同济大学
TONGJI UNIVERSITY

怡运动
运动爱好者的运动与社交平台

怡运动——运动爱好者的运动与社交平台

概要设计规约

2251093 冯伟航

2254300 王捷

2252721 韩坤甫

目录

1. 引言

1.1. 概要设计依据

- 本系统的需求规约
- 本系统的需求分析规约
- 课堂所讲内容

1.2. 参考资料

Roger S.Pressman&Bruce R.Maxim.软件工程—实践者的研究方法（第八版）.郑仁杰译 .北京:机械工业出版社.2016.12

《SpringBoot 更好的开发》

《HTML5+CSS3 从入门到精通》

《更好的软件架构，更好的设计》

《Design Pattern》

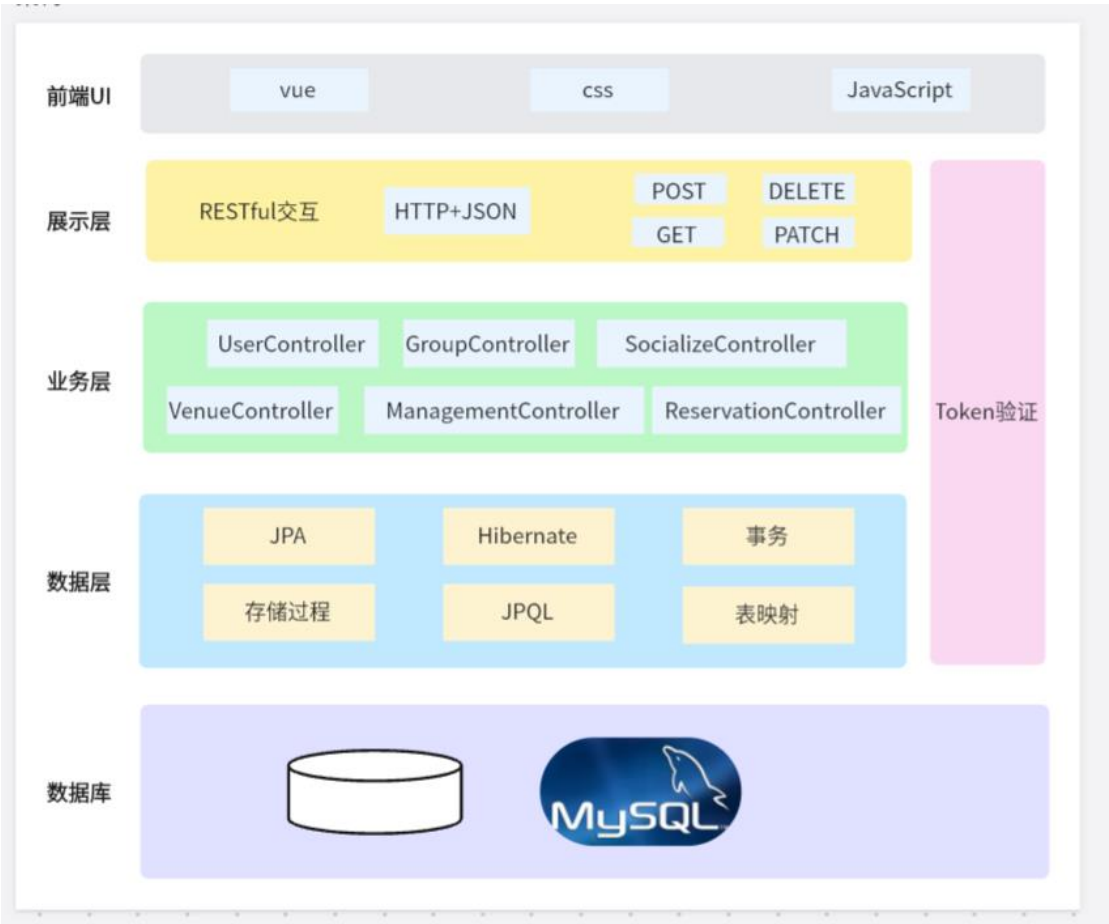
1.3. 假定和约束

1. 项目开发期限为 2 个月，时间为 2024 年 11 月~2025 年 1 月初，到期末答辩前为止；
2. 项目开发无经费，设备条件为 3 台 Windows 操作系统电脑以及阿里云平台等；
3. 项目在开发前线上通过问卷调研的方式收集了 74 份问卷，并据此制定了用户画像；
4. 在交流过程中，我们每周都会线下汇报工作进度，通过 Github 进行代码协作管理；我们还组建了微信群，实现有问题，随时线上沟通，以提高效率，加深团队成员之间的合作。

