中国大学生服务外包
创新创业大赛

慧博云通
HYDSOFT

智能排班系统

参赛学校：	西南石油大学
指导老师：	肖斌
团队名称：	脱贫脱单不脱发
团队成员：	孙贵林 韦富元 贾旺 王洋 郑源鹏

目录

1. 引言

1.1 项目简介

随着云计算、大数据和人工智能等技术的快速发展，劳动力管理正经历着由手工经验化管理到数字化、智能化管理的升级过程。作为服务和零售等行业的从业者，在日趋激烈的市场竞争中，如何通过科学的手段优化劳动力资源、控制劳动力成本、提高员工体验和客户满意度，将成为取得竞争优势的关键。

我们的客户慧博云通科技股份有限公司（股票简称：慧博云通 代码：301316）是一家专注于软件信息技术服务的国家高新技术企业。为了更好地实现劳动力的优化匹配，在市场变化快速的环境下，慧博云通决定开发一款劳动力管理系统。该系统的目标是通过业务预测、劳动力预测和智能排班等核心功能，实现员工的最优化分配和精细化管理，从而控制成本、提高客户满意度和员工体验。

我们的任务是基于给定的业务预测和劳动力预测数据，开发一套智能排班算法，并将其集成到智能排班的劳动力管理系统中。通过该算法的优化调度，我们将帮助慧博云通实现更高效的员工排班，提高客户满意度和员工工作体验，从而实现创造更大价值和取得竞争优势的目标。

1.2 项目目标

本项目旨在基于零售行业某公司的业务预测系统，开发一套智能排班系统，帮助店铺管理者更加高效地管理员工排班，满足不同时间段、不同技能岗位的门店劳动力需求，保证业务高峰期的服务质量，同时降低业务低谷时的人力浪费。

具体实现目标包括：

1. 根据给出的业务预测数据，精确合理地安排每个营业网点员工每天各个时间段的排班，员工排班的粒度以半小时为单位。

2. 把已经指派的班次重新分配给别的员工或者把未指派的开放班次分配给可用的员工，自动匹配可用的员工进行排班。

3. 提供方便的班次查看功能，支持按日、按周查看，日视图和周视图里可以按技能、岗位和员工分组查看、编辑排班表，支持手动的班次指派。

4. 在满足业务预测需求和所有规则的情况下，保证总体员工工作时间最低。

总之，本智能排班系统旨在通过自动排班、精细管理的方式，为店铺管理者带来更多便利和效益，提升服务质量和员工工作品质，同时降低用工成本和管理

难度。

1.3 读者对象

主要读者群体：本方案的主要读者群体为具有一定技术背景和开发经验的技术人员，包括开发人员、架构师、测试人员和数据库管理员等。

读者群体对方案的期望：开发人员期望方案能够提供明确的技术规范和文档说明；测试人员期望方案能够提供清晰的测试用例和测试规范；数据库管理员期望方案能够提供详细的数据库设计和架构说明。

2. 需求分析

2.1 客户需求和期望

客户需求：

本项目旨在开发一个智能排班系统，满足客户对人力资源管理方面的需求，提高排班的效率，减少排班错误导致的人力成本的损失，便于管理人员管理公司内部的员工，同时可以为员工提供更加合理舒适的工作安排，提高员工的工作满意度和忠诚度。

期望部分：

1. 精确排班：期望智能排班系统能够精确、自动地为门店的员工进行排班安排，遵循门店的工作规程和法定节假日的规定，且可以根据员工的特殊需求进行定制化安排。同时，支持员工各班次之间的换班，员工可以在系统内提出申请，经过管理员审批同意后，完成排班班次换班。

2. 智能管理：期望智能排班系统可以智能化地管理门店的排班信息，包括排班表、员工信息、换班信息、偏好信息等，对排班过程中的异常情况能给出警告，能够为管理人员提供准确详细的管理报表。

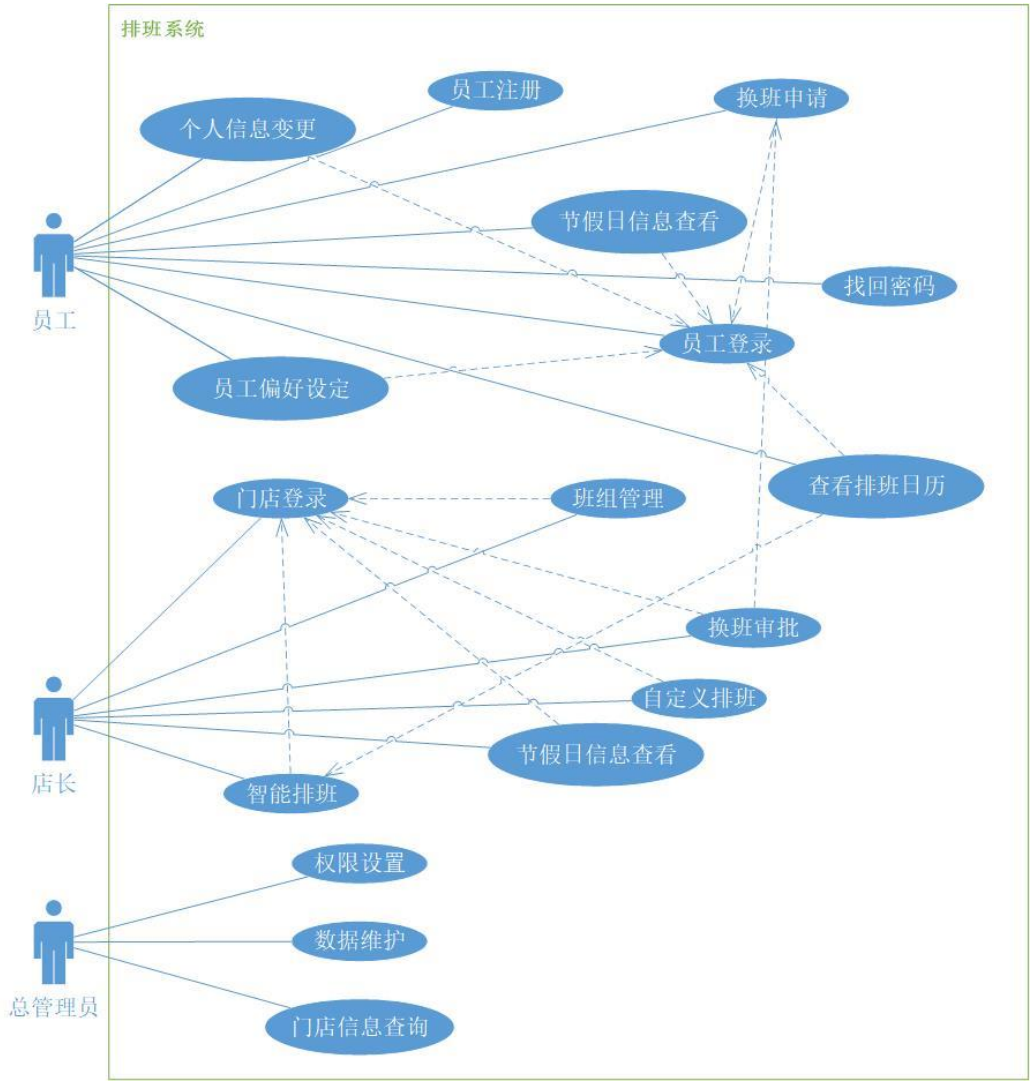
3. 方便查看：期望智能排班系统可以便捷地为管理人员提供查看员工排班情况，包括每个员工的出勤、请假及加班情况，可以根据时间轴进行排版，也支持根据员工技能或部门等信息进行筛选和查询。

4. 最小化总体员工时间：期望智能排班系统能够最大限度地减少门店总体员工的排班时间，同时较大限度保证员工的工作量和工作时长的合理性，以提高门店的运作效率。

5. 安全可靠：期望智能排班系统数据安全可靠，员工信息互不干扰，避免信息外泄，保证门店的商业机密和员工信息的安全。

6. 特殊需求：期望智能排班系统可以根据门店特殊需求进行定制化，包括多门店多部门管理、异地员工排班、灵活人员管理等等。同时，支持员工之间的排班换班，员工可以在系统内提出申请，经过管理员审批同意后，完成排班换班，方便员工在工作和个人事务之间调整安排。

2.2 用户场景和用例分析



员工角色：

场景一：员工查看自己的排班情况和门店信息

用例 1：员工登录系统后，进入自己的在职记录页面，查看自己的排班情况，包括工作日期、工作时间、工作地点、排班信息等详细信息。员工可以根据时间轴、日历等不同视图查看排班情况，同时可以发起排班换班申请和请假申请。

场景二：员工向管理员申请排班换班

用例：员工在系统中发起排班换班申请，选择要换班的日期、时间、对方员

工等相关信息，提交申请后，系统将自动将申请信息发送给管理员进行审批。管理员与对方员工确认后在系统中审批通过后，系统将自动完成排班换班。

场景三：员工修改编辑自己的偏好

用例：员工可以在系统中修改编辑自己的工作日偏好、工作时长偏好、班次时长偏好等个人偏好设置，以便于系统智能排班时参考员工的偏好优先进行调整安排。

店长（管理员）角色：

场景一：管理员查看门店员工的排班和门店信息

用例 1：管理员登录系统后，进入公司管理页面，可以查看本部门和全公司员工的排班情况。可以根据时间轴、日历等不同视图查看排班情况，同时还可以根据员工的个人信息、部门等进行筛选和查询。如果排班有异常情况（如漏排等），系统将自动给出提示信息。管理员可以根据对于本部门员工的信息进行手动调整。

用例 2：管理员登录系统后，进入门店管理页面，可以查看门店的各项信息，包括门店面积、门店各职位员工数量、职能树状图、当前客流承载量等等，方便进行门店的管理与优化。

场景二：管理员查看待审批的排班换班和请假申请以及门店规则管理

用例 1：管理员在系统中查看待审批的排班换班和请假申请，可以根据申请人、日期等条件进行筛选，便于管理员快速处理待审批事项。如果审批通过，系统将自动完成排班换班或请假操作。如果审批被拒绝，系统将自动将信息反馈给申请人，便于申请人进行下一步操作。

用例 2：管理员在系统中查看门店管理页面，可以进行门店规则的管理，例如设置门店的营业时间，门店客流量超过一定数值时，系统自动弹出提醒，门店各职位员工数量进行调整等等。

场景三：管理员查看排班统计报表

用例：管理员可以在系统中查看员工的出勤、请假、加班等排班信息的统计报表，包括员工工作日历、考勤报表、工作时长等。管理员可以根据员工、部门等条件进行筛选和查询，并导出相应的报表，以便于管理人员进行统计和分析。

场景四：管理员查看门店信息和规则

用例 1：管理员登录系统后，进入门店管理页面，可以查看门店的各项信息，

包括门店面积、门店各职位员工数量、职能树状图、当前客流承载量等等，方便进行门店的管理与优化。

用例 2：管理员在系统中查看门店规则，例如设置门店的营业时间、员工工作时间等等，以确保门店运营规范有序。

场景五：管理员在系统中添加、删除、修改员工信息

用例：管理员可以在系统中添加、删除、修改员工信息，包括员工的个人信息、在职状态、所属部门、职位等等，以保证员工管理和排班的有效性和准确性。

超级管理员角色：

场景一：管理员设置公司和门店的账户和权限

用例：超级管理员可以在系统中设置公司和门店的账户和权限，包括超级管理员、管理员、员工等不同角色的权限设置，以确保系统安全和操作规范。

场景二：管理员设置公司和门店的账户和权限

用例：超级管理员可以管理和查看系统所有的数据，包括门店，员工，权限的各种信息查看，超级管理员一般建议赋权给技术人员以便进行系统的调整。

2.3 功能和非功能需求

智能排班系统的功能需求：

员工端：员工可以查看自己的身份信息、排班计划、请假信息、换班信息，可以提出请假请求和换班请求。

管理员端：管理员可以查看每个门店的基本信息、员工信息，可以编辑、发布排班计划，审核和处理员工的请假和换班请求，能够添加、删除、修改员工和门店信息，配置排班规则。

超级管理员端：超级管理员可以对公司的各项信息进行管理，包括添加、删除、修改员工、门店信息和分配权限等。

智能排班系统的非功能需求：

- 系统的安全性要求高，确保数据的安全、防止数据泄露和攻击。
- 系统的可靠性要求高，确保系统的高可用性，避免系统的停机和崩溃。
- 系统的响应速度要求高，快速响应用户的操作请求。
- 系统的可扩展性要求高，能够支持规模和功能的扩展和升级。
- 系统的易用性和可维护性要求高，方便用户使用和日常的维护和升级，

减少用户的学习成本。

- 系统的准确性要求高，根据员工的工作状况和门店的客流量等信息，生成最优的排班计划。
- 系统的可定制性要求高，支持不同用户之间的个性化需求定制。
- 系统的性能稳定，需要能够支持大量的并发访问，避免系统出现卡顿和死机等情况。

2.4 限制和约束条件

1) 数据安全性：系统需要提供高度确保数据安全的技术措施，例如加密传输，防止遭受黑客攻击和数据泄露。
2) 实时性：系统需要能够实时地处理请求和生成新的排班计划。
3) 系统性能：系统需要保证响应时间和效率，尤其是针对大型门店和复杂排班规则的情况下，系统需要保持高可用性和卓越的性能。
4) 移动端设备兼容性：系统需要兼容不同种类的移动设备，例如 Android 和 iOS 等，确保用户可以轻松地在移动设备上查看和使用系统功能。
5) 定制性：系统需要支持满足用户需求的个性化定制，例如排班规则、员工信息、门店信息等等。
6) 系统稳定性：系统需要保证稳定，可靠并尽可能地防止系统崩溃，避免数据丢失和断电等情况发生导致的服务中断。
7) 界面易用性：系统需要提供友好的界面，使得用户可轻松地使用系统的功能，容易理解和快速掌握。
8) 运维要求：系统需要有良好的运维能力和后勤支持，确保系统运行安稳、可靠、高效和长时间运作。
9) 扩展性要求：系统需要具备可扩展性和灵活性，满足不断增长的业务需求，能够快速添加新的门店和员工数量还有排班规则。
10) 响应度：需要充分考虑系统并发性和效率，保证用户在任何时间任何地点都能简单、快速地使用系统。

3. 系统概要设计

3.1 算法设计

为了解决传统遗传算法出现的种种问题，本项目采用了遗传算法+模拟退火算法的设计方案，其大概思想是以遗传算法为基础生成初步最优排班结果，再用

模拟退火算法对排班结果进行优化，最终生成最优的排班表。

将遗传算法得到的结果，再用模拟退火算法进行优化，可以得到更好的优化结果。这种算法结合了遗传算法的全局寻优能力和模拟退火算法的全局和局部寻优能力，以达到更好的性能。有以下优点：

1. 收敛速度快

遗传算法和模拟退火算法都是启发式的全局最优解算法。遗传算法不依赖于任何先验知识，并且不易受到局部最优解的影响。模拟退火算法是在优秀的解空间中随意探索的算法，它可以从局部最优解中跳出来，提高全局搜索的能力。因此，将这两个算法结合起来可以加快搜索的收敛速度。

2. 改善局部最优解

由于遗传算法可能会陷入局部最优解，解决此问题需要使用一定的局部搜索策略。在模拟退火的策略下，将遗传算法的结果作为初始解，以改善选择的样本和交叉的个体为目标，逐渐逼近全局最优解。这样能有效地避免遗传算法的局限，提高搜索准确性和可靠性。

3. 增强寻优能力

遗传算法需要较长时间才能在一定程度上计算最终解。在遗传算法的初始种群中，可能有许多不正确的解，即使这些解在迭代运算的过程中有所改善，很有可能最终解仍然不是最优解。这时将模拟退火算法应用于优化过程，能够增强遗传算法的寻优能力。模拟退火算法在运行的过程中，可以一直寻找区域最优解，最终获得全局最优解。

4. 提高准确性

遗传算法无法保证在足够短的时间内找到全局最优解，特别是在某些复杂问题中。但是结合使用遗传算法和模拟退火算法，能够提高求解问题的准确性。这种结合方法能够获得最优的中间解，达到最优解或次优解。即使没有获得全局最优解，也可以得到最优的可接受解。

通过排班规则和员工偏好等信息，编写适应度函数和优化函数，以遗传算法为基础调用自定义的适应度函数进行第一轮排班，生成较为符合规则的初步排班表，再以模拟退火算法为基础，调用自定义的适应度函数和优化函数对初步排班表进行优化得到最终的排班表。

综上所述，将遗传算法得到的结果使用模拟退火算法进行优化，能够缩短收敛时间、改善局部最优解等弊端，增强寻优能力，适用于多种应用场景，提高求

解问题的准确性。同时，由于遗传算法和模拟退火算法的相互结合，非常灵活，可以根据具体问题进行调整，以获得更加精确的解决方案。

3.2 系统设计

1. 系统架构

智能排班系统的系统架构可以采用分层架构，将整个系统按照职责进行分层，每层之间通过接口对外提供服务。具体架构可以分为以下三层：

1.1 表现层（Presentation Layer）

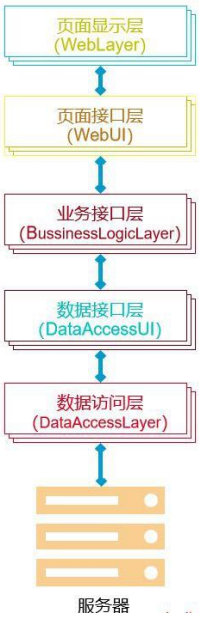
- 提供给用户界面操作程序，比如 Web 页面等。
- 主要负责前端页面的展示和用户输入的响应处理。
- 使用 React、Vue 框架。

