****

**本 科 毕 业 论 文（设计）**

|  |  |
| --- | --- |
| 课题名称 | 即时通信前端设计与实现 |
| 学 院 | 计算机科学与网络工程学院 |
| 专 业 | 软件工程 |
| 班级名称 | 软件162班 |
| 学生姓名 | 潘家炜 |
| 学 号 | 1606100136 |
| 指导教师 | 金政哲 |
| 完成日期 | 2020年5月1日 |

教 务 处 制

**[即时通信前端设计与实现](http://202.192.18.178:8080/manage/thesis/javascript:f_viewtopic('4a4092326eeb0d65016f1475687f4de1'))**

摘要 本文针对现今流行的即时通信软件进行调查和分析，对前端页面设计和操作流程进行深入研究，并说明各模块的功能设计与实现。在这基础上提供基于web即时通信好友管理功能，群组管理功能，基本文字交流功能，收发表情包和文件传输功能和维护在线用户状态等功能。为了客户切换设备也能访问到历史消息，所有的通讯消息聊天记录保存在服务器。即时通信系统采用前后端分离技术，前端页面采用HTML5+JavaScript+Vue框架和vue的周边库如集中式存储管理vuex和路由导航vue-router并采用WebSocket进行信息通信。后端使用MySQL和MongoDB进行持久层数据支持并使用redis管理在线用户的状态。

关键词 即时通信；web前端；websocket；

**ABSTRACT** This article investigates and analyzes the popular instant messaging software today, conducts in-depth research on the front-end page design and operation process, and explains the functional design and implementation of each module. On this basis, it provides web-based instant messaging friend management functions, group management functions, basic text communication functions, sending and receiving expression packages and file transmission functions, and maintaining online user status. In order for customers to switch devices to access historical messages, all communication message chat records are saved on the server. The instant messaging system uses front-end and back-end separation technologies. The front-end pages use HTML5 + JavaScript + Vue framework and vue peripheral libraries such as centralized storage management vuex and route navigation vue-router and use websocket for information communication. The backend uses MySQL and MongoDB for persistent layer data support and redis to manage the status of online users.

**KEY WORDS** Instant messaging; web front end; websocket；

目录

[即时通信系统前端的设计与实现 6](#_Toc5026)

[1. 前 言 6](#_Toc18774)

[1.1 课题研究背景 6](#_Toc23774)

[1.2 即时通信起源和发展 6](#_Toc5198)

[1.3 即时通信国内外现状 7](#_Toc10266)

[1.4 论文研究的意义 8](#_Toc19918)

[2. 开发环境和关键技术的研究 9](#_Toc19224)

[2.1 开发环境 9](#_Toc4856)

[2.2 vue前端框架 9](#_Toc6989)

[2.2.1 vue.js框架的简介 9](#_Toc17630)

[2.2.2 vue全家桶 10](#_Toc6448)

[2.3 vue网页布局 10](#_Toc7962)

[2.4 WebSocket协议 11](#_Toc25321)

[3. 即时通信系统需求分析 14](#_Toc15436)

[3.1系统设计目标 14](#_Toc4448)

[3.2系统目标用户分析 14](#_Toc16215)

[3.3核心功能点分析 14](#_Toc31502)

[3.3.1 用户管理功能模块 14](#_Toc20357)

[3.3.2 即时通信功能模块 15](#_Toc29174)

[3.3.3 文件传输功能模块 15](#_Toc18193)

[3.3.4 群组功能模块 15](#_Toc32424)

[3.4本章小结 16](#_Toc11931)

[4. 系统的总体设计 17](#_Toc26700)

[4.1系统概要设计 17](#_Toc13709)

[4.2系统业务模块设计 18](#_Toc6653)

[4.3数据库设计 21](#_Toc5770)

[4.3.1 MySQL数据库E-R图概览 21](#_Toc14330)

[4.3.2数据表 22](#_Toc4177)

[4.3.3 MongoDB消息JSON文档 24](#_Toc2946)

[4.4部署流程设计 24](#_Toc7871)

[5. 即时通信前端的详细设计与具体实现 26](#_Toc25162)

[5.1详细设计简要说明 26](#_Toc15569)

[5.2项目代码结构 26](#_Toc9971)

[5.3公用组件的实现 28](#_Toc32731)

[5.3.1 公共头像组件 28](#_Toc25069)

[5.3.2 个人资料组件 29](#_Toc7499)

[5.3.3 公共输入框组件 30](#_Toc3746)

[5.4模块功能的实现 31](#_Toc14260)

[5.4.1用户管理模块 31](#_Toc20204)

[5.4.2即时通信模块 38](#_Toc29936)

[5.4.3群组模块 42](#_Toc5904)

[总结 44](#_Toc11932)

[参考文献 45](#_Toc32481)

[致谢 47](#_Toc22340)

**即时通信系统前端的设计与实现**

1. 前 言
   1. 课题研究背景

随着网络技术的发展，互联网已经渗透在我们生活工作中的方方面面，网络已成为除了衣食住行之外必不可少的一部分。人们可以通过互联网随时随地获取千里之外的信息，并且不受区域的影响，开阔个人的眼界。其中，即时通信技术的出现，使人们与远处的家人和朋友或工作伙伴的沟通和交流获得极大的便利。

