w

廣東工業大學

QG 工作室项目报告

学　　院 计算机学院

专　　业 软件工程

班　　级 2018 级　2　班

组　　别 网络组

姓　　名 黄钰朝

学　　号 3118005005

2019 年 5 月

广东工业大学计算机学院制

目录

一．项目简介

1.1 设计要求

设计一款即时通讯软件，实现用户登陆注册，账户管理，在线聊天，好友管理，聊天记录管理，聊天群组，朋友圈等功能。

1.2 项目需求分析

作为互联网新生一代，我能够深刻体会到即时通讯软件带来的便利性，尤其是 QQ,微信两款社交产品，已经成为人们日常生活中重要的通讯软件，因此对于本项目的开发要求，我主要参考 QQ 和微信量款产品展开分析，根据项目的设计要求，从设计主题：“QG 版微信”，并结合实际的生活场景，可以分析得出本项目作为一个即时通讯网络程序，是一款社交产品，主要面向互联网用户，满足人们实时信息交互和社交活动的需求。

首先，作为一款面向个人用户的社交产品，产品的具体使用场景可以参考微信分析得出，使用过程需要涉及较多的用户个人信息，因此应该具备一定的隐私性，同时本项目作为一款面向人们日常生活需求的产品，日常的使用频次较高而且服务应该具备连续性，即系统应该有能力记住用户的使用记录，提供长期的服务，因此本项目系统对外具有一定封闭性，用户的个人信息和使用记录都不像普通的论坛类的产品一样对外开放，同时对内为个人用户保留一定的空间，主要面向熟人社交而不是社区式社交，因此要求具有用户的注册登陆功能，而对于游客用户，则不提供添加好友和朋友圈等功能。

其次，本软件作为实时通讯产品，具体的使用场景是用户之间进行实时聊天，提高人们信息交流的效率，互联网社交的关键在于互动，如果信息交互不及时，将会严重影响用户体验，也难以满足本项目的设计初衷，因此本系统需要提供基本的聊天功能，并且需要实现消息传送的即时性，要求系统具备即时处理用户的信息交互请求的能力，同时其要满足不同层次的社交需求，因此应该提供群聊的功能，还需要提供朋友圈的相关功能。

最后，本软件作为人们日常使用的社交工具，应该提供一些个性化的设置功能，满足人们在互联玩和人际关系中表达个性的需求，也就是用户更换头像，自定义昵称，发布个性签名，更换聊天背景等功能。由以上分析，可以得出本软件的功能需求具备隐私性，即时性，高效性，个性化等特点。除此之外，参考微信的成功和其设计理念，一款社交产品应该在设计上具有简洁性，因此在前端界面上，应该做到简洁清晰，符合人性，容易使用。

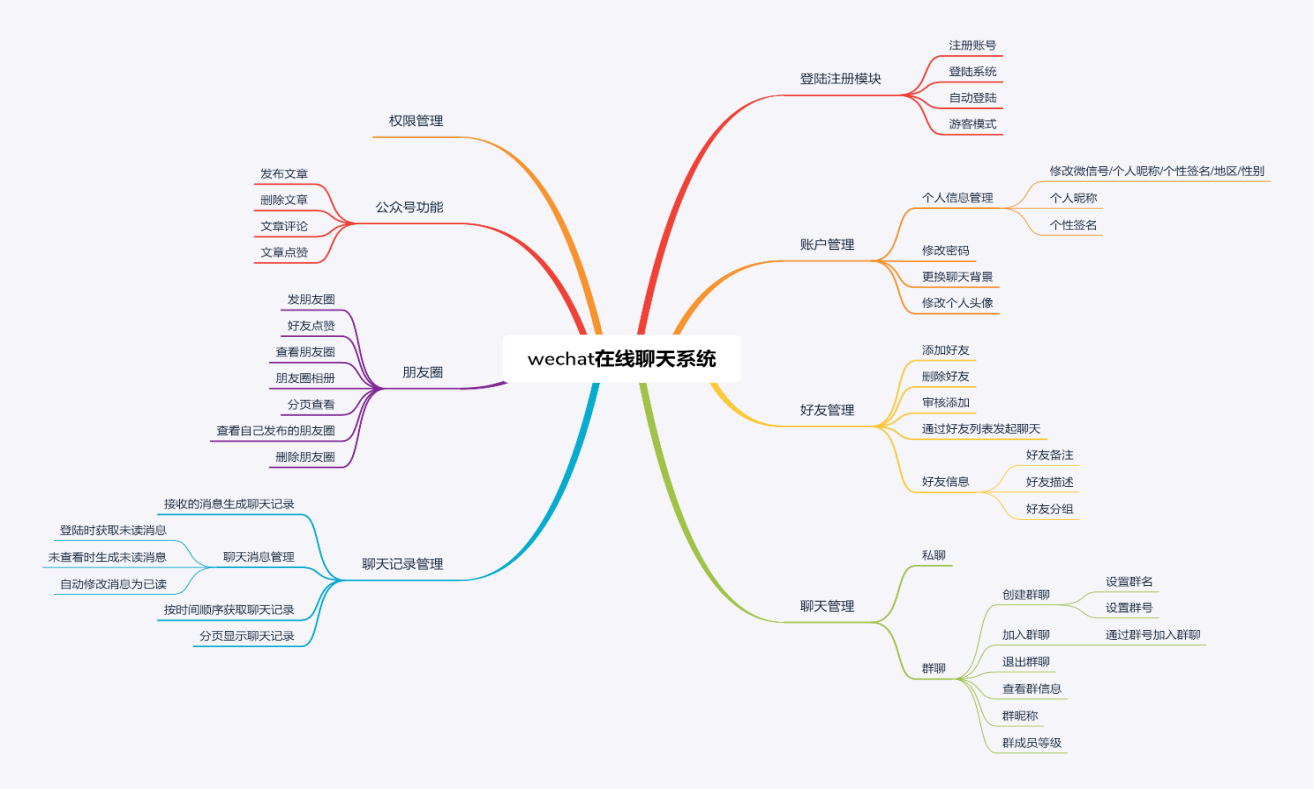
综上所述，本软件的开发应该围绕用户社交需求展开，提供高效的即时通讯功能，构建简洁清晰的界面和友好的人机交互体验。

1.3 功能模块划分

在具体的功能需求上分析的基础上，我将本软件的功能需求分为几大模块，如下图所示:

功能模块划分（大纲）

具体的功能模块



1. 登陆注册模块

1.1. 注册账号
1.2. 登陆系统
1.3. 自动登陆
1.4. 游客模式

2. 账户管理

2.1. 个人信息管理
2.1.1. 修改微信号/个人昵称/个性签名/地区/性别
2.1.2. 个人昵称
2.1.3. 个性签名
2.2. 修改密码
2.3. 更换聊天背景
2.4. 修改个人头像

3. 好友管理

3.1. 添加好友
3.2. 删除好友
3.3. 审核添加
3.4. 通过好友列表发起聊天
3.5. 好友信息
3.5.1. 好友备注
3.5.2. 好友描述
3.5.3. 好友分组

4. 聊天管理

4.1. 私聊
4.2. 群聊
4.2.1. 创建群聊
- 设置群名
- 设置群号

二．技术原理及设计方案

2.1 技术需求分析

在技术原理方面，在项目需求分析的基础上，可以得出本软件的功能需求具备隐私性，即时性，高效性，个性化等特点。因此在实现基础上，要求在数据传输时具有高效性，如果仅仅使用 http 协议进行通信，因为 http 是无状态协议，服务端只能被动地由客户端发起请求，并不能主动地向客户端发起通信，并且 http 属于 TCP 协议，每次连接都要进行三次握手，信息交互效率受到很大的制约，因此无法满足即时通信的需求，同时我了解到 http 协议可以使用 ajax 轮询或者 long poll 实现类似保持连接的效果，但是从其实现原理来看，显然这两种实现方式对于服务器的性能消耗都是非常巨大的，并不是很好的选择。而 WebSocket 协议就能够很好地满足技术需求，因为 WebSocket 只通过一次 http 握手就可以保持长久连接，并且可以由服务端主动向客户端推送消息，而不必由客户端主动请求，因此可以减轻服务器的负担，所以我最终决定选在 WebSocket 作为即时通信技术需求的实现方案。

在前后端数据传输方面，我主要考虑了作为一款通讯软件，必须具备使用上的高效性，因此在之前我所知道的通过表单提交数据的方式，已经无法满足本软件的技术需求，因此，根据在训练营中了解到的 ajax 技术，和 json 数据传输格式，认为这两者很好地满足本软件的技术需求，最终选择 ajax 技术为前后端交互形式，并且使用预加载等实现方式，以尽可能地在网页上模拟本地应用程序地使用体验。

在前端页面的设计方面，本着简洁清晰地设计理念，在界面上使用较为统一的设计风格，设计上也很大程度上参考了网页版微信的设计风格，在使用逻辑上力求简洁清晰。

2.2 确定技术选型

前端：

编程语言：HTML , CSS, Javascript，
通信协议：http, WebSocket
实现技术：jsp 动态网页，ajax 技术
数据交互格式：json

后台：

编程语言：Java

通信协议：http, WebSocket

IDE：IntelliJ IDEA 2019.1.1 (Ultimate Edition)

JRE: 1.8.0_202-release-1483-b44 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