1.2 业务逻辑层（Business Layer）

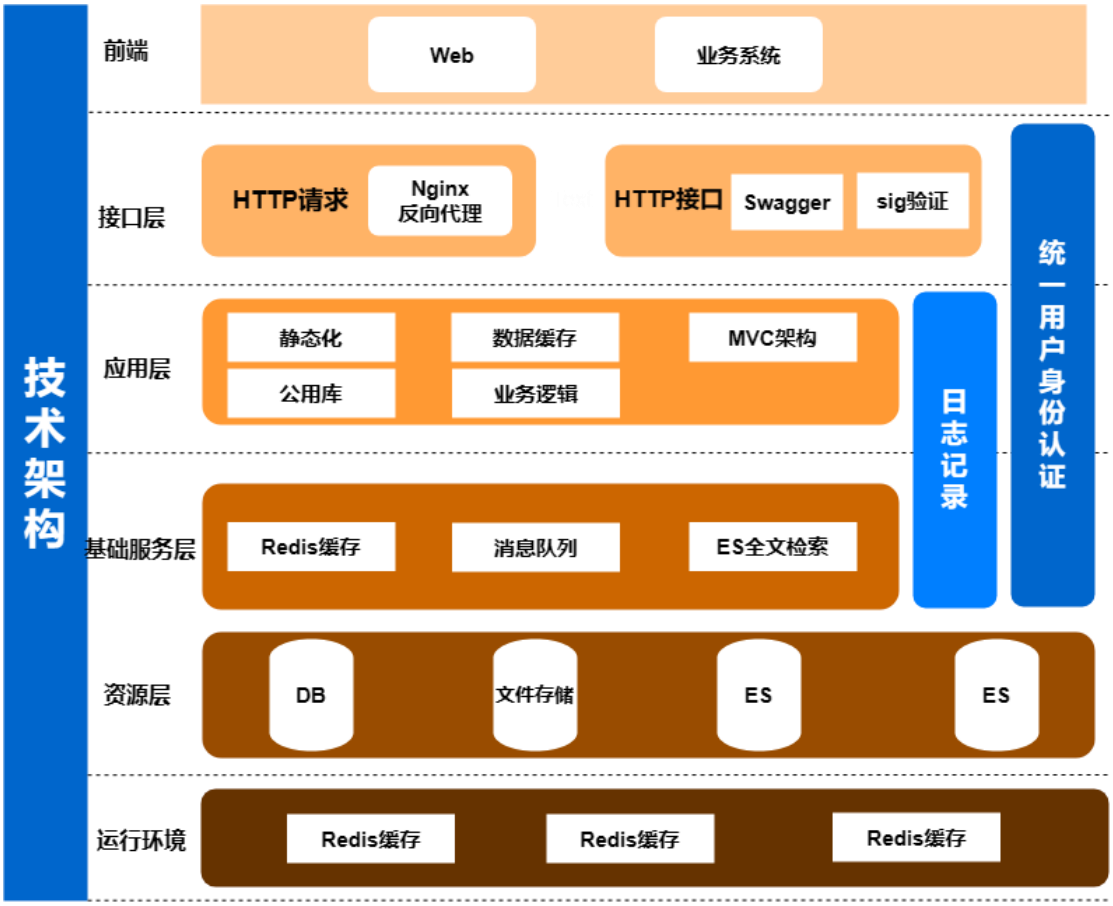
- 职责是处理应用程序的业务逻辑，提供接口给表现层、数据访问层使用。
- 包括员工信息管理、排班管理、请假审批等。
- 使用 Spring Boot、Spring Cloud 框架。

1.3 数据访问层（Data Access Layer）

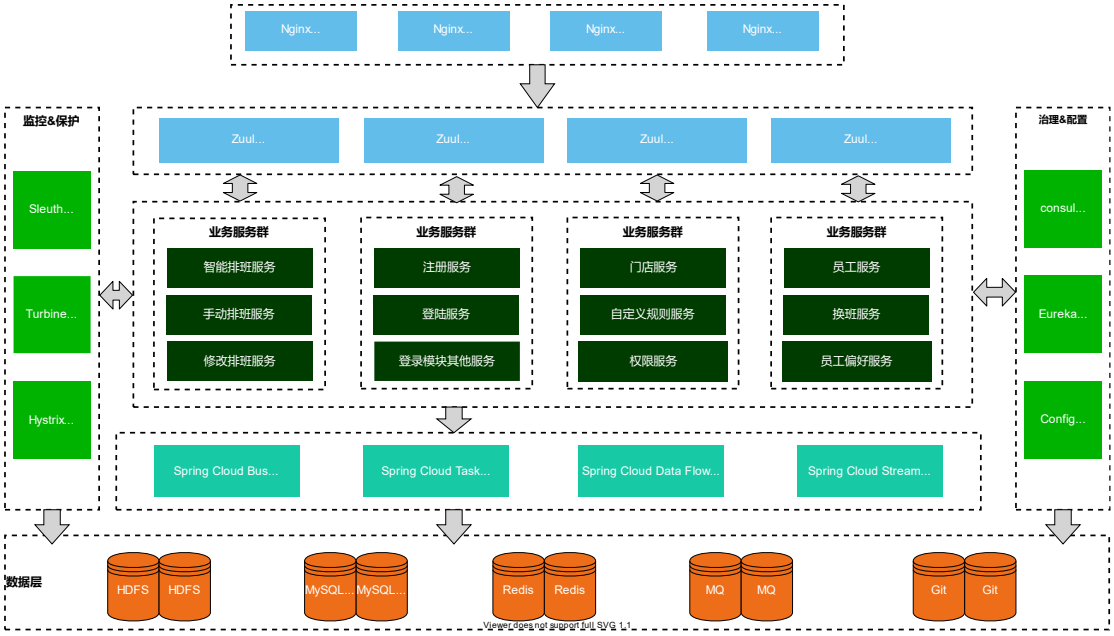
- 负责与所有数据存储系统交互，如关系型数据库、非关系型数据库等。
- 提供给业务逻辑层使用的数据接口，包括数据的增删改查等。
- 使用 MyBatis、Redis 等框架。
- 系统架构设计要考虑系统的性能、可扩展性、可维护性等因素，适用于适当的技术和框架。



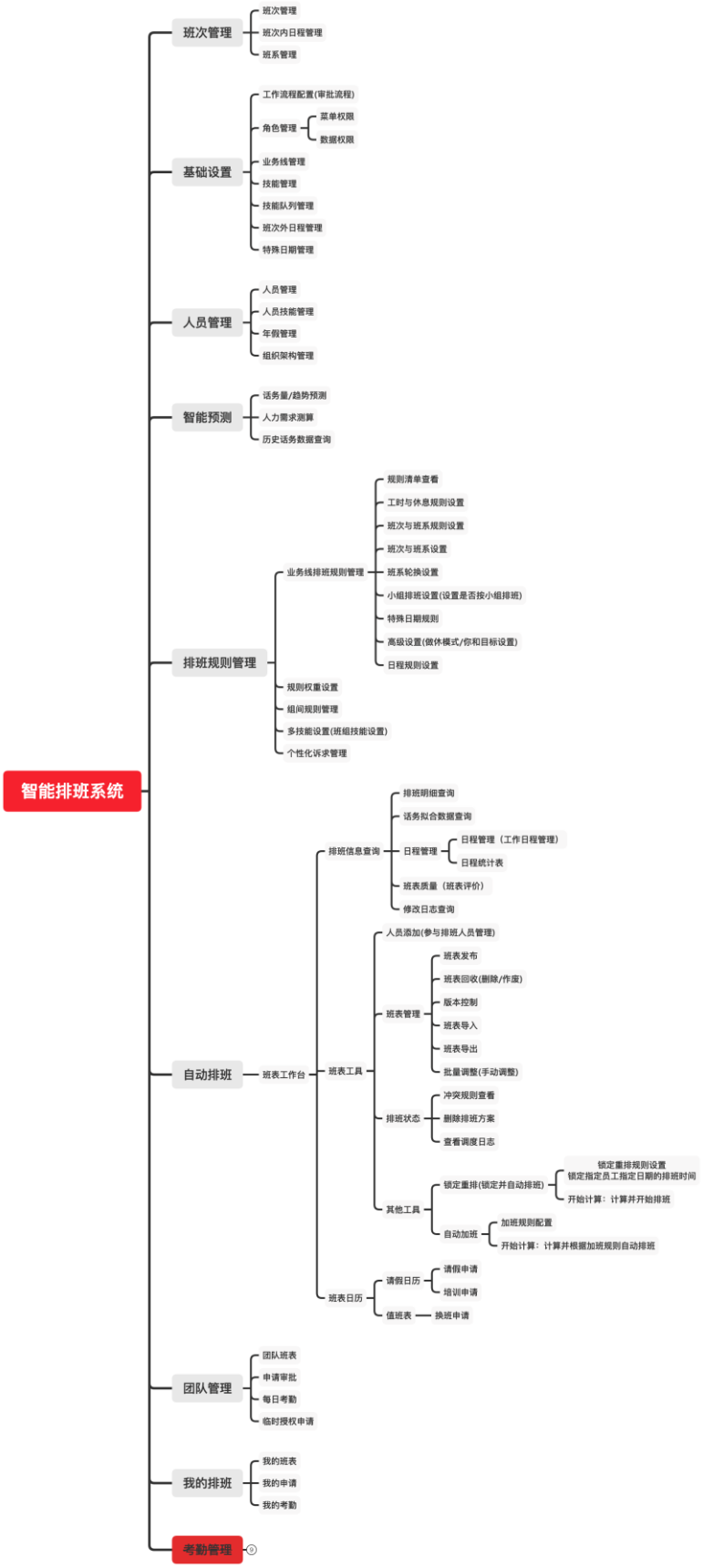
技术架构图：



微服务架构图



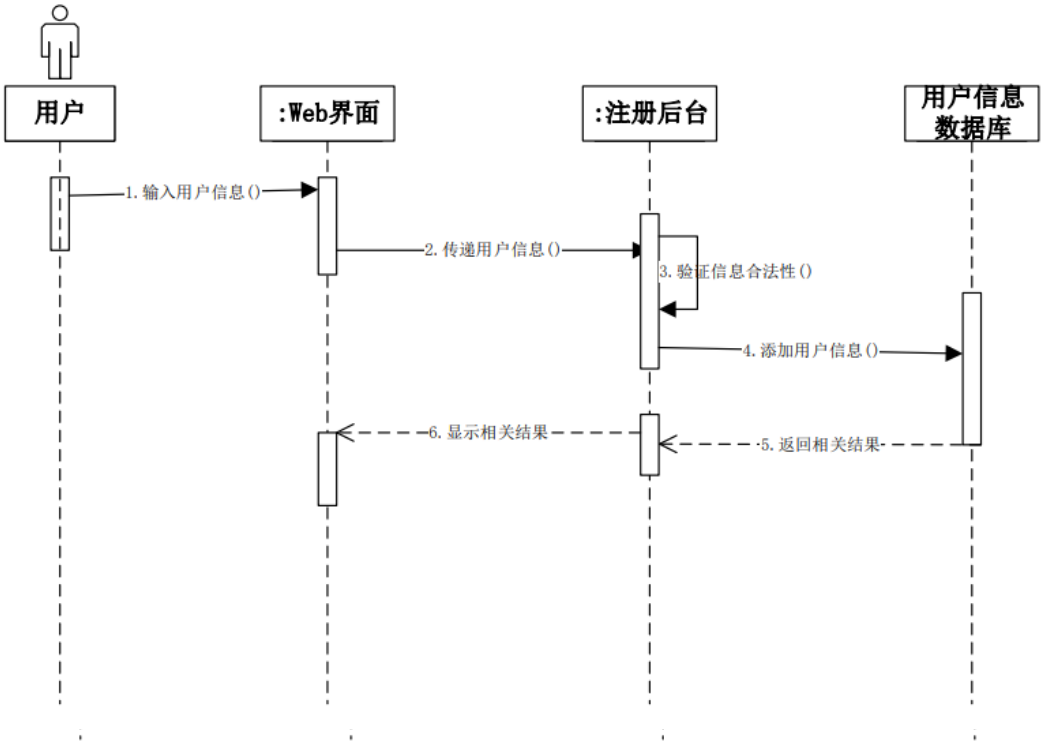
功能模块设计：


智能排班系统
班次管理
班次管理
班次内日程管理
班系管理
基础设置
工作流程配置(审批流程)
角色管理
菜单权限
数据权限
业务线管理
技能管理
技能队列管理
班次外日程管理
特殊日期管理
人员管理
人员管理
人员技能管理
年假管理
组织架构管理
智能预测
话务量/趋势预测
人力需求测算
历史话务数据查询
排班规则管理
业务线排班规则管理
规则清单查看
工时与休息规则设置
班次与班系规则设置
班次与班系设置
班系轮换设置
小组排班设置(设置是否按小组排班)
特殊日期规则
高级设置(做休模式/你和目标设置)
日程规则设置
规则权重设置
组间规则管理
多技能设置(班组技能设置)
个性化诉求管理
自动排班
班表工作台
排班信息查询
排班明细查询
话务拟合数据查询
日程管理
日程管理（工作日程管理）
日程统计表
班表质量（班表评价）
修改日志查询
班表工具
人员添加(参与排班人员管理)
班表管理
班表发布
班表回收(删除/作废)
版本控制
班表导入
班表导出
批量调整(手动调整)
排班状态
冲突规则查看
删除排班方案
查看调度日志
其他工具
锁定重排(锁定并自动排班)
锁定重排规则设置
锁定指定员工指定日期的排班时间
开始计算：计算并开始排班
自动加班
加班规则配置
开始计算：计算并根据加班规则自动排班
班表日历
请假日历
请假申请
培训申请
值班表
换班申请
团队管理
团队班表
申请审批
每日考勤
临时授权申请
我的排班
我的班表
我的申请
我的考勤
考勤管理