即时通信是目前网络最为流行的一种通讯方式。即时通信是一个终端服务，允许两人或多人通过网络服务传递与交换文字信息，图片信息，文件信息，也允许通过语音和视频进行交流。目前，即时通信技术的发展已经相对成熟，对于即时通信的各种使用场景的覆盖度也不断的扩大，比如基于个人的即时通信应用有国内的微信和QQ等，还有国外的Whatsapp和Telegram等。在办公场景下有国内的钉钉应用和字节跳动出品的飞书办公软件。但是原有的通讯方式渐渐达不到我们的要求。在界面设计的美观度上和用户体验也不够创新。Web端即时通信可以适应更多的通讯环境和场景，并根据用户的反馈可快速地进行迭代开发。这不仅给用户带来更好的使用体验，也降低了软件的开发成本，符合市场的需求。

本文主要针对即时通信系统进行设计和开发，对其在网络通讯中的特点和重点进行课题背景调查分析以及系统的需求分析和功能设计。本系统使用当今最流行的前端Vue框架，Golang后台语言，关系型和非关系型数据库技术以及缓存数据库技术，开发出一款适合年轻人社交的基于Web的即时通信系统。

* 1. 即时通信起源和发展

即时通信在互联网出现之前就已经被使用了。上世纪60年代，多用户操作系统的出现，用户很快就学会了与登录到同一台打印机或其他设备的其他人进行通信。随着网络、编程语言和协议的发展，电子公告牌系统在20世纪80年代出现，其中一些甚至包括聊天功能。

在1996年，四名以色列青年成立了一家公司，并开发出最早的即时通信软件ICQ。ICQ作为即时通信软件领域的开创者，不得不说他成就了一个辉煌的开端。软件一经推出，凭借着极为杰出的创意和便利的社交交流很快在全世界拥有大量的用户。从发布之后的6个月，ICQ成为当时世界上用户量最大的即时通信软件。

在1999年2月，腾讯自主研发并发布了一款的基于Internet的即时通信网络软件--腾讯QQ。QQ合理的设计，良好的用户体验，强大的功能，稳定高效的运行系统，支持中文的软件网络环境，赢得国内用户肯定。经过十几年的发展，腾讯QQ的功能越来越强大，用户体验越来越好，不仅支持在线聊天，还支持视频聊天以及语音聊天，再后来支持共享文件，网络硬盘和离线传输文件等功能。似乎腾讯QQ看起来已经足够完美，但是在2011年1月，腾讯又发布了一款即时通信软件--微信。微信支持跨通信运营商，跨操作系统，QQ的基本功能微信也有。如文字聊天，视频或语言聊天以及发送文件等。但微信还有额外的基于位置社交的功能，如“摇一摇”，“朋友圈”以及“公众平台”等。可以看出即时通信软件从最初的只注重即时消息传递与交换到现在把重心放在社交上的这一转变。随着互联网的发展，人们对实时互动的要求越来越高。其实，即时通信技术不止应用于QQ，微信这样的面向聊天软件，他其实有着宽广的应用场景和足够有想象力的前景。在不知不觉间，即时通信系统已经根植在我们的互联网生活中。

* 1. 即时通信国内外现状

无论在国内还是国外，即时通信软件都获得了极大的肯定和重视。在国内不止像腾讯QQ，微信等独立的即时通信软件，也有像淘宝天猫等电商类的嵌入式的即时通信软件。从百万级用户到亿级用户的发展，无疑给从多企业公司极大的挑战。微信的用户数量排在世界第一，用户来自200多个国家。最近，国内一款由阿里巴巴公司开发的，企业级的即时通信软件--钉钉，进入人们的视线。由于新冠疫情的影响，学生只能通过网络上课，而钉钉则提供一个给老师及学生上课的平台。这使得钉钉的用户量激增，无疑的是钉钉迎接了这一挑战并稳定持续地提供了服务。

在国外有WhatsApp，facebook,Viber等即时通信软件。这些软件都实现了文字聊天视频通话，发送图片，上传文件和定位信息的功能。周边功能的丰富度是他们唯一的不同之处。C/S架构的即时通信应用层出不穷，但是基于B/S架构的Web端即时通信应用相比之下数量并不多。微信网页版是其中最为流行的，使用Comet技术实现的Web端应用之一。在Web通讯技术中，有传统短轮训技术，长轮询技术和WebSocket协议。即时通信使用短轮训有诸多缺陷。一是消息无法及时送达，这取决于轮询的时间间隔，二是消耗不必要的服务器资源。短轮训和长轮询技术都不是最优解，后来出现WebSocket协议，允许服务器主动推送信息，解决了轮询的非即时性和消耗过多服务器资源的问题。

* 1. 论文研究的意义

随着社会发展脚步的加快，人和人之间的合作逐渐频繁，固然人和人的沟通效率成为关键的一环。互联网的出现是让我们的生活更加方便更加多姿多彩，是让我们办事的效率获得极大的改善。即时通信搭载着互联网的使命，让距离再也不是人和人之间沟通的阻碍，在任何地方都能和世界各地的人交流。即时通信无疑在生活和工作中扮演重要的角色。

