# 引言

1. 编写目的

该使用手册是为了帮助用户了解并且快速使用植毛机实时监控系统，讲解了一些基本的操作和需要注意的问题

1. 背景

广东雪洁日化用品有限公司是一家日用品生产公司，主要的产品有成人牙刷，儿童牙刷，儿童牙膏。

为了满足广东雪洁日化用品有限公司的发展需求，提高车间生产效率，广东雪洁日化用品有限公司委托广东三三智能科技有限公司进行数字化生产线车间改造

1. 项目需求
   1. 广东雪洁日化用品有限公司囊括了一下需求：
2. 对各个生产线的合格数量、不良数量、不良详细类型数量(错毛、脏污等..)未完成量、已完成量、开始时间、结束时间进行实时监控
3. 可以对工单单号、产品型号、客户单号、计划产量进行填写并保存，当订单完成后可以对订单进行手动提交
4. 可以查看历史记录并进行Excel表格导出
5. 通过大屏幕显示实时监控信息

# 项目设计

1. **系统概述**

植毛机实时监控系统采用硬件结合的方式，通过TCP协议进行通讯实现数据的交互并将数据保存至本地数据库，使用SpringBoot嵌入运行服务器进行部署，MySQL将数据保存至本地，并使用以下硬件进行交互：

1. 工控机
2. 无线键鼠
3. 网线
4. 交换机
5. 显示屏

**2.运行环境**

推荐配置

系统：win7 64位

语言：简体中文

内存：4GB

硬盘：128GB

显示屏：55-60寸大屏显示屏

**3.开发环境**

操作系统：win7 64位系统

开发工具：Eclipse

开发语言：java EE 8

开发框架：SpringBoot

数据库：MySQL 8

1. **部署环境**

数据库：MySQL 8

JDK版本：jdk 1.8

# 项目设计

1. **业务流程**

****

1. **数据库**

连接信息表

表名：ss\_connect\_info

作用：用于存储服务端的IP地址、端口号、线程连接状态

字段：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 是否允许为空 | 字段说明 |
| id | int(11) 自增 | NO | 连接信息表的id |
| ip | varchar(50) | NO | 生产线的IP地址 |
| port | int(10) | NO | 生产线的端口号，一般为6666 |
| thread\_state | tinyint(1) | NO | 线程状态 |
| modification\_time | timestamp | YES | 修改时间随时间戳进行改变 |

超出产量表

表名：ss\_exceeding\_production

作用：如果产量超出将超出的产量放入该表中

字段：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 是否允许为空 | 字段说明 |
| id | int(11) | NO | 主键，自增 |
| connect\_info\_id | int(11) | NO | 连接信息表的id |
| exceeding\_production | int(10) | NO | 超出的产量,默认为0 |

生产数据表  
表名：ss\_production\_data

作用：实时保存生产数据

字段：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 是否允许为空 | 字段说明 |
| id | int(11) | NO | 主键，自增 |
| name | varchar(100) | NO | 机器名称 |
| version | varchar(100) | NO | 产品型号 |
| customer\_number | varchar(100) | NO | 客户单号 |
| work\_order\_number | varchar(100) | NO | 工单单号 |
| scheduled\_production | Int(11) | NO | 计划产量 |
| production\_state | int(1) | NO | 生产状态  0 未完成 1 已完成 2 未开始  默认为2 |
| connect\_Info\_id | int(11) | NO | 连接信息id |
| ok | int(11) | NO | 合格数 |
| ng | int(11) | NO | 不合格数 |
| state | varchar(100) | NO | 机器状态  0 暂停 1 运行  2 未连接 |
| owe\_hole | int(11) | NO | 欠孔 |
| long\_hair | int(11) | NO | 长毛 |
| loose\_wool | int(11) | NO | 散毛 |
| Long\_short\_hair | int(11) | NO | 长短毛 |
| dirty | int(11) | NO | 脏污 |
| implantation\_error | varchar(255) | NO | 植错毛 |
| difference\_hair | varchar(255) | NO | 差毛 |
| total | int(11) | NO | 总数 |
| start\_time | timestamp | NO | 订单开始时间 |
| end\_time | timestamp | NO | 订单结束时间 |
| create\_time | timestamp | NO | 订单创建时间 |

用户表

表名：ss\_user

作用：用于记录用户登录信息,比如账号或者密码

字段：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 是否为空 | 字段说明 |
| id | int(11) | NO | 主键，自增 |
| user\_name | varchar(100) | NO | 用户名 |
| password | varchar(100) | NO | 密码 |