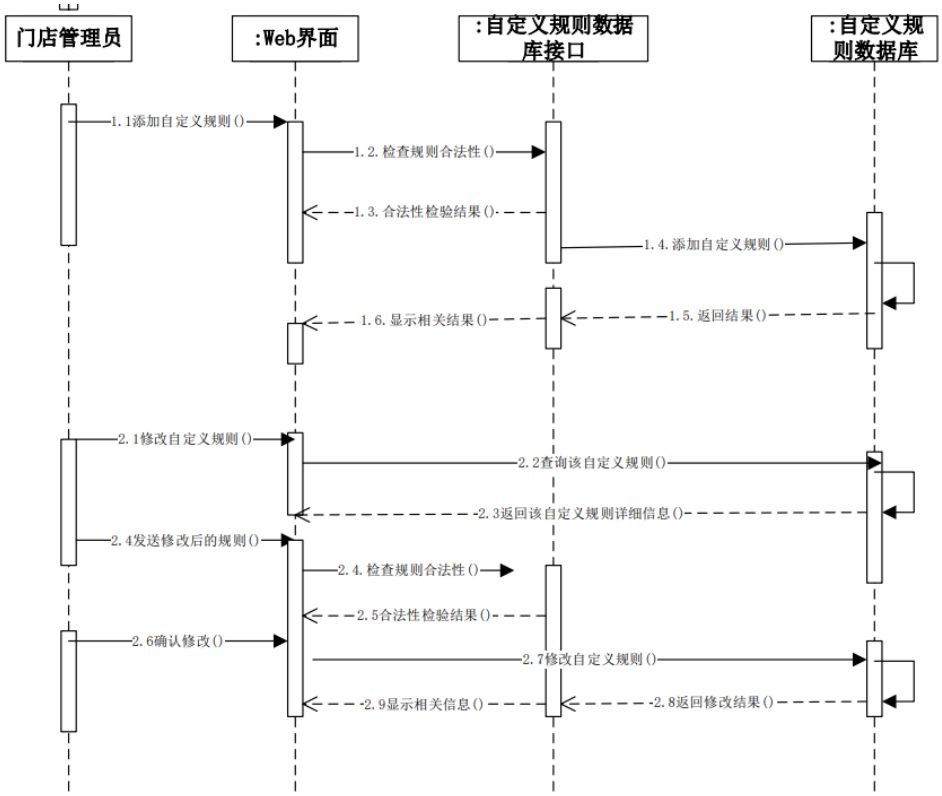
核心业务模型设计：

时序图建模：

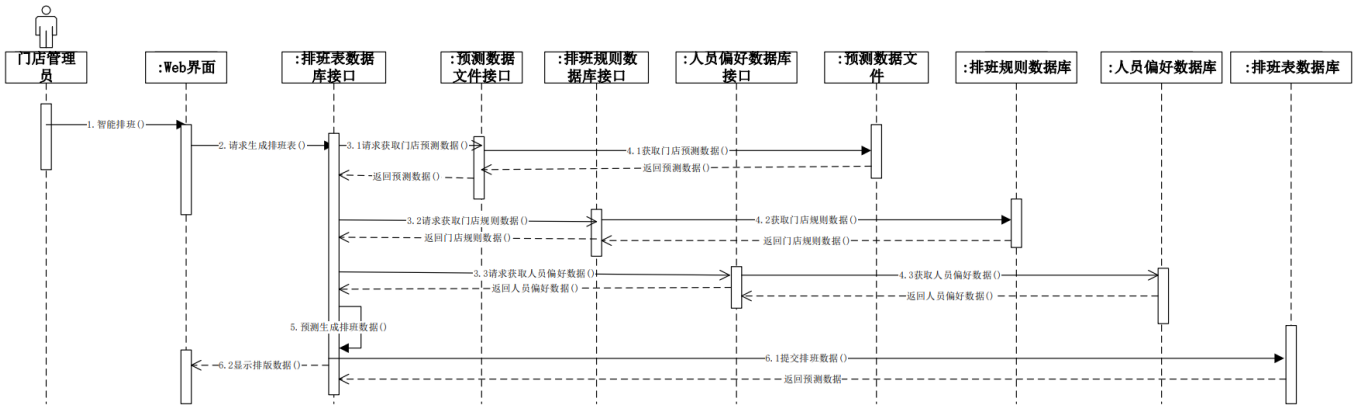
注册登录模块：



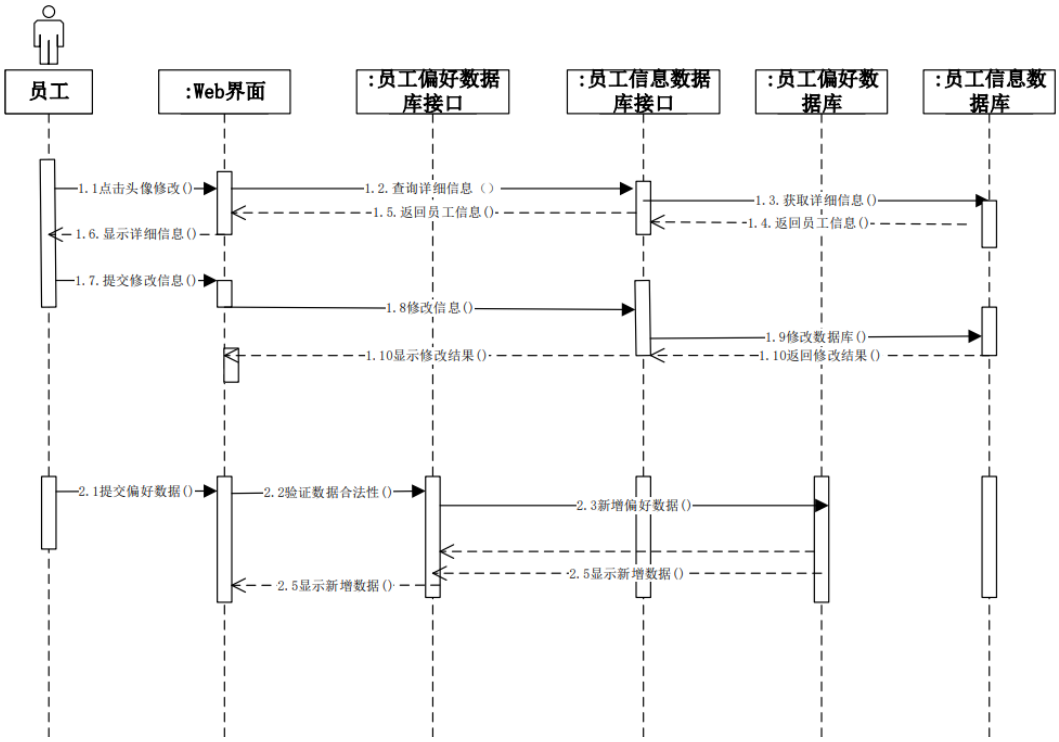
自定义规则模块：



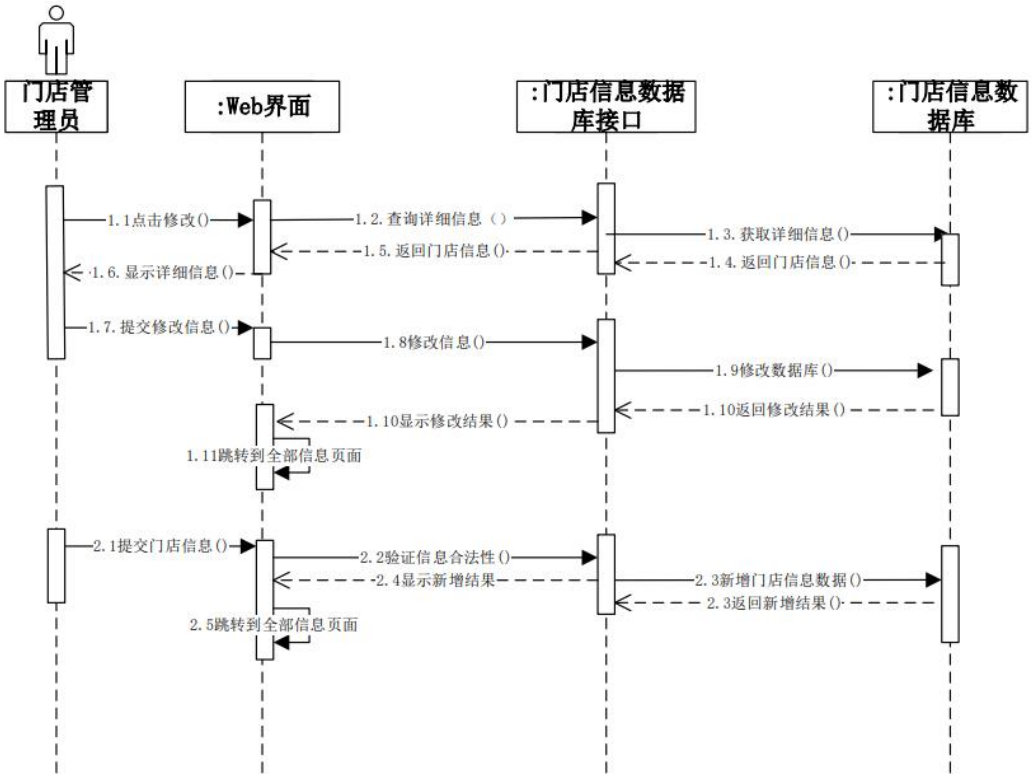
智能排班模块：


门店管理员
:Web界面
:排班表数据库接口
:预测数据文件接口
:排班规则数据库接口
:人员偏好数据库接口
:预测数据文件
:排班规则数据库
:人员偏好数据库
:排班表数据库
1.智能排班()
2.请求生成排班表()
3.1请求获取门店预测数据()
4.1获取门店预测数据()
返回预测数据()
返回预测数据()
3.2请求获取门店规则数据()
4.2获取门店规则数据()
返回门店规则数据()
返回门店规则数据()
3.3请求获取人员偏好数据()
4.3获取人员偏好数据()
返回人员偏好数据()
返回人员偏好数据()
5.预测生成排班数据()
6.1提交排班数据()
6.2显示排版数据()
返回预测数据

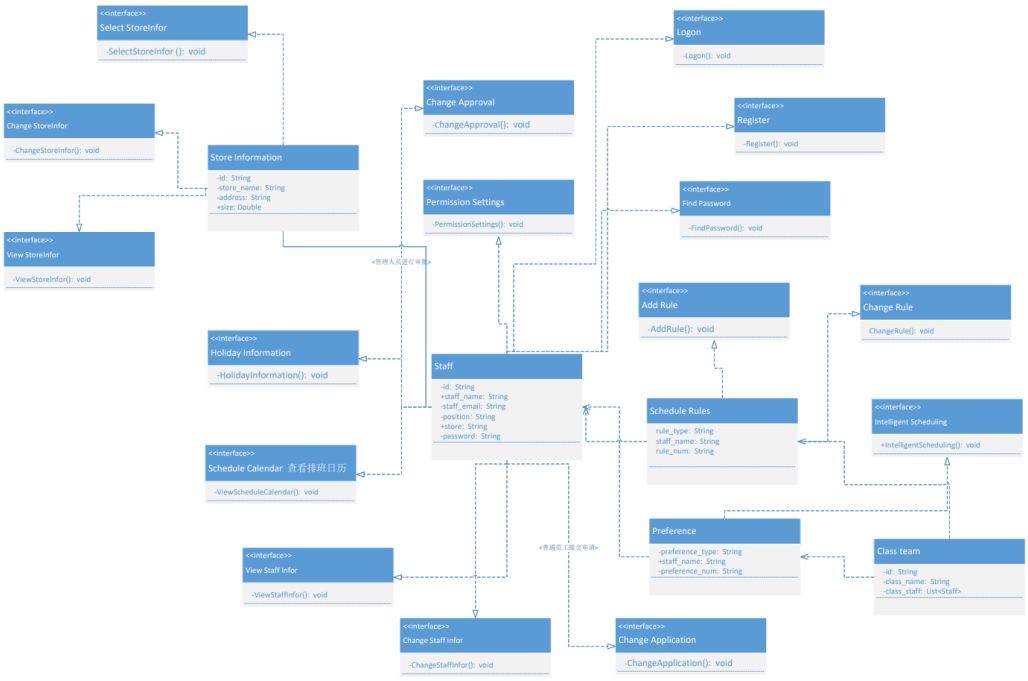
员工模块：


员工
:Web界面
:员工偏好数据库接口
:员工信息数据库接口
:员工偏好数据库
:员工信息数据库
1.1点击头像修改()
1.2.查询详细信息（）
1.3.获取详细信息()
1.5.返回员工信息()
1.4.返回员工信息()
1.6.显示详细信息()
1.7.提交修改信息()
1.8修改信息()
1.9修改数据库()
1.10显示修改结果()
1.10返回修改结果()
2.1提交偏好数据()
2.2验证数据合法性()
2.3新增偏好数据()
2.5显示新增数据()
2.5显示新增数据()

门店模块：


门店管理员
:Web界面
:门店信息数据库接口
:门店信息数据库
1.1点击修改()
1.2. 查询详细信息（）
1.3. 获取详细信息()
1.5. 返回门店信息()
1.4. 返回门店信息()
1.6. 显示详细信息()
1.7. 提交修改信息()
1.8修改信息()
1.9修改数据库()
1.10显示修改结果()
1.10返回修改结果()
1.11跳转到全部信息页面
2.1提交门店信息()
2.2验证信息合法性()
2.4显示新增结果
2.3新增门店信息数据()
2.3返回新增结果()
2.5跳转到全部信息页面

类图建模：


Select StoreInfor
-SelectStoreInfor (): void
Logon
-Logon(): void
Change StoreInfor
-ChangeStoreInfor(): void
Change Approval
-changeApproval(): void
Register
-Register(): void
Store Information
-id: String
-store_name: String
-address: String
+size: Double
Permission Settings
-PermissionSettings(): void
Find Password
-FindPassword(): void
View StoreInfor
-ViewStoreInfor(): void
Add Rule
-AddRule(): void
Change Rule
-ChangeRule(): void
Holiday Information
-HolidayInformation(): void
Staff
-id: String
+staff_name: String
-staff_email: String
-position: String
+store: String
-password: String
Schedule Rules
rule_type: String
staff_name: String
rule_num: String
Intelligent Scheduling
+IntelligentScheduling(): void
Schedule Calendar 查看排班日历
-ViewScheduleCalendar(): void
Preference
-preference_type: String
+staff_name: String
-preference_num: String
Class team
-id: String
-class_name: String
-class_staff: List<Staff>
View Staff Infor
-ViewStaffInfor(): void
Change Staff Infor
-ChangeStaffInfor(): void
Change Application
-ChangeApplication(): void

3.3 数据库设计

数据库规范

命名规范：

1、所有数据库对象名称必须使用小写字母并用下划线分割

2、所有数据库对象名称禁止使用 MySQL 保留关键字

3、数据库对象的命名要做到见名知意，并且最好不超过 32 个字符

4、临时表必须以 tmp_为前缀并以日期为后缀，备份表必须以 bak_为前缀并以日期（时间戳）为后缀

5、所有存储相同数据的列名和列类型必须一致

6、系统表（基础数据表）的前缀采用 sys_开始

7、表名前应该加上前缀，表的前缀一个用系统或模块的英文名称缩写。

8、在数据库表命名时应该用英文单词的单数形式。

数据库基本设计规范：

1、所有表和字段都需要添加注释

2、尽量做到冷热数据分离，减小表的宽度

3、禁止在表中建立预留字段

4、禁止在数据库中存储图片，文件等大的二进制数据

5、禁止从开发环境，测试环境直接连接生成环境数据库

数据库字段设计规范：

1、优先选择符合存储需要的最小的数据类型

原因： 列的字段越大，建立索引时所需要的空间也就越大

2、避免使用 TEXT,BLOB 数据类型

建议把 BLOB 或是 TEXT 列分离到单独的扩展表中（冷热分离）

3、尽可能把所有列定义为 NOT NULL

原因：索引 NULL 列需要额外的空间来保存，所以要占用更多的空间

4. 同财务相关的金额类数据必须使用 decimal 类型

Decimal 类型为精准浮点数，在计算时不会丢失精度

SQL 优化

1、禁止使用 SELECT * 必须使用 SELECT <字段列表> 查询

2、避免使用子查询，可以把子查询优化为 join 操作

3、充分利用表上已经存在的索引

4、避免使用 JOIN 关联太多的表

5、减少同数据库的交互次数

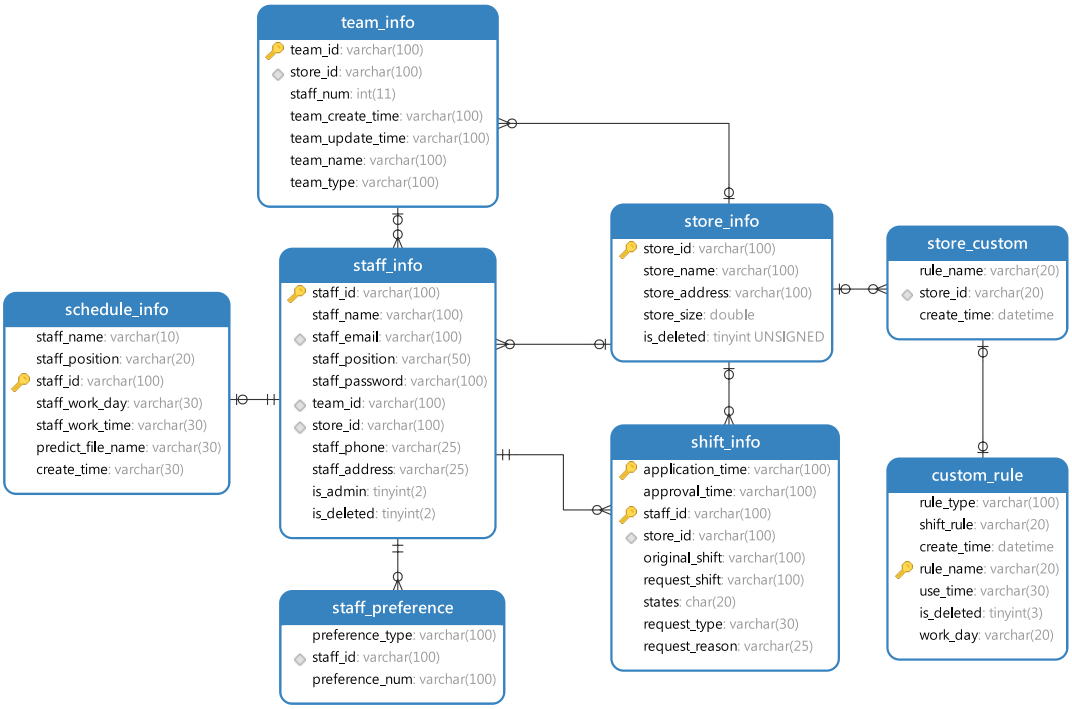
6、对应同一列进行 or 判断时，使用 in 代替 or

7、对应同一列进行 or 判断时，使用 in 代替 or

8、拆分复杂的大 SQL 为多个小 SQL

9、优化 groupBy 尽量在此之前把无用数据过滤掉，having 效率较低

ER 图



4. 系统详细设计

4.1 算法设计

4.1.1 数据预处理

xlsx 预测文件导入：通过自定义工具类把预测文件转化为预测文件实体类的 list。

自定义排班规则导入：通处理数据库里点自定义排班规则信息，转化为自定义的 CustomFinalRule 类。

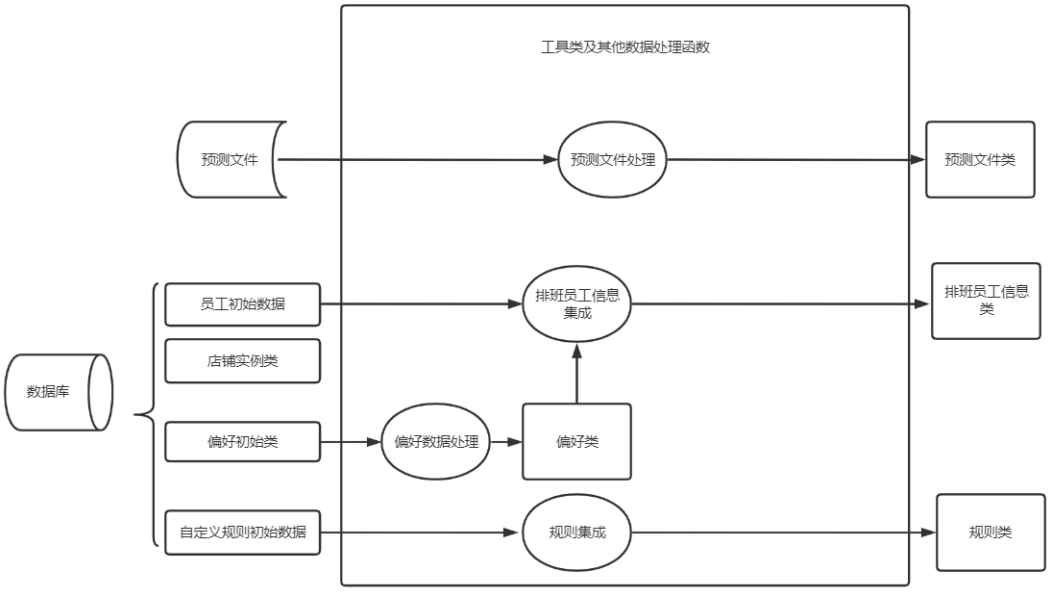
员工偏好导入：通处理数据库里点自定义排班规则信息，转化为自定义的

PreferenceValue 类。

员工信息导入：把员工信息存储到员工信息实体类 list，并通过一周最多工作 40 小时的机制选出可继续工作的员工 list，并把每个员工类实体和一个代表一天内工作时间的 Integer list 列表以及该员工偏好处理过后的 PreferenceValue 对象合并处理为自定义的 ScheduleStaff 类。