Windows 10 10.0

数据交互格式：json

数据存储方式：mysql数据库 5.7

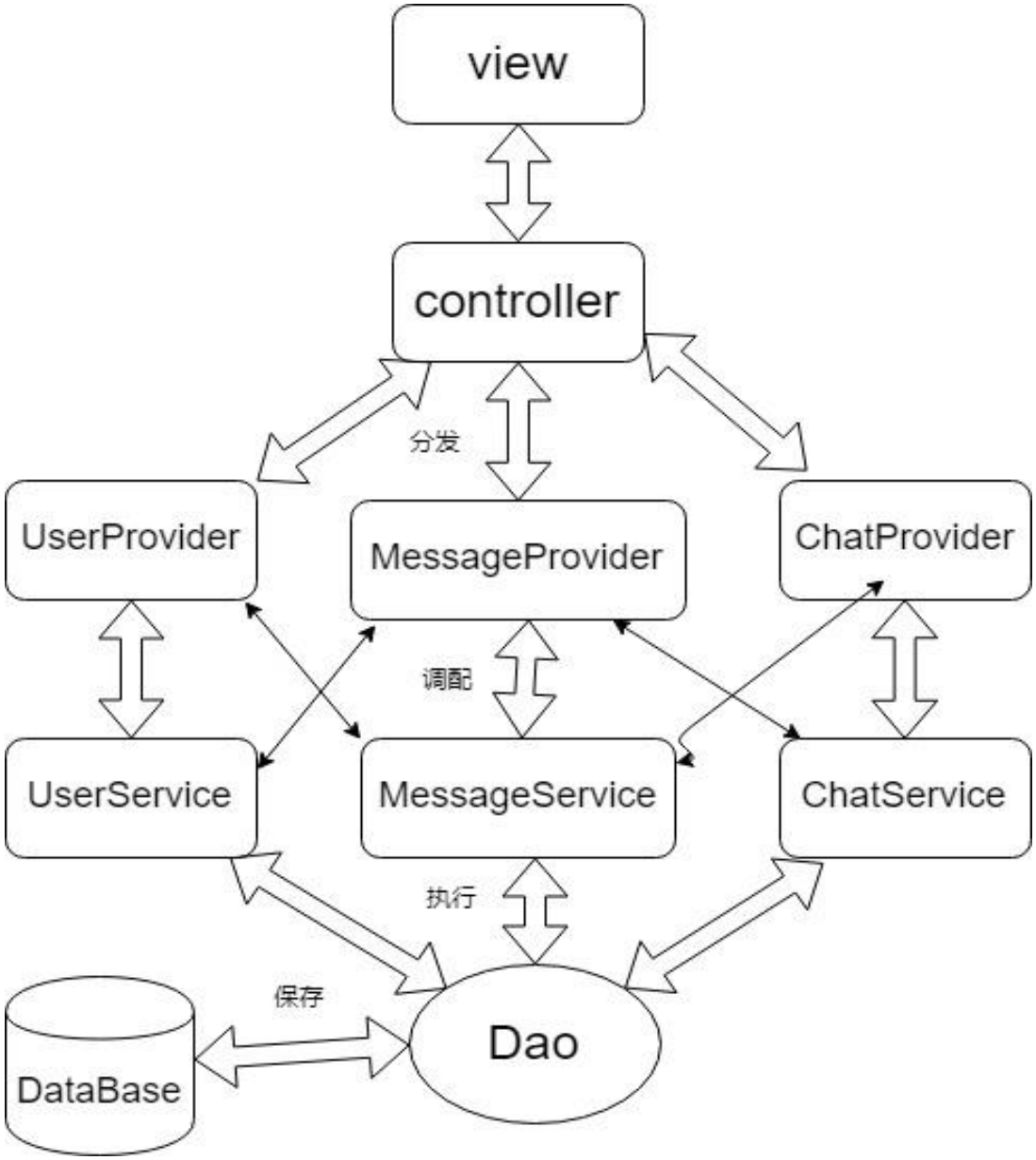
Web容器：Tomcat 9.0

版本管理：git

2.3 总体设计方案

在总体架构设计上，借鉴经典的 MVC 设计模式中，view-controller-model 模型，稍作改进， 设计了 view-controller-provider-model 的结构，将传统的 controller 分为两个部分；分发和调配，整个项目只有一个 servlet 作为集中分发者，分发者并不参与具体任务的执行过程，只负责根据前端的请求,将任务分发到负责相关业务流程的 provider 中，一个 provider 中包含相关的 action 方法,一个 action 方法代表一个请求所需执行一整套业务流程，每个 action 之间相对独立，不需要互相调用，就可以在方法内部完成一整套的业务流程，一个 action 完成后就可以给前端返回所需的结果，避免了在不同的方法或者类中反复跳转，从而简化编程逻辑，同时提高方法的内聚性。除此之外，在一个 action 方法中，action 层作为调配调用者，只负责业务的流程控制，具体的实现由下层的 model 来实现，model 层每个模块只需完成单一的功能而不关系业务流程的跳转，实现 model 层模块的复用。model 层分为逻辑实现层（service 层），数据持久化层（dao 层）。Service 层提供具体业务逻辑实现，dao 层负责对接数据库，与数据库表一一对应.

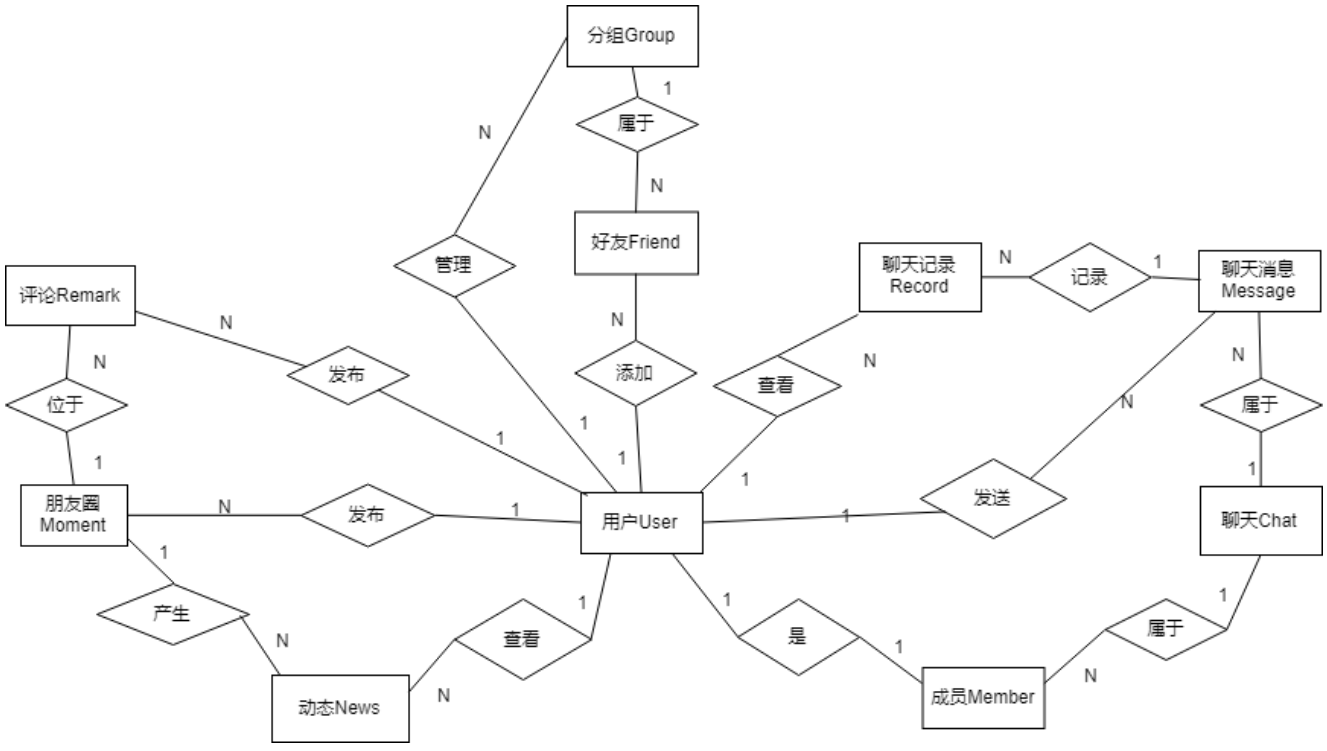
项目整体结构示意


view
controller
分发
UserProvider
MessageProvider
ChatProvider
调配
UserService
MessageService
ChatService
执行
保存
Dao
DataBase

2.5 数据模型设计

数据模型整体思路是，用户与用户之间通过好友表形成多对多好友关系，当一方加另一方为好友时插入一条好友记录，当双方都插入好友记录时两者达成好友关系，好友通过分组表记录分组关系，还能实现存储分组的优先级等信息，通过聊天表形成聊天关系，聊天可以是私聊，可以是群聊，用户通过成员表与聊天形成多对多的关系，一条聊天消息依赖于发送的用户 id 和聊天 id,一条消息只有发出者 id，没有接收者 id,聊天中的其他成员通过记录表与聊天消息形成多对多的关系，这样当一个用户发出消息时，他一定在一个聊天中，此时聊天中的所有成员都能收到这条消息，同时生成一份自己的记录，当一个用户删除记录时，不影响消息的状态。朋友圈也和消息同理，都是消息订阅模式，朋友通过动态表的记录获得朋友圈信息，当朋友对朋友圈点赞时，只影响自己的点赞状态。评论依赖于朋友圈 id，而回复依赖于评论 id,回复表记录回复之间的关系。

整体 ER 图结构



具体数据库表设计

User 用户表

用户表中 id 为自增主键，外部注册登陆使用的唯一标识是邮箱号，微信号同样要求全局唯一，但是允许用户在合理的范围内自定义微信号，将来业务扩展可用来可作为登陆用户名，其他字段为一些用户的基本信息。聊天背景将来如果扩展，这个字段可以降级为默认聊天背景，在聊天表上实现每个聊天窗口自己的聊天背景。

列名	说明	数据类型
id	主键	bigint
email	邮箱号	varchar
wechat_id	微信号	varchar
phone	手机号	varchar
password	账号密码	varchar
gender	性别（未设置/男/女）	varchar
signature	个性签名	varchar
name	用户昵称	varchar
photo	用户头像	varchar
chat_background	聊天背景	varchar
location	用户所在地区	varchar
online_status	在线状态（离线/在线/隐身/忙碌）	varchar
status	账户状态	smallint
gmt_create	插入时间	datetime
gmt_modified	最后修改时间	datetime

Friend 好友表

好友表中用来存储一个用户对另一个用户发起的好友关系，记录发起用户的 id，好友用户的 id，并且记录他们之间建立的聊天 id,和用户对好友的备注，用户对好友的描述，好友所处的分组等

column_name	column_comment	data_type
id	主键	bigint
user_id	用户id	bigint
friend_id	好友id	bigint
chat_id	聊天id	bigint
group_id	所处分组id	bigint
alias	好友备注	varchar
description	好友描述	varchar
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Grounp 好友分组表

好友分组表用来记录用户对他的好友的分组，好友和分组之间是一对多的关系，用户可以将好友分入不同的分组，并且可以对每个分组设置优先级，用以决定该分组位于好友列表的位置等信息。

列名	说明	数据类型
id		bigint
user_id	用户id	bigint
name	分组名称	varchar
priority	优先级	smallint
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Chat 聊天表

聊天表用来记录用户之间聊天关系，用户通过成员表而成为一个聊天中的成员，一个聊天中只能有一个群主，表中还记录了聊天的头像，聊天的成员数，聊天的名称。

列名	说明	数据类型
id	主键	bigint
number	聊天唯一编号	varchar
owner_id	群主id	bigint
name	聊天名称	varchar
photo	聊天头像	varchar
member	成员人数	smallint
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Member 成员表

成员表用来记录一个用户属于某个聊天，即一个用户是一个聊天的成员，同时还记录了这个用户在这个聊天中作为成员而具有的属性，包括群聊昵称，群等级，设置的聊天背景等信息。

列名	说明	数据类型
id	主键	bigint
user_id	用户id	bigint
chat_id	角色id	bigint
group_alias	群聊昵称	varchar
level	群等级	smallint
background	聊天背景	varchar
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Message 消息表

