selectUserList(user);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增用户信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增用户信息 | | | | 测试用例 | | JQ-TP-SM-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试新增用户功能 | | | | | | | |
| 测试用例 | @Autowired  private ISysUserService userService;  @Test  public void testUserAdd() {  SysUser user = new SysUser();  user.setUserName("里斯");  user.setNickName("李四");  user.setEmail("1326576@163.com");  user.setSex("1");  user.setDeptId(10101L);  user.setStatus("0");  int i = userService.insertUser(user);  System.out.println(i);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改用户信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改用户信息 | | | | 测试用例 | | JQ-TP-SM-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试修改用户功能 | | | | | | | |
| 测试用例 | @Autowired  private ISysUserService userService;  @Test  public void testUserUpdate() {  SysUser user = new SysUser();  user.setUserName("wangwu");  user.setNickName("王五");  user.setEmail("6699@163.com");  user.setSex("1");  int i = userService.updateUser(user);  System.out.println(i);    } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除用户信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除用户信息 | | | | 测试用例 | | JQ-TP-SM-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试删除用户信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysUserService userService;  @Test  public void testUserDel() {  Long userId = 2L;  int i = userService.deleteUserById(userId);  System.out.println("删除成功:>>>" + i);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 用户授权角色

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 用户授权角色 | | | | 测试用例 | | JQ-TP-SM-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试用户授权角色 | | | | | | | |
| 测试用例 | @Autowired  private SysUserRoleMapper userRoleMapper;  @Test  public void testUserRole() {  Long userId = 2L;  Long[] roleIds = {2L, 3L};  List<SysUserRole> list = new ArrayList<SysUserRole>();  for (Long roleId : roleIds) {  SysUserRole ur = new SysUserRole();  ur.setUserId(userId);  ur.setRoleId(roleId);  list.add(ur);  }  if (list.size() > 0) {  userRoleMapper.batchUserRole(list);  }  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取用户角色

