一、用户模块功能划分

1. 用户注册与登录

手机号/学号注册

密码登录

短信验证码登录

JWT认证

2. 校园认证

学号与学校邮箱验证

学校信息绑定

3. 个人信息管理

查看和编辑个人信息（昵称、头像、联系方式）

修改密码

注销账号

4. 信用评分体系

基于任务完成情况的评分

信用分计算与展示

二、技术栈

1. 后端技术栈

框架：Spring Boot

安全框架：Spring Security + JWT

数据库：MySQL

缓存：Redis（用于存储验证码、会话等）

短信服务：阿里云短信或腾讯云短信

邮件服务：SendGrid或本地SMTP服务器

2. 前端技术栈

框架：React或Vue

UI库：Ant Design或Element UI

HTTP库：Axios

三、数据库设计

1. 用户表

sql

复制

CREATE TABLE user (

id INT PRIMARY KEY AUTO\_INCREMENT,

username VARCHAR(50) UNIQUE, -- 用户名（手机号或学号）

password VARCHAR(100), -- 加密后的密码

nickname VARCHAR(50), -- 昵称

avatar VARCHAR(255), -- 头像URL

phone VARCHAR(20), -- 手机号

email VARCHAR(100), -- 邮箱

school\_id INT, -- 学校ID（外键）

credit\_score INT DEFAULT 100, -- 信用分

is\_verified BOOLEAN DEFAULT FALSE, -- 是否完成校园认证

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (school\_id) REFERENCES school(id));

2. 学校表

sql

复制

CREATE TABLE school (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) UNIQUE, -- 学校名称

email\_suffix VARCHAR(50) -- 学校邮箱后缀（如 @pku.edu.cn）);

3. 短信验证码表

sql

复制

CREATE TABLE sms\_code (

id INT PRIMARY KEY AUTO\_INCREMENT,

phone VARCHAR(20), -- 手机号

code VARCHAR(6), -- 验证码

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP);

四、接口设计

1. 用户注册

接口：POST /api/auth/register

参数：

json

复制

{

"username": "13800138000", // 手机号或学号

"password": "123456",

"phone": "13800138000",

"schoolId": 1 // 学校ID}

返回：

json

复制

{

"code": 200,

"message": "注册成功",

"data": {

"userId": 1

}}

2. 用户登录

接口：POST /api/auth/login

参数：

json

复制

{

"username": "13800138000",

"password": "123456"}

返回：

json

复制

{

"code": 200,

"message": "登录成功",

"data": {

"token": "xxxxxx", // JWT Token

"userId": 1

}}

3. 发送短信验证码

接口：POST /api/auth/send-sms

参数：

json

复制

{

"phone": "13800138000"}

4. 校园认证

接口：POST /api/auth/verify-school

参数：

json

复制

{

"studentId": "20210001", // 学号

"email": "20210001@pku.edu.cn" // 学校邮箱}

5. 修改个人信息

接口：PUT /api/user/profile

参数：

json

复制

{

"nickname": "小明",

"avatar": "https://xxx.com/avatar.jpg"}

五、核心代码实现

1. 用户注册

java

复制

@Servicepublic class UserService {

@Autowired

private UserRepository userRepository;

@Autowired

private PasswordEncoder passwordEncoder;

public void register(String username, String password, String phone, int schoolId) {

if (userRepository.findByUsername(username).isPresent()) {

throw new BusinessException("用户名已存在");

}

User user = new User();

user.setUsername(username);

user.setPassword(passwordEncoder.encode(password));

user.setPhone(phone);

user.setSchoolId(schoolId);

userRepository.save(user);

}}

2. 用户登录

java

复制

@Servicepublic class AuthService {

@Autowired

private UserRepository userRepository;

@Autowired

private JwtTokenUtil jwtTokenUtil;

public String login(String username, String password) {

User user = userRepository.findByUsername(username)

.orElseThrow(() -> new BusinessException("用户不存在"));

if (!passwordEncoder.matches(password, user.getPassword())) {

throw new BusinessException("密码错误");

}

return jwtTokenUtil.generateToken(user);

