乒乓球培训管理系统设计文档

一、系统概述

本系统为乒乓球培训机构提供学员管理、教练管理、课程预约、充值缴费、消息通知、校区管理等信息化服务。采用前后端分离架构，前端基于 Vue 3，后端基于 Kotlin + Ktor，数据库为 MySQL。

二、系统角色与权限

角色类型	核心权限描述
学员（student）	管理个人信息、查看课程安排、接收消息通知、查询充值记录、进行课程预约与反馈
教练（coach）	管理个人课程、查看名下学员信息、发送消息给学员、处理课程预约请求、记录训练反馈
校区管理员	管理本校区用户（学员 / 教练）、管控校区内课程与预约、查看校区财务数据、处理校区运营事务
超级管理员	拥有系统全权限、支持跨校区管理、配置系统基础参数、监控系统运行状态

三、主要功能模块设计

3.1 用户认证与权限管理模块

支持学员、教练、校区管理员、超级管理员四种角色；

登录 / 注册（学员、教练）；

Session/Cookie 认证，自动登录状态检查；

路由级、组件级、操作级权限控制。

3.2 用户管理模块

用户信息的增删改查（CRUD）；

个人中心页面；多角色切换与管理。

3.3 课程与课表管理模块

课程预约、取消、反馈；学员课表（周 / 月视图切换）；

教练课表管理；课程状态管理与时间线展示。

3.4 消息系统模块 (后端已实现，前端未实现)

站内消息推送（广播、个人、系统通知）；

消息中心页面；

消息已读 / 未读状态管理。

3.5 充值与钱包系统模块

学员账户充值；充值记录查询；

管理员可查看所有充值记录；

金额验证与操作日志。

3.6 校区管理模块

校区信息管理；

校区学员、教练管理；

校区课程与预约管理；

校区财务数据管理。

3.7 仪表盘与统计模块

多角色仪表盘（学员、教练、校区管理员、超级管理员）；

数据统计与可视化展示。

四、前端页面结构设计（核心页面）

4.1 基础入口页面

登录页、注册页（学员 / 教练）。

4.2 学员专属页面

学员仪表盘、课表、预约课程、找教练、账户充值。

4.3 教练专属页面

教练仪表盘、课程管理、学生双选。

4.4 管理员页面

校区管理员页面（学员 / 教练 / 预约管理、系统日志）；

超级管理员页面（校区管理、服务状态）。

4.5 通用页面

消息中心、

个人中心。

五、后端 API 接口设计

5.1 用户相关接口（/user）

接口路径	请求方式	接口功能描述	关联角色
/user/signup	POST	用户注册（支持学员、教练两类角色）	学员、教练
/user/login	POST	用户登录，获取认证凭证	所有角色

/user/logout	POST	用户登出，销毁当前会话	所有角色
/user/info	GET	获取当前登录用户的个人信息	所有角色
/user/info	PUT	修改当前登录用户的个人信息	所有角色
/user/change-password	PUT	修改当前登录用户的密码	所有角色
/user/allUsers	GET	获取系统内所有用户信息	超级管理员
/user/users	GET	分页获取用户列表，支持条件筛选	校区管理员、超级管理员
/user/totalUserNum	GET	获取当前校区的用户总数	校区管理员
/user/allTotalUserNum	GET	获取全系统的用户总数	超级管理员
/user/create	POST	创建新用户（支持指定角色）	校区管理员、超级管理员
/user/{userId}	DELETE	删除指定 ID 的用户	校区管理员（本校区）、超级管理员
/user/{userId}/reset-password	PUT	重置指定 ID 用户的密码	校区管理员（本校区）、超级管理员
/user/users/{username}	GET	根据用户名模糊查询或精确查询用户	校区管理员、超级管理员

5.2 教练相关接口（/coach）

接口路径	请求方式	接口功能描述	关联角色
/coach/coaches	GET	获取教练列表（支持分页与基础筛选）	所有角色（学员用于选教练，管理员用于管理）
/coach/queryCoach	POST	多条件查询教练（如性别、等级、校区等）	所有角色
/coach/queryCoachByUuid	POST	通过 UUID 查询指定教练的详细信息	所有角色
/coach/coaches/campusNotApproved	GET	获取当前校区内未审核的教练列表	校区管理员
/coach/coaches/allNotApproved	GET	获取全系统内未审核的教练列表	超级管理员
/coach/approve	POST	审核教练申请并标记为通过	校区管理员（本校区）、超级管理员