现今，有许多即时通信软件流行，像微信，腾讯QQ等，他们的技术水平也处于世界领先地位。同时，为了满足市场多样化的需求，Web即时通信应用应运而生。那些在手机或PC端需要安装才能使用的应用，往往还不够便捷。Web端即时通信应用只需要在浏览器打开一个网址就可以使用，同时功能也不输于pc和手机端。基于web的即时通信软件是该领域重要的组成部分，给用户带来更多的选择。

1. 开发环境和关键技术的研究

2.1 开发环境

硬件环境：内存容量8G以上，硬盘容量100G以上，2台主机

开发系统：Ubuntu 16.04，Windows10

开发前端框架：Vue

开发语言：JavaScript，HTML，JavaScript，CSS，Golang

容器化部署技术：Docker

数据库：Redis，Mysql，MongoDB

开发工具：VSCode

浏览器：Chrome最新版本

2.2 vue前端框架

2.2.1 vue.js框架的简介

Vue是一套构建前端界面的渐进式框架。自从2014年发布以来，Vue.js已经成为世界上最流行的JavaScript的框架之一。这种流行的原因之一是组件的广泛使用。开发人员能够创建整个应用程序中使用和重用简洁模块。它和其它的前端框架如Angular和React不同。Vue采用的设计是自底向上增量的开发设计。Vue的核心代码只关注视图层且十分容易和其它库或已有项目整合。它和传统的开发模式不同。传统的开发模式使用原生的JavaScript或者使用jQuery库来进行开发，并且没有数据驱动，数据的更新变化都需要操作原生的DOM元素，带来极大的不便。如今，人们很难想象没有任何JavaScript框架参与的前端开发。

Vue有两个核心要点：数据绑定和组件视图。数据绑定也可以说是数据驱动，意思是数据的变化会带动视图的变化。不需要使用传统的方法通过操作DOM更新视图。Vue把整一个网页的拆分成一个个区块，每个区块我们可以看作成一个组件，即组件视图。网页是由多个组件视图拼接或者嵌套组成。

使用Vue作为前端开发有很多好处。第一，Vue非常容易上手。很多现代框架都有其设计哲学：简单的且功能不太多的核心，再加上其他库来增强功能。正是因为简单好学，很多前端开发工程师可以快速掌握并用到实际的开发中。第二，Vue的性能高，占用内存小。Vue 核心代码库加上Vuex和Vue Router 的项目大小仅为 30KB。而要让 React 的架构达到相似的功能，需要 React DOM + React Router + React 和其他插件，总共是 48.8KB。

2.2.2 vue全家桶

**Vuex**是专门为Vue.js应用程序开发的状态管理工具。它使用一个集中的存储来管理应用程序的所有组件的状态，并根据相应的规则可以确保状态以可预测的方式变化。为了方便数据在组件间的共享和传递，Vuex应运而生，再也不必局限于数据是否在父子组件间传递。若是属于组件的私有数据就不必放在vuex保存。换句话说，只有需要共享的数据，才有必要放到Vuex中。组件内部私有的数据，放到组件的 data 中即可。相比浏览器本地存储，响应式的状态存储是Vue的一大亮点，且数据是直接从内存中取得，数据响应速度极快。当组件从 Vuex中读取数据的时，若 Vuex 中的数据发生变化，使用该数据的组件也会得到高效更新。

**Vue-Router**是Vue.js官方的路由管理器。路由分为软件路由和硬件路由，主要功能是对数据流量进行分发。这里的路由是软件路由，准确来说是单页应用的路径管理器。再通俗的说，vue Router就是前端应用的链接路径管理系统。它和 Vue.js 的核心高度集成，让构建单页面应用变得轻而易举。

2.3 vue网页布局

网页布局会影响网站的设计和用户的体验。网页布局常见有盒模型，flex布局和grid布局。

**盒模型**是使用CSS设计、布局的基石所在，CSS盒模型的本质是将HTML元素看做盒子。一个标准的盒模型由内容、内填充、边框和外边距四部分组成。在CSS中，所有的HTML元素都可以被看做一个盒子。对于初学者而言，深入理解盒模型这一重要概念，更有利于页面的排版和布局。

**Flex** 的缩略词是 Flexible Box，意思是"弹性布局"，用于为Box模型提供最大的灵活性。传统的布局解决方案基于盒模型，依赖于显示属性，位置属性和浮动属性，不易实现垂直居中，项目排列等其他复杂的布局。Flex带来极大的便利，几行代码即可实现复杂的需求。采用 Flex 布局的元素，称为 Flex 容器。它的所有子元素自动成为容器成员，称为 Flex 项目。