}}

3. 校园认证

java

复制

@Servicepublic class SchoolVerificationService {

@Autowired

private UserRepository userRepository;

public void verifySchool(Long userId, String studentId, String email) {

User user = userRepository.findById(userId)

.orElseThrow(() -> new ResourceNotFoundException("用户不存在"));

// 检查邮箱后缀是否匹配学校

School school = schoolRepository.findById(user.getSchoolId())

.orElseThrow(() -> new ResourceNotFoundException("学校不存在"));

if (!email.endsWith(school.getEmailSuffix())) {

throw new BusinessException("邮箱不属于该学校");

}

user.setVerified(true);

userRepository.save(user);

}}

4. 修改个人信息

java

复制

@Servicepublic class UserProfileService {

@Autowired

private UserRepository userRepository;

public void updateProfile(Long userId, String nickname, String avatar) {

User user = userRepository.findById(userId)

.orElseThrow(() -> new ResourceNotFoundException("用户不存在"));

user.setNickname(nickname);

user.setAvatar(avatar);

userRepository.save(user);

}}

六、扩展优化

1. 安全性

密码加密存储（使用BCrypt）。

敏感操作（如修改密码）增加二次验证。

2. 性能优化

使用Redis缓存用户信息，减少数据库查询。

短信验证码设置有效期（如5分钟）。

3. 用户体验

支持第三方登录（微信、QQ）。

提供忘记密码功能。

通过以上实现思路和代码示例，你可以快速搭建一个功能完善的用户模块。如果需要进一步优化或扩展功能，可以随时讨论！

一、任务模块

1. 功能划分

任务发布

任务接取

任务状态管理（待接取、进行中、已完成、已取消）

任务搜索与筛选

任务推荐

2. 技术栈

后端：Spring Boot + Spring Data JPA

数据库：MySQL

缓存：Redis（用于任务列表缓存）

搜索：Elasticsearch（可选，用于任务全文搜索）

3. 数据库设计

sql

复制

CREATE TABLE task (

id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(100), -- 任务标题

description TEXT, -- 任务描述

reward DECIMAL(10,2), -- 赏金

status ENUM('OPEN','IN\_PROGRESS','COMPLETED','CANCELED'), -- 任务状态

publisher\_id INT, -- 发布者ID（外键）

acceptor\_id INT, -- 接单者ID（外键）

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (publisher\_id) REFERENCES user(id),

FOREIGN KEY (acceptor\_id) REFERENCES user(id));

4. 接口设计

发布任务：POST /api/tasks

接取任务：PUT /api/tasks/{id}/accept

完成任务：PUT /api/tasks/{id}/complete

取消任务：PUT /api/tasks/{id}/cancel

任务列表：GET /api/tasks?status=OPEN&page=1&size=10

5. 核心代码实现

java

复制

@Servicepublic class TaskService {

@Autowired

private TaskRepository taskRepository;

@Transactional

public void createTask(String title, String description, BigDecimal reward, Long publisherId) {

Task task = new Task();

task.setTitle(title);

task.setDescription(description);

task.setReward(reward);

task.setStatus(TaskStatus.OPEN);

task.setPublisherId(publisherId);

taskRepository.save(task);

}

@Transactional

public void acceptTask(Long taskId, Long acceptorId) {

Task task = taskRepository.findById(taskId)

.orElseThrow(() -> new ResourceNotFoundException("任务不存在"));

if (task.getStatus() != TaskStatus.OPEN) {

throw new BusinessException("任务已被接取");

}

task.setStatus(TaskStatus.IN\_PROGRESS);

task.setAcceptorId(acceptorId);

taskRepository.save(task);

}}

二、交易模块

1. 功能划分

虚拟钱包管理

资金冻结与解冻

赏金支付

交易流水记录

2. 技术栈

后端：Spring Boot + Spring Data JPA

数据库：MySQL

事务管理：Spring @Transactional

3. 数据库设计

sql

复制

CREATE TABLE user\_wallet (

id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT UNIQUE, -- 用户ID

balance DECIMAL(10,2) DEFAULT 0.00, -- 可用余额

frozen\_balance DECIMAL(10,2) DEFAULT 0.00, -- 冻结金额

FOREIGN KEY (user\_id) REFERENCES user(id));

