乒乓球培训管理系统 · 开发总结报告

更新时间：2025-09-26（基于 dev 分支最新代码与文档）

本报告基于以下材料综合形成：

- 仓库代码（frontend/、backend/、docker-compose.yml 等）
- Git 历史与统计（近两日功能高频提交 + 主要贡献者概况）
- 第一次与第二次迭代的工作总结、技术文档与需求分析

报告目标：沉淀阶段性成果，复盘迭代过程，给出质量评估与后续建议，支撑下一阶段规划与验收。

一、项目概览

- 项目定位：为乒乓球培训机构提供学员、教练、校区、课程预约、赛事管理、消息中心与钱包充值的一体化信息化平台。
- 架构模式：前后端分离，容器化部署（Docker Compose），MySQL 持久化。
- 角色体系：学员（student）、教练（coach）、校区管理员（campus_admin）、超级管理员（super_admin）。

技术栈摘要

- 前端：Vue 3.5.x、Vite 7、Pinia 3、Vue Router 4.5、Element Plus 2.11、Axios、Vitest。
- 后端：Kotlin 2.1.10、Ktor 3.2.3、Exposed ORM、MySQL 8、HikariCP、Logback、BCrypt。
- 部署与运维：Docker Compose（bitnami/mysql）、.env 配置、日志文件输出（logs/）。

二、里程碑与时间线

- 2025-09-07 第一次迭代完成
  - 前后端基础骨架、登录注册、基础权限框架、核心 UI 组件初版。
- 2025-09-13 第二次迭代完成
  - 设计系统（Glassmorphism）建立与落地；消息中心、钱包充值、校区管理前后端打通；权限与路由守卫完善；环境配置与开发者工具页。
- 2025-09-15 ~ 2025-09-26 连续功能增强
  - 比赛管理：创建/报名、报名人数与状态统计、赛程安排与查看。
  - 预约管理：学生预约视图、教练审核预约/取消、我的预约菜单。
  - 管理端：super_admin/admin 页面补全（用户、校区、日志等）。
  - 后端：表结构与底层数据结构重构、权限缺陷修复、API 充实、球台资源与时间段查询、排序等。
  - 工程：日志系统引入/增强、Docker 配置优化、前端错误提示统一化。

注：以上节点来源于 documents/ 迭代文档与 2025-09-25/26 的高频提交记录。

三、功能实现现状（按角色/模块）

1. 通用能力

- 登录/注册（表单认证 + Session 持久化，前端以 cookie 维持登录，Pinia 管状态）。
- 权限与路由守卫：
  - 路由 meta + 角色校验（super_admin、campus_admin、student、coach）；
  - 组件/页面级权限判定与指引；
  - 游客允许访问的 Dashboard、设计系统及 DevTools 演示页。
- 消息中心：消息列表、未读计数、标记已读、删除与（管理员）发送。
- 日志与错误处理：后端 StatusPages/CallLogging；前端 errorHandler 与统一提示。

2. 学员端（student）

- 找教练/教练详情、我的教练。
- 预约训练、我的预约、我的课程表。
- 钱包充值与余额查询、充值记录（分页/筛选在服务端具备扩展基础）。
- 比赛报名、比赛/赛程查看、赛程安排浏览、我的比赛/赛程。
- 训练评价（页面与路由就绪，后端可逐步对接）。

3. 教练端（coach）

- 预约审批/取消审批、课程日程。
- 学员反馈查看、双选流程（与学员互选）。

4. 校区与超级管理员（campus_admin/super_admin）

- 校区管理（创建/列表）、用户管理、系统日志页面。
- 球台（球桌）资源管理、预约/比赛相关管理入口。
- 设计系统与开发工具页，便于体验一致性与调试。

四、架构与关键实现

前端

- 构建与工程化：Vite 7、ESLint 9 + Prettier、Vitest 单元测试脚手。
- 状态管理：Pinia 按域划分（用户、消息等），LocalStorage 做登录状态落标识（session-active）。
- 路由：角色权限嵌入 meta，beforeEach 统一守卫并支持游客访问的演示/工具页；动态标题设置。
- 设计系统：
  - CSS 变量与玻璃拟态（backdrop-filter、透明度层级、阴影规范）；
  - 通用组件（GlassCard/ModernButton/FormInput/TagBadge/StatusGrid 等）；
  - 响应式布局与移动端优先策略。

后端

- Ktor 插件：
  - Authentication（Session）
  - CORS
  - ContentNegotiation(JSON)

    - StatusPages
    - CallLogging
    - OpenAPI/Swagger
    - Sessions。
- ORM 与持久化：
    - Exposed（core/dao/jdbc/datetime）、
    - HikariCP 连接池
    - MySQL 8（docker容器化）
- 典型 API：
    - /user：signup/login/logout/profile 等（登录后设置 Session，前端携带 cookie）。
    - /messages：查询/未读数/标记已读/批量已读/删除；管理员可发送。
    - /wallet：余额、充值、充值记录（含管理员视角）。
    - /campus：创建、列表；
    - /tournament|/appointments|/tables：比赛/预约/球台查询与管理（近期提交中持续完善）。
- 安全：BCrypt 密码加密存储；多层权限校验；输入校验与错误码分层。

部署与运行

- Docker Compose：backend + bitnami/mysql，支持 .env.default/.env；健康检查、端口与数据卷映射已配置。
- 本地开发：前端 dev 代理 /api → VITE_API_BASE_URL；后端默认 8080；Node 20+/JDK 17+。