# **接口功能设计**

1. 主界面的实时查询接口

业务说明：

当打开植毛机实时监控系统后，系统首先判断是否当前的机器已存在订单数据，如果不存在则进行创建，如果已存在但是未填写基本数据(工单单号、产品型号、客户单号、计划产量) 则不对该订单进行数据的累计，而将未填写基本数据期间的数据保存至超出产量表中，当我们保存基本数据后将超出产量放入已完成量中并清零。当计划产量小于已完成量将超出的已完成量放入超出产量表中。

* 注意！！！

如果机器不发送数据或者未连接则不创建新的数据。

计算说明

* 已完成量的累计

|  |  |
| --- | --- |
| 计算目的 | 计算方式 |
| 获取实时已完成量 | 数据库中的已完成量 + ( 当前已完成量 - 上次已完成量 ) |

说明：

当启动程序时实时监控系统会创建两个变量，一个用来记录上一次的已完成量，一个用来记录当前从socekt获取到的已完成量，当前的已完成量减去上次的已完成量等于真正要累计到数据库中的已完成量，再加上数据库中的已完成量后更新到数据库中

* 未完成量的获取

1. 如果计划产量大于已完成量，计算方式如下所示：

|  |  |
| --- | --- |
| 计算目的 | 计算方式 |
| 获取未完成量 | 计划产量 - 合格数量 |

说明：

植毛机的**总数**相当于已完成量

* 注意！

计划产量如果小于已完成量，未完成量永远为0

* 超出产量的计算

超出产量的累计分为两种情况：

1. 当用户未填写基本数据是不进行数据的累计的，为了防止用户未填写基本数据期间的数据错漏，将未填写数据期间累计的数据存入数据库中防止数据错漏。

|  |  |
| --- | --- |
| 计算目的 | 计算方式 |
| 用户未填写基本数据期间遗漏的已完成量 | 数据库中的超出产量 + ( 上次的已完成量 - 当前的已完成量 ) |

1. 当已完成量超出计划产量后将超出的部分存入超出产量表中，当下一次基本数据填写完毕将超出的产量放入已完成量中并将超出产量清零。

|  |  |
| --- | --- |
| 计算目的 | 计算方式 |
| 超出产量 | 数据库中的超出产量 + (已完成量 - 计划产量) |



* **接口基本信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 生产数据查询 | ProductionData/findAll | GET | 无 |

1. 保存按钮接口

业务说明：

1. 该接口对工单单号，产品型号，客户单号，计划产量进行保存，如果工单单号，产品型号客户单号其中之一为空或者计划产量为0或空不允许对数据进行保存，并弹出提示窗口，但是将未累计数据期间的已完成量保存至超出产量表中

当基本数据(工单单号，产品型号，客户单号，计划产量)保存成功后，后台线程开始累计数据并将超出产量中的数据添加至已完成量中，如果已完成量小于计划产量则不允许修改订单基本数据，但是允许进行提交。



* **接口基本信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 基本数据保存接口 | ProductionData/saveBaseInfo | POST | ProductionDataBaseInfoReq req 生产数据请求对象 |

1. 提交按钮接口

业务说明

1. 同上，如果基本数据(工单单号，产品型号，客户单号，计划产量)未进行填写不可进行按钮的提交操作
2. 当基本数据填写完成后，无论订单是否完成都可以进行数据的提交，提交完成后如果线程未抛出异常则自动创建一条新的数据，线程状态暂时会显示为未连接，当成功获取到数据后线程状态改变为运行。线程抛出异常将数据库中的thread\_state（线程状态）改变为false也就是0然后利用break跳出while循环，正常结束线程，后台定时器实时监控线程状态每1秒一次如果为ture则继续进行监听，如果为false则创建新线程重新连接socket。

* 关于不合格数的说明

由于未完成量可以通过计划产量减去已完成量来获取，获取到的数据一定是准确的，但是当计划产量超出已完成量后累计的不合格数量将相当于第二单的不合格数量暂时无法进行计算累计，所以不合格数量在未提交订单以前会一直不断累计。