CREATE TABLE transaction (

id INT PRIMARY KEY AUTO\_INCREMENT,

from\_user INT, -- 支出用户ID

to\_user INT, -- 收入用户ID

amount DECIMAL(10,2), -- 交易金额

type ENUM('TASK\_REWARD','RECHARGE','WITHDRAW'), -- 交易类型

task\_id INT, -- 关联任务ID

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP);

4. 接口设计

充值：POST /api/wallet/recharge

提现：POST /api/wallet/withdraw

交易流水：GET /api/wallet/transactions

5. 核心代码实现

java

复制

@Servicepublic class WalletService {

@Autowired

private UserWalletRepository walletRepository;

@Autowired

private TransactionRepository transactionRepository;

@Transactional

public void recharge(Long userId, BigDecimal amount) {

UserWallet wallet = walletRepository.findByUserId(userId)

.orElseThrow(() -> new ResourceNotFoundException("钱包不存在"));

wallet.setBalance(wallet.getBalance().add(amount));

walletRepository.save(wallet);

Transaction transaction = new Transaction();

transaction.setToUser(userId);

transaction.setAmount(amount);

transaction.setType(TransactionType.RECHARGE);

transactionRepository.save(transaction);

}}

三、通知模块

1. 功能划分

站内消息通知

邮件/短信通知

系统公告

2. 技术栈

后端：Spring Boot + WebSocket

邮件服务：SendGrid

短信服务：阿里云短信

3. 数据库设计

sql

复制

CREATE TABLE notification (

id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT, -- 接收用户ID

content TEXT, -- 通知内容

type ENUM('TASK','SYSTEM'), -- 通知类型

is\_read BOOLEAN DEFAULT FALSE, -- 是否已读

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP);

4. 接口设计

发送通知：POST /api/notifications

获取未读通知：GET /api/notifications/unread

5. 核心代码实现

java

复制

@Servicepublic class NotificationService {

@Autowired

private NotificationRepository notificationRepository;

public void sendNotification(Long userId, String content, NotificationType type) {

Notification notification = new Notification();

notification.setUserId(userId);

notification.setContent(content);

notification.setType(type);

notificationRepository.save(notification);

}}

四、评价模块

1. 功能划分

任务完成后双方互评

信用分计算

评价展示

2. 技术栈

后端：Spring Boot + Spring Data JPA

数据库：MySQL

3. 数据库设计

sql

复制

CREATE TABLE review (

id INT PRIMARY KEY AUTO\_INCREMENT,

task\_id INT, -- 关联任务ID

from\_user INT, -- 评价者ID

to\_user INT, -- 被评价者ID

rating INT, -- 评分（1-5）

comment TEXT, -- 评价内容

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP);

4. 接口设计

提交评价：POST /api/reviews

获取用户评价：GET /api/users/{userId}/reviews

5. 核心代码实现

java

复制

@Servicepublic class ReviewService {

@Autowired

private ReviewRepository reviewRepository;

public void submitReview(Long taskId, Long fromUserId, Long toUserId, int rating, String comment) {

Review review = new Review();

review.setTaskId(taskId);

review.setFromUser(fromUserId);

review.setToUser(toUserId);

review.setRating(rating);

review.setComment(comment);

reviewRepository.save(review);

}}

五、管理模块

1. 功能划分

用户管理（查看、封禁）

任务管理（审核、删除）

数据统计（用户数、任务数、交易流水）

2. 技术栈

后端：Spring Boot + Spring Data JPA

前端：React + Ant Design（管理后台）

3. 接口设计

用户列表：GET /api/admin/users

封禁用户：PUT /api/admin/users/{userId}/ban

任务列表：GET /api/admin/tasks

删除任务：DELETE /api/admin/tasks/{taskId}

4. 核心代码实现

java

复制

@Servicepublic class AdminService {

@Autowired

private UserRepository userRepository;

public void banUser(Long userId) {

User user = userRepository.findById(userId)

.orElseThrow(() -> new ResourceNotFoundException("用户不存在"));

user.setBanned(true);

userRepository.save(user);

}}

通过以上设计，你可以实现一个功能全面、可扩展的校园学生互助系统。如果需要进一步优化或扩展功能，可以随时讨论！