门店信息导入：从数据库过得门店信息，并存储为门店实体类的 list。

规则整合：把固定排班规则和之自定义排班规则的自定义类 CustomFinalRule 合并到自定义的总规则类 TheRule

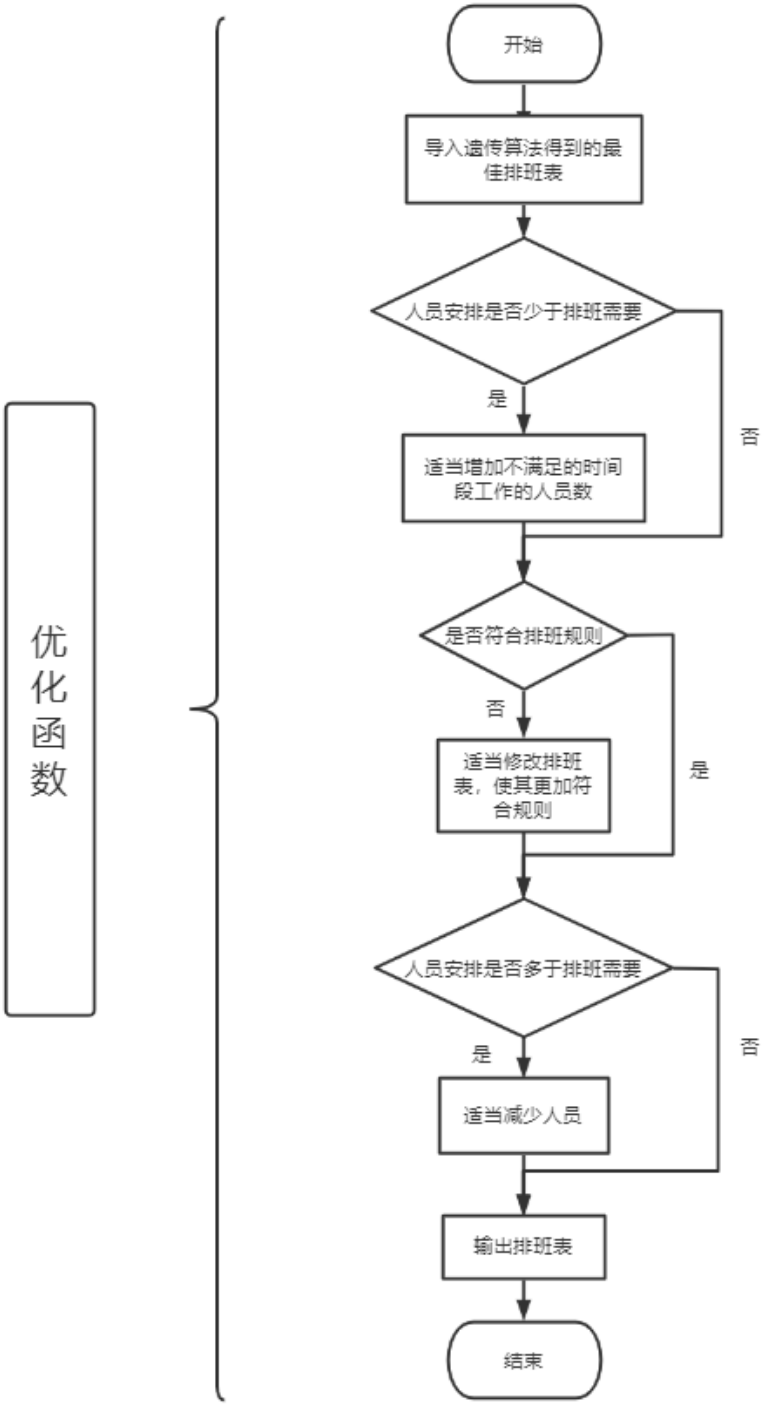


4.1.2 优化函数及适应度函数

适应度函数的实现：适应度函数主要作用是根据排班规则和员工偏好等软硬约束条件判断排班表是否合格，并根据一些评定细则进行打分。分值的计算公式为[需要的总时间数/(实际的总时间数+惩罚分数)]，惩罚分数的评定主要有以下几个方面的评分环节：是否符合员工需求数、是否符合固定排班规则、是否符合自定义排班规则、是否符合员工偏好。当违反的规则越多，员工工作总时间越长，适应度就越低。

优化函数的实现：算法生成返回的排班表通常会存在些许误差，因此，为了让排班表尽可能的符合规则，就自定义了一个优化函数，优化函数的主要功能就是将一个不符合硬约束排班规则的排班表变成符合排班规则的排班表，因此，其实现和适应度函数的部分内容相似，主要有三部分组成：不足员工数补足、按照规则修正排班表、时间段多余员工减少。具体流程为：排班优化函数内先对是否

满足预测数据进行了验证，员工数低于所需员工预测数据的时间段都会增加员工，然后验证各种排班规则，并对不满足的部分进行修改使其尽可能符合规则，最后判断每个时间段安排的员工数是否超过预测数据所需要的员工数，并在不违反排班规则的前提下进行员工工作时间的适当减少。



4.1.3 算法优化

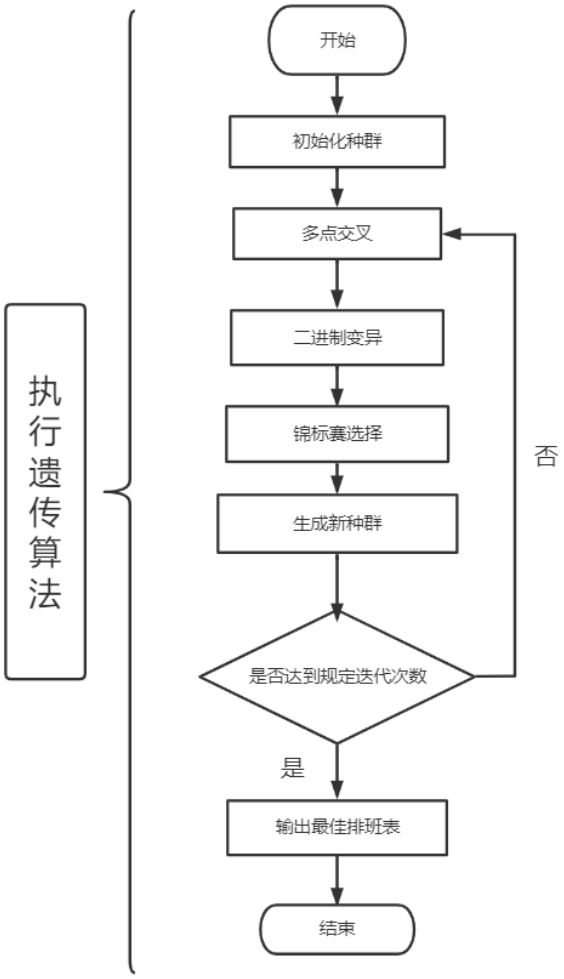
初始化种群：

初始化种群方式采用随机初始化的方式，随机初始化有以下优点：

随机性能够扩展搜索空间，随机初始化能够产生随机的特征，受遗传机制驱动的个体演变中，一些特征会在交叉、变异过程中得到保留或复制，而其他的特征则会被淘汰。这样的处理方式能够使搜索更具有多样性，相对于确定性的初始化过程，不容易陷入局部最优解。

算法具有鲁棒性，随机性初始化的遗传算法具有很好的鲁棒性能，因为它们具有一定的偏差和方差容忍度。它们不会因为初始解集合的差异而失去收敛能力，而且具有较高的容错率。

提高迭代次数效率，某些情况下，通过随机选择的初试解更有可能更接近全局最优解，比起一些基于先验知识选择的更加优秀。基于先验知识的初始化种群要求较高的专业知识水平，以最大限度地控制搜索空间，不能保证在每一个应用场景下都能获得最佳性能。而随机性初始化的种群则更加通用。



变异函数：

变异函数采用二进制变异，二进制变异有以下优点：

二进制变异可以降低局部最优解的概率，在遗传算法中随机选择变异基因，通过改变一些基因位的值，可以有效地增加全局搜索范围。而二进制变异是通过将某些随机选择的位从 0 翻转成 1 或从 1 翻转为 0 来实现的。因为二进制变异引起的变化是只有一个或少量位间的变化，相对于其他变异方式也更侧重于增加多样性。因此，二进制变异有效地降低了算法陷入局部最优解的概率，从而为算法的全局最优解提供了更大的机会。

二进制变异的统一性良好，二进制变异具有二元的特征，该特征显著地提高了算法的统一性和可扩展性。算法中出现的大量个体类型、适应度函数和种群规模不同的问题，使用二进制变异可以简化操作，不需要对个体类型做任何特殊的处理，不考虑适应度函数的大小，也不限制种群中的大小。二进制变异需要操作的实际上只是二元码的翻转过程，因此很容易对遗传算法进行改进和适应性拓展。

相对于其他变异方式，二进制变异的计算复杂度更低。只要指定一个变异位数，二元码就可以进行单一或多个基因分布的随机翻转。操作起来非常容易。

交叉函数：

交叉函数采用多点交叉，多点交叉有以下优点：

可以增加算法多样性，多点交叉是在一个或多个点上选择不同的基因片段进行交叉操作，这样就可以生成多个交叉后的后代。这种生成后代的方式可以带来多样性，从而增加遗传算法的多样性，增强算法的全局搜索能力。因为多点交叉会将不同的基因片段交换，从而产生新个体，可以使得个体的基因组更加多样化，有助于发现不同的适应度最优解。

可以避免陷入局部最优解，多点交叉可以产生更多的变化和更多的组合方式，因此有助于避免陷入局部最优解。通过选择不同的交叉点，可以增加交叉的深度，从而更好的发掘优化空间。相对于单点交叉的限制性，多点交叉可以更有效地生成多样性，从而避免算法在局部最优解陷入。

多点交叉可以增加算法的搜索性能，多点交叉可以利用更多的信息来创造新的个体，在遗传算法中经常用于增加算法的搜索性能。通过选择不同的交叉点，可以产生不同的交叉方式，从而创建出更多的后代。增加后代数量有助于增加算法的搜索能力，可更全面地搜索解空间，提高算法的全局最优解。

多点交叉可控性强，多点交叉是对个体基因片段进行选择性交换，交换方式很灵活，可以控制交换的片段长度、交换的数量以及交换点的位置，相对于其他交叉方式，更容易调节交叉的强度和效果。多点交叉也可以用来增加算法的多样性，但它也能够控制多样性的程度，以满足不同的搜索需求。

选择函数：

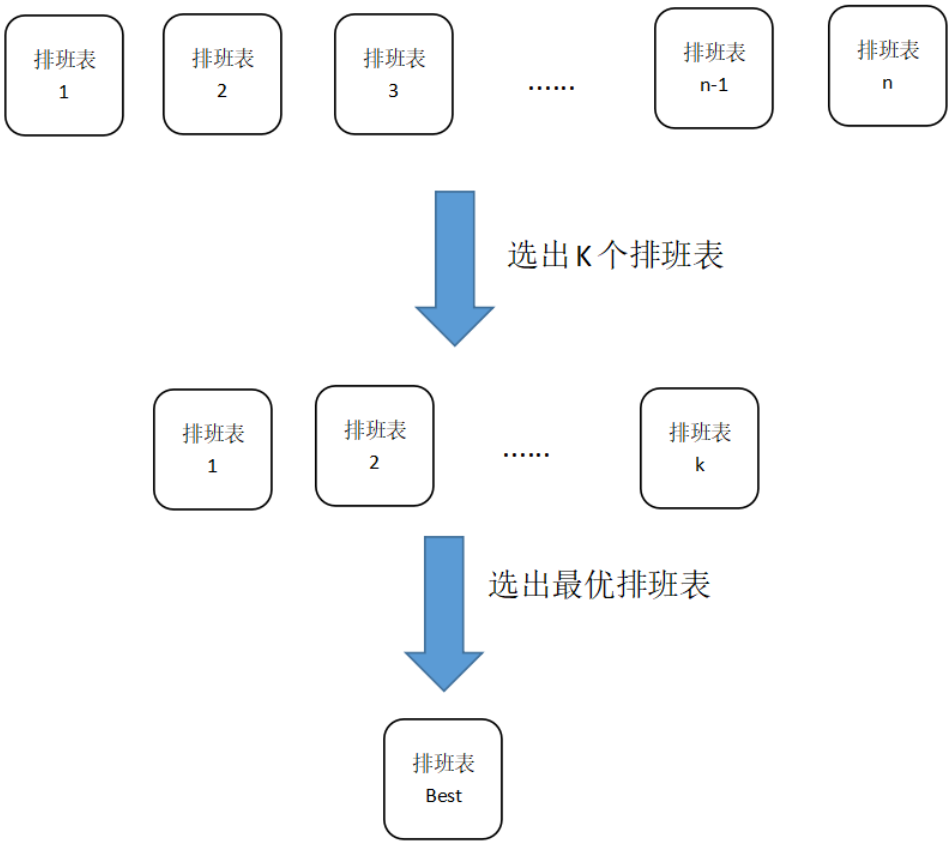
选择函数采用锦标赛选择法，锦标赛选择法有以下优点：

锦标赛选择法是一种非常简单的选择机制，易于实现。它只需要考虑各个个体之间基于适应度值大小的比较，方法简单粗暴，也不需要对个体进行排序、加权等处理。因此，锦标赛选择法具有实现简单、易于理解等优点。

锦标赛选择法不需要进行排序，因此相对于其它选择方法，不容易出现类似极端情况下选择不合理的问题。在选择时只比较多个个体中的最小适应度值，可以有效避免出现选择过小或太大的问题。同时，锦标赛选择法可灵活度调节的因素多，具有较高的适应性。

在锦标赛选择法中，我们可以自由调节每组参赛个体的个数，因此，它具有较好的可控性。因为我们可以尝试使用不同大小的组，在不同的数据集和问题情形，使用相应数量的组，可以更好地调整算法方案，适应更多的任务类型。

锦标赛选择法是相对激进的选择方法，相比于其他选择算法，其选择概率可能在单一条件复杂的情况下呈现较快的收敛速度。在某些情况下，锦标赛选择法可以使得选择效果更加突显，为最终找到更接近最优解的可能性提供了更高的概率。

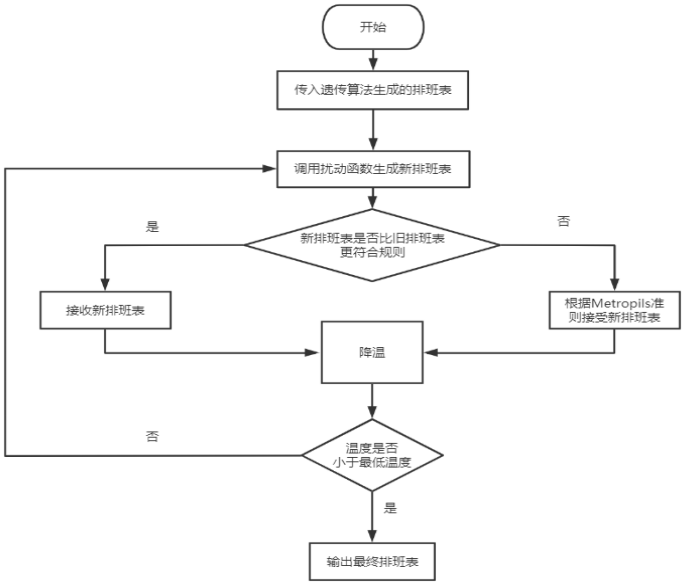


模拟退火算法：

扰动函数：采用自定义的优化函数

分数评估函数：采用自定义的适应度函数

参数设置：由于模拟退火算法处理的排班表为一个经过了遗传算法处理的初步排班表，满足一部分排班约束，因此，初始温度设置为30000，降温率设置为0.95。



4.2 数据库设计

store_info 门店表

该表存储记录门店基本信息

编号	名称	数据类型	主键	说明
1	store_id	varchar	Y	门店 id
2	store_name	varchar	N	门店名
3	store_address	varchar	N	门店地址
4	store_size	double	N	门店大小
5	is_deleted	tinyint	N	是否删除

说明：

1. store_id：门店 id，唯一性约束，通过 id 来查询获得门店信息，也通过门店 id 和其他表进行联系。
2. store_name：有些功能会需要用门店名来查询门店信息。
3. store_address：门店地址，记录门店地址信息。
4. store_size:门店面积,用 double 更加符合实际,并且要进行一些运算。
5. is_deleted：门店是否废弃的标记，0 表示正常运营，1 表示已废弃。

staff_info 员工表

该表存储员工基本信息

编号	名称	数据类型	主键	说明
1	staff_id	varchar	Y	员工 id
2	staff_name	varchar	N	在页面上显示员工姓名
3	staff_email	varchar	N	员工邮件地址。
4	staff_position	varchar	N	员工职位
5	staff_password	varchar	N	员工密码
6	team_id	varchar	N	班组编号
7	store_id	varchar	N	用于唯一标识门店，业务预测数据使用 ID 来关联门店
8	staff_phone	varchar	N	员工的电话号码
9	staff_address	varchar	N	员工的常用住址
10	is_admin	tinyint	N	员工的权限标记
11	is_deleted	tinyint	N	员工离职标记