**Grid**布局又称网格布局，Grid 布局是网站设计的基础，CSS Grid 是创建网格布局最强大和最简单的工具。Grid 布局实现复杂的网页结构布局不仅更加灵活而且开发效率更高。Grid 将网页分成网格，并可以任意组合不同的网格来制作不同的布局。网格是一组相交的水平线和垂直线，它定义了网格的列和行。我们可以将网格元素放置在与这些行和列相关的位置上。采用网格布局的区域，称为"容器"。容器内部采用网格定位的子元素，称为"项目"。

2.4 WebSocket协议

WebSocket是一种与HTTP协议兼容但与HTTP不同的网络传输协议。Websocket工作在HTTP 80和443端口之上，在防火墙阻止不是web环境中的连接非常有帮助。WebSocket位于网络层的应用层，可在单个TCP连接上进行全双工通讯。WebSocket握手使用HTTP升级头将HTTP协议转换为WebSocket协议。WebSocket协议支持浏览器与服务器之间的交互，开销低，方便实时数据传输。

WebSocket的通讯原理：

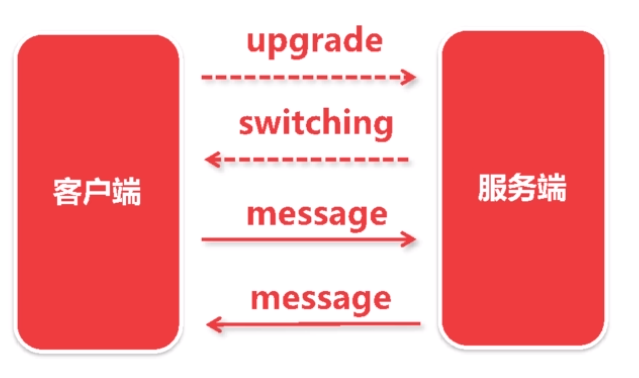


图 2-1 websocket原理图

首先，需要协议升级。下图是WebSocket协议请求头：

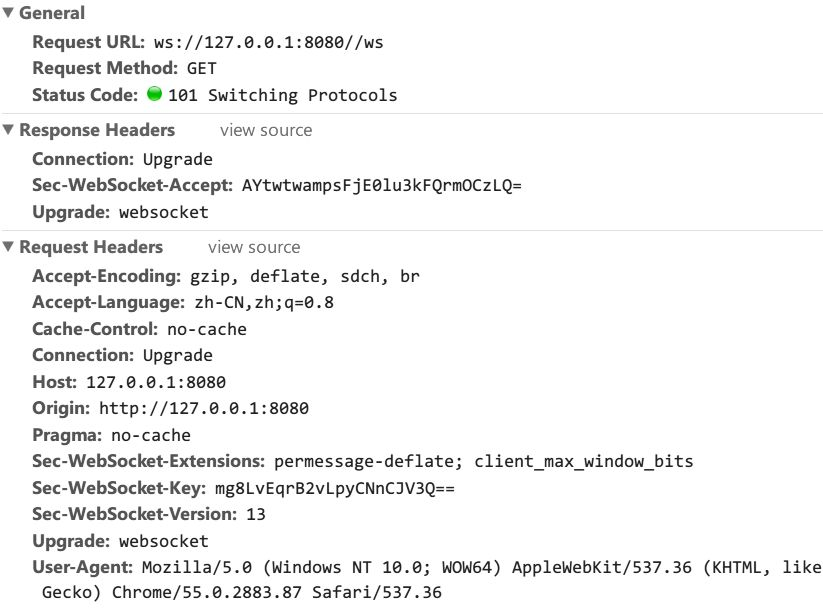


图 2-2 websocket请求包图

WebSocket 的握手是以 HTTP 的形式发起的。但是协议的标识符是ws，如果加密则为wws。Sec-WebSocket-Version表明客户端所使用的协议版本。响应的状态码101，表示切换了协议，说明利用http建立传输层的TCP连接，之后便与http协议无关的。Message底层会被切成多个Frame传输，编程时只需要操作Message，无需关心Frame。下图是WebSocket的frame：

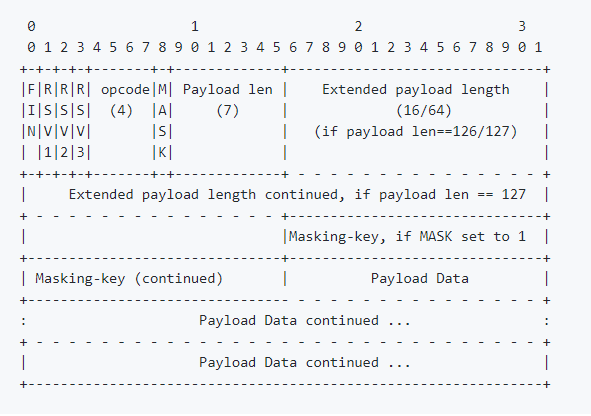


图 2-3 websocket协议图

按照RFC中的描述：FIN表示这是一个消息的最后一帧。RSV1/RSV2/RSV3表示这是一个扩展字段，只要经过协商赋予非零值才表示某种含义，否则必须为0。Opcode用来解释payload data的用途和功能。