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取用户角色 | | | | 测试用例 | | JQ-TP-SM-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试根据用户编号获取用户角色 | | | | | | | |
| 测试用例 | @Autowired  private ISysRoleService roleService;    @Test  public void testUserU4pdate() {  Long userId = 2L;  List<SysRole> roles = roleService.selectRolesByUserId(userId);    String jsonStr = JSONUtil.toJsonStr(roles);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取部门列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取部门列表数据 | | | | 测试用例 | | JQ-TP-SM-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试获取部门列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SysDeptMapper deptMapper;  @Test  public void testDeptList() {  SysDept dept = new SysDept();  dept.setDeptName("高速");  dept.setDelFlag("0");  List<SysDept> sysDepts = deptMapper.selectDeptList(dept);  String jsonStr = JSONUtil.toJsonStr(sysDepts);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取部门下拉树列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取部门下拉树列表 | | | | 测试用例 | | JQ-TP-SM-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试获取部门下拉树列表 | | | | | | | |
| 测试用例 | @Autowired  private ISysDeptService deptService;  @Autowired  private SysDeptMapper deptMapper;  @Test  public void testDeptList2() {  SysDept dept = new SysDept();  dept.setParentId(100L);  List<SysDept> depts = deptMapper.selectDeptList(dept);  List<TreeSelect> treeSelects = deptService.buildDeptTreeSelect(depts);  String jsonStr = JSONUtil.toJsonStr(treeSelects);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增部门信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增部门信息 | | | | 测试用例 | | JQ-TP-SM-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试新增部门信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysDeptService deptService;  @Test  public void testDeptAdd() {  SysDept dept = new SysDept();  dept.setParentId(100L);  dept.setDeptName("山东征程科技");  dept.setLeader("王大伟");  dept.setStatus("0");  dept.setCreateTime(new Date());  dept.setCreateBy("admin");  int i = deptService.insertDept(dept);  System.out.println();  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改部门信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改部门信息 | | | | 测试用例 | | JQ-TP-SM-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试修改部门信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysDeptService deptService;  @Test  public void testDeptUpdate() {  SysDept dept = new SysDept();  dept.setDeptId(10704L);  dept.setDeptName("比亚科技");  dept.setLeader("高宇");  dept.setStatus("0");  dept.setUpdateTime(new Date());  dept.setUpdateBy("tom");  int i = deptService.updateDept(dept);  System.out.println();  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除部门信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除部门信息 | | | | 测试用例 | | JQ-TP-SM-011 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试删除部门信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysDeptService deptService;  @Test  public void testDeptDel() {  Long deptId = 10704L;  Result<Object> result = null;  if (deptService.hasChildByDeptId(deptId)) {  result = Result.error("存在下级部门,不允许删除");  }  if (deptService.checkDeptExistUser(deptId)) {  result = Result.error("部门存在用户,不允许删除");  }  result = result == null ? Result.toResult(deptService.deleteDeptById(deptId)) : result;  System.out.println(result.getMsg());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取角色列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取角色列表数据 | | | | 测试用例 | | JQ-TP-SM-012 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取角色列表数据 | | | | | | | |
| 测试用例 | @Autowired  private SysRoleMapper roleMapper;  @Test  public void testRoleList() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(true);  SysRole role = new SysRole();  List<SysRole> list = roleMapper.selectRoleList(role);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增角色信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增角色信息 | | | | 测试用例 | | JQ-TP-SM-013 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试新增角色信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysRoleService roleService;  @Test  public void testRoleAdd() {  SysRole role = new SysRole();  role.setRoleName("超级管理员");  role.setRoleKey("material\_mg");  role.setCreateBy("admin");  role.setCreateTime(new Date());  Result result = null;  if (UserConstants.NOT\_UNIQUE.equals(roleService.checkRoleNameUnique(role))) {  result = Result.error("新增角色'" + role.getRoleName() + "'失败，角色名称已存在");  } else if (UserConstants.NOT\_UNIQUE.equals(roleService.checkRoleKeyUnique(role))) {  result = Result.error("新增角色'" + role.getRoleName() + "'失败，角色权限已存在");  }  result = result == null ? Result.toResult(roleService.insertRole(role)) : result;  System.out.println(result.getMsg());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改角色信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改角色信息 | | | | 测试用例 | | JQ-TP-SM-014 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试修改角色信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysRoleService roleService;  @Test  public void testRoleUpdate() {  SysRole role = new SysRole();  role.setRoleId(2L);  role.setRoleName("物资管理员");  role.setRoleKey("material\_mg");  role.setUpdateBy("李三");  role.setUpdateTime(new Date());  Long[] menuIds = {100L, 101L};  role.setMenuIds(menuIds);  roleService.checkRoleAllowed(role);  Result result = null;  if (UserConstants.NOT\_UNIQUE.equals(roleService.checkRoleNameUnique(role))) {  result = Result.error("修改角色'" + role.getRoleName() + "'失败，角色名称已存在");  } else if (UserConstants.NOT\_UNIQUE.equals(roleService.checkRoleKeyUnique(role))) {  result = Result.error("修改角色'" + role.getRoleName() + "'失败，角色权限已存在");  }  result = result == null ? Result.toResult(roleService.updateRole(role)) : result;  System.out.println(result.getMsg());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除角色信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除角色信息 | | | | 测试用例 | | JQ-TP-SM-015 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试删除角色信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysRoleService roleService;  @Autowired  private SysRoleMenuMapper roleMenuMapper;  @Autowired  private SysRoleMapper roleMapper;  @Autowired  private SysRoleDeptMapper roleDeptMapper;  @Test  public void testRoleDel() {  Long[] roleIds = {2L};  for (Long roleId : roleIds) {  roleService.checkRoleAllowed(new SysRole(roleId));  SysRole role = roleService.selectRoleById(roleId);  if (roleService.countUserRoleByRoleId(roleId) > 0) {  throw new ServiceException(String.format("%1$s已分配,不能删除", role.getRoleName()));  }  }  // 删除角色与菜单关联  roleMenuMapper.deleteRoleMenu(roleIds);  // 删除角色与部门关联  roleDeptMapper.deleteRoleDept(roleIds);  int i = roleMapper.deleteRoleByIds(roleIds);  System.out.println(i);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取岗位列表数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取岗位列表数据 | | | | 测试用例 | | JQ-TP-SM-016 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试获取岗位列表数据 | | | | | | | |
| 测试用例 | @Autowired  private ISysPostService postService;  @Test  public void testPostList() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(true);  SysPost post = new SysPost();  post.setPostCode("ceo");  post.setPostName("管理员");  List<SysPost> list = postService.selectPostList(post);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 新增岗位信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 新增岗位信息 | | | | 测试用例 | | JQ-TP-SM-017 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试新增岗位信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysPostService postService;  @Test  public void testPostAdd() {  SysPost post = new SysPost();  post.setPostName("管理员");  post.setPostCode("sdd");  post.setStatus("0");  post.setCreateBy("admin");  post.setCreateTime(new Date());  Result result = null;  if (UserConstants.NOT\_UNIQUE.equals(postService.checkPostNameUnique(post))) {  result = Result.error("新增岗位'" + post.getPostName() + "'失败，岗位名称已存在");  } else if (UserConstants.NOT\_UNIQUE.equals(postService.checkPostCodeUnique(post))) {  result = Result.error("新增岗位'" + post.getPostName() + "'失败，岗位编码已存在");  }  result = result == null ? Result.success(postService.insertPost(post)) : result;  System.out.println(result.getMsg());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 修改岗位信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 修改岗位信息 | | | | 测试用例 | | JQ-TP-SM-018 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试修改岗位信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysPostService postService;  @Test  public void testPostUpdate() {  SysPost post = new SysPost();  post.setPostId(1L);  post.setPostName("理发师");  post.setPostCode("ceo");  post.setStatus("0");  post.setUpdateBy("admin");  post.setUpdateTime(new Date());  Result result = null;  if (UserConstants.NOT\_UNIQUE.equals(postService.checkPostNameUnique(post))) {  result = Result.error("修改岗位'" + post.getPostName() + "'失败，岗位名称已存在");  } else if (UserConstants.NOT\_UNIQUE.equals(postService.checkPostCodeUnique(post))) {  result = Result.error("修改岗位'" + post.getPostName() + "'失败，岗位编码已存在");  }  result = result == null ? Result.success(postService.updatePost(post)) : result;  System.out.println(result.getMsg());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 删除岗位信息