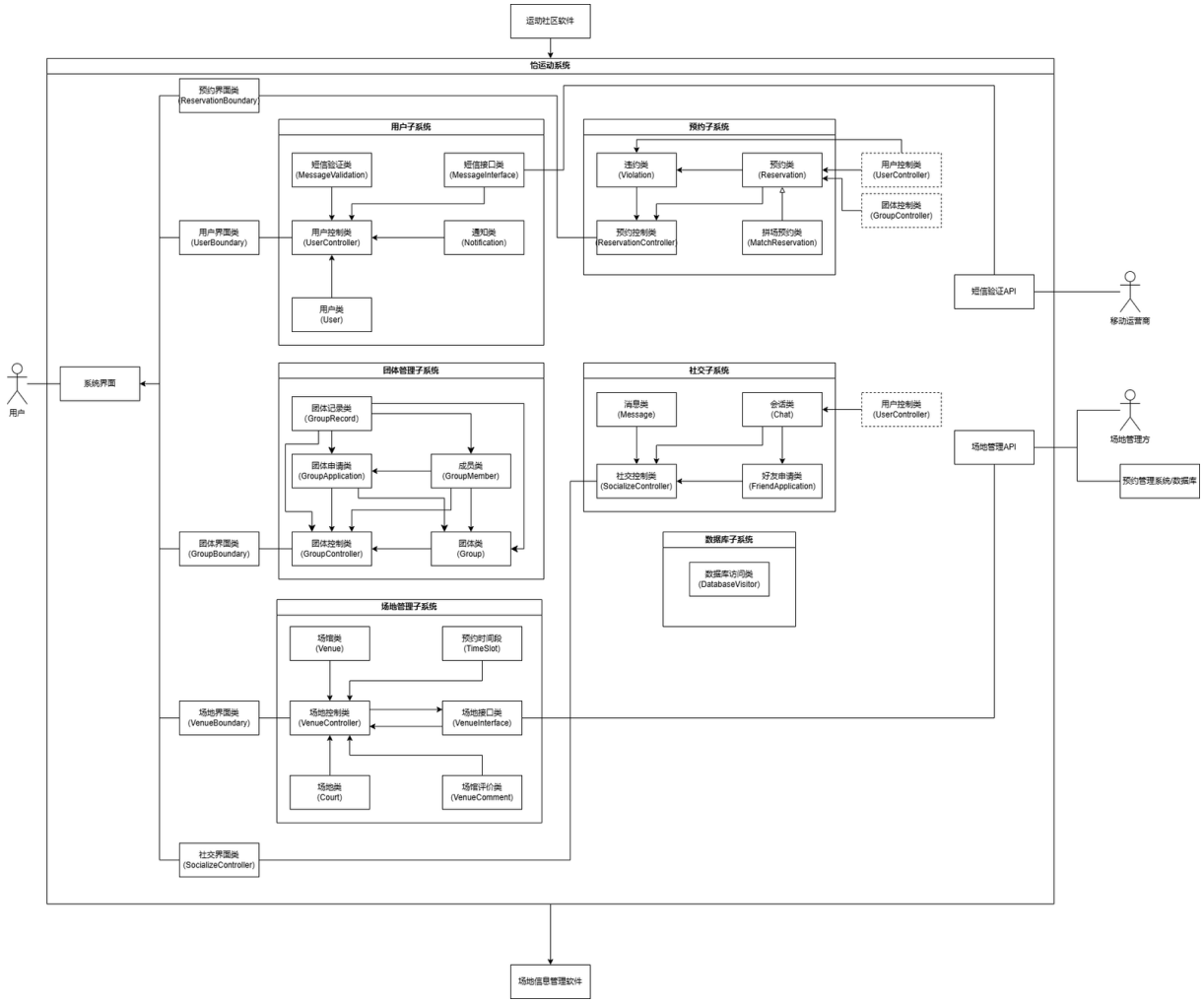
2. 概要设计

2.1. 系统体系架构设计

本系统采用前后端分离的架构设计，前端基于 Vue 3 构建，提供直观、响应式的用户界面与交互功能；后端以 Spring Boot 为核心框架，集成 Spring Security 实现用户认证与权限管理，使用 Spring Data JPA 负责数据持久化，并通过统一的接口规范为前端提供服务。数据存储采用 MySQL 数据库，保证数据的一致性与可靠性。系统设计注重扩展性与维护性，为用户提供流畅、高效的使用体验，同时支持灵活的功能扩展。



2.2. 软件（体系）结构设计


运动社区软件
怡运动系统
预约界面类
(ReservationBoundary)
用户子系统
短信验证类
(MessageValidation)
短信接口类
(MessageInterface)
用户界面类
(UserBoundary)
用户控制类
(UserController)
通知类
(Notification)
用户类
(User)
预约子系统
违约类
(Violation)
预约类
(Reservation)
用户控制类
(UserController)
团体控制类
(GroupController)
预约控制类
(ReservationController)
拼场预约类
(MatchReservation)
短信验证API
移动运营商
用户
系统界面
团体管理子系统
团体记录类
(GroupRecord)
团体申请类
(GroupApplication)
成员类
(GroupMember)
团体界面类
(GroupBoundary)
团体控制类
(GroupController)
团体类
(Group)
社交子系统
消息类
(Message)
会话类
(Chat)
用户控制类
(UserController)
社交控制类
(SocializeController)
好友申请类
(FriendApplication)
场地管理API
场地管理方
预约管理系统/数据库
数据库子系统
数据库访问类
(DatabaseVisitor)
场地管理子系统
场馆类
(Venue)
预约时间段
(TimeSlot)
场地界面类
(VenueBoundary)
场地控制类
(VenueController)
场地接口类
(VenueInterface)
场地类
(Court)
场馆评价类
(VenueComment)
社交界面类
(SocializeController)
场地信息管理软件

2.3. 接口设计

2.3.1. 前后端调用接口设计

2.3.1.1. 用户管理子系统

api 前缀：/api/users

1)

功能描述	用户登录
API	/login
请求方法	post
说明	用户通过输入账号密码，登录系统
参数	无
请求体	用户的用户名和加密后密码

	{
	"userName": string,
	"password": string
	}

成功返回	用户的登录 token 和过期时间

	{
	"token": string,
	"expiration_time": instant,
	"userId": int,
	"userName": string
	}

失败返回	登录失败的原因

	{
	"msg": string
	}

调用条件	用户进入系统且处于未登录状态

2)

功能描述	用户注册
API	/registration
请求方法	post
说明	用户输入信息，在系统中注册账号
参数	无

请求体	用户的基本信息，包括用户名，加密后的密码，联系电话，真实姓名，头像

	{
	"userName": string,
	"password": string,
	"phone": string,
	"realName": string,
	"photo": string
	}

成功返回	用户 ID

	{
	"userId": int
	}

失败返回	注册失败的原因

	{
	"msg": string
	}

调用条件	在登录界面选择“注册”选项

3)

功能描述	查询用户的详细信息
API	/info
请求方法	get
说明	根据用户 id 查找用户详细信息（需在 header 中配置用户 token）
参数	无
请求体	无
成功返回	用户的详细信息

	{
	"userId": int,
	"userName": string,
	"phone": string,
	"realName": string,
	"registrationDate": string,
	"photo": string
	}

失败返回	查询失败的原因

	{
	"msg": string
	}

调用条件	用户在已登录的情况下访问个人信息界面

4)

功能描述	修改用户信息
API	/info
请求方法	patch
说明	根据用户 id 修改用户详细信息（需在 header 中配置用户 token）
参数	无
请求体	修改后的用户信息

	{
	"userName": string,
	"phone": string,
	"realName": string,
	"photo": string
	}

成功返回	修改成功后的用户信息

	{
	"userId": int,
	"userName": string,
	"phone": string,
	"realName": string,
	"registrationDate": string,
	"photo": string
	}

失败返回	调用失败的原因

	{
	"msg": string
	}

调用条件	用户在个人信息界面修改个人信息并点击“确认修改”

5)

功能描述	修改密码
API	/password
请求方法	patch

说明	根据用户 id 修改用户密码（需在 header 中配置用户 token）
参数	无
请求体	用户的旧密码和新密码（SHA256 加密后）

	{
	"oldPwd": string,
	"newPwd": string
	}

成功返回	用户 ID

	{
	"user_id": int
	}

失败返回	修改失败的原因

	{
	"msg": string
	}

调用条件	用户在个人信息界面修改密码并点击“确认修改”

6)

功能描述	获取用户通知
API	/notifications
请求方法	get
说明	根据用户 id 获取用户通知（需在 header 中配置用户 token）
参数	无
请求体	无
成功返回	通知详细信息

	[
	{
	"notificationId": int,
	"type": enum("system", "reservation", "friend", "group"),
	"title": string,
	"content": string,
	"timestamp": date,
	"state": enum("read", "unread", "mark", "deleted"),
	}
	]

失败返回	获取失败的原因
	{ "msg": string

	}
调用条件	用户进入个人通知界面

7)

功能描述	修改通知状态
API	/newNotifications
请求方法	patch
说明	修改统治的状态为已读、标记、删除
参数	无
请求体	通知 ID 和修改操作

	{
	"notificationId": int,
	"operation": string
	}

成功返回	修改成功信息

	{
	"msg": string
	}

失败返回	修改失败信息

	{
	"msg": string
	}

调用条件	用户在通知界面修改通知的状态

8)

功能描述	向用户发送通知
API	/newNotifications
请求方法	post
说明	根据用户 id 向用户发送通知
参数	无
请求体	通知内容和用户 ID
	{ "type": enum("system", "reservation", "friend", "group"), "title": string, "content": string, "userId": int

	}
成功返回	发送成功信息

	{
	"msg": string
	}

失败返回	发送失败的原因

	{
	"msg": string
	}

调用条件	用户在个人信息界面修改密码并点击“确认修改”

9)

功能描述	通过用户名称搜索用户
API	/names
请求方法	get
说明	在输入框中输入用户名称，点击“搜索”按钮后通过用户名称搜索用户
参数	用户名称: userName: string
请求体	无
成功返回	筛选后的用户信息列表

	[
	{
	"userId": int,
	"userName": string,
	"phone": string,
	"realName": string,
	"registrationDate": date,
	"photo": string
	}
	]

失败返回	获取失败原因

	{
	"msg": string
	}

调用条件	用户添加好友时获取用户列表

10)

功能描述	获取用户列表
API	/list

请求方法	get
说明	获取所有用户信息，以列表形式展示
参数	无
请求体	无
成功返回	用户信息列表

	[
	{
	"userId": int,
	"userName": string,
	"phone": string,
	"realName": string,
	"registrationDate": date,
	"photo": string
	}
	]