WebSocke端点的有4个生命周期事件：打开事件，消息事件，错误事件和关闭事件。打开事件：事件发生在连接打开的时候。消息事件：如果服务器发送信息则会产生这个事件。他可以发生在连接打开到连接关闭这任意的时间。错误事件： WebSocket发生错误时则调用该事件。关闭事件：表示WebSocket做出关闭的动作，可以由任何一方关闭。

1. 即时通信系统需求分析

3.1系统设计目标

本系统的核心目标在于实现基于web的即时通信应用，用户不需要下载和安装就可以使用本系统进行远程即时交流。用户可发送文字消息，图片和文件消息，并且提供丰富的字符表情和动图表情包，也可以根据自己的需求对好友和群组进行管理。

3.2系统目标用户分析

本系统适用于全年龄段用户。经分析，年轻人比较喜欢使用表情图片进行交流，所以在页面设计和功能设计上偏向年轻人群体。

3.3核心功能点分析

基于WebSocket的即时通信系统通过B/S架构，在保留其他系统优秀的特性的前提下，主要实现了四个功能模块：用户管理模块，即时通信模块，群组模块和文件传输模块。

**用户管理模块**主要实现用户登录与注册功能和好友管理功能，注册成功后可登录进系统并进入设置页面完善个人信息，设置个人状态和自定义状态，让好友知道自己的此时此刻详细的动作与心情。好友管理主要实现对好友的增加，删除和屏蔽等操作。

**即时通信模块**主要实现消息的接收和发送功能。发送消息的类型包括文本型消息，图片型消息和文件型消息。消息的接收有友好的未读提示功能，提醒用户消息的到达。消息的展示因类型而异，且发送的消息可以随时删除撤回。

**群组模块**主要实现群组管理的功能。在即时通信模块的基础上，实现多对多的聊天。除此之外，加入了群成员的显示和管理，实现了创建群聊，修改群聊和解散群聊等功能。

**文件传输模块**主要实现图片或文件的上传和下载。图片文件以内容显示的方式展示，其他文件仅供下载使用。

3.3.1 用户管理功能模块

用户管理模块主要实现用户的登录和注册，用户个人信息变更和好友管等功能。其实现功能包括：

1. 用户登录。需要输入电子邮箱地址和密码。如果还没有账号，需要注册账号。
2. 用户注册。要使用系统功能，与好友进行交流，必须要注册一个账号，需要用户名，邮箱和密码的输入。
3. 变更用户个人信息。在初次登陆时，系统会设置默认的个人信息。在任何时候都可以变更自己的信息。
4. 添加好友。用户可根据用户的电子邮箱地址进行添加，经过对方同意后成为好友可进行聊天。
5. 删除好友。用户如果不喜欢该好友可以删除，可重新添加。
6. 屏蔽好友。用户可以屏蔽对自己造成困扰的好友。

3.3.2 即时通信功能模块

文本聊天模块负责最基础的Web即时通信功能，主要实现功能如下：

1. 收发信息。可发送文字，表情，图片或文件。好友收到消息会有提示信息。
2. 信息的展示。根据消息的不同类型，定制消息的展示形态。

3.3.3 文件传输功能模块

文件传输也是即时通信重要的一个功能，能及时地共享文件，不需要通过别的途径进而方便交流。主要功能实现如下：

1. 文件的上传。在聊天输入框点击加号可选择文件进行上传，上传的过程中可随时取消且有文件上传进度提示。
2. 文件的展示和下载。若是收到图片文件则直接将图片显示在聊天框，若是其他文件则提供下载按钮供用户下载。

3.3.4 群组功能模块

群聊模块主要实现多个好友间可创建一个群组或讨论组，进行多对多的通讯。主要功能实现如下：

1. 创建群聊。用户可以根据自己的需求创建群组，成员角色分为群主和普通群成员。
2. 修改群。可以更改群名称和头像，新增成员和移除成员。
3. 解散群聊和退出群聊。当用户不需要这个群聊时，可以解散或者退出。

3.4本章小结

本章介绍了即时通信系统的主要功能需求，并分析各个模块的功能。首先，提出了基于Web和WebSocket的即时通信设计思路并分析了传统的即时通信软件的不足。然后，说明每个模块的功能实现要求。在本章通过了分析和说明之后，为下一章的实现需求奠定良好的基础。

1. 系统的总体设计

4.1系统概要设计

基于Web的即时通信系统包含展示层，接入层，即时通信服务层和数据库层四个层次。

展示层：提供一个良好的操作界面给用户使用。用户通过操作前端页面，前端通过HTTP请求向后端获取数据，通过JSON数据格式，进行数据交换。通过WebSocket连接即时获取服务端推送的数据。

接入层：主要实现流量的转发和静态文件服务的功能。当前端构建之后，会生成对应的静态文件。前端请求静态文件则在接入层直接返回并且不需要请求服务层。如果URL的前缀是/api则直接转发到服务层进行处理。在接入层配置HTTPS也具备了一定的网络安全基础。若用户量越来越多，单台机器已经承受不住压力，只需要即时通信服务层扩容，接入层将流量分散打入多个服务层，提高服务的可用性。