消息表用来记录一条聊天消息的内容，消息的发送者，消息所处的聊天，消息的类型，消息的发送时间等信息。并不记录消息的接收者，接收者通过记录表来找到这条消息。

列名	说明	数据类型
id	主键	bigint
sender_id	发送者id	bigint
chat_id	聊天id	bigint
content	消息内容	varchar
type	消息类型	varchar
time	发送时间	timestamp
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Record 记录表

记录表用来记录一个用户是一条消息的接收者这样一个信息，记录表不保存消息的具体信息，只记录用户和消息之间的关系，和这条消息是否被这个用户阅读过的信息。

列名	说明	数据类型
id	主键	bigint
user_id	用户id	bigint
message_id	消息id	bigint
status	状态（0未读，1已读）	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Moment 朋友圈表

朋友圈表用于记录用户发布的朋友圈，记录朋友圈的具体内容，朋友圈的图片，朋友圈的发布时间，朋友圈的点赞数，评论数，收藏数，浏览量，转发量等信息。

列名	说明	数据类型
id	主键	bigint
owner_id	发布者id	bigint
content	内容	text
photo		varchar
time	发布时间	timestamp
love	点赞数	int
remark	评论数	int
share	转发数	int
view	浏览量	int
collect	收藏量	int
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

News 朋友圈动态表

朋友圈动态表用来记录一条朋友圈对于一个用户可见的信息，一个用户发布一条朋友圈时，将给他的所有好友每个人产生一条朋友圈动态，用来标识该好友可以看到这条朋友圈。同时记录了这个好友对于这条朋友圈的操作，包括点赞，评论，转发，收藏等操作。

列名	说明	数据类型
id	主键	bigint
user_id	发布者id	bigint
moment_id	朋友圈id	bigint
loved	是否点赞	tinyint
shared	是否转发	tinyint
viewed	是否浏览	tinyint
collected	是否收藏	tinyint
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

Remark 评论表

评论表用来记录一个用户对于一条朋友圈的评论，其中记录了评论的内容，评论的时间，以及这条评论的点赞数，收藏量，回复数等信息。

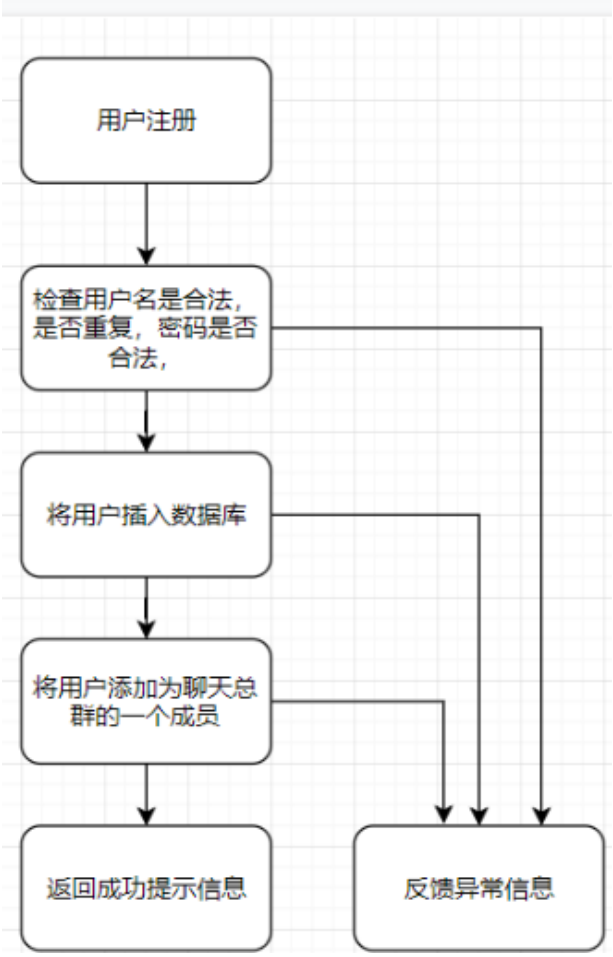
列名	说明	数据类型
id	主键	bigint
user_id	评论者id	bigint
moment_id	评论对象id	bigint
content	内容	varchar
like	点赞数	int
collect	收藏量	int
reply	回复数	bigint
status	状态	smallint
gmt_create	创建时间	datetime
gmt_modified	修改时间	datetime

三．功能模块详细设计

3.1 注册登陆模块

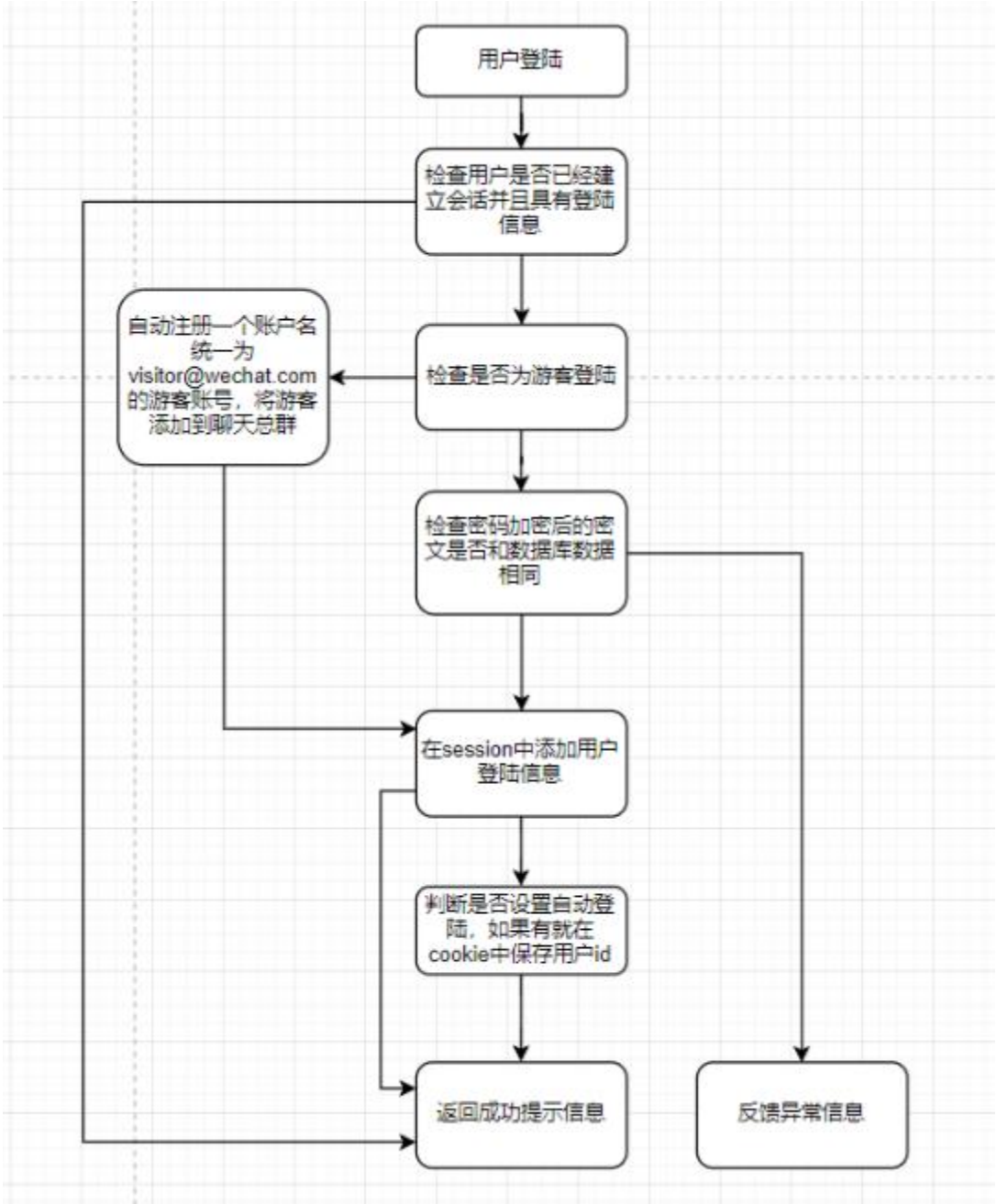
1.用户注册

用户注册模块提供获得系统中用户身份的功能，要求用户使用邮箱进行注册，并且一个邮箱不可以重复注册，邮箱作为用户的唯一标识，用于作为用户的登陆凭证，在注册的功能流程中需要检查邮箱（用户名）是否合法，并查询数据库检查该邮箱是否重复，以及密码是否合法，如果以上检查出现问题，要求用户重新输入。检查通过之后则可以插入到数据库中，并把用户添加为聊天总群的一个成员，然后返回成功信息，转到登陆界面，让用户使用刚刚注册用户名进行登陆。



2.登陆系统

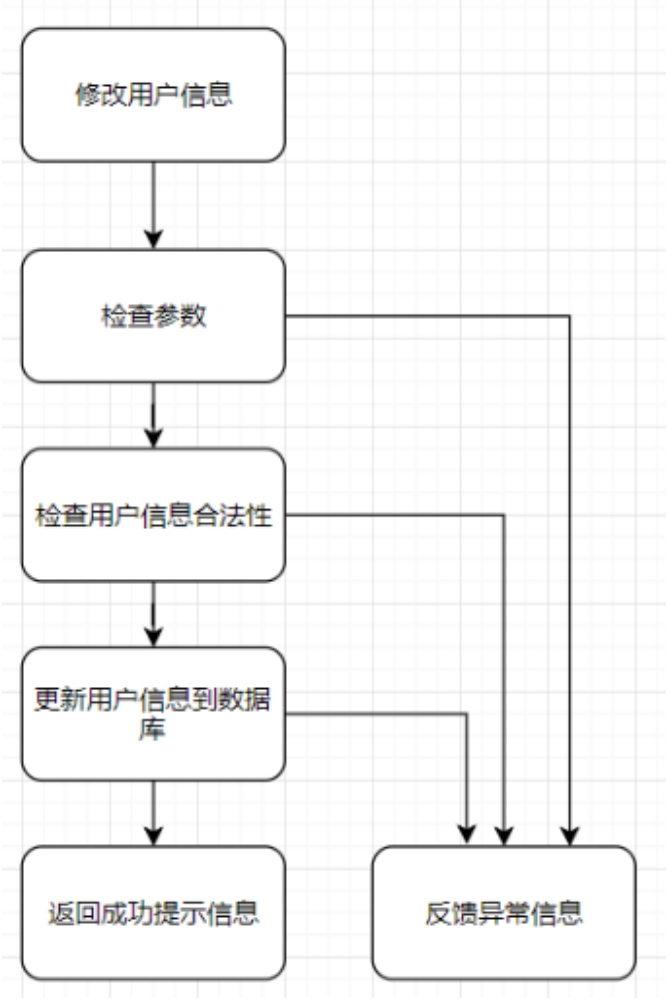
登陆系统功能提供用户登陆系统的功能，要求验证用户输入的密码和数据库中的对应字段匹配。当执行登陆功能时先检查该用户是否有 session 建立了会话，再检查该 session 中是否含含有登陆的信息，如果有登陆信息，则表示该用户在本次会话中已经登陆，则直接跳过验证程序，通过验证，如果没有 session,则先检查该请求是否是游客登陆的请求，如果是则自动注册一个游客账号，并自动登陆成功，然后向 session 中添加登陆信息。如果是普通用户登陆，则验证该用户的密码。错误时跳转到登陆页面。