失败返回	获取失败原因

	{
	"msg": string
	}

调用条件	用户添加好友时获取用户列表

2.3.1.2. 场地管理子系统

类型说明：

enum[venueState]：场馆开放状态枚举类型，包括开放、未开放，取值为以下字符串之一：“open”, “closed”

enum[courtState]：场地开放状态的枚举类型，包括开放、关闭，取值为以下字符串之一：“open”, “closed”

enum[availability]：场地可预约状态的枚举类型，包括可预约、拼场中、已满、关闭，取值为以下字符串之一：

“reserveable”, “matching”, “full”, “closed”

date 表示日期类型的数据，使用 UTC 字符串格式，即”yyyy-mm-ddThh:mm:ssZ”

场馆评分应当为保留 1 位小数的浮点型数据，十分位为 0 或 5，最小值为 1.0，最大值为 5.0

预约价格应当为保留 2 位小数的浮点型数据，值必须大于 0

api 前缀：/api/venues

1)

功能描述	获取场馆列表或根据场馆名称查找符合条件的场馆

API	/list
请求方法	GET
说明	参数中传递场馆名称时，表示查找名称中包含该参数的场馆
参数	获取场地列表的页码：page: int 场地名称（可选）：name: string
请求体	无
成功返回	符合条件的场馆总数、返回的页码、场馆列表信息，包含的信息为：场馆ID、场馆名称、场馆位置、场馆状态、场馆联系方式和场馆图片的 url

	{
	“total”: int
	“page”: int
	“venues”:
	[
	{
	“venueId”: int
	“venueName”: string
	“location”: string
	“state”: enum[venueState]
	“contactNumber”: string
	“image”: string
	}
	]
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	venueBoundary 类（前端）向 venueController 类（后端）请求场馆列表信息

2)

功能描述	获取场馆详细信息
API	/detail
请求方法	GET
说明	无

参数	要获取信息的场馆 ID：venueId: string
请求体	无
成功返回	场馆信息，包括：场馆 ID、场馆名称、场馆图片 url、场馆描述、场馆地址和场馆开放状态

	{
	“venueId": int
	“venueName”: string
	“image”: string
	“description”: string
	“location”: string
	“state”: enum[venueState]
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	venueBoundary 类（前端）向 venueController 类（后端）请求场馆详细信息

3)

功能描述	获取场馆的场地信息
API	/courts
请求方法	GET
说明	无
参数	要获取信息的场馆 ID：venueId: string
请求体	无
成功返回	场地信息数组，包括：场地 ID、场地名称、场地类型、场地开放状态和场地描述
	[{ “courtId”: int “courtName”: string “capacity”: int

	“type”: string “state”: enum[courtState] “location: string }]
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	venueBoundary 类（前端）向 venueController 类（后端）请求某一场馆的场地信息

4)

功能描述	获取场馆某一天的开放时间段信息
API	/timeslots
请求方法	GET
说明	参数中的 date 字段应当符合”yyyy-mm-dd”的格式，分别表示请求日期的年、月和日
参数	要获取开放时间段的场馆 ID：venueId: string 要获取时间段的日期：date: string
请求体	无
成功返回	开放时间段数组，每个数组除了包含开放的时间段信息以外，还包含该时间段中该场馆内所有场地的可预约状态信息，即场地 ID、预约价格和可预约状态

	[
	{
	“timeslot”: {
	“timeslotId”: int
	“startTime”: date
	“endTime”: date
	}
	“courtAvailabilities”:
	[
	{
	“availiabilityId”: int
	“courtId”: int

	“price”: real “state”: enum[avaliability] }] }]
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	venueBoundary 类（前端）向 venueController 类（后端）请求某一天的开放时间段信息

5)

功能描述	获取某一场馆的评价
API	/comments
请求方法	GET
说明	获取的评价以时间从最新到最晚作为排序依据。
参数	要获取评价的场馆 ID：venueId: string 获取评价的页码：page: int
请求体	无
成功返回	场地评价的总数、请求的页码、场地评价数组，其中的信息包括：评价 ID、评价内容、发布评价的用户 ID、用户名、用户头像 url、评价时间、评分

	{
	“total”: int
	“page”: int
	“comments”: [
	{
	“commentId”: int
	“content”: string
	“userId”: string
	“userName”: string
	“userAvatar”: string
	“commentTime”: date
	“rating”: real
	}
	]

	}
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	venueBoundary 类（前端）向 venueController 类（后端）请求场地评价信息

6)

功能描述	用户发布场地评价
API	/comments
请求方法	POST
说明	需要使用 token 验证用户身份，并通过 token 获取发布评价的用户 ID 信息
参数	无
请求体	要评价场地的 ID、场馆评分、评价内容、发布时间

	{
	“venueId”: int
	“rating”: real
	“content”: string
	“commentTime”: string
	}

成功返回	state 字段设置为 1 表示发布成功

	{
	“state”: int
	“msg”: string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	“state”: int
	msg: string
	}

调用条件	用户发送评论时 venueBoundary 类（前端）向 venueController 类（后端）传递发布评论请求

2.3.1.3. 预约管理子系统

API 前缀：/api/reservations

1)

功能描述	用户进行个人预约
API	/individual
请求方法	POST
说明	需要使用 token 验证用户身份，并通过 token 获取的用户 ID 信息
参数	无
请求体	进行预约的信息，包括：可预约项的 ID 和预约用户数组

	{
	“availabilityId”: int
	“users”: [
	int
	]
	}

成功返回	返回该预约中创建的预约和用户信息

	{
	reservationInfo:
	{
	“reservationId”: int,
	“availabilityId”: int,
	“type”: enum[reservationType]
	“state”: enum[reservationState]
	}
	“users”: [
	{
	“userReservationId”: int,
	“userId”: int,
	“userName”: string,
	“photo”: string,
	}
	]
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	用户进行个人预约时 reservationBoundary 类（前端）向 reservationController 类（后端）传递预约请求

2)

功能描述	用户进行团体预约
API	/group
请求方法	POST
说明	需要使用 token 验证用户身份，并通过 token 获取的用户 ID 信息
参数	无
请求体	进行预约的信息，包括：可预约项的 ID、团体 ID 和预约用户数组

	{
	“availabilityId”: int
	“groupId”: int
	“users”: [
	int
	]
	}

成功返回	返回该预约中创建的预约信息、用户信息和团体预约信息

	{
	reservationInfo:
	{
	“reservationId”: int,
	“availabilityId”: int,
	“type”: enum[reservationType]
	“state”: enum[reservationState]
	}
	“users”: [
	{
	“userReservationId”: int,
	“userId”: int,
	“userName”: string,
	“photo”: string,
	}

	]
	“groupInfo”:
	{
	“groupReservationId”: int,
	“groupId”: int,
	}
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	用户进行团体预约时 reservationBoundary 类（前端）向 reservationController 类（后端）传递预约请求

3)

功能描述	用户进行拼场预约
API	/match
请求方法	POST
说明	需要使用 token 验证用户身份，并通过 token 获取的用户 ID 信息
参数	无
请求体	进行预约的信息，包括：场馆 ID、开放时间段 ID、预约场地的类型、预约用户数组和预约人数

	{
	“venueId”: int
	“timeslotId”: int
	“courtType”: string
	“users”: [
	int
	]
	“reservationCount”: int
	}

成功返回	返回该预约中创建的预约信息、用户信息和拼场预约信息
	{ reservationInfo: { “reservationId”: int, “availabilityId”: int,

	“type”: enum[reservationType]
	“state”: enum[reservationState]
	}
	“users”: [
	{
	“userReservationId”: int,
	“userId”: int,
	“userName”: string,
	“photo”: string,
	}
	]
	“matchInfo”:
	{
	“matchReservationId”: int,
	“expirationDate”: date,
	“reservedCount”: int,
	}
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	用户进行拼场预约时 reservationBoundary 类（前端）向 reservationController 类（后端）传递预约请求