即时通信服务层：主要提供登录注册，好友管理，即时通信和文件传输等服务。向前端提供HTTP协议的API和WebSocket，在服务器监听请求并进行路由分发到相应的逻辑处理函数。

数据库层：在本系统中，通过文件系统，关系型数据库MySQL，文档型数据库MongoDB，内存数据库Redis分别存储用户上传的图片和文件，用户和群组信息，聊天消息以及用户的在线状态。

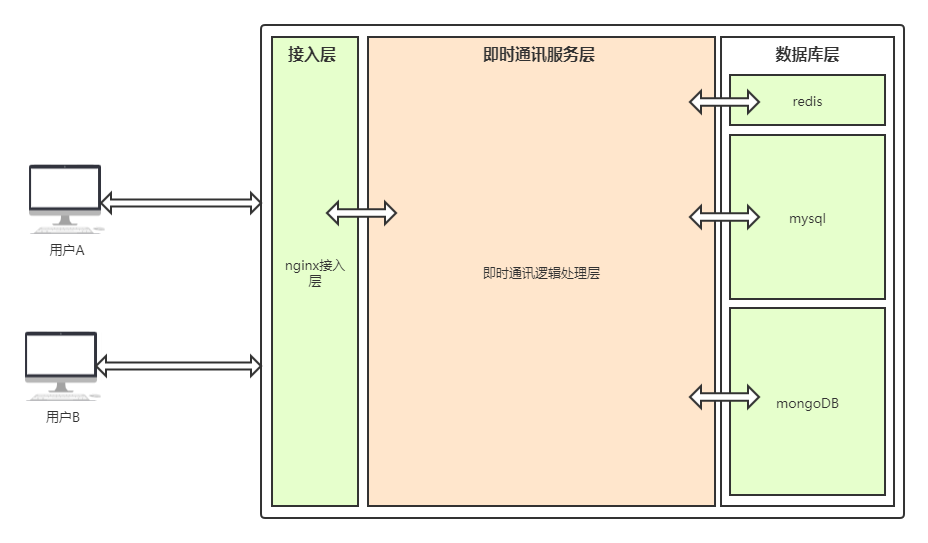


图 4-1 系统架构图

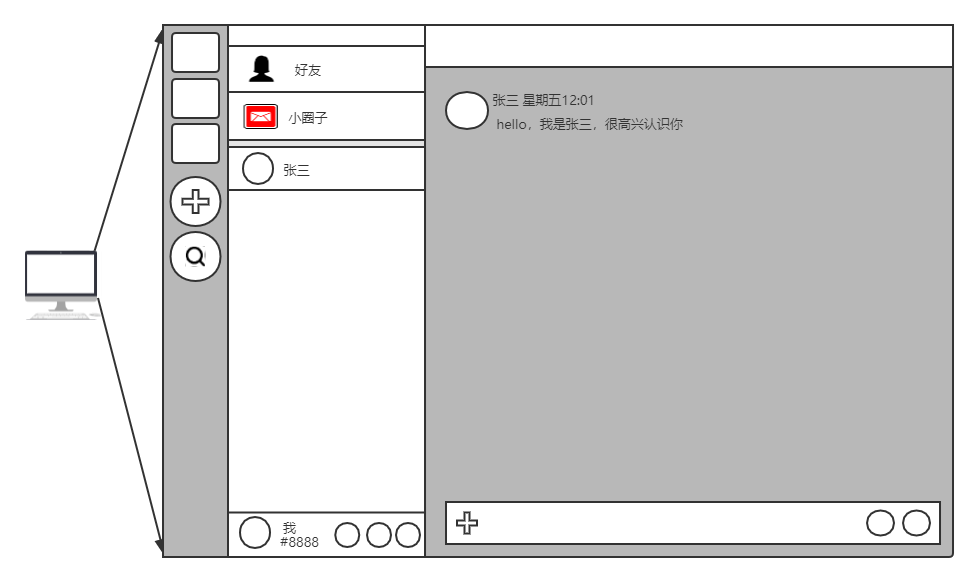


图 4-2 系统前端界面概览图

4.2系统业务模块设计

**用户管理模块：**

对于用户管理模块，主要实现的是用户的登录，注册和好友管理等功能。在实现功能的同时，为了用户数据的安全，在数据库中不直接存储用户的明文密码，而是将用户的明文密码用MD5算法加密并存储。

用户注册：用户通过在Web客户端输入电子邮箱地址，用户名和密码进行注册。一个电子邮箱地址对应一个用户账号，不能同时申请多个账号。在某些系统上需要用户名唯一不能有冲突，用户只能无奈选择自己不喜欢的用户名。在本系统中，当用户注册时会生成4位标识数字，当用户名+标识数字唯一标识用户名可用，注册成功。

用户登录：用户注册成功之后就可以根据电子邮箱地址和密码进行登录，密码认证采用MD5算法进行对比认证。登录成功后，后台维护用户的在线状态信息，并开始建立WebSocket连接。

好友管理：好友管理主要包含新增好友，删除好友和屏蔽好友三个功能。新增好友需要知道对方的电子邮箱地址。首先检查对方是否是自己的好友，如果是则结束。其次检查对方是否同样请求自己为好友，如果是则直接成为好友。如果不是则将请求插入双方的好友待定列表。