说明：

1. staff_id，员工 id，唯一性约束，通过员工 id 对员工信息进行查询修改等操作
2. staff_name：员工名，记录员工姓名
3. staff_email：员工邮件地址，唯一性约束，可以作为用户登录名。
4. staff_position：员工职位，可选值：门店经理，副经理，小组长，店员，登陆时，通过验证不同职位，可以获取到不同的权限。
5. staff_password：员工密码
6. team_id：班组编号，外键约束，和 team_info 表建立外键约束。
7. store_id：外键约束，用于唯一标识门店，业务预测数据使用 ID 来关联门店。
8. store_phone：员工的电话号码
9. staff_address：员工的常住地址
10. is_admin：员工的权限标记，0 为员工，1 为管理员，2 为超级管理员
11. is_deleted：员工的离职标记，0 表示在岗，1 表示离职

team_info 班组表

该表将员工分为多个班组，通过班组对员工进行便捷的管理

编号	名称	数据类型	主键	说明
1	team_id	varchar	Y	班组 ID
2	store_id	varchar	N	门店 ID
3	staff_num	varchar	N	班组人数
4	team_create_time	varchar	N	班组创建时间
5	team_update_time	varchar	N	班组更新时间
6	team_name	varchar	N	班组名称
7	team_type	varchar	N	班组类型

说明：

1、team_id：班组 id，主键约束，通过班组 id 来对班组进行操作

2、store_id：班组所属的门店 id，外键约束，7.用于唯一标识门店，业务预测数据使用 ID 来关联门店。

3、staff_num：班组的人数，统计班组中员工的人数。

4、team_create_time：班组创建时间，后端在创建新班组时通过 Date 获取当时的时间而生成。

5、team_update_time：班组更新时间，后端在更新班组时通过 Date 获取当时的时间而生成。

6、team_name：班组名称，有些功能会用班组名称进行班组查询。

7、team_type：班组类别，比如早班，晚班等类别。

shift_info 换班申请表

基础数据维护，设置系统有什么角色，角色在系统使用前，由甲方确定。

编号	名称	数据类型	主键	说明
1	application_time	varchar	Y	申请换班的时间
2	approval_time	varchar	N	审核通过的时间
3	staff_id	varchar	N	申请换班的员工 Id
4	store_id	varchar	N	申请换班的员工工作地点 Id
5	original_shift	varchar	N	原来的班次
6	request_shift	varchar	N	申请调换的班次
7	states	char	N	审核状态（0 待审核 1 通过 -1 拒绝）
8	request_type	varchar	N	申请换班的类型
9	request_reason	varchar	N	申请原因

说明：

1、 application_time:员工申请换班的时间，当员工提交申请时，申请时间会由后端自动生成。

2、 approval_time：审核通过的时间，当员工申请换班被管理员通过时，后端会自动生成。

3、 staff_id：申请换班员工的 id，是一个外键，当员工信息删除时，该表中关于此员工的数据也将被删除。

4、 store_id:申请换班员工所在的门店的 id，为了方便查看该员工的部分信息。

5、 original_shift：记录员工原来需要值班的班次，方便日后进行信息查询。

6、 request_shift：记录员工申请调换的值班班次，方便日后进行信息查询。

7、 states：表示员工是否成功换班的状态，由管理员审核，0 表示待审核，表示管理员还未审核此申请，1 为审核通过，表示管理员同意该员工进行换班，-1 表示审核未通过表示员工的换班申请被管理员拒绝了。

8、 request_type：员工申请的类型，可能有普通请假，紧急申请，换班申请等，具体可由甲方自定义，针对不同的类型在系统中使用不同级别的提示。

9、request_reason：员工申请的原因，具体原因由员工提交。

staff_preference 员工偏好表

该表用于管理有偏好的员工的偏好信息

编号	名称	数据类型	主键	说明
1	preference_type	varchar	N	员工偏好类型
2	staff_id	varchar	Y	员工 id
3	preference_num	varchar	N	员工偏好具体内容

说明：

1、preference_type：表示员工的偏好类型，包括但不限于工作日偏好，工作时间偏好，班次市场偏好。

2、staff_id：员工的 id，作为主键，表示员工身份的唯一性，同时当员工信息表中的员工被删除时，该表关于该员工的信息也会删除。

3、preference_num：表示偏好的具体信息，例如工作日偏好的值：1，2，3 表示偏好在周一，二，三工作工作时间偏好的值：08:00-12:00,18:00:22:00 表示偏好上午 8 点到 12 点和晚上 6 点到 10 点工作。

schedule_info 排班信息表

该表记录员工排班的基本信息

编号	名称	数据类型	主键	说明
1	staff_name	varchar	N	员工姓名
2	staff_position	varchar	N	员工职位
3	staff_id	varchar	Y	员工 id

4	staff_work_day	varchar	N	员工工作日
5	staff_work_time	varchar	N	员工工作时间
6	predict_file_name	varchar	N	预测文件名
7	create_time	varchar	N	创建时间

说明：

1、staff_name：在排班表中的员工的姓名信息

2、staff_position：在排班表中的员工的职位信息

3、staff_id：排班表中员工的 id，是一个外键，当员工信息删除时，该表中关于此员工的数据也将被删除

4、staff_work_day:记录员工在一周中某天进行值班

5、staff_work_time：记录员工某天的某些时间段进行值班

6、predict_file_name：预测文件名，预测文件名表示进行一键智能排班的员工是通过哪个预测文件获得的客流数据从而进行的排班，当预测文件名为空时表示管理员进行自定义排班

7、create_time：表示该员工排班信息是何时创建的，在管理员进行一键排班或者自定义排班后生成。

custom_rule 自定义规则表

该表存储了初固定规则以外某些门店还有特殊的规则来规定排班的要求

编号	名称	数据类型	主键	说明
1	rule_type	varchar	N	规则类型
2	rule_name	varchar	Y	规则名称
3	rule_value	varchar	N	自定义排班规则
4	create_time	datetime	N	创建时间
5	use_time	varchar	N	生效时间
6	is_deleted	tinyint	N	是否弃用
7	work_day	varchar	N	工作日选择

说明：

1、 rule_type：可选包括但不限于准备时间规则，营业时间规则，最少员工数规则，打烊时间规则，最大排班次数规则。

2、 rule_name:规则名称，用于快捷检索。

3、rule_value：准备时间默认值为 1 表示每天开店之前需要 1 小时做准备工作(如做清洁)，所需员工使用上取整[门店面积/100]来获得，打烊时间默认 2 每天关店之后需要 2 小时做收尾工作(如盘点、清算、清洁)。最少员工数默认为 1，营业时间使用上取整[预测客流/3.8]来获得，最大排班次数表示员工每天最多的排班次数，默认为 4。

4、create_time：用于记录规则的创建时间。

5、use_time：用于设置规则的生效时间，用以设置定时任务。

6、is_deleted：是否弃用标记，0 为使用中 1 为弃用。

7、work_day：动态调整上班的工作日，设置调休

store_custom 门店规则映射表

该表存储门店与规则之间的映射关系

编号	名称	数据类型	主键	说明
1	stroe_id	varchar	N	门店的 id
2	rule_name	varchar	N	规则名称
3	create_time	datetime	N	创建时间

说明：

1、 rule_name：规则名称，外键依赖于 custom_rule 表中的 rule_name。

2、 store_id：门店 id,外键依赖于 store_indo 表中的 store_id。

3、 create_time：创建时间，用于区分每条映射

4.3 接口设计

1、 接口名称：发送注册验证码

接口类型：POST

接口路径：/admin/common/code/email

请求参数：

参数名	类型	描述
staffEmail	字符串	员工邮箱
staffPassword	字符串	员工密码
staffPhone	字符串	员工手机号码

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST /admin/common/code/email
请求参数：

{
  "staffEmail": "jane@example.com",
  "staffPassword": "123456",
  "staffPhone": "13812345678"
}

返回参数：
{
  "code": 200,
  "data": {},
  "message": "验证码已发送至员工邮箱，请及时查收。",
  "ok": true
}
```

说明：

该接口用于向员工邮箱发送注册验证码。当请求成功时，返回的数据为空对象，提示信息包含了成功发送的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。

2、 接口名称：员工登录

接口类型：POST

接口路径：/admin/lr/login

请求参数：

参数名	类型	描述
code	字符串	验证码
staffEmail	字符串	员工邮箱
staffPassword	字符串	员工密码

staffPhone	字符串	员工手机号码

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST /admin/lr/login
请求参数：
{
  "code": "123456",
  "staffEmail": "jane@example.com",
  "staffPassword": "123456",
  "staffPhone": "13812345678"
}
返回参数：
{
  "code": 200,
  "data": {},
  "message": "登录成功",
  "ok": true
}
```

说明：

该接口用于实现员工登录功能，需要员工提供邮箱、密码和手机号码。对于验证码，则需要添加短信或邮件验证的机制，确保登录的安全性和可信度。当成功登录后，返回的数据为空对象，提示信息包含了成功登录的信息；当登录失败时，返回的数据为空对象，提示信息包含了失败原因。

3、 接口描述：管理员找回密码接口

请求路径：/admin/lr/findPassword

请求方法：POST

请求参数：

参数名	类型	必填	描述
code	string	是	验证码

confirmPassword	string	是	确认密码
staffAddress	string	否	员工地址，可为空
staffEmail	string	是	员工电邮
staffName	string	是	员工姓名
staffPassword	string	是	员工密码
staffPhone	string	是	员工电话
staffPosition	string	否	员工职位，可为空
storeName	string	是	店名

返回参数：

参数名	类型	描述
code	整型	返回状态码，0 表示成功
data	对象	返回的数据对象
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST  /admin/lr/findPassword
请求参数：
{
  "code": "123456",
  "confirmPassword": "mypassword",
  "staffAddress": "No. 123, Main Street, New York",
  "staffEmail": "staff@example.com",
  "staffName": "John Doe",
  "staffPassword": "mypassword",
  "staffPhone": "1234567890",
  "staffPosition": "Sales Associate",
  "storeName": "ABC Store"
}
返回参数：
{
  "code": 0,
  "data": {},
```

```
  "message": "密码修改成功",
  "ok": true
}
```

说明：

该接口用于员工找回密码。当请求成功时，返回的数据为空对象，提示信息包含了成功找回密码的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。。

4、 接口名称：退出登录

接口类型：POST

接口路径：/admin/lr/loginOut

请求参数：无

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST /admin/lr/loginOut
请求参数：无

返回参数：
{
"code": 200,
"data": {},
"message": "退出登录成功",
"ok": true
}
```

说明：

该接口用于员工退出登录。当请求成功时，返回的数据为空对象，提示信息包含了成功退出登录的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。

5、 接口名称：注册

接口类型：POST

接口路径：/admin/lr/register

请求参数：

参数名	类型	描述
code	字符串	注册验证码
confirmPassword	字符串	确认密码
staffAddress	字符串	员工地址
staffEmail	字符串	员工邮箱
staffName	字符串	员工姓名
staffPassword	字符串	员工密码
staffPhone	字符串	员工手机号码
staffPosition	字符串	员工职位
storeName	字符串	所属门店名称

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST /admin/lr/register
请求参数：

{
"code": "1234",
"confirmPassword": "123456",
"staffAddress": "address",
"staffEmail": "jane@example.com",
"staffName": "jane",
"staffPassword": "123456",
"staffPhone": "13812345678",
"staffPosition": "position",
"storeName": "store"
}
```

```
返回参数：
{
"code": 200,
"data": {},
"message": "注册成功",
"ok": true
}
```

说明：

该接口用于员工注册。当请求成功时，返回的数据为空对象，提示信息包含了成功注册的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。

6、 接口名称：查询所有员工信息

接口类型：GET

接口路径：/admin/Staff/queryAllStaff

请求参数：无

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
GET /admin/Staff/queryAllStaff
请求参数：无
返回参数：
{
"code": 200,
"data": [
 {
      "staffId": "0",
      "staffName": "admin",
      "staffEmail": "admin@qq.com",
      "staffPosition": "无",
      "storeId": "1",
      "staffPassword": "507f513353702b50c145d5b7d138095c",
```

```
      "teamId": null,
      "staffPhone": "17690575215",
      "staffAddress": "总公司",
      "isAdmin": 2,
      "isDeleted": 0
    },
    {
      "staffId": "1",
      "staffName": "sd",
      "staffEmail": "1849192989@qq.com",
      "staffPosition": "店员",
      "storeId": "1",
      "staffPassword": "e10adc3949ba59abbe56e057f20f883e",
      "teamId": null,
      "staffPhone": null,
      "staffAddress": null,
      "isAdmin": 0,
      "isDeleted": 0
    },
],
"message": "查询员工信息成功",
"ok": true
}
```


说明：

该接口用于查询所有员工的信息。当请求成功时，返回的数据包含了所有员工信息，提示信息包含了成功查询员工信息的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。

7、 接口名称：通过员工 ID 查询员工信息

接口类型：GET

接口路径：/admin/Staff/queryStaffById

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	查询到的员工信息

message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
GET /admin/Staff/queryStaffById?staffId=123

请求参数：
{
"staffId": "123"
}

返回参数：
{
  "staffId": "0",
  "staffName": "admin",
  "staffEmail": "admin@qq.com",
  "staffPosition": "无",
  "storeId": "1",
  "staffPassword": "507f513353702b50c145d5b7d138095c",
  "teamId": null,
  "staffPhone": "17690575215",
  "staffAddress": "总公司",
  "isAdmin": 2,
  "isDeleted": 0
},
"message": "查询员工信息成功",
"ok": true
}
```

说明：

该接口用于查询指定员工 ID 的员工信息。当请求成功时，返回的数据包含了查询到的员工信息，其中包括员工 ID、姓名、年龄、性别、职位和部门信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。同时，为了确保查询的准确性，员工 ID 是必填项。

8、 接口名称：根据员工 ID 更新员工信息

接口类型：PUT

接口路径：/admin/Staff/updateStaffByStaffId

请求参数：

参数名	类型	描述
staffAddress	字符串	员工地址
staffEmail	字符串	员工邮箱
staffName	字符串	员工姓名
staffPassword	字符串	员工密码
staffPhone	字符串	员工手机号码
staffPosition	字符串	员工职位
storeName	字符串	所属门店名称

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
PUT /admin/Staff/updateStaffByStaffId
请求参数：
{
  "staffAddress": "new address",
  "staffEmail": "new email",
  "staffName": "new name",
  "staffPassword": "new password",
  "staffPhone": "new phone number",
  "staffPosition": "new position",
  "storeName": "new store"
}
返回参数：
{
  "code": 200,
  "data": {},
  "message": "更新员工信息成功",
  "ok": true
}
```

说明：

该接口用于根据员工 ID 更新员工的信息。当请求成功时，返回的数据为空

对象，提示信息包含了成功更新员工信息的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。

9、 接口名称：新增员工

接口类型：POST

接口路径：/admin/Staff/insertStaff

请求参数：

参数名	类型	描述
staffAddress	字符串	员工地址
staffEmail	字符串	员工邮箱
staffName	字符串	员工姓名
staffPassword	字符串	员工密码
staffPhone	字符串	员工手机号码
staffPosition	字符串	员工职位
storeName	字符串	所属门店名称

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
POST /admin/Staff/insertStaff

请求参数：

{
"staffAddress": "new address",
"staffEmail": "new email",
"staffName": "new name",
"staffPassword": "new password",
"staffPhone": "new phone number",
"staffPosition": "new position",
"storeName": "new store"
```

```
}

返回参数：
{
"code": 200,
"data": {},
"message": "添加员工成功",
"ok": true
}
```

说明：

该接口用于新增员工信息。当请求成功时，返回的数据为空对象，提示信息包含了成功添加员工的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。在添加员工时，必须确保员工信息的完整性，每个字段都是必填项。

10、 接口名称：根据员工 ID 删除员工信息

接口类型：DELETE

接口路径：/admin/Staff/deleteStaffByStaffId

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
DELETE /admin/Staff/deleteStaffByStaffId?staffId=123

请求参数：
{
"staffId": "123"
}

返回参数：
```

```
{
"code": 200,
"data": {},
"message": "删除员工信息成功",
"ok": true
}
```

说明：

该接口用于根据员工 ID 删除员工信息。当请求成功时，返回的数据为空对象，提示信息包含了成功删除员工信息的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。同时，为了确保删除的准确性，员工 ID 是必填项。

11、 接口名称：将员工权限降级

接口类型：GET

接口路径：/admin/Staff/deprivePermissionToStaff

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

GET /admin/Staff/deprivePermissionToStaff?staffId=123

请求参数：

```
{
"staffId": "123"
}

返回参数：
{
"code": 200,
"data": {},
```

```
"message": "员工权限剥夺成功",
"ok": true
}
```

说明：

该接口用于将员工的权限降级。当请求成功时，返回的数据为空对象，提示信息包含了成功将员工的权限剥夺的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。同时，为了确保操作的准确性，员工 ID 是必填项。