功能描述	获取用户预约列表
API	/list
请求方法	GET
说明	需要使用 token 验证用户身份，并通过 token 获取的用户 ID 信息
参数	无
请求体	无
成功返回	用户预约信息数组，包括：预约 ID、场馆名称、场地名称、预约时间、预约类型和预约状态
	[

	{
	“reservationId”: int
	“venueName”: string
	“courtName”: string
	“startTime”: date
	“endTime”: date
	“type”: enum[ReservationType]
	“state”: enum[ReservationState]
	}
	]

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	用户进入预约页面时 reservationBoundary 类（前端）向 reservationController 类（后端）传递获取预约记录的请求

5)

功能描述	获取预约详细信息
API	/detail
请求方法	GET
说明	需要使用 token 验证用户身份
参数	要获取预约信息的 ID：reservationId: int
请求体	无
成功返回	预约详细信息，其中包括： 基本信息：预约 ID、场馆 ID 和名称、场地 ID 和名称、开始和结束时间、预约类型、预约状态、（团体预约）团体 ID 和名称、（拼场预约）过期时间和当前人数 用户信息数组：用户 ID、用户名称、用户头像 操作记录数组：记录 ID、操作时间、预约记录状态、
	{ “basicInfo”: { “reservationId”: int “venueId”: int

	“venueName”: string “courtId”: int “courtName”: string “startTime”: date “endTime”: date “type”: enum[ReservationType]; “state”: enum[ReservationState]; “groupId”: int | null “groupName”: string | null “expirationTime”: date | null “reservedCount”: date | null } “users”: [“userReservationId”: int “userId”: int “userName”: string “photo”: string] records: [“reservationRecordId”: int “state”: enum[ReservationState] “time”: date “userId”: int “reservationId”: int] }
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	用户进入预约页面时 reservationBoundary 类（前端）向 reservationController 类（后端）传递获取预约记录的请求

6)

2.3.1.4. 团体管理子系统

api 前缀：/api/groups

1)

功能描述	创建团体
API	
请求方法	POST
说明	需要使用 token 验证用户身份，从 token 中获取用户 id
参数	无
请求体	团体的详细信息，包括名字，团体描述，所有团体成员的 userId 和角色。

	{
	"groupName": "string",
	"description": "string",
	"photo": "string"，
	"members":[int]
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户创建团体时 GroupBoundary 类（前端）向 GroupController 类（后端）传递创建团体请求。

2)

功能描述	获取团体列表
API	
请求方法	GET
说明	无

参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及团体列表

	[
	{
	"groupId":int,
	"groupName":string,
	"description":string,
	"chatId":int
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户查看团体列表时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

功能描述	获取用户加入的列表
API	/byUser
请求方法	GET
说明	需要使用 token 验证用户身份，从 token 中获取用户 id
参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及团体列表
	[{

	"groupId":int, "groupName":string, "description":string, "chatId":int }]
失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户查看团体列表时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

3)

功能描述	根据搜索某一团体
API	/byName/{groupName}
请求方法	GET
说明	无
参数	团体的名字：groupName:string
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及团体名称中包含该名字的团体列表

	[
	{
	"groupId":int,
	"groupName":string,
	"description":string,
	"chatId":int
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因
	{ state: int，

	msg: string }
调用条件	用户搜索团体时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

4）

功能描述	获取某一团体的详细信息
API	/byId/{groupId}
请求方法	GET
说明	无
参数	团体的 id：groupId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及团体的详细信息

	{
	"groupId":int,
	"groupName":string,
	"description":string,
	"chatId":int
	"members":[
	"userId":int,
	"userName":string,
	"photo":string,
	"role": string
	]
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户查看团体详细信息时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

5)

功能描述	解散团体
API	
请求方法	DELETE
说明	需要使用 token 验证用户身份。从 token 中获取用户 id
参数	操作的团体的 Id：groupId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户解散团体时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

6）

功能描述	发起加入申请
API	/application
请求方法	POST
说明	需要使用 token 验证用户身份。从 token 中获取用户 Id
参数	无
请求体	加入的团体的 ID，申请理由

	{
	"groupId": int,
	"applyInfo": string
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户发起加入申请时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

7)

功能描述	邀请好友加入团体
API	/application/by
请求方法	POST
说明	需要使用 token 验证用户身份，从 token 中获取进行操作的用户 id
参数	
请求体	加入的团体的 ID，被邀请人 ID

	{
	"groupId": int,
	"inviteeId": int
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户发起邀请时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

8)

功能描述	获取所有需要该用户审核的申请
API	/application
请求方法	GET
说明	需要使用 token 验证用户身份，从 token 中获取用户 id
参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及申请列表

	[
	{
	"groupApplicationId": int,
	"type": string,
	"applyInfo": string,
	"state": "waiting",
	"operationTime": date,
	"expirationTime": date,
	"applicantId":int,
	"applicantName": string,
	"groupId": int,
	"reviewerId": int,
	"reviewerName": int,
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户查看申请列表时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

9)

功能描述	审核申请
API	/application

请求方法	PATCH
说明	需要使用 token 验证用户身份。从 token 中获取执行操作的用户 ID，
参数	无
请求体	审核的申请的 ID，审核结果

	{
	"result": boolean,
	"auditObjectId": int
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	管理员审核完申请后 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

10)

功能描述	退出团体
API	/members
请求方法	DELETE
说明	需要使用 token 验证用户身份。从 token 中获取执行操作的用户 ID。
参数	退出的团体的 ID：groupId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户退出团体时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

11)

功能描述	管理员将团员移出团体
API	/members/by
请求方法	DELETE
说明	需要使用 token 验证用户身份。从 token 中执行操作的管理员 ID，
参数	无
请求体	移出的用户 ID，移出团体的 ID

	{
	”memberId“:int,
	"groupId":int
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户将团员移出团体时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

12)

功能描述	管理员授予用户权限
API	/members
请求方法	PATCH

说明	需要使用 token 验证用户身份。从 token 中获取执行操作的管理员 ID。
参数	无
请求体	被操作的用户 ID，授予的权限

	{
	"targetId":int,
	"role":string,
	"groupId":int
	}

成功返回	state 字段设置为 1 表示发布成功，以及成功信息

	{
	“state”: int，
	"msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	管理员对团员进行授权时 GroupBoundary 类（前端）向 GroupController 类（后端）传递请求。

13)

功能描述	获取团体成员的操作记录
API	/records
请求方法	GET
说明	需要使用 token 验证用户身份。从 token 中获取操作的管理员 ID
参数	被获取记录的用户的 ID:targetId:int 所在团体 ID: groupId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功，以及操作记录的列表
	[{

	"groupRecordId":int, "operatorId":int, "targetId":int, "groupId":int, "time":date, "operateType":string }]
失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	state: int，
	msg: string
	}

调用条件	用户查看团员记录时 GroupBoundary 类（前端）向 GroupController 类（后端）传递查询请求。

2.3.1.5. 社交管理子系统

api 前缀：/api/socialize

1)

功能描述	创建群聊
API	/chats
请求方法	POST
说明	需要使用 token 验证用户身份，从 token 中获取 userId。
参数	无
请求体	群聊的详细信息和成员信息。

	{
	"chatName": "string",
	"photo": "string",
	"members": [int]
	}

成功返回	state 字段设置为 1 表示发布成功

	{
	"chatId":int,
	"chatName":"string",
	"type":"string",
	"creationTime":date,
	"photo":"string"
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户创建群聊时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递创建群聊请求。

2)

功能描述	获取用户参与的全部群聊
API	/chats
请求方法	GET

说明	需要使用 token 验证用户身份，根据 token 中的 userId 进行查询
参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功,data 为包含群聊全部信息的数组。

	[
	{
	"chatId":int,
	"chatName":"string",
	"type":"string",
	"creationTime":date,
	"photo":"string"
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户获取群聊列表时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递查询请求。