删除好友则删除单方面的好友关系，并设置对方的好友关系的状态为已删除。屏蔽好友则在好友列表删除该好友，将该好友移入屏蔽列表并设置好友的关系状态。被删除和被屏蔽的好友则无法发送消息。

**即时通信模块：**

即时通信模块是基于WebSocket网络协议实现的，主要实现消息的接收和发送，如发送文字，表情，图片和文件。当用户成功登陆，前端系统自动初始化WebSocket连接。WebSocket成功连接之后会调用onopen()函数，提示Web客户端连接成功并且可以初始化后续的步骤。如果客户端调用了onerror()或者onclose()函数则新建连接，如果三次之后连接仍让失败则提示用户相关的信息并退出登录。在WebSocket成功，用户在界面进行聊天，前端会自动构建消息体，掉用WebSocket的send()函数以JSON的数据形式发送给服务端。服务端收到消息并对消息进行解析,存储和转发。前端则监听WebSocket的onmessage()函数，若服务器有新的消息推送过来，则前端收到消息并对消息进行解析暂存和展示。

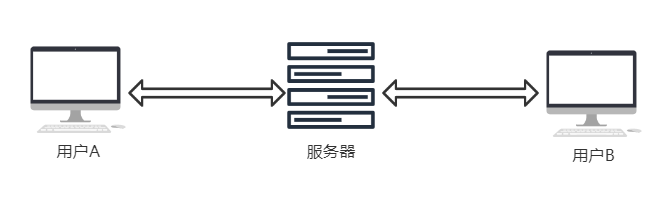


图 4-3 系统通讯原理图

当用户A发送一条消息给用户B，只要这条消息到达服务器并且存入数据库，用户A就完成了发送消息这一步骤。服务器会从消息体中找出接收者的ID也就是用户B的ID,如果用户B有连接上服务器，则把这条消息通过WebSocket方式推送给用户B，否则丢弃这条消息，等待用户B上线拉去消息。

用户可以发送文本消息，图片消息和文件消息，发送的数据虽然并不一样，但是消息体的JSON格式是一致的。一条消息可以表示text,image和file这三种类型的其中一种，image和file都有额外的信息，比如文件名，文件大小和数据源URL。Image消息类型比其他类型多了图片高度和宽度信息，若图片加载较为缓慢，则以占位背景展示，提供良好的排版体验。

表情是Unicode编码格式，可以看做是一类文字，不需要做其他处理。发送文字或表情活动图如下：

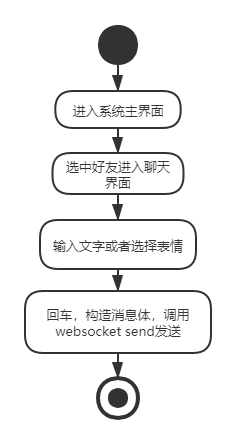


图 4-4 发送文字消息活动图

图片信息比文字信息多了url，宽度，高度等信息。发送图片之前，首先需要获取图片的文件名，文件大小，图片的高度和宽度，再讲图片上传到服务器，服务器返回图片的URL地址，最后把图片信息组合构造消息体打包发送。发送文件信息和发送图片信息的步骤基本是一样的，唯一的差别是只有图片消息需要获取图片的宽度和高度。发送图片活动图如下：

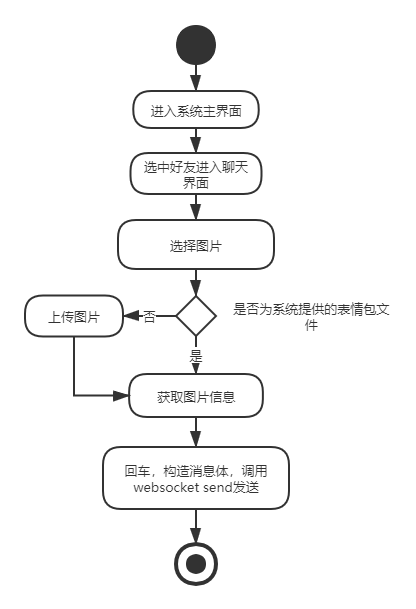


图 4-5 发送文件消息活动图

**文件传输模块：**

文件的传输也是即时通信重要的一环，一个良好的人性化的文件传输对于用户体验显得尤为重要。上传文件时，显示进度条告诉用户当前的上传情况。如果网络状况不尽人意，用户可随时取消上传，放弃这一条消息。上传文件的大小控制在100MB以下，如果超过100MB则拒绝发送。为了节省文件服务器的空间，上传的文件采用MD5算法进行分析，文件名采用MD5算法运算后的字符串。若服务器有相同的文件则不再另外存储一份。

**群组模块：**

群组模块主要解决多个用户的交流沟通需求，目的是为了可以在群组里面共同探讨同一个问题或者同一个感兴趣的领域，让多个成员参与进来。群组成员不必须是好友关系，不受是否为好友的限制。群组的角色只有两类，一是管理人员，而是普通成员。管理人员可以对群成员列表进行操作。