12、 接口名称：给员工授权

接口类型：GET

接口路径：/admin/Staff/givePermissionToStaff

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	成功时的数据
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
GET /admin/Staff/givePermissionToStaff?staffId=123
请求参数：
{
"staffId": "123"
}
返回参数：
{
"code": 200,
"data": {},
"message": "给员工授权成功",
"ok": true
}
```

说明：

该接口用于给员工授权。当请求成功时，返回的数据为空对象，提示信息包

含了成功将员工授权的信息；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。同时，为了确保操作的准确性，员工 ID 是必填项。

13、 接口名称：查询员工偏好列表

接口类型：GET

接口路径：/admin/StaffPreference/getAllStaffPreference

请求参数：无

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象数组	员工偏好列表
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

返回的**对象数组**每个元素包含以下参数：

参数名	类型	描述
preferenceType	字符串	偏好类型
staffId	字符串	员工 ID
workDayPreference	字符串数组	工作日偏好
workTimePreference	字符串数组	工作时间偏好
shiftDurationPreference	字符串数组	班次时长偏好
customPreference	字符串数组	自定义偏好
staffName	字符串	员工名字
staffEmail	字符串	员工邮箱

返回**字符串数组**包含以下参数：

数组名	类型	描述
workDayPreference	字符串	1-7 的字符串
workTimePreference	字符串	24H 制的时间段
shiftDurationPreference	字符串	N 小时的字符串
customPreference	字符串	用户自定义

接口示例：

```
GET /admin/StaffPreference/getAllStaffPreference
```

返回参数：


```
{
"code": 200,
"data": [
    {
      "preferenceType": 011,
      "staffId": "1637838846851772418",
      "workDayPreference": [
        "1",
        "  5",
        "  6",
        " 2",
      ],
      "workTimePreference": [
        "07:00-11:00",
        "     12:00-16:00",
        " 09:00-13:00",
        " 12:00-16:00"
      ],
      "shiftDurationPreference": [
        "5 小时"
      ],
      "customPreference": ["中午休息两小时"],
      "preferenceNum": null,
      "staffEmail": null,
      "staffName": "程诚成"
    },
    {
      "preferenceType": 021,
      "staffId": "1638353475772338178",
      "workDayPreference": [
        "1",
        " 5",
        " 6"
      ],
      "workTimePreference": [
        "07:00-11:00",
        "    12:00-16:00"
      ],
      "shiftDurationPreference": [
        "5 小时"
      ],
      "customPreference": ["四点之前下班"],
      "preferenceNum": null,
```

```
      "staffEmail": null,
      "staffName": "李四"
    },
  ],
"message": "查询员工偏好列表成功",
"ok": true
}
```


说明：

该接口用于查询所有员工的偏好设置列表。当请求成功时，返回的数据包含了员工列表及其对应的偏好设置；当请求失败时，返回的数据为空数组，提示信息包含了失败的原因。

14、 接口名称：添加员工偏好设置

接口类型：POST

接口路径：/admin/StaffPreference/addStaffPreference

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID
staffName	字符串	员工姓名
staffEmail	字符串	员工邮件地址
preferenceType	字符串	偏好类型
preferenceNum	整型	偏好设置数量
customPreference	数组	自定义偏好设置
workTimePreference	数组	工作时间偏好设置
workDayPreference	数组	工作日偏好设置
shiftDurationPreference	数组	轮班时间偏好设置

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	新增员工偏好设置
message	字符串	请求返回的提示信息

ok	布尔型	请求是否成功

接口示例：

```
POST /admin/StaffPreference/addStaffPreference
请求参数：
{
"staffId": "123",
"staffName": "张三",
"staffEmail": "zhangsan@example.com",
"preferenceType": "custom",
"preferenceNum": 3,
"workDayPreference": ["1","6","4"],
"workTimePreference": ["10:.00-14:00","16:30-18:00","14:30-15:30"],
"shiftDurationPreference": ["2 小时","3.5 小时"],
"customPreference": ["早上十点上班"]
}
返回参数：
{
"code": 200,
"data": {},
"message": "新增员工偏好设置成功",
"ok": true
}
```

说明：

该接口用于新增员工的偏好设置。当请求成功时，返回的数据包含了新增的员工偏好设置；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。其中，staffId、staffName 和 staffEmail 是必填项；preferenceType 为偏好类型，这里可以选择 custom、workTime、workDay 或 shiftDuration；preferenceNum 为偏好设置个数，具体个数跟偏好类型相关；customPreference、workTimePreference、workDayPreference 和 shiftDurationPreference 是对应的偏好设置数组。

15、 接口名称：删除员工偏好设置

接口类型：DELETE

接口路径：/admin/StaffPreference/deleteStaffPreference

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	删除的员工偏好设置
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
DELETE /admin/StaffPreference/deleteStaffPreference?staffId=123
返回参数：
{
"code": 200,
"data": [],
"message": "删除员工偏好设置成功",
"ok": true
}
```

说明：

该接口用于删除员工的偏好设置。当请求成功时，返回的数据包含了被删除的员工偏好设置；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。注意，该接口仅能删除员工的偏好设置，无法删除员工数据。

16、 接口名称：修改员工偏好设置

接口类型：PUT

接口路径：/admin/StaffPreference/updateStaffPreference

请求参数：

参数名	类型	描述
staffId	字符串	员工 ID
staffName	字符串	员工姓名
staffEmail	字符串	员工邮件地址
preferenceType	字符串	偏好类型
preferenceNum	整型	偏好设置数量
customPreference	数组	自定义偏好设置
workTimePreference	数组	工作时间偏好设置

workDayPreference	数组	工作日偏好设置
shiftDurationPreference	数组	轮班时间偏好设置

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	修改后的员工偏好设置
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
PUT /admin/StaffPreference/updateStaffPreference
请求参数：
{
"staffId": "123",
"staffName": "张三",
"staffEmail": "zhangsan@example.com",
"preferenceType": "custom",
"preferenceNum": 3,
"workDayPreference": [
        "1",
        " 5",
        " 6",
        " 2",
      ],
"workTimePreference": [
        "07:00-11:00",
        "12:00-16:00",
        " 09:00-13:00",
        "12:00-16:00"
      ],
"shiftDurationPreference": [
        "5 小时"
      ],
"customPreference": ["早上十点上班"]
}
返回参数：
{
```

```
"code": 200,
"data": {},
"message": "员工偏好设置更新成功",
"ok": true
}
```


说明：

该接口用于修改员工的偏好设置。当请求成功时，返回的数据包含了修改后的员工偏好设置；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。注意，该接口会覆盖原有的偏好设置，是一次完整的修改。

17、 接口名称：查询员工偏好设置

接口类型：GET

接口路径：/admin/StaffPreference/getStaffPreference

请求参数：

参数名	类型	描述
staffName	字符串	员工姓名

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	查询到的员工偏好设置
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
GET /admin/StaffPreference/getStaffPreference?staffName=程诚成
返回参数：
{
"code": 200,
"data": {
"staffId": "123",
"staffName": "程诚成",
"staffEmail": "zhangsan@example.com",
"preferenceType": "custom",
"preferenceNum": 3,
    "workDayPreference": [
        "1",
```

```
        "  5",
        "  6",
        " 2",
      ],
      "workTimePreference": [
        "07:00-11:00",
        "     12:00-16:00",
        " 09:00-13:00",
        " 12:00-16:00"
      ],
      "shiftDurationPreference": [
        "5 小时"
      ],
      "customPreference": ["中午休息两小时"],
},
"message": "查询员工偏好设置成功",
"ok": true
}
```


说明：

该接口用于查询员工的偏好设置。当请求成功时，返回的数据包含了查询到的员工偏好设置；当请求失败时，返回的数据为空对象，提示信息包含了失败的原因。注意，该接口查询员工偏好设置的依据是员工姓名，因此在请求时需传入员工姓名。

18、 接口名称：查询排班信息

接口类型：GET

接口路径：/admin/pb/getPbInf

请求参数：无

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象数组	查询到的员工信息列表
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

员工信息对象包含的属性：

参数名	类型	描述
title	String	员工信息
start	Datatime	班次开始时间
end	Datatime	班次结束时间
position	Stirng	员工职位

接口示例：

```
GET /admin/pb/getPbInf

返回参数：
{
"code": 200,
"data": [
    {
      "title": "员工 0",
      "start": "2023-03-25T07:30:00",
      "end": "2023-03-25T10:30:00",
      "position": "店员"
    },
    {
      "title": "员工 0",
      "start": "2023-03-25T11:00:00",
      "end": "2023-03-25T13:00:00",
      "position": "店员"
    },
    {
      "title": "员工 0",
      "start": "2023-03-25T13:30:00",
      "end": "2023-03-25T14:00:00",
      "position": "店员"
    }
],
"message": "查询排班信息成功",
"ok": true
}
```

说明：

该接口用于查询所有员工的信息。当请求成功时，返回的数据包含了所有员工的信息列表；当请求失败时，返回的数据为空数组，提示信息包含了失败的原因。

19、 接口名称：删除排班信息

接口类型：DELETE

接口路径：/admin/pb/deletePbInfoByCreateTime

请求参数：

参数名	类型	是否必须	描述
createTime	字符串	是	需要删除排班信息的创建时间
storeId	字符串	是	排班信息所属门店编号

返回参数：

参数名	类型	描述
code	整型	返回状态码
data	对象	返回空对象
message	字符串	请求返回的提示信息
ok	布尔型	请求是否成功

接口示例：

```
DELETE /admin/pb/deletePbInfoByCreateTime?storeId=001&createTime=2021-01-01

返回参数：
{
"code": 200,
"data": {},
"message": "该排班信息删除成功",
"ok": true
}
```

说明：

该接口用于删除指定创建时间下属于指定门店的排班信息。当请求成功时，返回的数据为空对象，提示信息为删除成功；当请求失败时，返回的数据为空对象，提示信息为删除失败的原因。注意，在删除排班信息时需要提供所属门店编号和排班信息的创建时间。

注：

① 由于门店信息管理、门店规则管理、换班管理的接口与员工管理的

逻辑一致，本文档对重复度较高的接口不再做单独展示，若要查看详请请参考：门店信息管理接口文档，门店规则管理接口文档，换班管理接口文档。

② 所有的接口返回请求中使用 code 字段，code 的值可取 200、401、402、403、404，其中 200 表明客户端请求已成功并且结果已正常返回，是一个重要的成功状态码。

401 Unauthorized（未授权）是一个 HTTP 状态码，表示客户端尝试访问需要身份验证的资源，但没有提供有效的凭据。

403 Forbidden（禁止访问）是一个 HTTP 状态码，表示服务器已经收到请求，但拒绝授予客户端访问所请求的资源的许可。

404 Not Found（未找到）是一个 HTTP 状态码，表示服务器无法找到所请求的资源。

4.4 前端设计

1. 页面模块化设计

- 顶部导航栏：

在页面顶部固定，包含系统名称、个人头像、信息提示、回到主页等元素。高度为 60px，背景色为蓝色，字体颜色为白色。当鼠标在搜索框内输入内容时，显示自动下拉选择列表。

- 侧边栏：

在页面左侧固定，宽度为 200px，包含页面菜单和相关操作按钮。菜单折叠时宽度自适应至 50px，支持菜单展开/折叠，以及菜单选中状态的变化。按钮样式为矩形带圆角，并附有图标和文字说明。

- 主体内容区：

位于页面右侧，宽度自适应。包含页面主体业务功能、数据展示区、操作按钮等元素。背景色为白色，边框线为细黑线，按钮样式为圆角矩形，鼠标悬停时按钮颜色变深、字体颜色变白，点击后按钮颜色变浅。

- 页面底部：

在页面底部固定，包含版权信息、帮助链接和其他相关内容。高度为 50px，背景色为灰色，字体颜色为白色。版权信息居左，帮助链接和其他相关内容居右，位置自适应。

2. 界面设计

- 页面结构设计：

头部：包含网站 Logo、菜单导航、用户登录/注册按钮等元素。

主体：包含页面主要内容、图文列表、分页等元素。

底部：包含版权信息、友情链接等元素。

- 布局设计：

主体区域采用两栏式布局，左边是导航列表，右边是具体内容展示区域。

- 颜色、字体、样式设计：

颜色以蓝色、灰色为主，字体选择简洁、清晰的 Sans-serif 字体，所有元素采用纯色背景，悬停时进行微小颜色调整，并加上透明效果。

- 图片和图标设计：

网站标志设计采用简洁的单色图标，导航图标全部采用扁平化风格，并且具有视觉可识别性。

- 表单和交互设计：

表单元素使用延时提示浮框，提供有效的辅助信息，帮助用户快速输入。

- 动态交互操作设计：

鼠标悬停在导航菜单时产生弱淡阴影，鼠标悬停在页面主体区域时会产生高亮显示。

3. 状态变化的设计

1. 交互设计：

状态变化的交互设计应该尽量简洁、清晰，避免让用户感到迷惑或不知所措。例如，当用户点击一个按钮时，应该在点击后立即反馈对应的操作，而不是等待服务器响应。如果需要等待响应，则应该给出明确的提示信息和进度条等反馈，告知用户操作正在进行中。

2. 视觉设计：

状态变化的视觉设计应该采用视觉反馈的方式来提示用户不同状态下的变化。例如，当按钮被点击时，按钮应该有一个视觉上的变化，如背景颜色变化、边框变化、文字颜色变化等，以方便用户知道按钮已被点击。

3. 用户体验：

状态变化的设计应该考虑用户体验，即在用户操作时，应该给予用户足够的反馈和时间，让用户知道操作的进程和结果，避免用户产生担心、不信任等负面情绪。例如，在网页加载过程中，可以添加进度条等反馈，让用户知道加载进程

和时间，避免用户产生等待过长和不知道进程的困惑。

一个比较好的状态变化设计示例是：提交订单按钮。

当用户在填写订单信息后，点击提交订单按钮时，按钮应该发生状态变化，比如背景颜色变深、文字颜色变白，转动的进度条出现等，以给用户明确的反馈。此外，需要对订单信息进行校验，如果有问题，应该给出明确的提示信息。

当跳转到下一个页面时，应该显示进度条并禁用提交按钮，以防止用户重复提交或取消操作。在跳转过程中，应该给出明确的提示信息，告知用户操作正在进行中。

当订单提交成功时，应该给出成功的提示信息，并跳转到对应的页面，以让用户知道操作已成功完成。如果提交失败，应该给出失败原因和解决方法，并允许用户重新填写和提交订单。

4. 页面交互设计

1. 列表的交互

列表是 Web 应用中最基本、用得最多的元素之一，因此其交互设计也相当重要。常见的交互方式有：

- 悬停变色：当用户鼠标移至列表项上时，该项背景色或者边框颜色变化，以为用户提供反馈。
- 显示操作项：当用户将鼠标放至列表项上时，会显示一个操作菜单，如删除、编辑等。
- 拖动排序：用户可以通过拖动列表项的方式进行排序或者交换位置。

2. 表单的交互

- 根据需求动态增加表单项：当用户需要添加新的表单项时，可以通过点击“新增”按钮来实现。
- 检查输入项：当用户在输入表单数据时，应及时给出提示信息并检查输入项。
- 表单联动：当表单输入项存在联动关系时，需要相互关联，以加强用户体验。

3. 动画效果

动画效果对于页面设计来说非常重要，可以增强页面的视觉冲击力和用户体验。将动画效果融入到页面过渡、交互事件和滚动效果中，可以让整个界面更具生命力，提高用户参与感和用户体验。

4. 图片的交互

- 点击放大：当用户想要查看更大、更清晰的图片时，可以通过点击图片来实现。
- 轮播切换：当要展示多张图片时，可以采用轮播的方式展示。

5. 设计时的约定和限制

1. 统一 UI 元素和布局：设计师要统一 UI 元素的风格、颜色、图标、按钮等等细节，以确保页面的风格一致性，减少开发人员对 UI 的疑惑和修改。
2. 变量和标准化：在设计时需要考虑到变量和标准化，包括颜色、代码命名以及 CSS 规范等等，这样可以更方便地让团队开发人员理解设计。
3. 可复用性：在设计时要考虑到可复用性，需要判断 UI 元素是否可以灵活地移植到其他页面，或者在不同页面重复使用，这样可以大大减少制作大量相似页面的时间和工作量。
4. 响应式设计：在设计时需要考虑到响应式设计，这样可以确保网站可以适应不同设备和屏幕大小，让用户在不同设备上都能够舒适的使用应用程序。
5. 与后端开发的配合：在前端设计时，需要考虑与后端开发人员的配合。设计师应该充分了解后端 API 的数据模型和请求方法，并设计符合 API 数据模型的前端页面。
6. 兼容性和性能：设计师在设计前端页面时，需要考虑到兼容性和性能，例如使用优化图片大小、懒加载、最小 HTML 模板等等优化手段来优化页面性能，比如使用兼容性强的 jQuery 插件可以减少浏览器兼容性的问题。

4.5 后端设计

2. 模块划分

智能排班系统的模块划分可以从以下几个方面进行：

1.1 员工信息管理模块

- 包括员工基本信息、工作经历、培训记录等内容。
- 提供员工信息维护、查询等接口。
- 有员工信息审批子模块。

1.2 排班管理模块

- 负责排班信息的管理，包括起止时间、员工分配等。

- 提供排班信息录入、删除、查询、导出等接口。
- 有排班审批子模块。

1.3 请假审批模块

- 负责请假申请、请假审批等业务。
- 提供请假申请、请假审批、查询请假记录等接口。

1.4 系统管理模块

- 包括用户管理，角色与权限管理等。
- 提供用户注册、登录等接口。
- 可视化界面，方便管理员使用。

2.5 门店信息管理模块

- 包括门店基本信息、门店排班规则、门店职能列表等内容。
- 提供门店信息维护、查询等接口。

3. 技术栈和框架选择

2.1 后端开发技术栈

选择 Java、Python 等语言开发后端。

2.2 框架选择

Spring Boot：可以快速创建基于 Spring Framework 的业务应用程序，提供丰富的开箱即用的特性。

Spring Cloud: 一系列框架的有序集合，它利用 Spring Boot 的开发便利性简化了分布式系统的开发，比如服务发现、服务网关、服务路由、链路追踪等。