3)

功能描述	用户退出群聊
API	/chats
请求方法	DELETE
说明	需要使用 token 验证用户身份。
参数	发起操作的用户的 ID,退出的群聊 ID：chatId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“msg”:"string"
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户退出群聊时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递请求。

4)

功能描述	邀请好友加入群聊
API	/chats
请求方法	PATCH
说明	需要使用 token 验证用户身份。
参数	无
请求体	被邀请的用户的 ID,群聊的 ID

	{
	"inviteeId":int,
	"chatId":int
	}

成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“msg”:"string"
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户邀请好友加入群聊时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递请求。

5)

功能描述	用户在群聊中发消息
API	/messages
请求方法	POST
说明	需要使用 token 验证用户身份。
参数	无
请求体	消息的具体内容

	{
	”chatId":int,
	"content": "string",
	}

成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“msg”:"string"
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户在群聊中发送消息时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递发布请求。

6)

功能描述	获取群聊的历史消息
API	/messages
请求方法	GET
说明	需要使用 token 验证用户身份。
参数	请求对象 chat 的 Id：”chatId":int,
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及历史消息数组

	[
	{
	"messageId": int,
	'time': date,
	"content": string,
	“userId”:int,
	"photo":string,
	"userName":string
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户在群聊中获取历史消息时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递查询请求。

7)

功能描述	删除某群聊消息
API	/messages
请求方法	DELETE
说明	需要使用 token 验证用户身份。
参数	删除消息的 ID messageId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	"msg": string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因
	{ "state": int， "msg": string

	}
调用条件	用户在群聊中删除消息时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递删除请求。

8)

功能描述	获取 ID 为{id}的聊天详细信息
API	/chats/{chatId}
请求方法	GET
说明	需要使用 token 验证用户身份。
参数	群聊的 ID：chatId:int
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	"chatId": int,
	"chatName": string,
	"type": string,
	"creatingTime": date,
	"photo": string,
	"members": [
	{
	"userId": int,
	"userName": string,
	"photo": string
	}
	]
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户获取群聊详细信息时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递查询请求。

9)

功能描述	获取用户的好友列表

API	/friends
请求方法	GET
说明	需要使用 token 验证用户身份，获取 token 中的用户 Id
参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及好友群聊的信息

	[
	{
	"chatId":int,
	"userId":int,
	"userName":int,
	"photo":"string"
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户获取好友列表时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递查询请求。

10)

功能描述	发送添加好友申请
API	/application
请求方法	POST
说明	需要使用 token 验证用户身份，
参数	无
请求体	发起申请的用户和接受邀请的用户
	{ "reviewerId":int， “applyInfo”:string

	}
成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户发送好友申请时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递请求。

11)

功能描述	审核好友申请
API	/application
请求方法	PATCH
说明	需要使用 token 验证用户身份。
参数	无
请求体	审核的申请记录的 ID，审核结果：同意/拒绝

	{
	"auditObjectId"： int,
	“result”:boolean
	}

成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“msg":string
	}

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户审核好友申请，得到审核结果时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递请求。

12)

功能描述	获取好友申请列表
API	/application
请求方法	GET
说明	需要使用 token 验证用户身份。从 token 中获取用户 Id
参数	无
请求体	无
成功返回	state 字段设置为 1 表示发布成功.以及申请的信息

	[
	{
	"friendApplicationId": int,
	"applyInfo": "string",
	"state": "string",
	"operationTime": date,
	"expirationTime": date,
	"applicantId": int,
	"applicantName": string,
	"reviewerId"： int,
	"reviewerName": string,
	}
	]

失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

	{
	"state": int，
	"msg": string
	}

调用条件	用户查看好友申请时，得到审核结果时 SocializeBoundary 类（前端）向

	SocializeController 类（后端）传递查询请求。

13)

功能描述	删除好友
API	/friends
请求方法	DELETE
说明	需要使用 token 验证用户身份。
参数	无
请求体	被删除的用户的 ID，对应的会话的 ID

	{
	"targetId":int，
	“chatId”:int,
	}

成功返回	state 字段设置为 1 表示发布成功.以及成功信息

	{
	“state”: int,
	“mas":string
	}

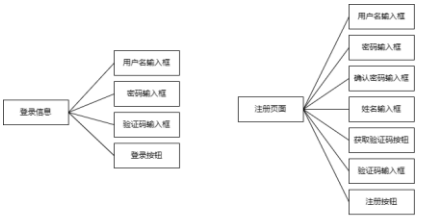
失败返回	state 字段设置为 0 表示发布失败，以及失败的原因

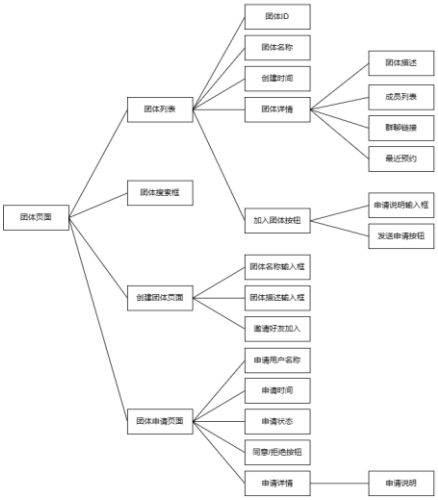
	{
	"state": int，
	"msg": string
	}

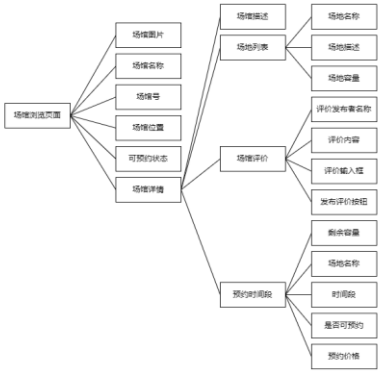
调用条件	用户删除好友时，得到审核结果时 SocializeBoundary 类（前端）向 SocializeController 类（后端）传递删除请求。

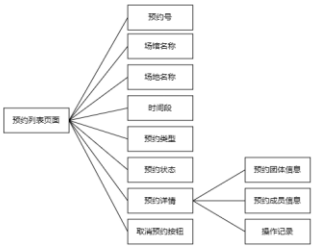
2.4. 界面设计

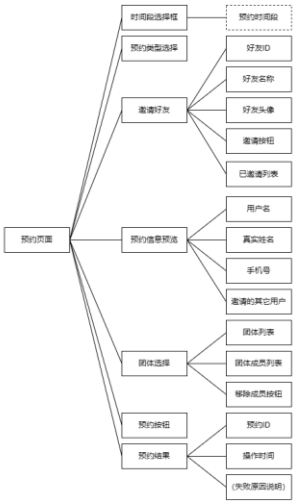
我们通过 data-tree 分析了怡运动页面中包含的元素，如图所示：

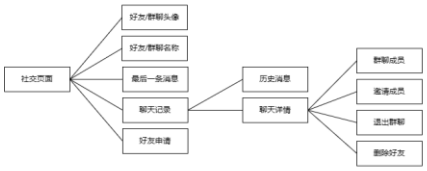


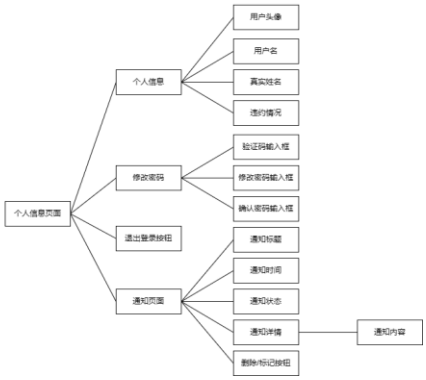












我们使用 js.design 工具绘制了页面高保真度模型，见[link]

2.5. 外部接口设计

2.5.1 怡运动向外部提供的接口

2.5.1.1 场地管理接口

1.