5.3 课程相关接口（/courses）

接口路径	请求方式	接口功能描述	关联角色
/courses/my-schedule	GET	获取当前学员的个人课表（周 / 月视图）	学员

/courses/coach-schedule/{coachId}	GET	获取指定教练的课表	学员（选课时）、管理员
/courses/my-ordered-courses	GET	获取当前学员已预约的课程列表	学员
/courses/feedback	POST	提交课程反馈（训练收获、建议等）	学员
/courses/coach-schedule	GET	教练获取自己的课表	教练
/courses/{courseId}/status	PUT	修改指定课程的状态（如待确认→已确认）	教练、管理员
/courses/create	POST	创建新课程（指定教练、时间、球台等）	教练、管理员
/courses/query	POST	多条件查询课程（如时间、状态、教练等）	所有角色（按需筛选）
/courses/book	POST	预约课程（需已建立师生双选关系）	学员
/courses/cancel	POST	发起课程取消申请	学员、教练
/courses/querycancel	GET	查询当前用户可取消的课程列表	学员、教练
/courses/judgecancel	POST	审核课程取消申请（同意 / 拒绝）	教练（学员发起时）、学员（教练发起时）

/courses/querypending	GET	查询待审核的课程（如待确认、待取消）	教练、管理员
/courses/coach-judge	POST	教练审核学员的课程预约申请	教练
/courses/available-tables	GET	查询指定时间段内可用的球台列表	学员（预约时）、教练（创建课程时）

5.4 钱包 / 充值相关接口（/wallet）

接口路径	请求方式	接口功能描述	关联角色
/wallet/balance	GET	查询当前用户的账户余额	学员（查个人）、管理员（查指定用户）
/wallet/recharge	POST	学员账户充值（支持微信、支付宝、线下支付）	学员
/wallet/recharge/history	GET	获取当前学员的个人充值记录	学员
/wallet/recharge/records	GET	获取全系统的所有充值记录	超级管理员
/wallet/recharge/records/campus	GET	获取当前校区的充值记录	校区管理员

/wallet/recharge/records/{username}	GET	获取指定用户名用户的充值记录	校区管理员（本校区）、超级管理员

5.5 消息相关接口（/messages）

接口路径	请求方式	接口功能描述	关联角色
/messages/unread-count	GET	获取当前用户的未读消息数量	所有角色
/messages/{id}/read	PUT	将指定 ID 的消息标记为已读	所有角色
/messages/read-all	PUT	将当前用户的所有未读消息标记为已读	所有角色
/messages/{id}	DELETE	删除指定 ID 的消息	所有角色

5.6 师生双选相关接口（/mutual-selection）

接口路径	请求方式	接口功能描述	关联角色
/mutual-selection/apply	POST	学员发起绑定教练的申请	学员
/mutual-selection/student-applications	GET	学员查看自己发起的双选申请记录	学员
/mutual-selection/withdraw	POST	学员撤回未审核的双选申请	学员

/mutual-selection/cancel-approved	POST	学员解除已绑定的教练关系	学员
/mutual-selection/coach-applications	GET	教练查看收到的双选申请列表	教练
/mutual-selection/review	POST	教练审核学员的双选申请（同意 / 拒绝）	教练
/mutual-selection/admin-create	POST	管理员手动创建师生双选关系	校区管理员、超级管理员
/mutual-selection/all	GET	管理员查看系统内所有师生双选关系	校区管理员（本校区）、超级管理员
/mutual-selection/student/current-coaches	GET	学员查看当前绑定的教练列表	学员
/mutual-selection/coach/current-students	GET	教练查看当前绑定的学员列表	教练
/mutual-selection/coach/historical-students	GET	教练查看历史绑定的学员列表	教练
/mutual-selection/admin/relation/{relationId}	GET	管理员查看指定 ID 的师生关系详情	校区管理员、超级管理员
/mutual-selection/admin/relation/{relationId}	POST	管理员修改指定 ID 的师生关	校区管理员、超级管理员

		系（如解除、重新绑定）	

5.7 校区相关接口（/campus）

接口路径	请求方式	接口功能描述	关联角色
/campus/create	POST	创建新校区（指定名称、地址、负责人等）	超级管理员
/campus/names	GET	获取系统内所有校区的名称列表	所有角色（注册、选课时选择校区）
/campus/addTable	POST	为指定校区添加球台（设置球台编号、状态）	校区管理员、超级管理员
/campus/queryFreeTables	GET	查询指定校区、指定时间段内的空闲球台	校区管理员、超级管理员

