乒乓球培训管理系统

本项目是吉林大学软件工程课程设计的作业，旨在为乒乓球培训机构提供一套完整的信息化管理解决方案。经过两次迭代开发，已实现了完整的用户管理、消息系统、充值功能、校区管理等核心业务功能。

项目概述

乒乓球培训管理系统是一个现代化的Web应用程序，采用前后端分离架构，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、校区管理等全方位的信息化服务。

✨ 核心功能

- **多角色权限管理** - 支持学员、教练、校区管理员、超级管理员四种角色
- **实时消息系统** - 支持广播消息、个人消息、系统通知
- **充值缴费管理** - 完整的充值流程，支持余额管理
- **校区管理系统** - 多校区支持，分级管理
- **现代化UI设计** - 基于Glassmorphism设计风格的美观界面
- **响应式布局** - 完美适配桌面端和移动端

技术架构

前端技术栈

- **Vue 3.5.18** - 现代JavaScript框架，提供响应式用户界面
- **Vue Router 4.5.1** - 单页面应用路由管理
- **Pinia 3.0.3** - 轻量级状态管理库
- **Element Plus 2.11.1** - 基于Vue 3的组件库
- **Vite 7.0.6** - 快速的前端构建工具
- **自研设计系统** - 基于Glassmorphism的15+组件库

后端技术栈

- **Kotlin 2.1.10** - 现代编程语言，完全兼容Java
- **Ktor 3.2.3** - 轻量级协程Web框架
- **Exposed 0.56.0** - Kotlin ORM框架
- **MySQL 8.0** - 关系型数据库
- **HikariCP 5.1.0** - 高性能数据库连接池
- **BCrypt** - 密码加密算法

设计系统

项目采用Glassmorphism设计风格，包含：

- 42个CSS变量统一管理主题色彩
- 15+个可复用组件（GlassCard、ModernButton、StatusGrid等）
- 完整的响应式设计规范

- 统一的动画和交互效果

项目结构

```
TableTennisTrainingSystem/
├── backend/              # 后端项目目录 (Kotlin + Ktor)
│   ├── src/main/kotlin/io/
│   ├── src/main/resources/
│   ├── build.gradle.kts    # 构建配置文件
│   └── README.md
├── frontend/             # 前端项目目录 (Vue 3)
│   ├── src/
│   │   ├── assets/         # 静态资源
│   │   ├── components/     # Vue组件库
│   │   ├── composables/    # 组合式函数
│   │   ├── layouts/        # 布局组件
│   │   ├── stores/         # Pinia状态管理
│   │   ├── styles/         # 设计系统样式
│   │   ├── utils/          # 工具函数
│   │   └── views/          # 页面视图
│   ├── package.json
│   └── vite.config.js
├── documents/            # 项目文档目录
│   ├── 迭代01_20250907/    # 第一次迭代文档
│   └── 迭代02_20250913/    # 第二次迭代文档
├── tests/                # 测试用例目录
└── docker-compose.yml      # Docker 部署配置
```

快速开始

环境要求

- Node.js >= 20.19.0
- Java JDK >= 17
- MySQL >= 8.0
- Git

数据库配置

1. 创建MySQL数据库：

```
CREATE DATABASE table_tennis_system;
```

2. 配置数据库连接： 在 `backend/src/main/resources/application.conf` 中修改数据库配置

后端启动

```
cd backend
./gradlew run
```

后端服务将在 `http://localhost:8080` 启动

🎨 前端启动

```
cd frontend
npm install
npm run dev
```

前端应用将在 `http://localhost:3000` 启动

🏗️ 生产环境构建

```
# 前端构建
cd frontend
npm run build

# 后端构建
cd backend
./gradlew build
```

📈 项目进展

📊 技术指标

- **代码提交数**: 93+ commits
- **前端组件**: 15+ 个可复用组件
- **API接口**: 20+ 个RESTful API
- **页面覆盖**: 15+ 个功能页面
- **测试覆盖**: 单元测试 + 集成测试

👥 角色权限

角色	权限描述
🎓 **学员**	查看个人信息、课程安排、消息通知、充值记录
🏃 **教练**	管理课程、查看学员信息、发送消息通知
🏢 **校区管理员**	管理本校区用户、课程、财务数据
👑 **超级管理员**	系统全权限、跨校区管理、系统配置

🎨 设计特色

Glassmorphism设计语言

- 🔮 玻璃质感背景
- 🌈 渐变色彩系统
- ✨ 毛玻璃模糊效果
- 💎 半透明卡片设计

响应式布局

- 📱 移动端优先设计
- 💻 桌面端完美适配
- 🔄 灵活的栅格系统

🔧 开发规范

Git 分支规范

- 🌿 主分支: `master`
- 🔄 开发分支: `dev`
- ⭐ 功能分支: `feature/*`
- 🔥 热修复分支: `hotfix/*`

代码规范

- 📝 ESLint + Prettier 代码格式化
- 🧪 Vitest 单元测试
- 📚 JSDoc 文档注释
- 🔍 TypeScript 类型检查（部分）

📚 文档结构

- 📋 **需求分析文档** - 详细的功能需求和技术需求
- 🔧 **技术文档** - 架构设计和API文档
- 📊 **工作总结** - 开发过程和成果总结
- 🎯 **项目概览** - 项目整体介绍和规划

🚀 部署指南

Docker 部署

```
docker-compose up -d
```

手动部署

详见各模块的 README 文档

🤝 贡献指南

1. Fork 项目
2. 创建功能分支 (`git checkout -b feature/AmazingFeature`)
3. 提交更改 (`git commit -m 'Add some AmazingFeature'`)
4. 推送到分支 (`git push origin feature/AmazingFeature`)
5. 开启 Pull Request

许可证

本项目采用 GPL3 许可证 - 查看 LICENSE 文件了解详情

开发团队

- **前端开发** - Vue.js + 设计系统开发
- **后端开发** - Kotlin + Ktor API开发
- **产品设计** - UI/UX设计和需求分析
- **测试工程** - 质量保证和测试用例

让乒乓球培训管理更简单高效

Made with by JLU Software Engineering Team