3.2 账户管理模块

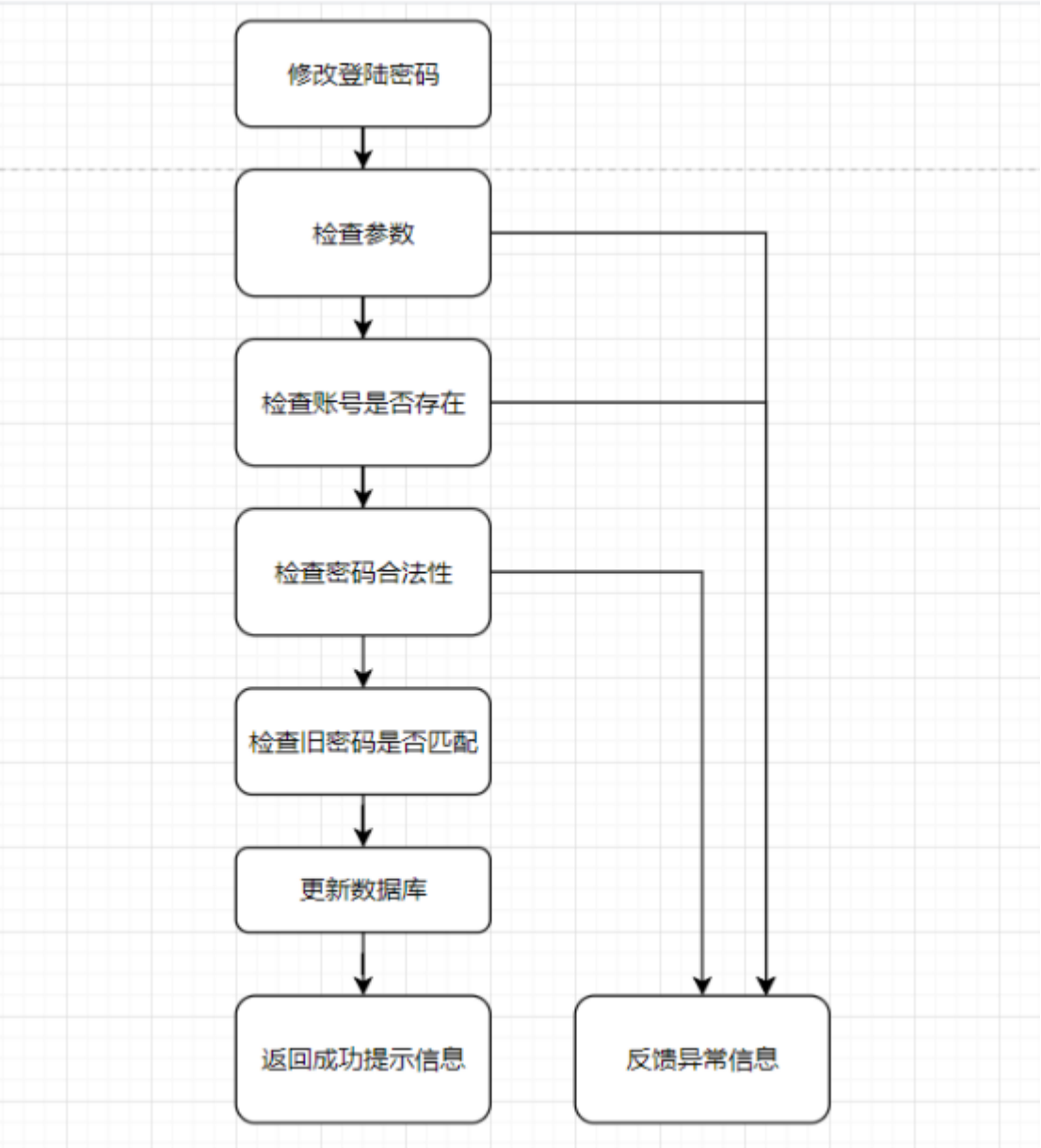
1.更新个人信息

这个功能负责更新用户的个人信息需要检查用户的微信号是否合法，是否重复，通过后可以更新到数据库。



2．更新登陆密码

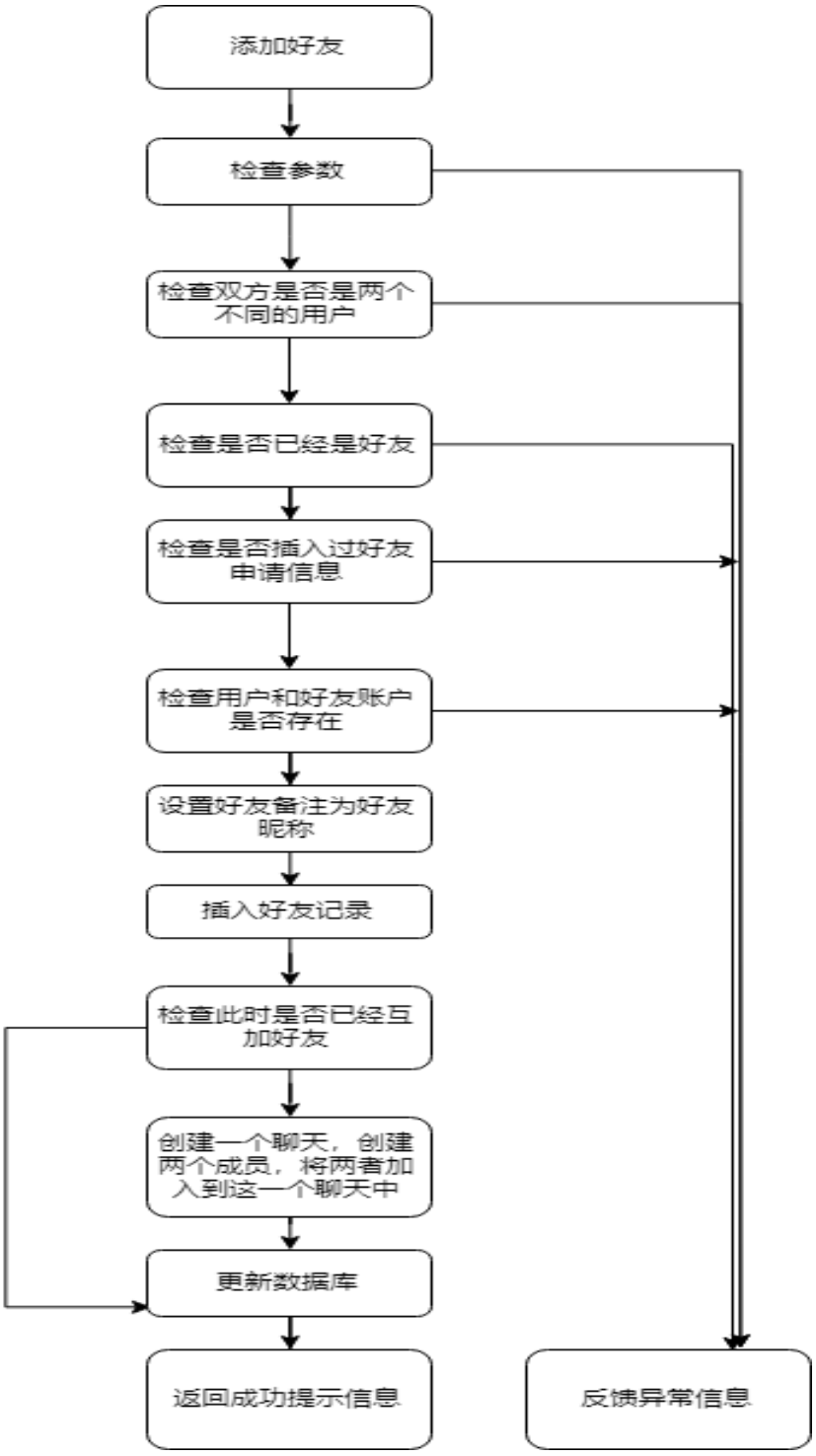
提供用户修改登陆密码的功能。需要验证用户输入的旧密码，并检查新密码是否合法。



3.3 好友管理模块

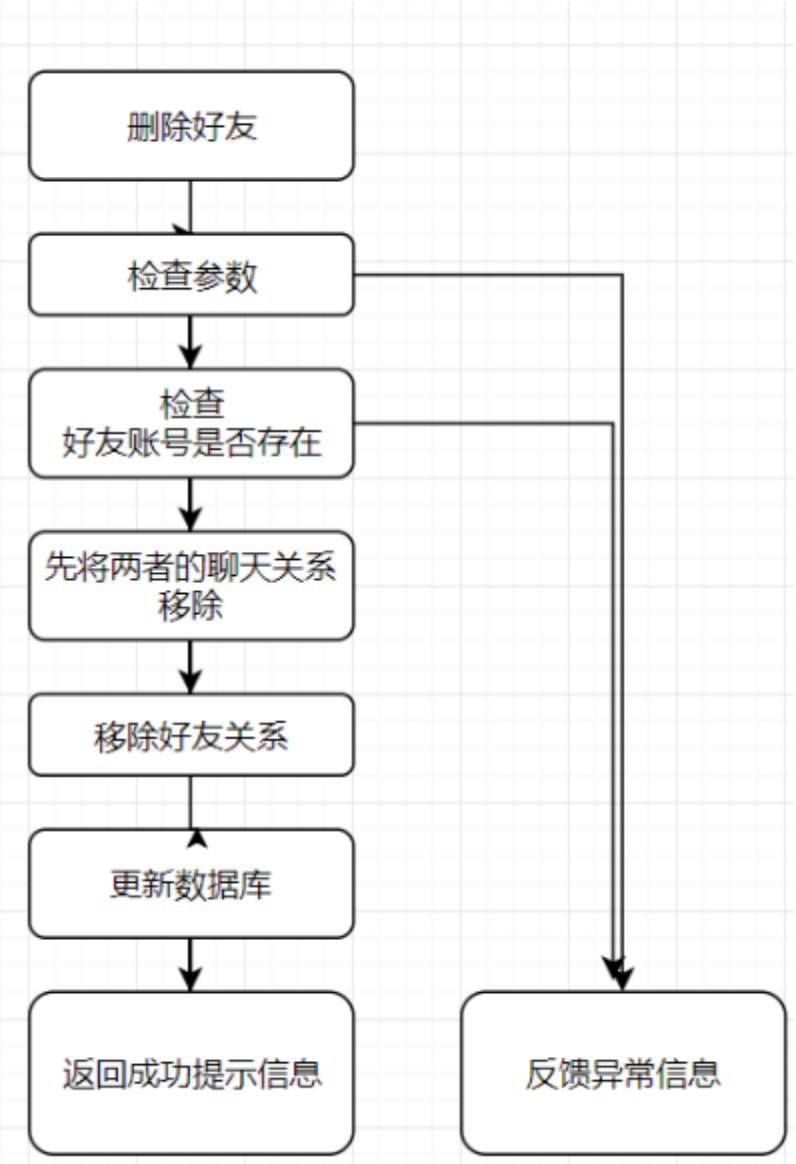
1.添加好友

提供用户添加好友的功能，先检查是否是添加自己为好友，这种情况则返回不允许添加自己为好友。然后检查是否重复添加，在添加完好友后还需判断是否已经是好友了，如果是，则建立聊天关系