五、工程质量与度量

团队分工

余绍缘

- 角色与范围：全栈/架构促进者，覆盖设计系统、前端核心页面与组件，及后端多个领域路由与服务
- 主要产出（节选）：
    - 前端：设计系统（variables/design-system.css、GlassCard/ModernButton/TagBadge/StatusGrid 等）、路由与权限、消息中心、学生侧预约/课表/充值、校区/管理员页面等
    - 后端：认证/会话/异常、User/Message/Wallet/Campus/Coach/Course/Competition/Admin/MutualSelection 等路由与 Service、OpenAPI 文档与基础设施插件
- 贡献数据：
    - 提交：155 次（含未指明账户合并）
    - 变更行：总计 ≈ 900,074
- 亮点：
    - 自研设计系统沉淀，统一视觉与交互、显著提升复用与效率
    - 前后端架构完善与打通，业务模块覆盖广、基础设施健全

王邺

- 角色与范围：全栈开发，前后端均衡投入
- 主要产出（节选）：
    - 前端：消息/校区/钱包等业务对接与页面完善、权限守卫/状态管理协作

    - 后端：消息、钱包、校区、用户管理等接口实现与参数校验、分页与记录查询
- 贡献数据：
    - 提交：44 次
    - 变更行：总计 778,145
- 亮点：
    - 以业务交付为导向的前后端协同，接口与页面闭环推进

杨元枭

- 角色与范围：前端与文档侧贡献，参与页面与说明文档完善
- 主要产出（节选）：
    - 文档：需求/技术/工作总结体系化整理，降低沟通成本
    - 前端：部分页面与组件增强、路由/状态协作、设计系统使用落地
- 贡献数据：
    - 提交：24 次
    - 变更行：总计 770,037
- 亮点：
    - 文档体系与可读性提升显著，助力团队对齐与复盘

代码与规范

- 前端 ESLint + Prettier 一致化；Vitest 测试基建可用（tests/ 与 scripts 已就绪）。
- 后端 Kotlin/Gradle 规范化，Logback 日志输出；StatusPages 统一异常处理。
- 文档体系：项目与设计系统说明、迭代文档（01/02）较完善。

提交与贡献（git shortlog 摘要）

- 主要贡献者（含 merge，统计至今）：
    - YSY：155 次
    - Yewargg：44 次
    - yanfeng：24 次

日志与监控

- logs/ 中已生成运行日志（application.*.log）；后端启用 CallLogging；可在下阶段补充健康探针/指标端点与可视化。

测试与 CI/CD

- 单元测试：前端 Vitest 可直接启用；后端建议引入 JUnit + Ktor Test Host 扩展覆盖。
- 集成与端到端：当前以人工联调为主；可引入 Playwright/Cypress 做关键路径 E2E。
- CI/CD：仓库未见现成流水线配置（建议后续接入 GitHub Actions）。

六、问题复盘与解决策略

来源：迭代文档与近期提交信息。

1. 学习曲线与协作磨合：

- 问题：Kotlin/Ktor 与 Vue 3 生态上手成本、前后端契约反复。
- 解决：拆解任务与接口契约、内置 DevTools 与 OpenAPI/Swagger、建立权限与设计系统规范，缩短沟通路径。

2. 数据建模复杂度（多角色、多业务）：

- 问题：用户/教练/学员/校区/预约/比赛等实体间约束复杂，存在阶段性重构。
- 解决：Exposed 分层建模与迁移；引入枚举与严格校验；逐步补齐分页/筛选与关联查询。

3. 权限与边界问题：

- 问题：部分接口早期权限校验缺口（Git 有多次“修复权限问题”提交）。
- 解决：后端多 Session realm/路由守卫，前端 meta 角色白名单与降级导航。

4. 设计一致性与体验：

- 解决：建立 Glassmorphism 设计系统、统一组件与 token、响应式与弱网/错误提示一致化。

七、当前短板与风险

- 自动化测试覆盖不足（尤其是后端与关键业务流程）。
- 并发与资源冲突风险（预约/球台/赛程的时间段冲突与校验）。
- 性能与容量评估欠缺（高并发预约与赛事编排场景）。
- 移动端深度体验仍有空间（触控/离线/推送）。
- 运维与可观测性基础薄弱（指标/告警/追踪未完善）。

八、持续集成开发路线图

1. 可靠性与质量

- 建立最小集回归用例：登录/注册、预约创建与审核、比赛创建/报名/编排、钱包充值。
- 后端引入 JUnit + Testcontainers（MySQL）做集成测试；前端补 3~5 个关键组件单测。
- 接入 GitHub Actions：lint + unit test + build，提交门禁。

2. 业务完善与治理

- 预约/赛程的时间冲突校验与并发控制（唯一键 + 业务锁/查询幂等）。
- 钱包与支付：分账户流水、限额与风控规则抽象；与实际支付网关的对接预研。
- 消息中心：批量操作/分页查询落服务端，推送与未读策略优化。
- 比赛编排：支持分组/淘汰赛制，导出赛程与结果复核。

3. 可观测性与运维

- 健康检查与指标：/health、/metrics（Micrometer/Prometheus）与日志结构化。
- 任务/审计日志：关键操作与管理员行为留痕。
- Docker Compose 分环境参数化（dev/test/prod），最小化线上启动文档。

4. 体验与设计

- 继续沉淀设计系统（空状态、加载、错误、可访问性规范），移动端表单与手势优化。

- 统一空/错文案与异常码到位的提示映射。