1. 历史查询接口

历史记录查询有两种方式：

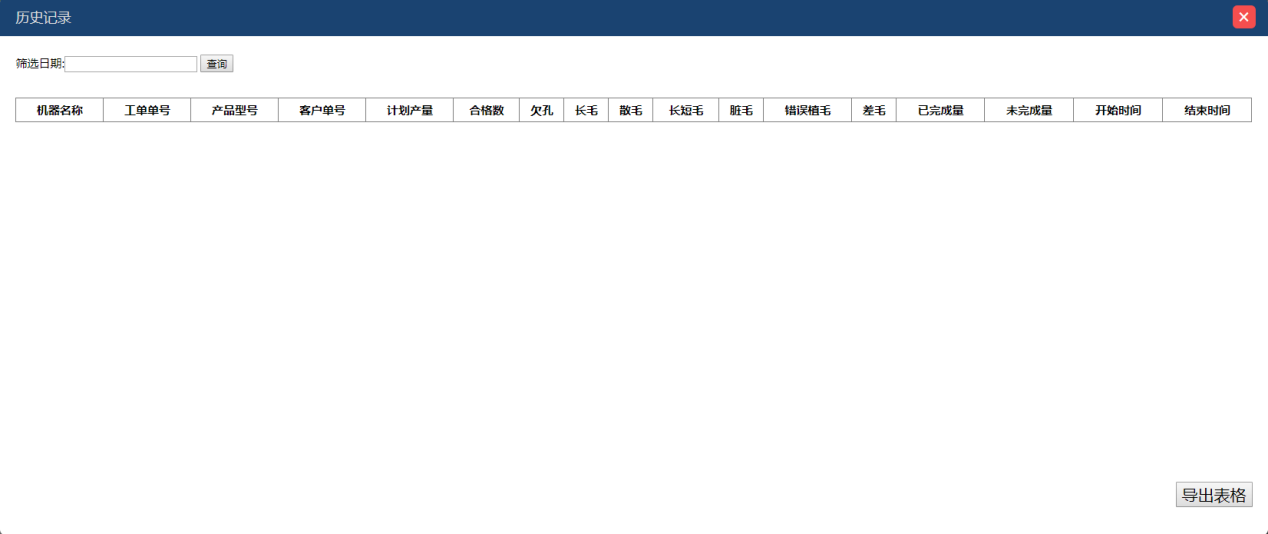
业务说明：

1. 当用户提交数据后则将生产状态改变为已完成，并变为历史记录。
2. 当我们点击历史记录按钮后会显示默认查询出当天的已完成的订单数据，并弹出历史记录页面进行显示，如果当天不存在数据则不进行显示

* 注意！！

这里并没有历史记录表，只是查询当天已提交的订单，如果是指定日期查询则查询已提交的指定日期的订单。

1. 无参数历史记录查询



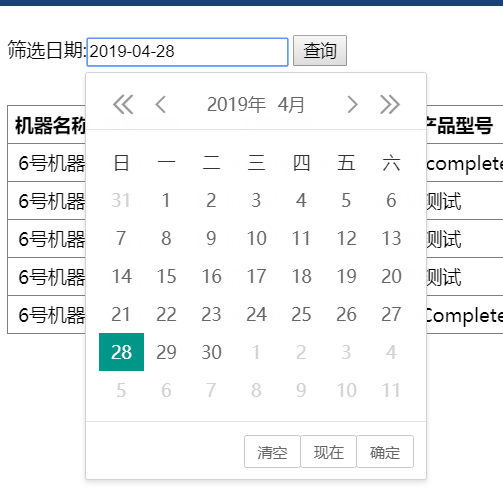
1. 日期模糊查询

业务说明：

当显示历史记录页面后，可以点击日期查询文本框进行日期查询，这里的日期查询使用的是模糊查询，与历史记录查询是同一个接口,默认为查询当天的已完成的订单，但是如果传入指定的日期参数会查询指定日期的所有订单信息。

* 注意！！！

如果已完成量大于计划产量则未完成量为0



* **接口基本信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 历史记录日期查询接口 | /ProductionData/findByPageHistory | GET | 指定的日期,  日期格式：yyyy-MM-dd |

1. 用户登录接口

业务说明：

管理员通过输入正确密码进行登录，并对登录进行Session状态登记。如有配

置或密码修改请求，在检查Session是否登录后实现允许或禁止访问。



* **接口基本信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 基本数据保存接口 | /login | POST | User usr：包括了用户名和密码 |