2.删除好友

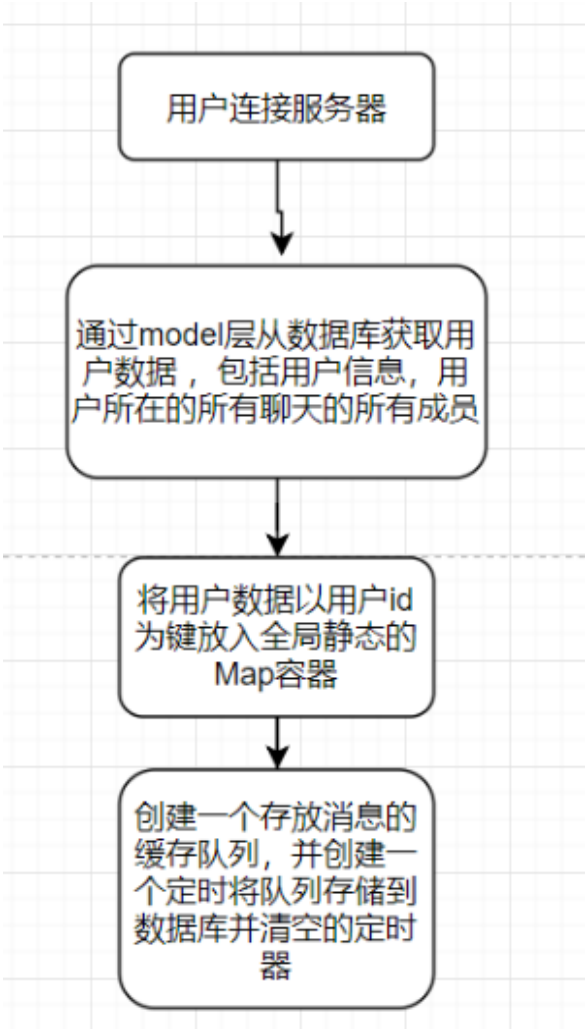
提供删除好友的功能，首先减产好友的账号是否存在，如果存在则移除两者之间的聊天关系，然后移除两者之间的好友关系。



3.4 实时聊天功能模块

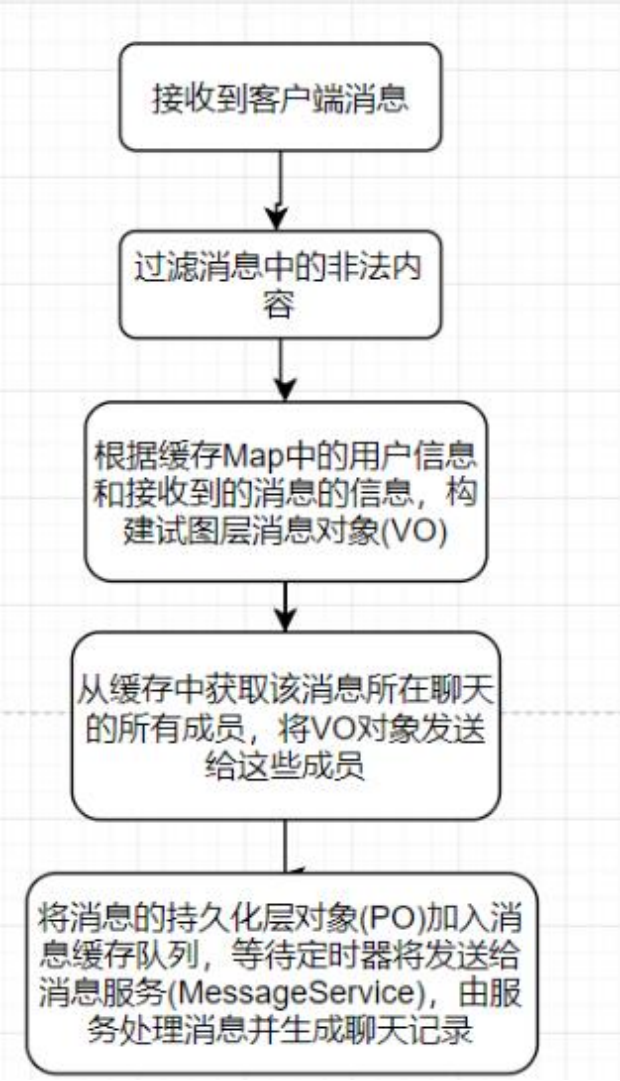
1.用户连接服务器

当用户连接服务器时，从数据库中加载用户的个人数据，放在缓存中，然后启动一个消息缓存队列，等待聊天消息存入队列，并且定时地将消息存入数据库，清空队列



2.接收到客户端消息

当接收到用户发来地消息时，先查看类型，根据类型使用不同的过滤级别，将消息内容进行过滤之后，插入消息缓存队列等待被存入数据库，然后将该消息和缓存中的用户信息一起，转化成消息的视图层对象，再从用户缓存中获取该消息所在聊天的所有成员，向每一位成员尝试发送消息。

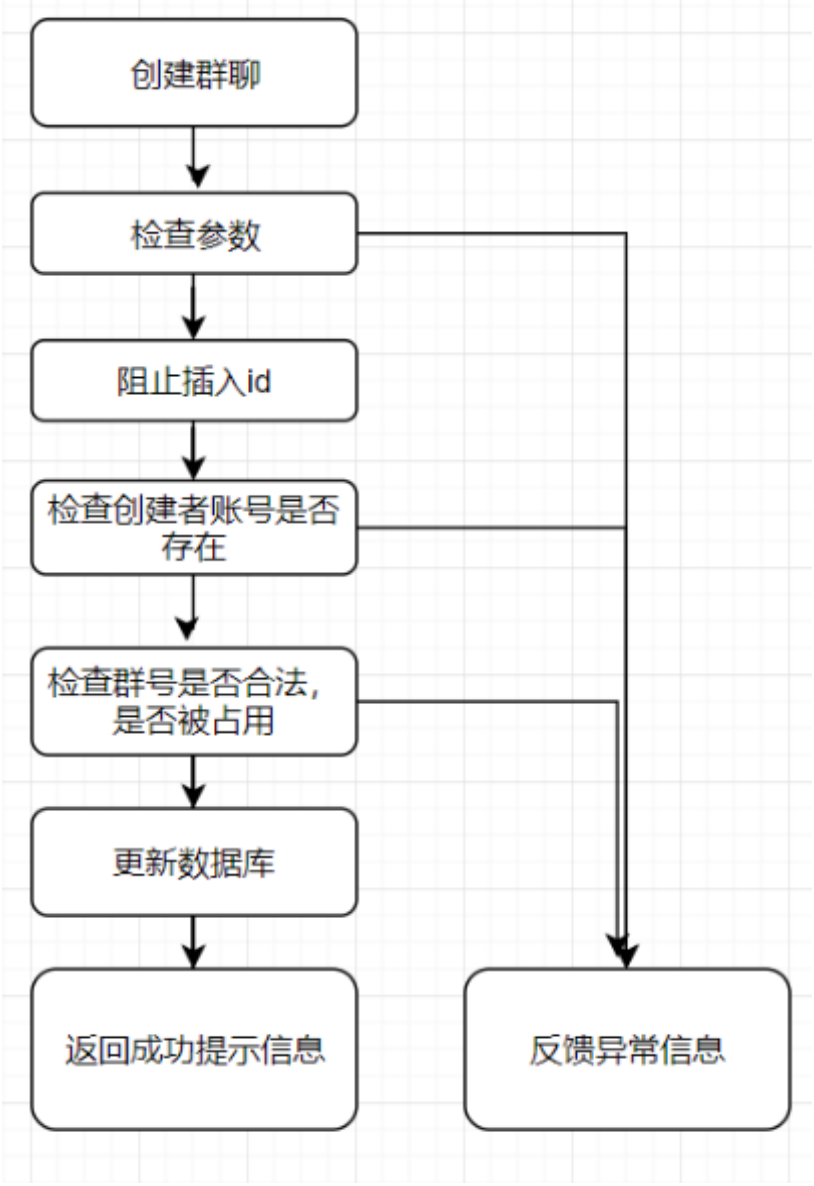


3.5 聊天管理模块

1.创建群聊

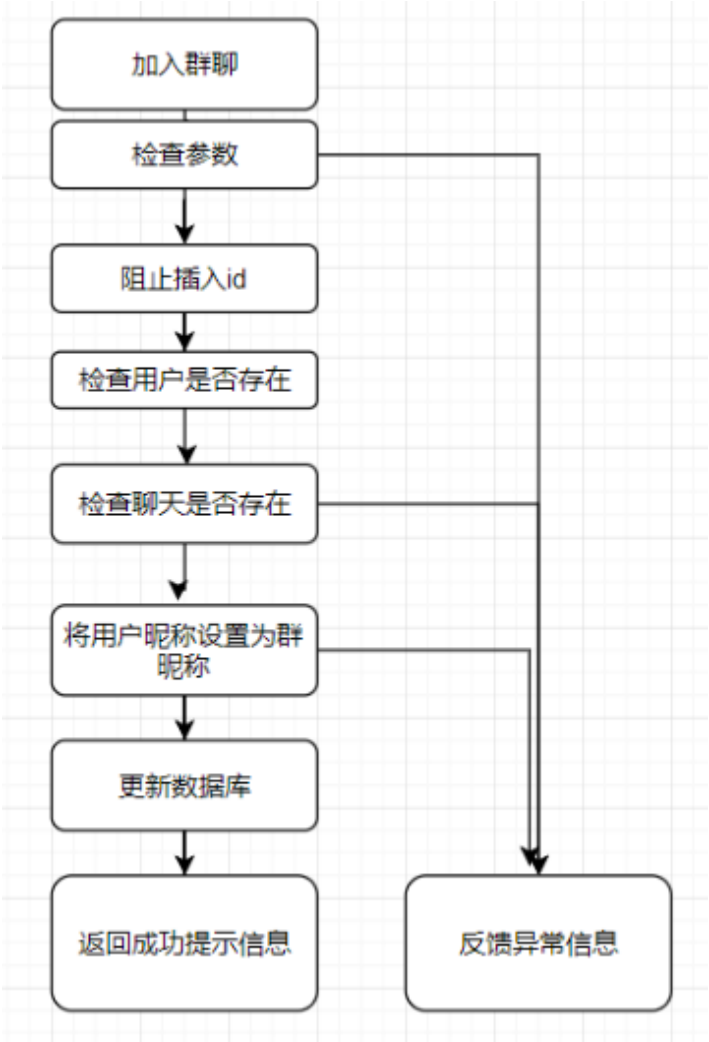
提供创建一个群聊的功能，需要检查用户输入的唯一标识“群号”是否重复，是否合法等，通过后可以更