5.8 管理员相关接口（/admin）

接口路径	请求方式	接口功能描述	关联角色
/admin/students	GET	获取系统内所有学员信息	超级管理员
/admin/coaches	GET	获取系统内所有教练信息	超级管理员

5.9 日志相关接口（/logs）

接口路径	请求方式	接口功能描述	关联角色
/logs	多方式	系统日志相关操作（如查询日志、筛选日志、导出日志等，具体功能需结合实现）	校区管理员、超级管理员

5.10 比赛相关接口（/competition）

接口路径	请求方式	接口功能描述	关联角色
/competition/create	POST	创建比赛（指定时间、组别、校区等）	校区管理员、超级管理员
/competition/self-campus	GET	获取当前校区的比赛列表	校区管理员、本校区学员 / 教练
/competition/all	GET	获取全系统的比赛列表	超级管理员
/competition/{competitionId}	GET	获取指定 ID 比赛的详细信息（赛程、参赛人员等）	所有角色（参赛人员、管理员）
/competition/signup	POST	学员报名参加指定比赛（扣除报名费）	学员
/competition/signup	GET	获取指定比赛的报名信息（报名人数、名单等）	教练、管理员

/competition/arrange/{competitionId}	POST	为指定比赛安排赛程（分组、对阵等）	校区管理员、超级管理员
/competition/my-schedule/{competitionId}	GET	获取当前学员在指定比赛中的个人赛程安排	学员

六、数据库设计（核心表结构）

6.1 用户相关表

用户表（支持多角色）class UserEntity(uuid: EntityID<UUID>) : UUIDEntity(uuid) {

companion object : UUIDEntityClass<UserEntity>(UserTable)

var username by UserTable.username

var encryptedPassword by UserTable.encrypted_password

var realName by UserTable.real_name

var gender by UserTable.gender

var age by UserTable.age

var phoneNumber by UserTable.phone_number

var email by UserTable.email

var campus by CampusEntity referencedOn UserTable.campus

var role by UserTable.role

var status by UserTable.status

var createdAt by UserTable.created_at

var lastLoginAt by UserTable.last_login_at

};

学员表 class StudentEntity(id: EntityID<UUID>) : Entity<UUID>(id) {

companion object : EntityClass<UUID, StudentEntity>(StudentTable)

var userId by UserEntity referencedOn StudentTable.id

var balance by StudentTable.balance

var maxCoach by StudentTable.max_coach

var currentCoach by StudentTable.current_coach

}、

教练表 class CoachEntity(id: EntityID<UUID>) : Entity<UUID>(id) {

companion object : EntityClass<UUID, CoachEntity>(CoachTable)

var userId by UserEntity referencedOn CoachTable.id

var photoUrl by CoachTable.photo_url

var achievements by CoachTable.achievements

var level by CoachTable.level

var hourlyRate by CoachTable.hourly_rate

var balance by CoachTable.balance

var maxStudents by CoachTable.max_students

var currentStudents by CoachTable.current_students

var isApproved by CoachTable.is_approved

var approvedBy by CoachTable.approved_by

}。

6.2 校区与课程相关表

校区表: class CampusEntity(id: EntityID<Int>) : IntEntity(id) {

companion object : IntEntityClass<CampusEntity>(CampusTable)

var campusName by CampusTable.campus_name

var address by CampusTable.address

var contactPerson by CampusTable.contact_person

var phone by CampusTable.phone

var email by CampusTable.email

var balance by CampusTable.balance

var isCentral by CampusTable.is_central

var tableNumber by CampusTable.tableNumber

var createdAt by CampusTable.created_at

var lastLoginAt by CampusTable.last_login_at

}

课程表:class CourseEntity(uuid: EntityID<UUID>) : UUIDEntity(uuid) {

companion object : UUIDEntityClass<CourseEntity>(CourseTable)