功能描述	场地管理方首次接入怡运动系统进行初始化
API	/management/initialization
请求方法	POST
说明	无
参数	无
请求体	场馆信息，包括场馆名称、场馆位置、场馆状态（默认为 closed），联系方式和场馆图片链接；场地信息数组，包括场地名称、场地容量、场地类型、场地状态和场地描述

	{
	“venueInfo”:
	{
	“venueName”: string
	“location”: string
	“state”: enum[venueState]
	“contactNumber”: string
	“image”: string
	},
	“courts”:
	[
	{
	“courtName”: string
	“capacity”: int
	“type”: string
	“state”: enum[courtState]
	“description”: string
	}
	]

	}
成功返回	返回怡运动系统为场馆以及场地分配的 ID

	{
	“venueId”: int,
	“courts”: [
	{
	“courtId”: int,
	“courtName”: string,
	}
	]
	“msg”: string,
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方希望接入怡运动系统时向 ManagementController 发起请求

2.

功能描述	场地管理方获取自己场馆的信息
API	/management/venueinfo
请求方法	GET
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息
参数	无
请求体	无
成功返回	怡运动中记录的场馆信息，包括：场馆 ID、场馆名称、场馆图片 url、场馆描述、场馆地址和场馆开放状态
	{ “venueId": int “venueName”: string “image”: string “description”: string “location”: string

	“state”: enum[venueState] }
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要查看在怡运动系统中记录的场馆信息时向 ManagementController 发起请求

3.

功能描述	场地管理方修改场馆信息
API	/management/venueinfo
请求方法	PATCH
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息；不支持修改场馆 ID
参数	无
请求体	要编辑的场馆信息，可选字段包括：场馆名称、场馆图片 url、场馆描述、场馆地址和场馆开放状态

	{
	“venueId": int
	“venueName”: string
	“image”: string
	“description”: string
	“location”: string
	“state”: enum[venueState]
	}

成功返回	state 字段设置为 1 表示编辑成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要修改在怡运动系统中记录的场馆信息时向

	ManagementController 发起请求

4.

功能描述	场地管理方添加场地信息
API	/management/courts
请求方法	POST
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息
参数	无
请求体	要创建的场地信息，包括：场地名称、场地容量、场地类型、场地状态和场地位置

	{
	“courtName”: string
	“capacity”: int
	“type”: string
	“state”: enum[courtState]
	“location”: string
	}

成功返回	系统为场地分配的 ID 信息

	{
	“courtId”: int,
	“courtName”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中创建新的可预约场地时向 ManagementController 发起请求

5.

功能描述	场地管理方修改场地信息
API	/management/courts
请求方法	PATCH
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息

参数	无
请求体	必填字段为场地 ID，此外还可选填写要编辑的场地信息，可选字段：场地名称、场地容量、场地类型、场地状态和场地位置

	{
	“courtId”: int
	“courtName”: string
	“capacity”: int
	“type”: string
	“state”: enum[courtState]
	“location”: string
	}

成功返回	state 字段设置为 1 表示编辑成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中修改的可预约场地的信息时向 ManagementController 发起请求

6.

功能描述	场地管理方删除场地
API	/management/courts
请求方法	DELETE
说明	需要使用 token 验证场地管理方身份；参数中的 ID 和名称需要至少填写一个，优先使用 ID。注意删除场地应当在确定场地永久不可预约时调用，如果是场地暂时不可用应当使用前面的 PATCH 方法修改场地状态为“关闭”
参数	要删除的场地 ID（可选）：courtId: int 要删除的场地名称（可选）：courtName: string
请求体	无
成功返回	state 字段设置为 1 表示删除成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中删除的可预约场地时向 ManagementController 发起请求

7.

功能描述	场地管理方添加开放时间段信息
API	/management/timeslots
请求方法	POST
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息
参数	无
请求体	要创建的时间段信息，包括：开始时间和结束时间

	{
	“startTime”: date,
	“endTime”: date,
	}

成功返回	系统成功创建的时间段信息

	{
	“timeslotId”: int,
	“startTime”: date,
	“endTime”: date,
	“venueId”: date,
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中创建新的开放时间段时向

	ManagementController 发起请求

8.

功能描述	场地管理方删除开放时间段
API	/management/timeslots
请求方法	DELETE
说明	需要使用 token 验证场地管理方身份；请求参数中应当填写 ID 或者填写开始时间和结束时间，优先使用 ID。场地管理方在调用该接口时应当保证该时间段没有可预约项
参数	要删除的时间段 ID（可选）：timeslotId: int 要删除的时间段开始时间（可选）：startTime: date 要删除的时间段结束时间（可选）：endTime：date
请求体	无
成功返回	state 字段设置为 1 表示删除成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中删除开放时间段时向 ManagementController 发起请求

9.

功能描述	场地管理方添加可预约项
API	/management/availabilities
请求方法	POST
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息，仅支持通过 ID 添加预约项，具体 ID 可以通过查询场地列表和查询时间段获取
参数	无
请求体	要创建的可预约项，包括：场地 ID、开放时间段 ID、可预约状态、价格

	{
	“courtId”: int,
	“price”: real
	“state”: enum[avaliability]
	“timeslotId”: int
	}

成功返回	系统为可预约项分配的 ID 信息

	{
	“availablityId”: int,
	“courtId”: int,
	“price”: real
	“state”: enum[avaliability]
	“timeslotId”: int
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中创建新的可预约项目时向 ManagementController 发起请求

10.

功能描述	场地管理方修改预约项信息
API	/management/availabilities
请求方法	PATCH
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息；该接口优先通过预约项 ID 查找，再尝试通过场地 ID 和开放时间段 ID；不支持修改场地 ID 和开放时间段 ID
参数	无
请求体	需要填写预约项 ID 或场地 ID 和时间段 ID，此外可选填写要编辑的开放时间段信息，可选字段：价格、可预约状态
	{ “availablityId”: int, “price”: real “state”: enum[avaliability] “courtId”: int “timeslotId”: int

	}
成功返回	state 字段设置为 1 表示编辑成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中修改的可预约项目的信息时向 ManagementController 发起请求

11.

功能描述	场地管理方删除可预约项
API	/management/availabilities
请求方法	DELETE
说明	需要使用 token 验证场地管理方身份；参数中应当填写可预约项 ID 或者场地 ID 和开放时间段 ID，优先使用可预约项 ID
参数	要删除的可预约项 ID（可选）：availabilityId: int 要删除的可预约项的场地 ID（可选）：courtId: int 要删除的可预约项的时间段 ID（可选）：timeslotId: int
请求体	无
成功返回	state 字段设置为 1 表示删除成功

	{
	“state”: int,
	“msg”: string
	}

失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要在怡运动系统中删除的可预约项目时向 ManagementController 发起请求

12.

功能描述	获取场地管理方场馆的场地信息
API	/management/courts
请求方法	GET
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息
参数	无
请求体	无
成功返回	同场地管理子系统接口 3
失败返回	调用失败的原因

	{
	“msg”: string
	}

调用条件	场地管理方需要查看在怡运动系统中记录的场馆信息时向 ManagementController 发起请求

13.