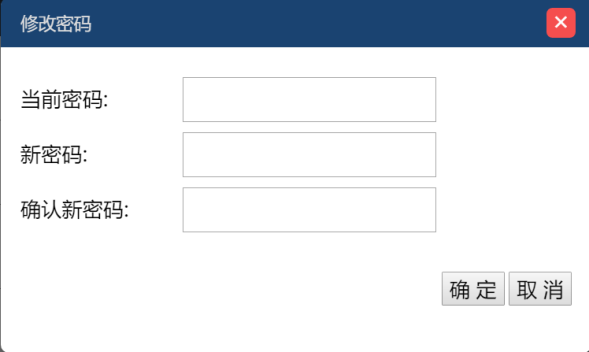
1. 修改密码接口

业务说明:

管理人员输入原密码和新密码，提交服务端接口进行修改。后台对于原密码先进

行验证；如果原密码错误，则返回密码错误提示；否则，修改数据库中管理员密码

并返回修改成功提示。



|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 基本数据保存接口 | /modifypsw | POST | String psw:原密码  String newpsw：新密码 |

1. 配置接口

业务说明：

当点击配置按钮后，后台会检查session的登录状态，如果为已登录则查询出所有配置信息并进行保存，用户可以对IP地址、端口进行信息的修改，当点击保存按钮对数据进行保存。

如果session未记录登录状态，则为未登录，则不弹出窗口配置窗口并弹出提示信息。

配置接口分为查询配置接口和修改配置接口

1. 查询配置接口

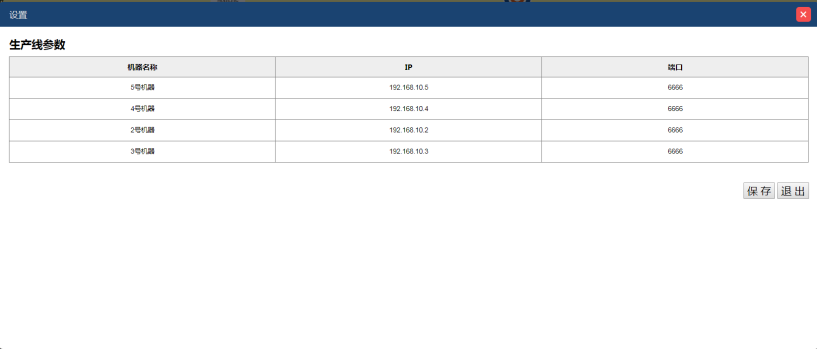
当点击登录成功后点击配置按钮可打开配置页面，但是在打开的同时查询所有配置信息



|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 基本数据保存接口 | /config/list | GET | 无 |

1. 修改配置接口

当用户修改完成ip地址、端口号点击保存按钮进行保存



|  |  |  |  |
| --- | --- | --- | --- |
| 接口名称 | 访问地址 | 请求方式 | 请求参数 |
| 基本数据保存接口 | /config/modify | POST | ConnectInfo config:包含了所有连接信息的对象 |

# **五、其他**

1. **通讯方式**

系统通讯通过socket进行连接通讯并获取数据,生产线为服务器而实时监控系统为客户端，客户端利用死循环实时监控服务器是否发送数据，如果收到数据接收数据并进行累计。

业务说明：

当程序启动并加载完成JVM后socket会自动调用初始化连接方法，并在初始化方法中初始化线程连接状态将状态更改为正常，之后从数据库中获取所有的端口号，IP地址信息并循环遍历连接，如果连接成功不进行其他处理，同时启动线程状态定时器，两秒监听一次线程状态，如果连接失败！将线程状态改变为false，这时线程监听定时器会尝试重新启动线程(两秒进行一次)，如果启动成功将线程状态更改为true，定时器会继续进行监听。

1. **数据的读取方式**

系统读取数据是通过JSON格式进行获取的读取的，格式如下：

{"OK":"1","NG":"0","Dirt":"0","LooseHair":"0","LongHair":"0","WrongHair":"0","Hole":"0","CrushHair":"0","UnevenHair":"0","State":"0"}

参数说明

|  |  |
| --- | --- |
| **参数名称** | **对应key** |
| 合格数 | OK |
| 不合格数 | NG |
| 脏污 | Dirt |
| 散毛 | LooseHair |
| 长毛 | LongHair |
| 错毛 | WrongHair |
| 欠孔 | Hole |
| 叉毛 | CrushHair |
| 长短毛 | UnevenHair |
| 状态 | State |

状态说明：

植毛机会发送两个状态，0 为暂停 1为运行 当socket 未连接成功状态将变为未连接

* 注意！

这里获取到的数据首先是JSON格式的字符串，之后将字符串转换为json对象获取数据