var coach by CoachEntity referencedOn CourseTable.coach

```
    var student by StudentEntity referencedOn CourseTable.student
    var campus by CampusEntity referencedOn CourseTable.campus
    var table by TableEntity referencedOn CourseTable.table
    var title by CourseTable.title
    var date by CourseTable.date
    var startTime by CourseTable.startTime
    var endTime by CourseTable.endTime
    var status by CourseTable.status
    var price: Float by CourseTable.price
    var paymentStatus by CourseTable.paymentStatus
    var attendanceStatus by CourseTable.attendanceStatus
    var studentFeedback by CourseTable.studentFeedback
    var studentRating by CourseTable.studentRating
    var createdAt by CourseTable.createdAt
}
双选关系表:class MutualSelectionEntity(uuid: EntityID<UUID>) : UUIDEntity(uuid) {
    companion object : UUIDEntityClass<MutualSelectionEntity>(MutualSelectionTable)
    var coachID by CoachEntity referencedOn MutualSelectionTable.coach_id
    var studentID by StudentEntity referencedOn MutualSelectionTable.student_id
    // 状态
    var status by MutualSelectionTable.status
    // 申请时间
    var applicationTime by MutualSelectionTable.application_time
    // 预期开始时间
    var expectedStartTime by MutualSelectionTable.expected_start_time
    // 实际开始时间
    var actualStartTime by MutualSelectionTable.actual_start_time
    // 结束时间
    var endTime by MutualSelectionTable.end_time
}
球桌表 class TableEntity(id: EntityID<UUID>) : Entity<UUID>(id) {
    companion object : EntityClass<UUID, TableEntity>(TableTable)
```

```
    var status by TableTable.status
    var group by TableTable.group
    var indexInCampus by TableTable.index_in_campus
    var campus by CampusEntity referencedOn TableTable.campus
}
```

6.3 业务关联表

```
消息表 class MessageEntity(uuid: EntityID<UUID>) : UUIDEntity(uuid) {
    companion object : UUIDEntityClass<MessageEntity>(MessageTable)

    var sender by UserEntity optionalReferencedOn MessageTable.senderId
    var recipient by UserEntity referencedOn MessageTable.recipientId
    var title by MessageTable.title
    var content by MessageTable.content
    var type by MessageTable.type
    var isRead by MessageTable.isRead
    var createdAt by MessageTable.createdAt
    var readAt by MessageTable.readAt
};
钱包 / 充值记录表
class RechargeEntity(id: EntityID<Int>) : IntEntity(id) {
    companion object : IntEntityClass<RechargeEntity>(RechargeTable)
    var userId by RechargeTable.userId
    var amount by RechargeTable.amount
    var createdAt by RechargeTable.createdAt
};
```

6.4 比赛相关表

```
class CompetitionArrangementEntity(id: EntityID<Int>) : IntEntity(id) {
    companion object :
IntEntityClass<CompetitionArrangementEntity>(CompetitionArrangementTable)
    var competition by CompetitionEntity referencedOn
CompetitionArrangementTable.competition
```

```
    var turnNumber by CompetitionArrangementTable.turnNumber
    var table by TableEntity referencedOn CompetitionArrangementTable.table
    var playerA by UserEntity referencedOn CompetitionArrangementTable.playerA
    var playerB by UserEntity referencedOn CompetitionArrangementTable.playerB
    var status by CompetitionArrangementTable.status
    var result by CompetitionArrangementTable.result
    var winner by CompetitionArrangementTable.winner
}
class CompetitionEntity(uuid: EntityID<UUID>) : UUIDEntity(uuid) {
    companion object : UUIDEntityClass<CompetitionEntity>(CompetitionTable)
    var name by CompetitionTable.name
    var type by CompetitionTable.type
    var campus by CampusEntity referencedOn CompetitionTable.campus
    var date by CompetitionTable.date
    var registrationDeadline by CompetitionTable.registrationDeadline
    var status by CompetitionTable.status
    var fee by CompetitionTable.fee
    var description by CompetitionTable.description
}
class CompetitionSignupEntity(id: EntityID<Int>) : IntEntity(id) {
    companion object : IntEntityClass<CompetitionSignupEntity>(CompetitionSignupTable)
    var competition by CompetitionEntity referencedOn CompetitionSignupTable.competition
    var user by UserEntity referencedOn CompetitionSignupTable.user
    var group by CompetitionSignupTable.group
    var status by CompetitionSignupTable.status
    var createdAt by CompetitionSignupTable.createdAt
}
```

七、技术架构细节

7.1 前端技术栈

Vue 3 + Vite + Pinia + 现代化 UI 设计。

7.2 后端技术栈

Kotlin + Ktor + Exposed ORM + MySQL；

权限与认证：Session/Cookie、路由守卫；

API 文档：OpenAPI/Swagger；

日志与监控：系统日志页面。

八、已实现核心功能总结

多角色权限与认证体系；

用户、课程、校区、消息、钱包等主要业务模块；

完整的前后端页面与 API 对接；

响应式布局与现代化 UI；

统一的错误处理与 API 响应格式。