HDFS：适合运行在通用硬件(commodity hardware)上的分布式文件系统，提供高吞吐量的数据访问。

Kafka：一个分布式、高吞吐量、高扩展性的消息队列系统。

Zuul：实现动态路由，监视，弹性和安全性。包含了对请求的 路由 和 过滤 两个最主要的功能。

4. OOP 和设计模式

3.1 面向对象编程（OOP）

采用面向对象的编程思想，将程序分解为功能独立的对象，提高代码的可扩展性和可维护性。

3.2 设计模式

采用合适的设计模式可以提高代码质量和可复用性推荐使用：

工厂模式：用于创建对象，可以将复杂对象的构造过程封装起来。

单例模式：保证对象在程序中只被创建一次。

装饰器模式：动态地扩展对象的功能，而不必修改现有的代码。

观察者模式:观察者对象监听主题对象的变化,并在其状态改变时作出响应。

5. 可读性、可维护性和扩展性。

4.1 代码的可读性

编写文档：给出描述代码功能、参数、返回类型等的文档，方便其他人理解代码。

善用注释：在代码中添加清晰的注释，解释代码的作用，在维护代码时有助于理解。

规范命名：变量、函数和类等拥有易于理解的、具有意义的名称，并且要注意大小写规范。

4.2 代码的可维护性

减少代码耦合：采用接口和抽象类等工具进行松耦合设计，避免代码紧密耦合，以便重构和修改。

解耦业务逻辑:将业务逻辑代码分层,在结构化的模块中使用类来解耦部件。

常见数据集中管理：常见的数据（如常量、配置、静态数据）集中存储在配置文件或数据库中，而不是在代码中硬编码，方便修改和维护。

4.3 代码的扩展性

面向接口编程：面向接口开发可以将代码形成松耦合的对象，方便扩展和修改。

合理抽象：将相同功能的代码聚合成独立的、可扩展的单元，方便功能的扩展和重构。

模块化：采用模块化设计和依赖注入，可以让不同的模块之间不紧密耦合，方便后期的拓展。

6. 异常监控机制

在智能排班系统中，通过以下方式监控运行时异常：

5.1 日志记录

在系统中记录日志，记录每个异常的信息，包括时间、错误类型、错误信息、异常栈等。记录详细的异常信息,有助于开发人员快速定位异常并进行问题排查。

5.2 监控工具

采用监控工具对系统性能、异常进行监控。不同的监控工具可以针对不同的异常类型进行监控，如：

APM 工具：可以监控系统性能、CPU 使用率、内存使用情况等。

日志分析工具：可以实时监控系统日志，并根据错误日志的关键字或规则生成告警或报警信息。

异常监控工具：用于监控系统的异常情况，比如未捕获的异常、http 错误、数据库错误等。

7. 告警机制

在系统出现异常时，必须及时通知开发团队并进行相应的处理。常见的告警方式包括：

6.1 邮件通知

发送邮件给相应的开发人员，提醒他们处理异常。邮件中应包含异常类型、发生时间、异常堆栈等信息。

6.2 短信通知

将异常信息发送到指定的手机，通知相应的开发人员及时处理。

6.3 告警处理

开发团队收到异常通知后，应当及时处理异常，快速定位问题并进行相应的修改和优化。

5. 技术选型

5.1 编程语言

Java

Java 是一种广泛使用的面向对象编程语言，具有稳定性、安全性、跨平台特性等，是构建中大型软件系统的理想语言。在 Java 开发中，可以使用 Spring Boot、Hibernate 等框架，以及 Apache Maven、Gradle 等构建工具。

Python

Python 是一种简单、易学、易读、开发效率高的面向对象编程语言，广泛用于 Web 开发、数据科学等领域。在 Python 开发中，可以使用 Flask、Django 等 web 框架，以及 pip、Anaconda 等构建工具。

Node.js

Node.js 是一种开源的跨平台 JavaScript 运行环境，常用于构建可扩展的

服务器端项目。在 Node.js 开发中，可以使用 Express、Vue 等 web 框架，以及 npm 等构建工具。

5.2 开发框架及组件

1. Java 框架

- Spring Boot：简化 Spring 应用的开发、测试、部署等操作的框架。
- Spring Cloud：构建分布式系统、微服务架构的框架。
- Hibernate：面向对象的关系型数据库 ORM 框架，可以简化数据访问层的开发。

2. Python 框架

- Flask：轻量级 Web 框架，可以快速构建 RESTful API 等应用。
- Django：全能型 Web 框架，提供 ORM、模板引擎等工具，适用于构建大型 Web 应用。

3. Node.js 框架

- Express：灵活的 Web 应用程序框架，提供路由、中间件等功能，适合构建 RESTful API 应用。
- Koa：轻量级 Web 框架，可以构建高效的 Web 应用程序。

4. Vue.js 相关

- Vue：渐进式 JavaScript 框架，用于构建用户界面，实现数据驱动和组件化。
- Element UI:基于 Vue.js 的前端组件库,提供了一系列常用的 UI 组件。
- Tippy.js：轻量级的原生 JavaScript 工具提示库，提供了交互性更强的工具提示。
- Echarts.js：可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器，底层依赖轻量级的 Canvas 类库 ZRENDER，提供直观，生动，可交互，可高度个性化定制的数据可视化图表。
- fullcalendar.js: 基于 jQuery 的全日历、事件日历。

5. 服务发现和注册

- Nacos：维护服务的注册和配置中心，包括服务的注册、发现和配置管理等功能。

6. 其他

- Apache Kafka：分布式流数据传输平台和发行/订阅消息系统，可以处理来自多个数据源的实时数据流和批量数据。

- Elasticsearch：分布式全文搜索和分析引擎，可以快速地进行复杂的搜索和数据分析。
- Docker：容器化平台，可以用来封装应用程序和其依赖项，并且可以轻松地进行部署和运行。
- Kubernetes：容器编排系统，可以自动化地管理应用程序的部署、扩展和运行。
- Sentry：实时错误跟踪和异常监控平台，可以帮助开发人员及时发现和解决应用程序中的异常情况，并提供错误日志和统计数据。

5.3 数据库管理系统

Redis：高性能的键值对存储数据库，支持持久化、集群等特性。

Mysql：访问数据库的最常用的标准化语言。由于 MySQL 数据库体积小、速度快，适用于本项目这样的小型系统使用。

5.4 代码管理

Gitee: 一个功能齐全、容易上手的代码托管和开发协作平台，非常适合国内开发者或团队使用。

5.5 运行环境

- 操作系统： Windows 11
- Web 服务器： Nginx 1.15.9
- 应用服务器： Tomcat 9.0.17
- 数据库管理系统： MySQL 5.7.24
- 缓存系统： Redis 5.0.3
- 开发语言： Java 8
- 开发框架： Spring Boot 2.1.4.RELEASE
- 前端框架：Vue.js 2.6.10

6. 开发过程

6.1 项目规划和管理

1. 进度管理的日常工作：

每日或每周固定召开项目进度会议，评估任务进展情况、解决任务中遇到的问题及风险预警。

按时定期跟进各个任务的进度，采用甘特图等工具对任务进度进行可视化管理，以便及时调整和安排。

定期检查每个任务的质量和完成情况，并在任务完成后为整个项目汇总报告。

2. 风险管理的日常工作：

对可能出现的风险进行梳理、量化和分类，制定有效的风险防范措施。在项目进展中及时识别、评估、跟踪和监控风险。

定期更新风险管理计划，随时跟进风险的进展情况，借助项目管理工具、邮件等方式提醒参与者注意风险状况。

建立并完善项目风险管理手册，确保所有团队成员都能够理解、执行和持续改进风险管理措施。

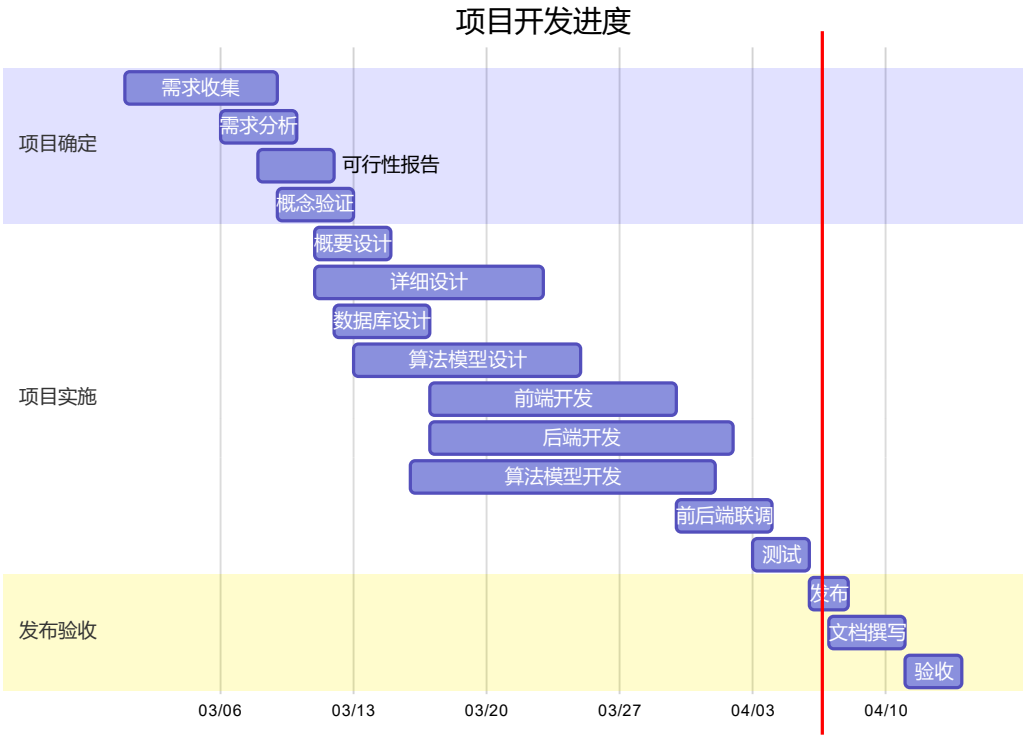
3. 质量管理的日常工作：

针对每个任务和阶段的标准和细节，建立并实施测试计划、质量计划和控制计划。

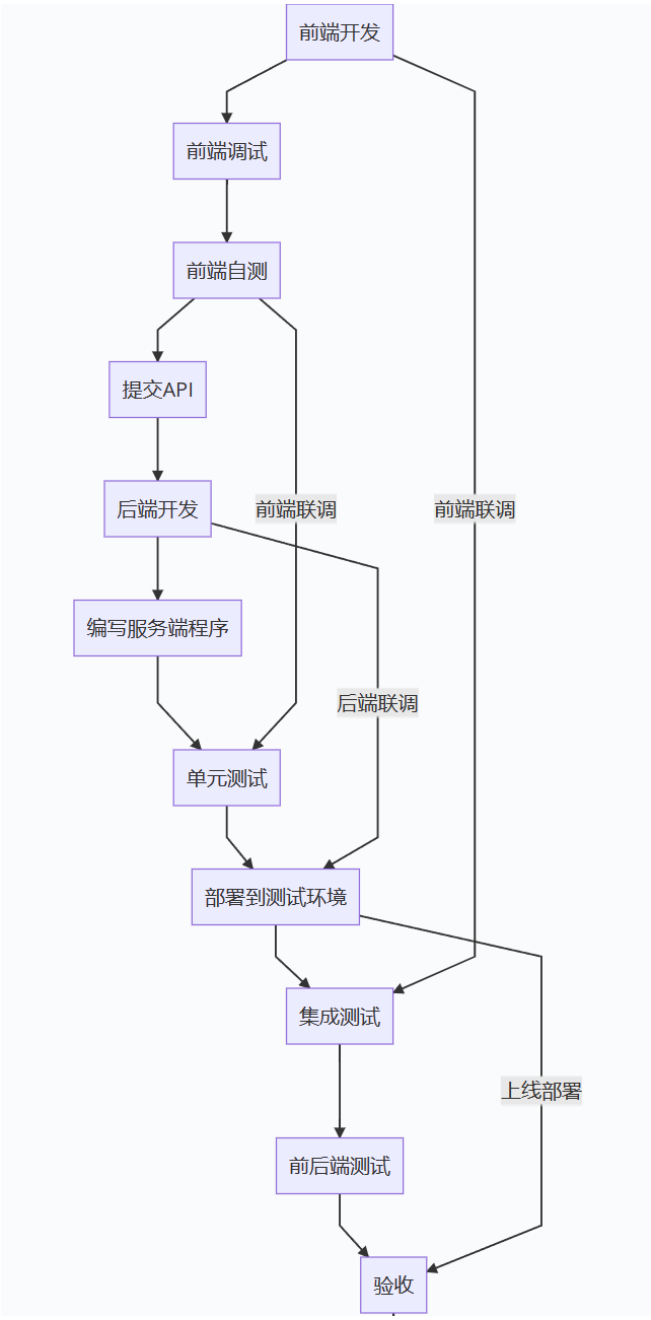
制定各阶段的度量、评估和报告机制，以评估任务的质量和成果，并为工作团队和合作伙伴提供合适的反馈意见。

定期进行质量审核和质量评估，确定项目中的风险，并采取适当的控制方法，以保证整个项目的质量。

4. 项目进度预期：



6.2 开发流程和分工



人员分工：

姓名	人员分工
孙贵林	需求分析、用户界面设计、前端页面开发和维护
贾旺	云服务器搭建、数据库管理、系统部署和实施方案评估
韦富元	后台程序逻辑设计、代码开发、测试和 Bug 修复
王洋	系统性能测试、压力测试、安全测试和结果分析
郑源鹏	算法的设计和开发

6.3 开发规范

一、 命名规范

1.1、类名、接口名使用大驼峰式命名法，如：ClassName、TestInterface。

1.2、变量名、方法名使用小驼峰式命名法，如：variableName、testMethod。

1.3、常量名使用全大写字母，单词间用下划线分隔，如：CONSTANT_NAME。

二、 编码规范

2.1、Java 代码缩进使用四个空格而不是 Tab，不要使用回车作为行尾，文件以 UTF-8 编码。

2.2、if、while、for、switch 等保留字与括号之间必须有一个空格空格，如 if (condition) {}。

2.3、在 Java 文件顶部添加文件描述、版权、作者等注释信息。

三、 注释规范

3.1、方法的注释必须包括方法名、功能、输入输出描述，如：

```
/**
 * 用户登录方法
 * @param username 用户名
 * @param password 密码
 * @return boolean  成功返回 true，否则返回 false
 */
public boolean login(String username, String password) {
  // TODO: 实现用户登录
}
```

3.2、变量的注释必须包括变量名、类型、含义，如：

```
int age;  // 用户年龄
```

6.4 算法对比和评价

对比算法：

初始遗传算法：单纯的遗传算法，调用的自定义的适应度函数。

遗传算法+优化函数：在单独遗传算法基础上加上了优化函数，对最终结果进行优化。

遗传算法+模拟退火算法+优化函数：以遗传算法为基础生成初步最优排班结

果，再用模拟退火算法对排班结果进行优化，优化部分调用自定义优化函数。

评价指标：

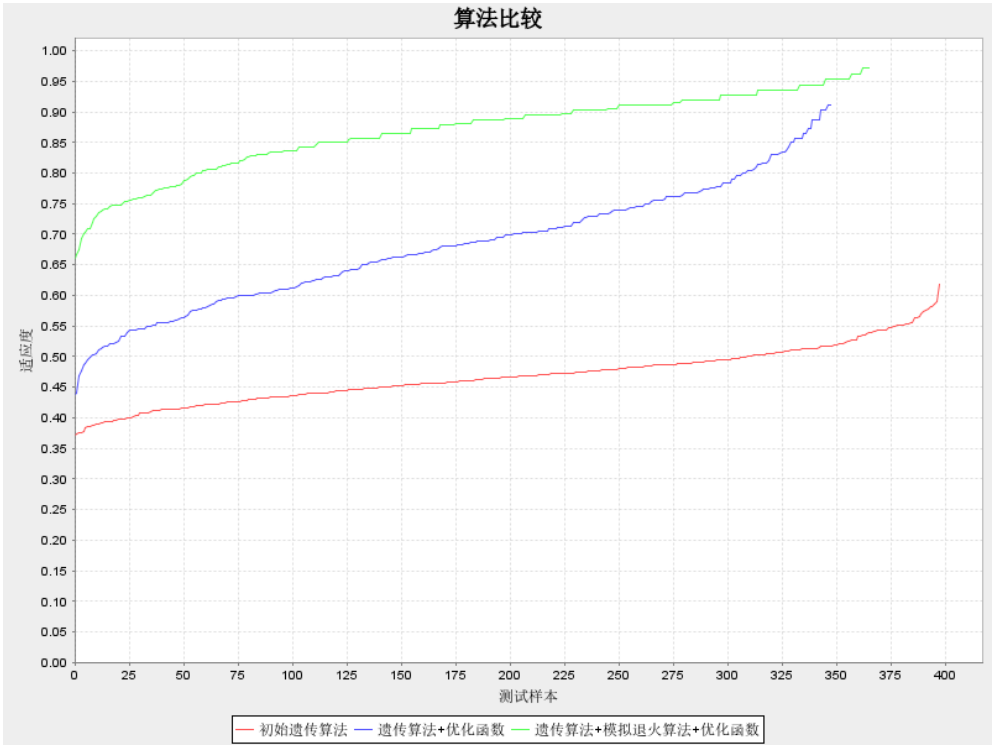
适应度 Fitness：计算公式为[需要的总时间数/(实际的总时间数+惩罚分数)]，违反的规则越多，惩罚分数越大，适应度越小。从总体上反映了排班表是否满足需要。