新到数据库中



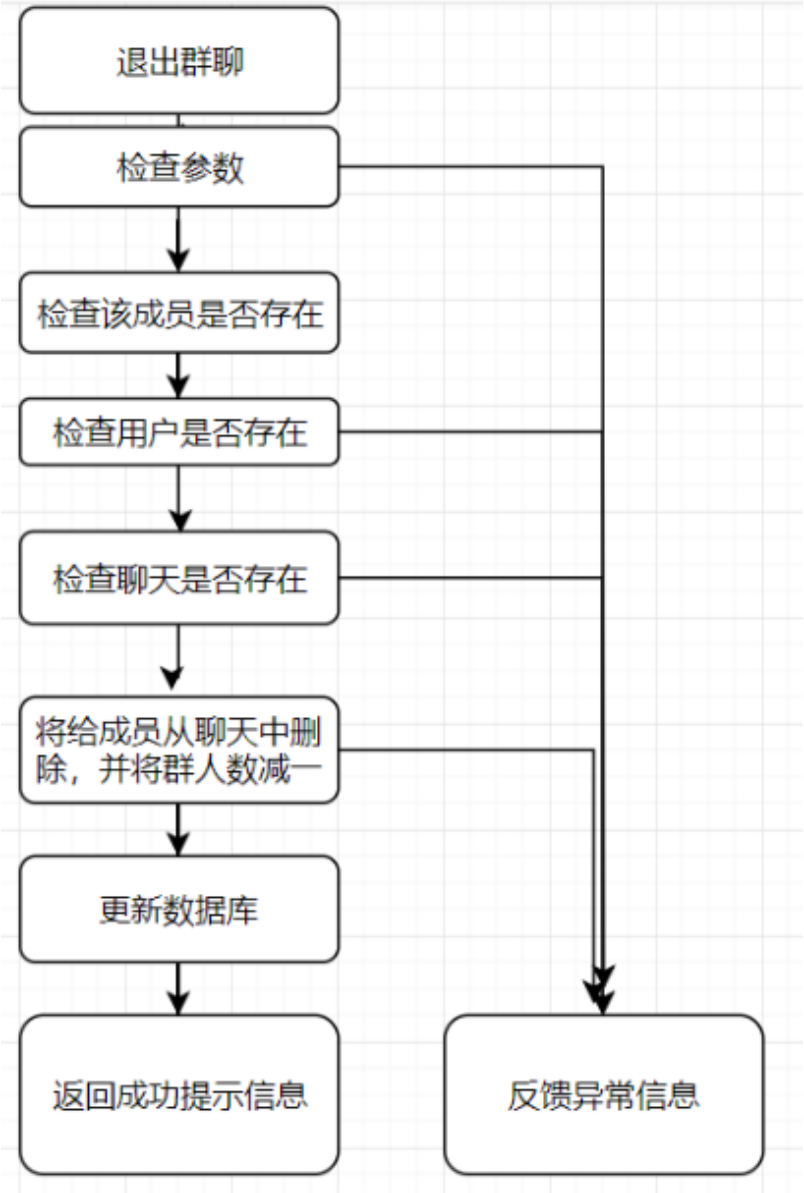
2.加入群聊

提供加入群聊的功能，根据用户输入的群号，查找该群的 id，和该用户的信息一起，生成一个成员记录，并插入到数据库，需要检查用户和群聊都存在才可以执行


加入群聊
检查参数
阻止插入id
检查用户是否存在
检查聊天是否存在
将用户昵称设置为群昵称
更新数据库
返回成功提示信息
反馈异常信息

3.退出群聊

提供一个用户退出群聊的功能，检查该用户和群聊都存在的情况下可以执行，并将群人数减一



四．程序测试

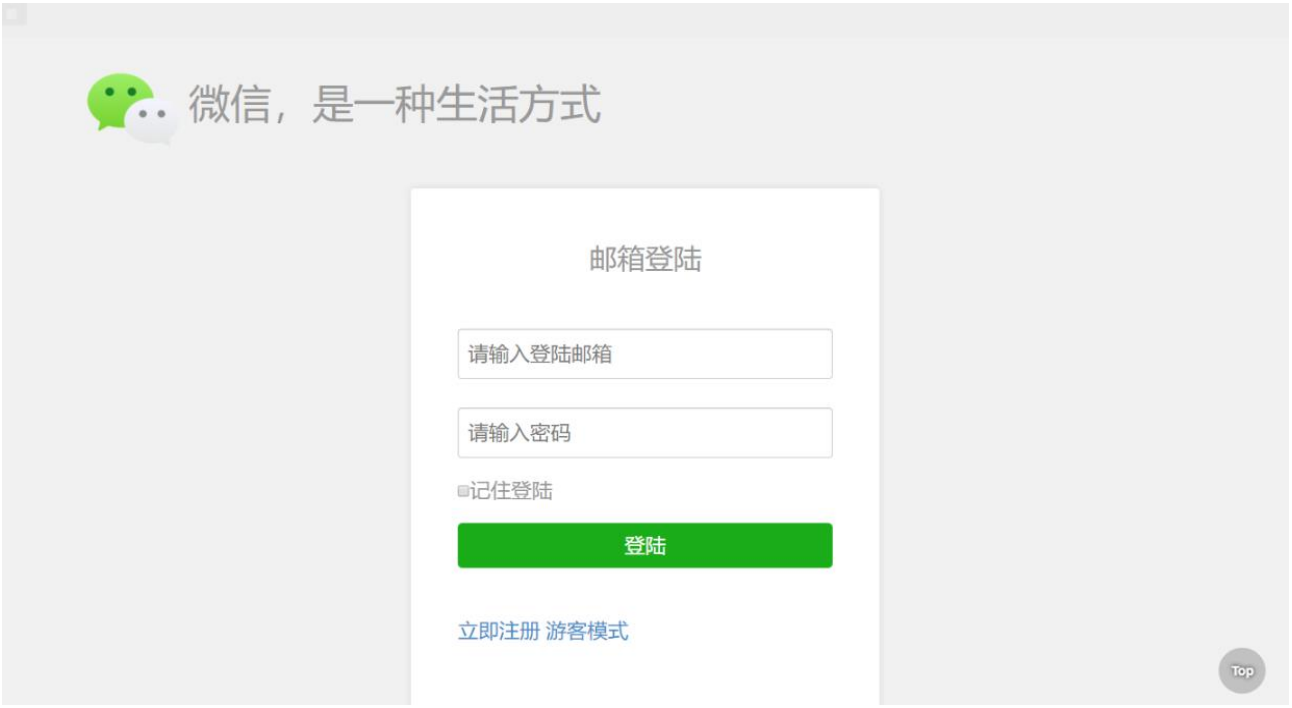
4.1 测试列表

测试账号	邮箱	密码
1	3205579341@qq.com	test1234
2	test5@qq.com	test1234
3	test2@qq.com	Test1234
n	游客模式	无密码

4.2 测试截图

1. 登陆页面

登陆页面可以选择输入邮箱号和密码进行登陆，登陆之后将进入程序的主页面，也可以选择跳转到注册页面进行账号注册，还可以选择游客模式快速登陆，这种方式无须用户操作，系统将自动注册一个游客账号，并自动通过登陆验证，同样会进入程序的主页面，但是进入的是功能受限模式，无法使用好友和朋友圈的功能。输入框的左下角还有一个自动登陆选项，如果用户选择该选项，则在 30 天内在此浏览器上会执行自动登陆功能。