功能描述	获取场地管理方场馆的时间段信息
API	/management/timeslots
请求方法	GET
说明	需要使用 token 验证场地管理方身份，并通过 token 获取场地管理方的管理的场馆 ID 信息
参数	venueId 通过 token 获取，其余参数同场地管理子系统接口 4
请求体	无
成功返回	同场地管理子系统接口 4
失败返回	调用失败的原因

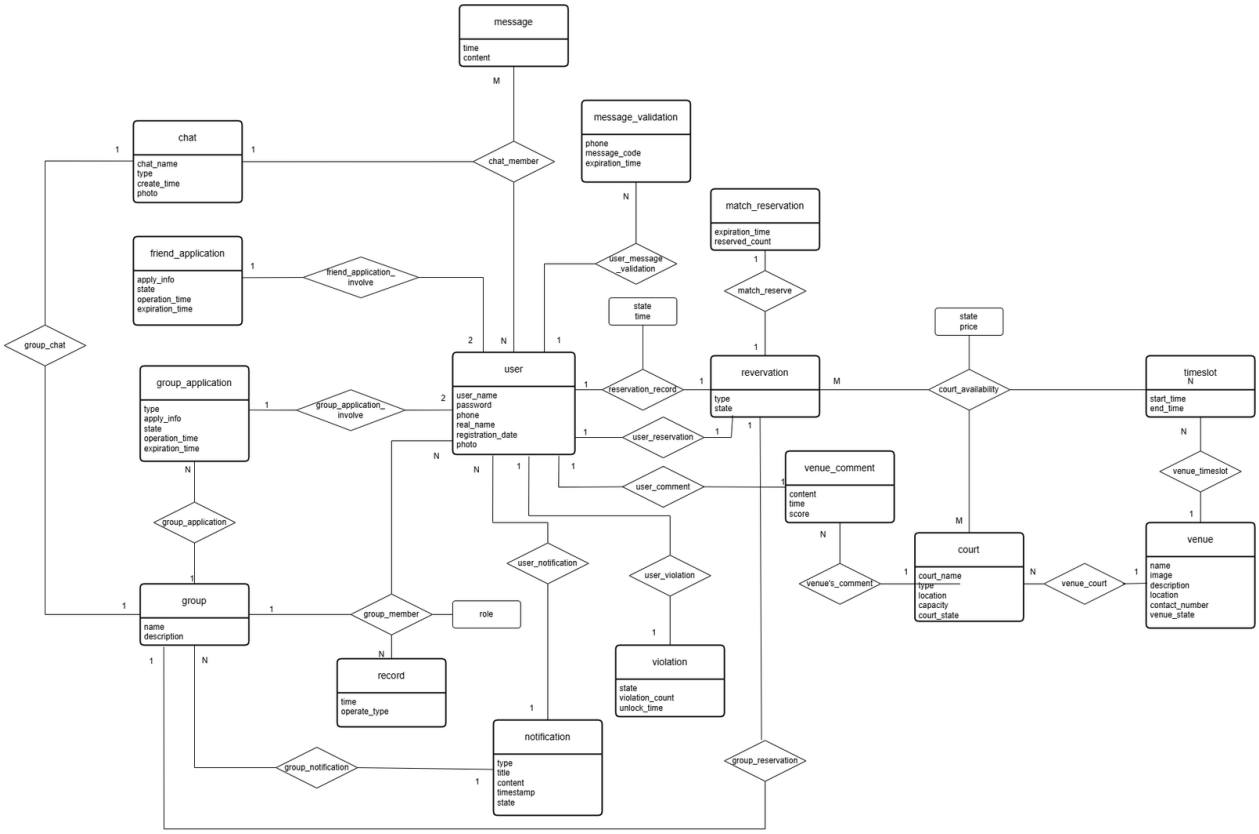
	{
	“msg”: string
	}

调用条件	场地管理方需要查看在怡运动系统中记录的开放时间段信息时向 ManagementController 发起请求

2.6 数据库设计

2.6.1 数据库逻辑设计

2.6.1.1 数据库 ER 图



2.6.2 数据库物理设计

2.6.2.1 数据库表设计

1. group 表（团体的信息）

字段名	数据类型	备注	说明
group_id	int	主码，自增	团体的唯一标识
description	text		团体描述
name	varchar(100)		团体名称
photo	varchar(100)		团体的照片
chat_id	int	外键，引用自 chat 表的 chat_id	团体专属的聊天

2. user 表（用户的信息）

字段名	数据类型	备注	说明
user_id	int	主键，自增（主码）	用户唯一标识
user_name	varchar(50)		用户名
password	varchar(255)	存储的密码已经过 SHA256 加密	用户密码
phone	varchar(20)		用户电话
real_name	varchar(100)		用户真实姓名
registration_date	datetime		注册时间
photo	varchar(100)		用户照片的 URL

3. friend_application 表（好友申请的信息）

字段名	数据类型	备注	说明
friend_application_id	int	主键，自增（主码）	好友申请的唯一标识
apply_info	varchar(100)		申请信息
state	enum('accepted', 'rejected', 'waiting', 'expired')	默认为 'waiting'	申请状态（“通过”，“拒绝”，“审核中”,“已过期”）
operation_time	datetime		操作时间
expiration_time	datetime		过期时间
applicant_id	int	外键，引用自 user 表的 user_id	申请人的 ID
reviewer_id	int	外键，引用自 user 表的 user_id	审核人的 ID

4. group_application 表（加入团体申请的信息）

字段名	数据类型	备注	说明
group_application_id	int	主键，自增（主码）	群组申请唯一标识
type	enum('invited', 'inviting')	默认为 'invited'	申请的类型（“受邀”，“申请加入”）
apply_info	varchar(100)		申请信息
state	enum('accepted', 'rejected', 'waiting', 'expired')	默认为 'waiting'	申请状态（“通过”，“拒绝”，“审核中”,“已过期”）
operation_time	datetime		操作时间
expiration_time	datetime		过期时间
applicant_id	int	外键，引用自 user 表的 user_id	申请人的 ID
reviewer_id	int	外键，引用自 user 表的 user_id	审核人的 ID
group_id	int	外键，引用自 group 表的 group_id	团体的 ID

5. group_member 表（团体成员的信息）

字段名	数据类型	备注	说明
group_member_id	int	主键，自增（主码）	团体成员的唯一标识
user_id	int	外键，引用自 user 表的 user_id	团体成员的用户 ID
group_id	int	外键，引用自 group 表的 group_id	群组 ID
role	enum('member', 'leader')	默认为'member'	成员在团体中的角色(普通成员，团体管理员)

6. group_record 表（团体记录）

字段名	数据类型	备注	说明
group_record_id	int	主键，自增（主码）	团体成员管理操作记录的唯一标识
operator_id	int	外键，引用自 user 表的 user_id	操作发起成员的 ID
target_id	int	外键，引用自 user 表的 user_id	操作对象成员的 ID
group_id	int	外键，引用自 group 表的 group_id	操作发生的团体的 ID
time	datetime		操作发生的时间
operate_type	varchar(10)		操作的类型

7. chat 表（聊天的信息）

字段名	数据类型	备注	说明
chat_id	int	主码，自增	聊天的唯一标识
chat_name	varchar(50)		聊天名称
type	enum("friendChat", "friendGroup", "matchChat", "groupChat")	默认为 'friendChat'	聊天类型(群聊聊天，好友聊天)
creation_time	datetime		创建时间
photo	varchar(100)		聊天照片的 URL

8. chat_member 表（群聊的成员信息）

字段名	数据类型	备注	说明
chat_member_id	int	主码，自增	成员关系的唯一标识

chat_id	int	外键，引用自 chat 表的 chat_id	所属聊天的 ID
user_id	int	外键，引用自 user 表的 user_id	成员的 ID

9. message 表（消息）

字段名	数据类型	备注	说明
message_id	int	主键，自增（主码）	消息的唯一标识
time	datetime		消息时间
content	varchar(255)		消息内容
chat_id	int	外键，引用自 chat 表的 chat_id	所属聊天的 ID
user_id	int	外键，引用自 user 表的 user_id	用户的 ID

10. message_validation 表（验证消息）

字段名	数据类型	备注	说明
message_validation_id	int	主键，自增（主码）	验证信息的唯一标识
phone	varchar(20)		手机号
message_code	varchar(10)		验证码
expiration_time	datetime		过期时间
user_id	int	外键，引用自 user 表的 user_id	用户的 ID