偏差率：计算公式为[各时间段所需员工数和实际安排员工数差值的平方和/时间段数]，主要反应了排班表是否满足用工人数，总工时超出实际需要的量。

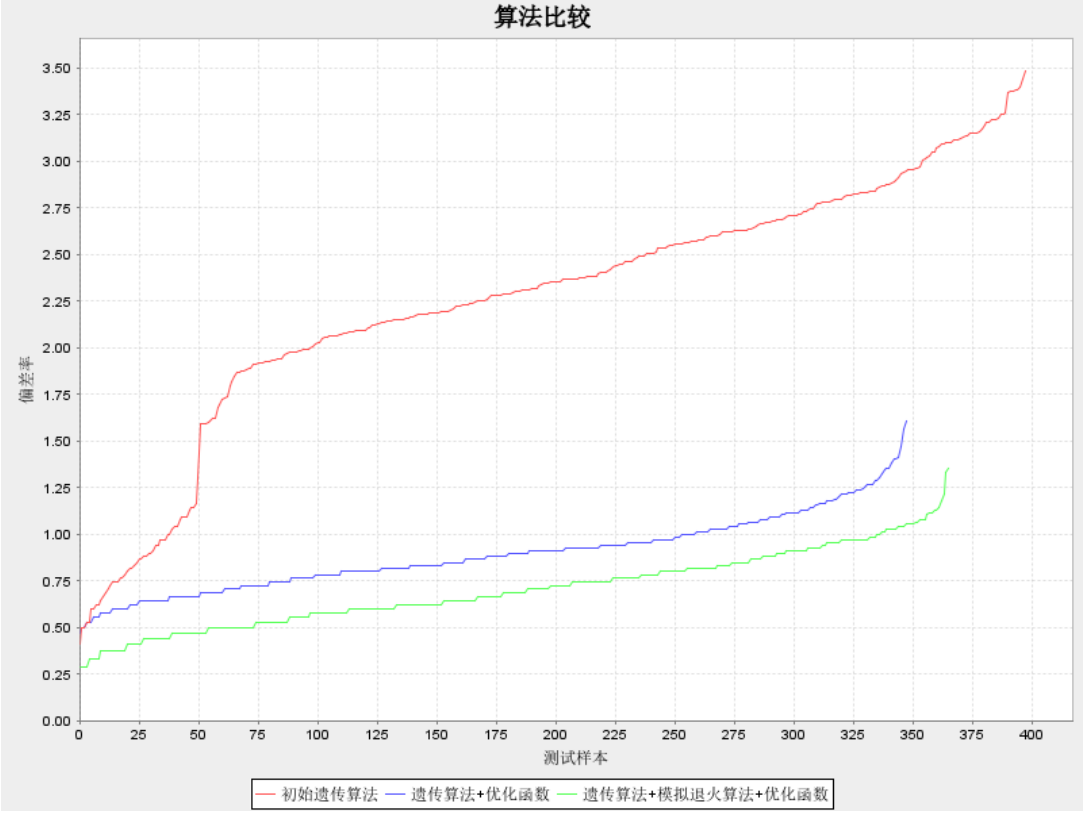
适应度越大，偏差率越小时算法生成的排班表越满足规定。

算法	平均适应度	平均偏差率
初始遗传算法	0.46782549137553675	2.2629839543743233
遗传算法+优化函数	0.6785208789786978	0.8933080944727184
遗传算法+模拟退火算法+优化函数	0.8678767012124087	0.703054773432896

各算法适应度比较：



各算法偏差率比较：



算法发展方向改进：

性能方面：

性能方面，在一些复杂计算层面，采用多线程及并行计算等计算方式或者采用多线程的方式来提升速度。

准确率方面：

种群规模的选择，种群大小是对算法表现有很大影响的一个因素。种群大小过小会导致搜索空间不充分，可能会陷入到局部最优解而无法跳出，而种群大小过大则会增加算法的运行时间，同时也有可能导致算法过度拟合数据。可以考虑采用一些自适应调节策略，根据问题的复杂度和搜索空间大小等因素来选择合适的种群规模。

算法参数的选择，遗传算法和模拟退火算法都有很多参数需要调节，例如交叉率、变异率、退火降温速率等等。这些参数的不同取值会影响算法的表现。可以考虑引入一些更高效的调参工具和算法策略，来优化参数值。

可以更好的优化目标函数，可以考虑采用一些启发式算法结合原有的优化函数，比如粒子群算法、蚁群算法等。也可以将多个优化算法结合起来，形成“集成优化算法”，以便更好地处理更为复杂的排班规则。

7. 系统测试

7.1 系统测试计划

在本次软件测试中，主要采用了黑盒测试法，采用了等价类划分法、边界值分析法对软件界面、软件功能进行测试。

7.2 单元测试和集成测试

1. 员工注册模块

员工注册模块，运用黑盒测试的等价类划分法。

输入数据	有效等价类	无效等价类
姓名 staff_name	（1）姓名不为空	（2）姓名为空
密码 passwpord	（3）密码不为空且符合要求	（4）密码为空 （5）密码不为空但不不符合要求
邮箱 email	（14）邮箱唯一	（6）邮箱不唯一 （7）邮箱为空
职位 position	（8）职位为门店经理、副经理、小组长、店员	（9）职位为其他 （10）职位为空
门店 store_id	（11）门店为存在的门店 id	（12）门店为空 （13）门店 id 不合理

员工注册测试用例

测试数据	期望结果	测试范围	测试结果
姓名输入：张三 **姓名输入：**	姓名输入有效 姓名为空	（1） （2）	正确 正确
密码输入：A123456 **密码输入：** **密码输入：451** **密码输入：A123**	密码输入有效 密码为空 密码不符合要求 密码不符合要求	（3） （4） （5） （5）	正确 正确 正确 正确
邮箱输入：18491@qq.com（邮箱不存在） **邮箱输入：1841@qq.com（已经存在）** **邮箱输入：**	邮箱输入有效 邮箱已注册 邮箱为空	（14） （6） （7）	正确 正确 正确
职位输入：门店经理 **职位输入：副经理** **职位输入：小组长** **职位输入：店员** **职位输入：店长** **职位输入：**	职位输入有效职位输入有效职位输入有效职位输入有效职位为其他 职位为空	（8） （8） （8） （8） （9） （10）	正确 正确 正确 正确 正确 正确
门店输入：1 **门店输入：**	门店输入有效 门店为空	（11） （12）	正确 正确

门店输入：9999	门店 id 无效	（13）	正确

2. **登录模块**

门店管理者、员工、系统管理者登录账号后可进入系统，运用黑盒测试的等价类划分法

输入数据	有效等价类	无效等价类
邮箱和密码	（1）邮箱和密码对应	（2）邮箱为空 （3）密码为空 （4）邮箱错误 （5）密码错误

登录测试用例：

测试数据	期望结果	测试范围	测试结果
邮箱：18491@qq.com 密码：A123456	输入有效，登录成功	（1）	正确
邮箱：密码：123	邮箱为空	（2）	正确
邮箱：18491@qq.com 密码：	密码为空	（3）	正确
邮箱：1849 @qq.com 密码：122	邮箱不存在	（4）	正确
邮箱：18491@qq.com 密码：12333	密码错误	（5）	正确

3. **员工偏好设置**

员工登录成功进入系统后，可自行设置工作偏好，通过黑盒测试划分等价类如下。

输入数据	有效等价类	无效等价类
工作日偏好	（1）1~7	（2）其他数字
工作时间偏好	（3）08:00-12:00,18:00:22:00	（4）其他时间段
工作班次偏好	（5）N 小时（0<N<6）	（6）其他输入

员工偏好设置测试用例

测试数据	期望结果	测试范围	测试结果
工作日偏好输入：1，3，4	工作日偏好有效	（1）	正确
工作日偏好输入：8	工作日偏好无效	（2）	正确
工作时间偏好输入：08:00-12:00,18:00-:22:00	工作时间偏好有效	（3）	正确
工作时间偏好输入：18:00-25:00	工作时间偏好无效	（4）	正确
工作班次偏好输入：2 小时，5 小时	工作班次偏好有效	（5）	正确
工作班次偏好输入：-3 小时，8 小时	工作班次偏好无效	（6）	正确

4. 员工换班申请

员工可提交换班请求。通过黑盒测试划分等价类如下。

输入数据	有效等价类	无效等价类
旧班次	（1）已有的归属自己的班次	（2）其他班次
新班次	（3）同等时间段的班次	（4）其他班次
申请理由	（5）不少于 10 字	（6）其他输入

员工换班申请测试用例

测试数据	期望结果	测试范围	测试结果
旧班次输入：20230221T14:00:00	旧班次有效	（1）	正确
旧班次输入：20240221T14:00:00	旧班次无效	（2）	正确
新班次输入：20230223T14:30:00	新班次有效	（3）	正确
新班次输入：2020221	新班次无效	（4）	正确
申请理由：家中突发紧急事件，需要更换到后天的班次	申请理由有效	（5）	正确
申请理由：12346##$!@#@#$$1321QWE	申请理由无效	（6）	正确

员工换班 UI 测试

测试用例	具体操作	预期结果
test1	点击换班申请	进入换班请求界面
test2	点击换班申请，输入有效的换班请求信息，点击提交	换班申请提交成功
test3	点击换班申请，输入无效的换班请求信息，点击提交	提示换班信息有误

5. 员工查看排班日历

员工查看排班情况 UI 测试。

测试用例	具体操作	预期结果
test1	点击查看排班日历	进入排班信息日历展示界面

6. 门店管理者智能排班

门店管理者登录系统后，系统可根据门店排班规则、员工需求个性化生成排班计划，通过黑盒测试划分等价类如下。

输入数据	有效等价类	无效等价类
预测文件	（1）xlsx、xls 格式数据	（2）其他班次

门店管理者的测试用例

测试数据	期望结果	测试范围	测试结果
预测文件输入：3-25.xlsx　3-26.xls	预测文件有效	（1）	正确
预测文件输入：2022.txt 2023.word	预测文件无效	（2）	正确

门店管理者智能排班 UI 测试

测试用例	具体操作	预期结果
test1	门店管理者上传有效门店客流量预测文件，点击智能排班	进入排班界面，输出排班信息表
test2	门店管理者直接点击智能排班	提示请上传客流量预测文件，排班失败

7. 门店管理者自定义排班

门店管理者可以自定义排班规则，进行排班。

测试用例	具体操作	预期结果
test1	门店管理者上传有效门店客流量预测文件，输入自定义排班规则，点击排班	进入排班界面，输出排班信息表
test2	门店管理者输入自定义排班规则，点击排班	提示请上传客流量预测文件，排班失败

8. 门店管理者换班审批

店内员工可申请换班，提交换班审批后，由门店管理者审批是否通过，换班审批通过即可完成换班。

测试用例	具体操作	预期结果
test1	门店管理者点击换班审批，点击第一条换班审批申请，点击同意	换班状态更新，申请人班次更新
test2	门店管理者点击换班审批，点击第一条换班审批申请，点击拒绝	换班状态更新，申请人弹出提示

9. 门店管理者班组管理

门店管理者可以新建班组，将某些员工加入某班组，点击班组可以查看班组的信息，包含班组名称、班组成员等。

测试用例	具体操作	预期结果
test1	点击新建班组，输入班组名称，选中员工，点击提交	新建班组成功
test2	点击班组名称	进入班组显示界面，显示班组信息和班组成员的信息
test3	点击班组名称，选中某成员，点击删除	班组删除该成员

10. 更改个人信息

员工可以更改个人信息，可修改个人的邮箱、密码、职位、所属门店的信息。

测试用例	具体操作	预期结果
test1	员工点击更改个人信息，输入修改后的未注册的邮箱，点击提交	员工邮箱修改成功
test2	员工点击更改个人信息，输入修改后的已经注册的邮箱，点击提交	邮箱已存在，修改失败
test3	员工点击更改个人信息，输入修改后的密码	员工密码修改成功
test4	员工点击更改个人信息，职位输入副经理，点击提交	员工职位修改成功
test5	员工点击更改个人信息，职位输入店长，点击提交	员工职位修改失败
test6	员工点击更改个人信息，门店输入 1，	员工所属门店修改成功

	点击提交	
test7	点击更改个人信息，门店输入 999，点击提交	员工所属门店修改失败

11. **找回密码**

员工忘记密码后可以通过邮箱找回密码。

测试用例	具体操作	预期结果
test1	在登录界面，点击找回密码，输入邮箱，输入新密码，点击获取验证码，输入邮箱收到的验证码，点击提交	密码修改成功
test2	在登录界面，点击找回密码，输入邮箱，输入新密码，点击提交	密码修改失败

注：原型中未制作非核心业务的 UI 设计暂无法进行测试，后续完整测试文档中会予以补充。[原型参考]

8. 项目交付和维护

8.1 项目交付计划和过程

在项目初期，我们根据大赛官方的需求文档深入的沟通分析，了解其显现和隐性需求，对业务流程有个初始化的规划并设计数据结构。在实施过程中，我们根据官方的需求文档不断进行调整，确认需求变化和潜在问题，并及时解决。

同时，我们需要关注客户的需求和优化要求，并根据其合理性和可行性分析后给出合理的方案。根据大赛的需求，我们需要确定智能排班的具体功能，并将其转化为技术需求，包括方案、数据要求、规模估算、用户体验等，并为客户提供相应的方案和建议。

阶段	起始时间	工作内容
需求分析	2023.3.1-2023.3.12	详细确认业务流程和数据结构 分析整个排班的流程
设计方案	2023.3.13-2023.3.20	确认系统功能、前端页面设计 数据库设计、接口设计 服务端设计、系统架构设计
技术实现	2023.3.21-2023.4.11	编写前端页面、搭建数据库 实现接口 后端开发、系统架构实现
测试优化	2023.4.11-2023.4.17	功能测试、性能优化、 系统稳定性测试、收集反馈 进行问题修复和优化
上线运营	2023.3.31-2023.4.17	部署上线、监控运营 维护优化、持续改进

8.2 后续需求和升级计划

在初步评估业务流程后，我们需要确定智能排班系统的具体功能，并将其转化为技术需求。在系统投入使用后，客户对系统的需求会随着业务的发展，而不断产生变化。因此，我们需要快速响应客户的变化需求，不断提供新的升级和功能改进。下面是智能排班系统的后续需求和升级计划：

- 基于移动端的排班修改和审核：增加手机端排班修改和审核功能，方便客户在外出办公期间，随时修改和审核员工排班计划；
- 实时数据监控与提醒：提供实时的数据监控和提醒功能，帮助客户及时发现异常数据和问题，保证排班系统的稳定运行；
- 生产计划排班：针对具体业务，增加生产计划排班功能，将生产计划与排班计划结合起来，更加优化排班效率；
- 智能预测算法：优化排班预测算法效率，并逐步应用到页面显示中，提供更加智能的排班计划；
- 员工自助服务：增加自助服务功能，包括预留休假和调换班次等，解放管理员的工作压力。

9. 总结和反思

9.1 开发过程中算法问题与解决方案

排班的约束条件：根据具体业务要求，需考虑早班、晚班、轮班等排班方式，同时需要考虑休息日、假期等约束条件，避免排班冲突。解决方案：通过建立约束条件算法，保证排班的合理性，同时考虑员工的个人特殊情况，如需别分配工作量等；

排班算法性能问题：针对大规模数据和高并发的情况，排班计算可能会由于算法效率等问题，导致系统运行缓慢或者崩溃。解决方案：通过使用分布式计算、优化算法等方式，提高排班算法的性能，确保算法更加高效、稳定；

排班质量问题：排班的结果直接影响人员的工作效率、质量等目标，因此算法要保证排班质量，避免盲目的优化，优先保证服务效率。解决方案：将系统性能优化结果作为衡量标准，相对稳定的算法一直是所要追求的目标。

9.2 系统达成的目标和展望

本项目系统通过遗传算法进行优化排班，可以进行高效快速的制定合理的排班方案，充分利用人力资源，提高工作效率，同时 PC 端页面的设计提供友好的

用户体验，允许管理员快速编辑员工排班和员工信息，还为员工提供个人信息和社交交流功能，通过假期、休息日、预排班等的高效利用，优化人力资源的利用，从而降低企业成本，为企业提供精细的人力资源管理，包括员工工作时长、工作质量、休息日/假期安排等，同时提供数据操作和分析功能，为企业管理做出科学决策。

未来可以向更加智能化方向进行发展，通过机器学习、人工智能等技术手段，实现智能化的排班分配和预测，为企业提供更加准确和个性化的服务。随着企业和员工需求的变化，本排班系统也将提供更加合理和符合实际需求的服务，更加便利的员工管理，希望最终能够自动不断解决用户应用过程中遇到的问题，不断更新系统，适应不断变化的用户需求。

9.3 提议和建议

- 系统的可维护性和可扩展性要得到充分考虑；
- 提高系统的性能，保证实时性的处理能力；
- 排班系统需要良好的用户界面，并考虑在手机等移动端上可用性的问题；
- 提供更加细致的数据分析功能和优化分析算法，帮助客户更好地理解员工排班现状，以提高企业的效益。
- 在设计算法实现的时候，应当尽可能地进行顺序化处理，避免在高并发的情况下导致系统崩溃的问题。同时，应尽量使用分布式部署的方式，提高系统的稳定性。