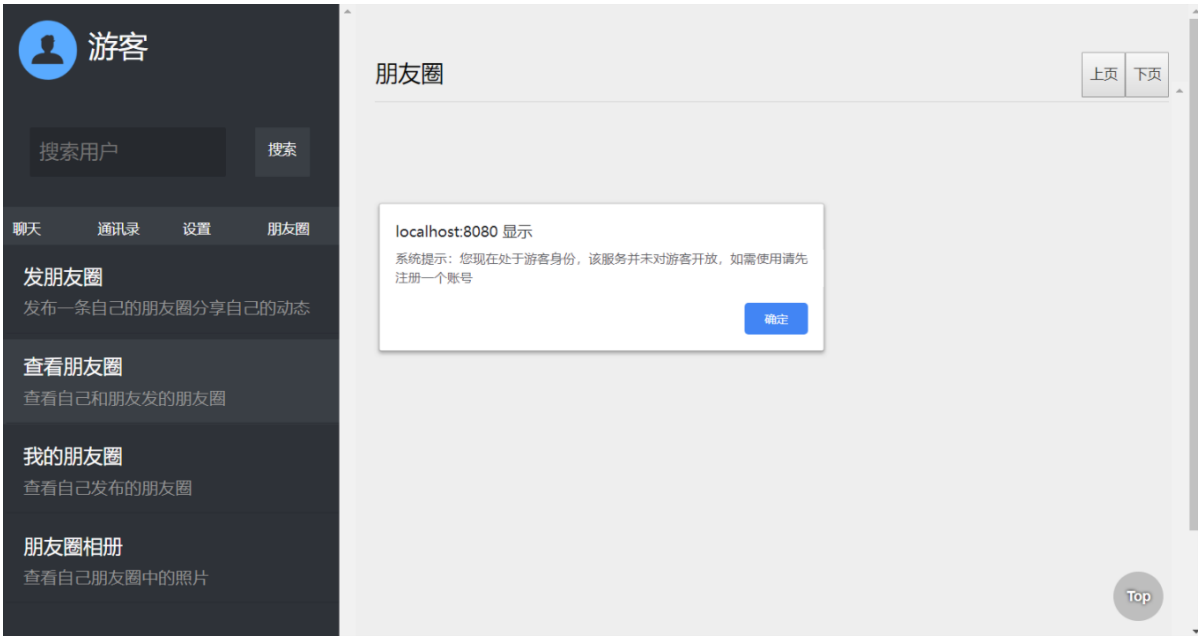


2.游客模式

这是游客模式，游客模式自动分配的用户昵称都是“游客”，并且被自动加入到聊天总群中，自动和“微信团队”账号建立好友关系，

3.游客模式

游客的好友功能和朋友圈功能被限制，当他尝试使用该功能时，会弹出如图所示的提示



4.用户模式

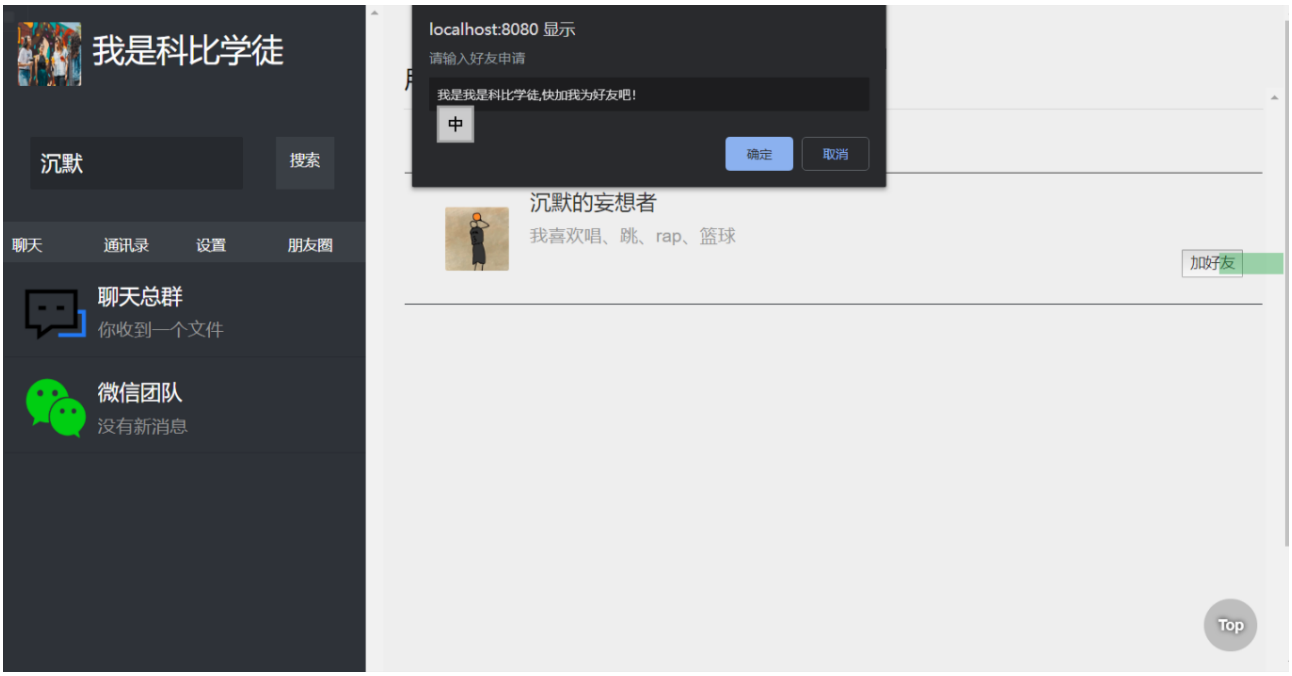
使用邮箱注册之后的初始个人昵称是邮箱账号

5.修改个人信息

6.在聊天中发送不同类型的信息

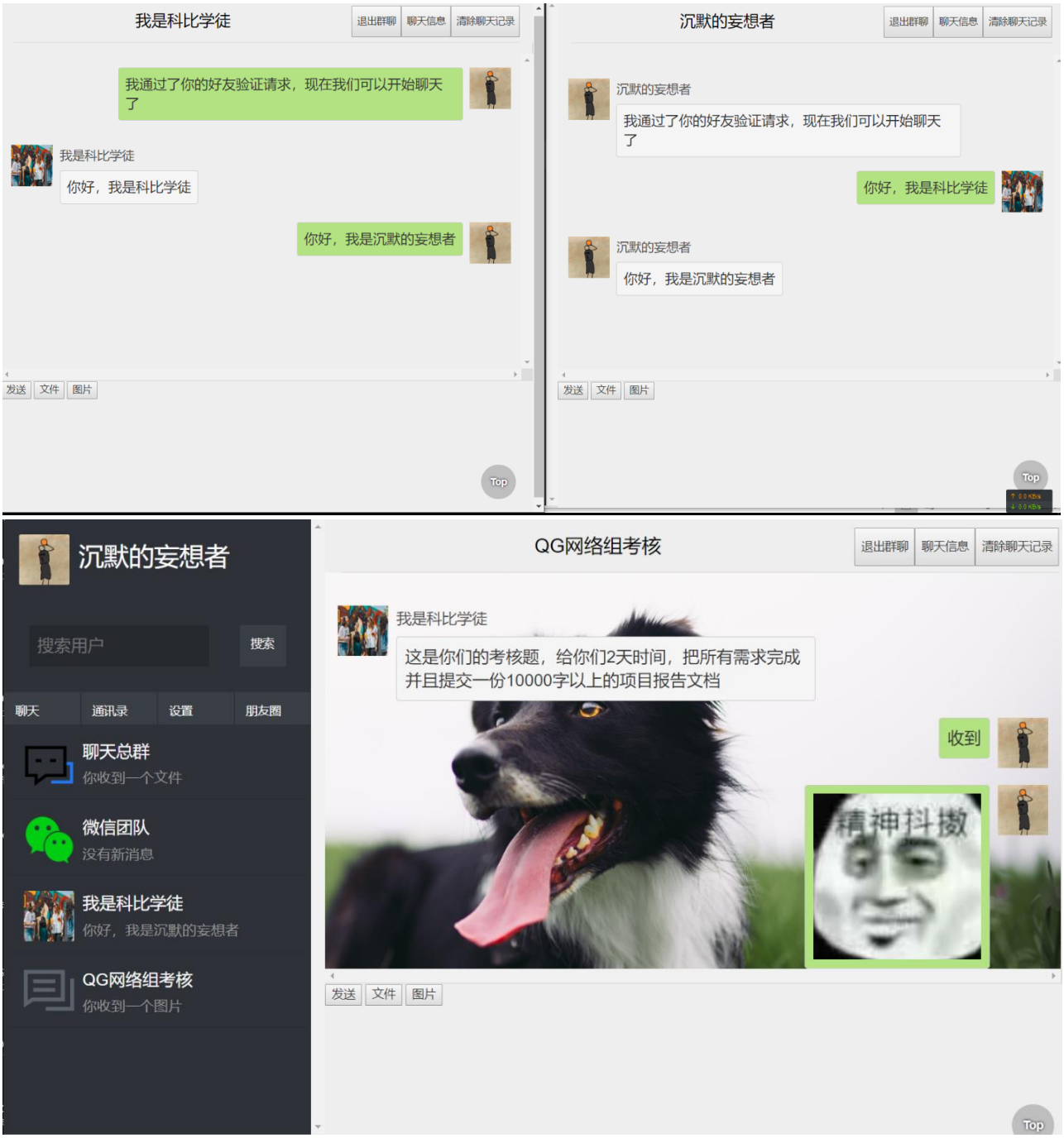
7.朋友圈

8.发送好友申请


我是科比学徒
沉默
搜索
聊天
通讯录
设置
朋友圈
聊天总群
你收到一个文件
微信团队
没有新消息
localhost:8080 显示
请输入好友申请
我是我是科比学徒,快加我为好友吧!
中
确定
取消
沉默的妄想者
我喜欢唱、跳、rap、篮球
加好友
Top


搜索用户
搜索
聊天
通讯录
设置
朋友圈
添加好友
搜索并添加系统中的用户为好友
创建群聊
创建一个群聊，邀请好友加入
加入群聊
通过群号加入一个群聊
微信团队
发消息
删除好友
我是科比学徒
好友申请：我是我是科比学徒,快加
同意
微信团队
退出群聊
聊天信息
清除聊天记录
我是科比学徒
好友申请：我是我是科比学徒,快加我为好友吧!
发送
文件
图片

9.实时聊天测试


我是科比学徒
退出群聊
聊天信息
清除聊天记录
我通过了你的好友验证请求，现在我们可以开始聊天了
我是科比学徒
你好，我是科比学徒
你好，我是沉默的妄想者
发送
文件
图片
Top
沉默的妄想者
退出群聊
聊天信息
清除聊天记录
沉默的妄想者
我通过了你的好友验证请求，现在我们可以开始聊天了
你好，我是科比学徒
沉默的妄想者
你好，我是沉默的妄想者
发送
文件
图片
Top

沉默的妄想者
搜索用户
搜索
聊天
通讯录
设置
朋友圈
聊天总群
你收到一个文件
微信团队
没有新消息
我是科比学徒
你好，我是沉默的妄想者
QG网络组考核
你收到一个图片
QG网络组考核
退出群聊
聊天信息
清除聊天记录
我是科比学徒
这是你们的考核题，给你们2天时间，把所有需求完成并且提交一份10000字以上的项目报告文档
收到
精神抖擞
发送
文件
图片
Top

10.朋友圈测试


沉默的妄想者
搜索用户
搜索
聊天
通讯录
设置
朋友圈
发朋友圈
发布一条自己的朋友圈分享自己的动态
查看朋友圈
查看自己和朋友发的朋友圈
我的朋友圈
查看自己发布的朋友圈
朋友圈相册
查看自己朋友圈中的照片
朋友圈
返回
我是科比学徒 发布于 2019/5/15 下午3:10:45
I know that the day will come when my sight of this earth shall be lost, and life will take its leave in silence, drawing the last curtain over my eyes.Yet stars will watch at night, and morning rise as before, and hours heave like sea waves casting up pleasures and pains.When I think of this end of my moments, the barrier of the moments breaks and I see by the light of death thy world with its careless treasures. Rare is its lowliest seat, rare is its meanest of lives.Things that I longed for in vain and things that I got---let them pass. Let me but truly possess the things that I ever spurned and overlooked.
点赞(0)
评论(1)
收藏(0)
转发(0)
浏览量(0)
沉默的妄想者 发布于 2019/5/15 下午3:12:39
我是沉默的妄想者，我评论了你的朋友圈，快来看一下吧
Top

五．程序亮点

5.1 性能优化

1．数据库连接池

在 dao 层编写数据库连接池提高访问数据库的性能，与每次都关闭连接相比，数据库连接池在性能的优化上十分明显尤其对于即时通讯软件，对实时访问的能力要求更高，数据库连接池的作用更为明显

2．对实时聊天用户数据使用缓存

在与 websocket 建立连接时会 将用户数据加载到内容，包括用户信息和用户所在聊天的所有成员，避免了在每次发送消息时再去查询数据库，相比之下对用户数据进行缓存的功能能够较为明显地提高用户体验

3．使用消息缓存队列

连接时启动消息缓存队列，将消息先缓存到队列再定时缓存到数据库，对于即时通讯软件而言，频繁地发送消息，使用缓存机制能够明显降低服务器资源的消耗。

4．线程池

在系统中创建一个线程池，每个客户端连接时会从中获取一个线程用于启动消息队列

5.优化网页加载

优先加载消耗性能少的网页，是网页加载速度大大提升

5.2 用户体验

1 提供友好的界面

用户使用界面友好，各个地方具有使用上的提示信息

2.简化用户操作

各个输入框组，都设置相应的键盘监听，让用户可以使用回车代替点击发送消息等操作

3.图片上传预加载

用户上传图片时，会立即加载图片，再传回后台

4.前端使用 ajax 进行预加载

在前端界面上尽可能提供本地应用程序的体验，在首次加载时请求用户的聊天数据和消息记录，用户在打开新的聊天窗口时不需要重新加载，关闭聊天窗口时消息也不会被清空，包括输入框中已经输入的内容。

5.自动加载已读消息

当请求到没有未读消息时，会自动请求每个聊天最近的 20 条已读聊天记录

6.重复登陆提醒

当用户在别处登陆导致当前 session 失效时，会收到断开连接的提示，并且在发消息时也会收到无连接的提示

5.3 安全性

1.登陆过滤器：登陆校验，游客权限

2.后台数据检查

3.聊天记录独立

4.密码使用 md5 加密,修改登陆密码.