11. notification 表（通知消息）

字段名	数据类型	备注	说明

notification_id	int	主键，自增（主码）	通知的唯一标识
type	enum('system','reservation','friend','group')	默认为 'system'	通知类型(系统通知、预约通知、好友申请通过通知、团体加入成功通知)
title	varchar(20)		通知标题
content	varchar(255)		通知内容
timestamp	datetime		通知时间
state	enum('read','unread','mark','deleted')	默认为'unread'	通知当前状态（已读、未读、标记）
user_id	int	外键，引用自 user 表的 user_id	通知用户的 ID
group_id	int	外键，引用自 group 表的 group_id	团体 ID

12. venue 表（场地管理方）

字段名	数据类型	备注	说明
venue_id	int	主键，自增（主码）	场馆管理唯一标识
name	varchar(20)		场馆名称
image	varchar(100)		场馆图片
description	text		场馆描述
location	varchar(100)		场馆位置
contact_number	varchar(20)		场馆联系方式
venue_state	enum('open','closed')	默认为 'closed'	场馆状态（开放，关闭）

13. court 表（场地）

字段名	数据类型	备注	说明

court_id	int	主键，自增（主码）	场地唯一标识
court_name	varchar(50)		场地名称
type	varchar(20)		场地类型
location	varchar(100)		场地位置
capacity	int		场地容量
court_state	enum('open', 'closed')	默认为 'closed'	场地状态（开放，关闭）
venue_id	int	外键，引用自 venue 表的 venue_id	所属的场馆

14. timeslot 表（场地开放的时间段）

字段名	数据类型	备注	说明
timeslot_id	int	主键，自增（主码）	时间段的唯一标识
start_time	datetime		开始时间
end_time	datetime		结束时间
venue_id	int	外键，引用自 venue 表的 venue_id	所属场馆管理方

15. court_availability 表（可预约项）

字段名	数据类型	备注	说明
availability_id	int	主键，自增（主码）	可预约项的唯一标识
timeslot_id	int	外键，引用自 timeslot 表的 timeslot_id	可预约项的时间段
court_id	int	外键，引用自 court 表的 court_id	可预约项的场地
state	enum('reservable', 'matching', 'full',	默认为 'closed'	当前状态（可预约，拼场预约中，已满，

	'closed')		关闭）
price	decimal(10,2)		可预约项的价格

16. reservation 表（预约信息）

字段名	数据类型	备注	说明
reservation_id	int	主键，自增（主码）	预约唯一标识
type	enum('individual', 'group', 'match')	默认为 'individual'	预约类型
availability_id	int	外键，引用自 court_availablity 表的 availability_id	预约的时间段

17. reservation_record 表（预约的记录）

字段名	数据类型	备注	说明
reservation_record_id	int	主键，自增（主码）	记录唯一标识
state	enum('reserved','signed','matching','cancelled','violated')	默认为 'reserved'	预约状态(已预约,已签到、拼场中、已取消、违约)
time	dateTime		操作的时间
user_id	int	外键，引用自 user 表的 user_id	操作用户的 ID
state	enum('reserved','signed','matching','cancelled','violated')	默认为 'reserved'	预约状态(已预约,已签到、拼场中、已取消、违约)
reservation_id	int	外键，引用自 reservation 表的 reservation_id	预约的 ID

18. group_reservation 表（团体预约时团体的信息）

字段名	数据类型	备注	说明
group_reservation_id	int	主键，自增（主码）	团体预约唯一标识

group_id	int	外键，引用自 group 表的 group_id	预约的团体的 ID
reservation_id	int	外键，引用自 reservation 表的 reservation_id	预约记录的 ID

19. match_reservation 表（拼场预约信息）

字段名	数据类型	备注	说明 ID
match_reservation_id	int	主键，自增（主码）	拼场预约的唯一标识
reservation_id	int	外键，引用自 reservation 表的 reservation_id	预约记录的 ID
expiration_time	datetime		过期时间
reserved_count	int		已预约人数

20. user_reservation 表（预约的用户的信息）

字段名	数据类型	备注	说明
user_reservation_id	int	主键，自增（主码）	用户预订唯一标识
user_id	int	外键，引用自 user 表的 user_id	预约用户的 ID
reservation_id	int	外键，引用自 reservation 表的 reservation_id	预约记录的 ID

21. venue_comment 表（场地评价）

字段名	数据类型	备注	说明
venue_comment_id	int	主键，自增（主码）	场地评论唯一标识
content	varchar(255)		评论内容

time	datetime		评论时间
score	float	精确到小数点后一位，最小 1.0，最大 5.0	评价分数
user_id	int	外键，引用自 user 表的 user_id	评价用户的 ID
venue_id	int	外键，引用自 venue 表的 venue_id	评价场馆的 ID

22. violation 表（用户违约记录）

字段名	数据类型	备注	说明
violation_id	int	主键，自增（主码）	违规记录唯一标识
state	enum('normal','locked')	默认为 'normal'	账户状态(正常、封禁)
violation_count	int		违规次数
unlock_time	datetime		解锁时间
user_id	int	外键，引用自 user 表的 user_id	用户 ID

2.7 系统出错处理设计

2.7.1 出错信息

以下以一览表的形式说明每种可能出现的软错误和硬错误发生时，系统输出信息的形式，含义及处理方法。

错误类型	错误信息	处理方法	系统输出信息
TCP 连接错误	连接超时或断开	尝试重新连接，特定时间后输出错误信息	尝试失败后输出对应的网络错误信息，显示在日志中
系统部分自定义错误	用户名密码错误	不予登录	系统前端提示输入错误，登陆失败
	注册时用户名已存在	不予注册	系统前端提示该用户名已经被注册过
	日期信息错误	不予填入	系统前端提示日期错误

	在没有权限的情况下管理团体	不予执行	系统前端提示用户没有权限执行该操作。
程序错误	执行数据库的操作时违反约束条件	该操作被终止	系统根据预设的事务回滚，输出错误信息。
	内部程序错误	服务重启	系统提示内部错误并重启服务
其他错误	服务器故障	无	系统提示服务器故障信息并等待人为修复重启

2.7.2 补救措施

由于错误类型不尽相同，错误的原因也各有差异，因此对于不同的错误我们采取不同的变通措施，列举如下：

- **对于软错误**，如用户输入非法信息，可能会导致后端数据库的操作错误和程序内部发生错误，系统直接在前端的输入或修改操作中对数据本身进行验证，分析错误类型，并给出相应的错误提示语句，从用户输入层面避免输入信息的非法，从而从一定程度上避免因输入非法带来的故障。

  如果存在前端未检测出来的错误信息，被传递给了后端，后端在执行业务逻辑也会进行一定的检验，对于错误数据，会抛出错误，经过捕捉后，将错误信息返回给前端以提示。

  如果这些保护措施都为能拦截该错误数据或操作，导致程序运行发生异常，我们也为程序设置了回滚机制，以最大可能确保已有数据的安全和程序的恢复。

- **对于硬错误**，如数据库连接错误和网络连接错误而言，由于错误类型较少且原因明确，可在前端输出对应的提示信息，或直接将相应日志输出在前端页面中；而对于程序内部的错误，可以在程序编写阶段设置对应的异常捕获程序和抛出异常语句，在出错时输出相应的错误语句，将服务或程序重启，避免整个业务的故障停滞。维护人员可在输出日志中查看错误信息及时修正。

- 采取适当的后备技术，如当数据库受到攻击或删库时，通过定期转储数据库，对数据库定期备份避免非法攻击带来的不可逆故障。

- 对于用户密码等信息，将提供 MD5 加密，在数据库中将密码加密存储。