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 删除岗位信息 | | | | 测试用例 | | JQ-TP-SM-019 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 测试删除岗位信息 | | | | | | | |
| 测试用例 | @Autowired  private ISysPostService postService;  @Autowired  private SysPostMapper postMapper;  @Test  public void testPostDel() {  Long[] postIds = {1L, 3L, 10L,};  for (Long postId : postIds) {  SysPost post = postService.selectPostById(postId);  if (postService.countUserPostById(postId) > 0) {  throw new ServiceException(String.format("%1$s已分配,不能删除", post.getPostName()));  }  }  int i = postMapper.deletePostByIds(postIds);  System.out.println(i);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-07 | 测试地点 | 智能交通产业园 | | 测试人员 | | 卢强 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

## 系统交互

### 获取雷达感知数据列表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取雷达感知数据列表 | | | | 测试用例 | | JQ-TP-SI-001 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取雷达感知数据列表 | | | | | | | |
| 测试用例 | @Autowired  private ISdRadarDetectDataService sdRadarDetectDataService;  @Test  public void testTunnel() {  Integer pageNum = 1;  Integer pageSize = 10;  String orderBy = "";  Boolean reasonable = true;  PageHelper.startPage(pageNum, pageSize, orderBy).setReasonable(reasonable);  SdRadarDetectData sdRadarDetectData = new SdRadarDetectData();  List<SdRadarDetectData> list = sdRadarDetectDataService.selectSdRadarDetectDataList(sdRadarDetectData);  String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取设备状态数据集合

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取设备状态数据集合 | | | | 测试用例 | | JQ-TP-SI-002 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取设备状态数据集合 | | | | | | | |
| 测试用例 | @Autowired  private RadarEventService service;  @Test  public void testTunnel(Map<String,Object> map) {  String tunnelId = (String) map.get("tunnelId");  Object sdRadarDevices = service.selectDevice(tunnelId);  return AjaxResult.success(sdRadarDevices); String jsonStr = JSONUtil.toJsonStr(list);  System.out.println(jsonStr);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取雷达感知数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取雷达感知数据 | | | | 测试用例 | | JQ-TP-SI-003 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取雷达感知数据 | | | | | | | |
| 测试用例 | @KafkaListener(topics = {RadarEventConstants.MATCHRESULTDATA}, containerFactory = "kafkaOneContainerFactory")  public void topicMatchResultData(ConsumerRecord<String, String> record, Acknowledgment item, Consumer<?,?> consumer) throws Exception {  String value = record.value();  Map<String,Object> map = (Map<String, Object>) JSON.parse(value);  String participantNum = map.get("participantNum")+"";  if (Integer.parseInt(participantNum)>0){  service.insertRadarDetect(map);  } else {  List<SdRadarDetectData> dataList = new ArrayList<>();  JSONObject object = new JSONObject();  object.put("radarDataList", dataList);  );  }  consumer.commitSync();  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 获取雷达设备运行状态数据