5.过滤标签注入

6.过滤脚本攻击

7.防止 sql 注入

六．程序难点

1. 前端独立窗口设计难点

难点：聊天窗口互相覆盖，输入框，发送按钮 id 重复
解决：将聊天窗口和数据库表的聊天 id 一一对应，形成唯一窗口，并使用 css+javascript 实现窗口切换

2. 后台数据库表设计难点

难点：前端需求的数据难以用一张表的数据提供，容易产生数据冗余
解决：构造视图层 VO 对象，对接前端需求，并且使用关联查询对接后台数据库，既实现向前端提供数据，又无需后台数据库做出妥协。

3. 后台数据库表设计难点

难点：聊天消息只存一份，但是每个用户都可以进行管理，要求互不影响
解决：使用 Record 记录表作为中间表，连接用户和消息，实现聊天记录独立管理

4.后台服务器设计难点

难点：实时聊天要求对消息及时转发，要求尽可能降低性能开销
解决：查询用户数据的操作，使用用户数据缓存，保存消息记录的操作，使用定时消息缓存队列，实现聊天过程无需数据库操作

5.后台服务器设计难点

难点：加好友和加入聊天群组希望实现实时通知，并且无需重载用户数据缓存
解决：聊天服务器对外提供动态地指定哦那过户发送消息通知，动态添加聊天成员地接口，实现用户缓存热更新。

七．体会总结

7.1 学习与收获

三月初

实践:

刚开始学 Java，尝试过几行代码加深理解

学习:

刚开学，只会 C 语言，也只有寒假 C 语言课程设计项目的经验，其他的编程语言都没有学过，开学后知道工作室考核要用 java，赶紧开始学 Java。

三月中旬

实践:

学了一些 Java 基础语法，只写过几行简单的代码，没有 Java 项目经验

学习:

工作室一面招新，Java 学到了接口和抽象类

三月底

实践:

3 月 21 号开始做第一个 Java 项目

学习:

工作室一面之后开始学数据库和 git,二面时掌握基本的 mysql 数据库操作，git 的基本操作，学了 jsp 和 servlet,以及 tomcat,并且慢慢学会使用基本的 html 标签。

四月初

实践:

第一个 Java 项目已完成，总结项目经验：

1.了解到有 el 表达式和 jstl 标签库可以代替 jsp 中的 java 代码

2.有正则表达式可以高效地进行字符匹配和数据检查。

3.没有异常处理，只是简单打印堆栈

4.常量没有管理，随处编写

学习:

在经验总结基础上，学习 Java 的 io 流，泛型，反射的知识，以及 rbac 权限管理，思考了异常的处理方案，单例模式，静态类和面向对象编程的关系，线程安全问题的出现和处理，几乎保持每天思考一个问题的节奏，在训练营中，了解到 postman 测试工具和 chrome 的调试工具，知道如何对前后端分别测试，了解到 md5 加密和 uuid。在导师的提示下，学习枚举类进行常量管理 。

四月中旬

实践:

4 月 7 号开始做第二个 Java 项目，这次真正使用了 git 进行版本管理，并且尝试给项目添加阿帕奇开源许可证，给项目生成 javadoc 文档，在学习了反射知识的基础之上，对项目的 dao 层进行了大幅度的改进，不再每个 dao 层实现类去取连接，执行，处理结果集，而是封装在一个基础 dao 实现类统一处理，用反射实现结果集的映射。并且尝试写了一个简单的数据库连接池。

学习:

学习了数据库的条件查询，模糊查询，分页查询，数据库的事务隔离级别，正则表达式，Java 的内存分区，反射和类加载器，了解 gc，过滤器，监听器，session 和 cookie。

四月底

实践:

第二个 Java 项目完成，租了一个阿里云服务器，买了个域名，尝试在远程 linux 系统上部署了自己的项目，在此过程中踩了不少坑，但是也对主机，域名，端口等概念有了更深的理解，并且在配置 tomcat 的过

程中，也对 tomcat 的目录结构更加了解，通过大量阅读 tomcat 的日志文件，处理错误，也体会到日志文件的好处。

总结项目经验:

1.控制层没有对请求做处理，直接下放到逻辑层，导致逻辑层接口不明确，与控制层严重耦合，不利于代码维护和复用。

2.没有业务逻辑设计，流程简单，大部分只在做增删查改，代码的价值不高。

3.没有采用后台数据检查，依靠前端阻止不合理数据，不可靠。这个在部署项目之后发布到朋友圈，被各种注入攻击的过程中深有体会。

4.没有采用后台数据校验，相信前端传回的登录用户名等信息，部署项目之后这个漏洞被利用，导致数据被篡改。

5.没有限制用户的操作次数合理范围，部署项目之后发现一个用户不断发送内容相同的评论，导致服务器资源被浪费，破坏评论的有效性。

学习:

训练营的培训中，了解到 javascript,css,json,ajax 等一些知识，小组培训中了解到了可以使用动态代理和自定义注解实现更好的 dao 层，因此花几天学习了动态代理，注解的知识，凭借当时小组培训时的一点印象，使用动态代理，注解，工厂方法，实现了 dao 层无实现类，只有接口，让动态代理去解析注解和执行。使得 dao 层代码大大简化，效率大大提高。借此也改进了数据库连接池，使用代理连接对象，使得在关闭连接时自动将连接放回连接池。了解数据库的内连接，联表查询，联表删除。

五月初

实践:

进入考核阶段，因为在做小组作业的过程中不断实践和积累，写 Java 代码的能力一直慢慢提升，不断地踩坑和摸索，发现和解决问题，经验也越来越多，此时 Java 已经具备了一定的基础，代码上的实现很容易就有思路，这个时候考虑更多的就是架构的设计上，业务逻辑设计和业务流程设计上，异常处理上，代码规范上，性能优化和用户体验上的一些问题。技术上也开始尝试前后端分离，除了登录注册之外，全部在前端使用 ajax 和 json 实现前后端数据的交互，并且尝试进行前端架构的设计，用 js 实现了前端页面展示的逻辑。开发的过程中使用 postman 对后台数据的接口进行测试，真正体会到前后端分离带来的好处，并且从前端，后端，两个角度理解了整个 web 程序的构建过程，和前后端协调交互，分工协作的过程。

学习:

在项目的技术需求下，学习了websocket技术,了解ajax轮询，长连接等知识。

7.2 项目体会

技术上的体会:

1.出错处理很重要，我们的程序不可能一直在理想的状态下运行，不合理的数据，意外的环境变化等都有可能影响程序的执行，如果没有对数据做检查，没有做好异常处理的话，

2.架构设计很重要，一开始就要做到需求分析和架构设计，架构设计的合理的话可以实现各个模块充分解耦，很容易扩展，到后面在架构之上增加功能的话不容易产生耦合，并且能够复用代码，就很快

3.编码规范很重要，一个是代码注释，注释写好到后面很容易维护，一个是范式，比如数据库建表范式，遵循范式可以让设计更合理，少走弯路，一个是接口规范，前后端交互的时候按照接口来写，效率很高

生活上的体会:

1.第一个是对自己要有信心

刚开始拿到考核题的时候觉得很难，很多东西没有学过，很多需求也没有做过，觉得是不可能完成的，但是坚持下来，也发现还是可以做完。

2.第二个是做事要细心

很多时候错误只要细心一点就可以避免

3.第三个是要有恒心

从训练营开始到现在，遇到许多困难，学习的知识非常多，有些问题是一遍又一遍的测试的过程中才发现和解决的。还需要解决很多的技术难题，有恒心，坚持，很重要。

7.3 总结

这个学期比较忙，学的东西非常多，生活上的空闲时间很少，大部分的时间，一直在不停地思考所学的新知识，也不停地反思之前所写的项目，存在什么问题。但是偶尔还是要抽空去打打篮球，如果连续两周没有运动，加上大量时间在学习，或者遇到难题迟迟不能解决，往往会觉得身心俱疲，这个时候就会放松一下，下午的时间拿去打一下篮球，打球的过程中专注于运动，也就忘记了学习的压力，困难，烦恼，运动之后一身大汗淋漓，身体的疲劳代替了心灵的烦恼，反而倍感轻松，第二天就重新充满了活力，也更加镇定，很多问题也迎刃而解。尽管生活过得忙碌，偶尔也会在吃饭的时候给妈妈发个微信，发一发饭菜的照片，说几句家常。真的体会到了平时真的很忙也会熬夜，却跟家里人说一切都好的滋味。训练营的经

历使我认识了许多优秀的人，也领悟到许多道理。每周的培训都能学到很多的知识，更重要的是，我发现 QG 的师兄非常的负责，每周在全体培训之后，还有大组培训，这个时候后台组能学到前端的知识，前端组的能学到后台的知识，培训之后还有答疑，非常负责地解决我们的问题。小组培训还会讲很多后台实用的知识，培训之后导师还了解我们的学习情况和解决遇到的问题。每次陪训我总能感觉到 QG 的师兄是真正在帮助我们学到知识，并且学得更快更好，因此也都十分珍惜每次陪训的机会。平时的时间我也会和导师交流遇到的问题，这个过程也学到很多东西，少走很多弯路。

训练营的过程让我领悟到，做一件事，要么不做，要么认真负责地完成。也让我明白，交流是一种很好的学习方式，交流让知识更好地融合，交流促进对知识更深的理解，交流让灵感迸发。训练营的经历将成为不可磨灭的一段记忆。