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 获取雷达设备运行状态数据 | | | | 测试用例 | | JQ-TP-SI-004 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 获取雷达设备运行状态数据 | | | | | | | |
| 测试用例 | @Autowired  private RadarEventService service;  @KafkaListener(topics = {RadarEventConstants.WJDEVICERUNNINGINFO}, containerFactory = "kafkaOneContainerFactory")  public void topicWjDeviceRunningInfo(ConsumerRecord<String, String> record, Acknowledgment item, Consumer<?,?> consumer) throws ParseException {  String value = record.value();  Map<String,Object> map = (Map<String, Object>) JSON.parse(value);  service.saveRedis(map);  consumer.commitSync();  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 万集设备运行状态数据发送

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 万集设备运行状态数据发送 | | | | 测试用例 | | JQ-TP-SI-005 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 万集设备运行状态数据发送 | | | | | | | |
| 测试用例 | @Autowired  private RadarEventService service;  @PostMapping("wjData/sendBaseDeviceStatus")  public String sendBaseDeviceStatus(@RequestBody Map<String,Object> map){  service.sendBaseDeviceStatus(map);  return "向主题发送数据信息";  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 物联中台指令数据模拟发送

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 物联中台指令数据模拟发送 | | | | 测试用例 | | JQ-TP-SI-006 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 物联中台指令数据模拟发送 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @PostMapping("/direct")  public Result direct(@RequestParam("devNo") String devNo, @RequestParam("devType") String devType) {  String s = sendMsgService.sendDirect(devNo, devType);  return Result.success(s);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 物联中台事件数据模拟发送

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 物联中台事件数据模拟发送 | | | | 测试用例 | | JQ-TP-SI-007 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 物联中台事件数据模拟发送 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @PostMapping("/event")  @ApiOperation("发送事件数据")  public Result event(){  return Result.success(sendMsgService.sendEvent());  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 物联中台设备数据模拟发送

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 物联中台设备数据模拟发送 | | | | 测试用例 | | JQ-TP-SI-008 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 物联中台设备数据模拟发送 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @PostMapping("/devicesdata")  @ApiOperation("发送设备数据")  public Result devicesdata(@RequestParam("devId") String devId,@RequestParam("state") String state){  return Result.success(sendMsgService.devicesdata(devId,state));  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 物联中台发送状态数据模拟发送

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 物联中台发送状态数据模拟发送 | | | | 测试用例 | | JQ-TP-SI-009 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 物联中台发送状态数据模拟发送 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @PostMapping("/devStatus")  @ApiOperation("发送状态数据")  public Result devStatus(@RequestBody RadarMsgTopicVo vo){  String ss = sendMsgService.sendDevStatus(vo);  return Result.success(ss);  } | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 接收设备状态数据测试

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 接收设备状态数据测试 | | | | 测试用例 | | JQ-TP-SI-010 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 接收设备状态数据测试 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @KafkaListener(topics = {"wq\_devStatusTopic"}, containerFactory = "kafkaTwoContainerFactory")  public void devStatusData(ConsumerRecord<String,Object> record, Acknowledgment acknowledgment, Consumer<?,?> consumer){  log.info("{}", record.value());  if (authorizeName != null && !authorizeName.equals("") && authorizeName.equals("GSY")) {  if (record.value() != null || !record.value().toString().equals("")) {  System.out.println(record.value());  JSONObject jsonObject = JSONObject.parseObject(record.value().toString());  }  }  consumer.commitSync(); | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |

### 接收事件数据测试

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 待测功能 | | 接收事件数据测试 | | | | 测试用例 | | JQ-TP-SI-011 | |
| 测试  类型 | | □静态分析 □接口测试 ☑单元测试 □功能测试  □性能测试 □结构覆盖 □内存缺陷 □边界测试  □人机界面 □强度测试 □安全性 □可恢复性 | | | | | | | |
| 测试用例设计 | 描述 | 接收事件数据测试 | | | | | | | |
| 测试用例 | @Autowired  private SendMsgService sendMsgService;  @KafkaListener(topics = {"wq\_tunnelEvent"}, containerFactory = "kafkaTwoContainerFactory")  public void tunnelEventData(ConsumerRecord<String, Object> record, Acknowledgment acknowledgment, Consumer<?, ?> consumer) {  System.err.println("authorizeName:" + authorizeName);  log.info("{}, {}, {}, {}", record.topic(), record.partition(), record.offset(), record.value());  }  }  consumer.commitSync(); | | | | | | | |
| 用例执行情况 | 测试时间 | | 11-20 | 测试地点 | 智能交通产业园 | | 测试人员 | | 吴昊阳 |
| 测试结果 ☑通过 □未通过 □可重现 □不可重现  故障现象描述： | | | | | | | | |