4.3数据库设计

4.3.1 MySQL数据库E-R图概览

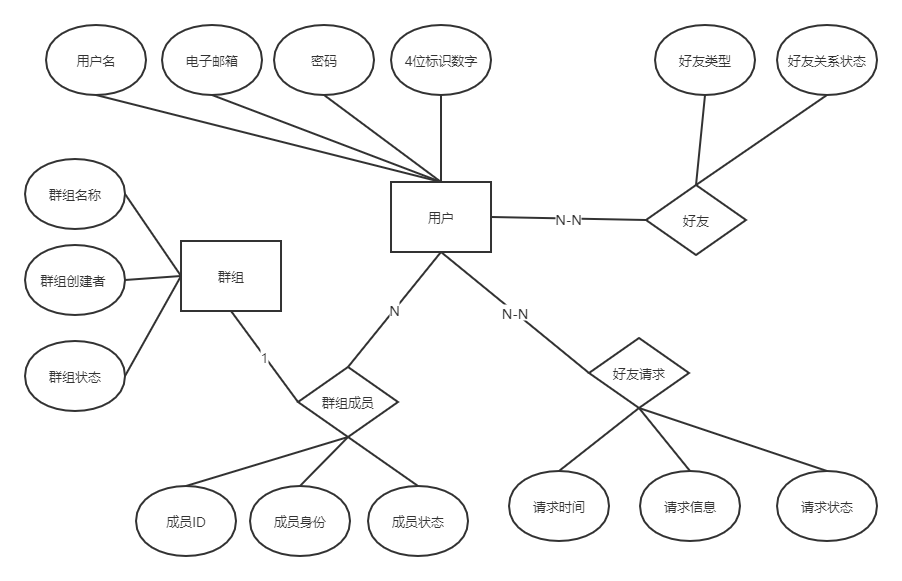


图 4-6 数据库关系图

4.3.2数据表

t\_user表：用于记录用户信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 类型（） | 说明 |
| 1 | uid | varchar(255) | 用户ID |  |
| 2 | name | varchar(255) | 用户名 |  |
| 3 | tailnum | chat(4) | 用户4位标识 |  |
| 4 | passwd | varchar(255) | 密码 | MD5算法加密 |
| 5 | sex | varchar(255) | 性别 |  |
| 6 | age | int | 年龄 |  |
| 7 | city | varchar(255) | 所在城市 |  |
| 8 | email | varchar(255) | 电子邮箱 |  |
| 9 | telephone | varchar(255) | 电话号码 |  |
| 10 | avatar | varchar(255) | 头像URL |  |
| 11 | self\_defind\_status | varchar(255) | 自定义状态 |  |
| 12 | self\_defind\_status\_deadline | bigint | 自定义状态结束时间 |  |

表4-1 t\_user表：用于记录用户信息

t\_friend表：用于记录用户好友信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 字段名（中文） | 说明 |
| 1 | user\_id | varchar(255) | 用户ID |  |
| 2 | friend\_id | varchar(255) | 好友ID |  |
| 3 | friend\_type | int | 好友类型 | 0：表示普通好友；1：表示该好友被屏蔽 |
| 4 | comments | varchar(255) | 备注 |  |
| 5 | createtime | bigint | 创建时间 |  |
| 6 | unreadcount | int | 未读数 |  |
| 7 | latest\_msg\_time | bigint | 最新消息时间 |  |
| 8 | status | int | 好友关系状态 | 0：表示状态正常；1：表示不可发送消息状态 |

表4-2 t\_friend表

t\_frien\_req好友请求表：记录发送的邀请信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 字段名（中文） | 说明 |
| 1 | from\_userid | varchar(255) | 发出请求的用户ID |  |
| 2 | to\_userid | varchar(255) | 被邀请的用户ID |  |
| 3 | createtime | bigint | 请求时间 |  |
| 4 | message | varchar(255) | 发送的消息 |  |
| 5 | status | int | 状态 | 0:表示正常；1：表示被拒绝 |

表 4-3 t\_friend\_req表

t\_group群组表：记录群组的信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 字段名（中文） | 说明 |
| 1 | group\_id | varchar(255) | 群组ID |  |
| 2 | group\_name | varchar(255) | 群组名称 |  |
| 3 | group\_creater\_id | int | 群组创建者ID |  |
| 4 | group\_avatar | varchar(255) | 头像URL |  |
| 5 | group\_member\_count | int | 成员数量 |  |
| 6 | group\_create\_datetime | bigint | 创建时间 |  |
| 7 | group\_status | int | 群组状态 |  |

表 4-4 t\_group表

t\_group\_member群成员表：记录群组成员的信息

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 字段 | 类型 | 字段名（中文） | 说明 |
| 1 | p\_group\_id | varchar(255) | 群组ID |  |
| 2 | p\_group\_member\_id | varchar(255) | 用户ID |  |
| 3 | p\_group\_member\_identity | int | 用户身份 | 0：表示普通成员；1：表示管理员 |
| 4 | p\_group\_member\_join\_datetime | bigint | 加入时间 |  |
| 5 | p\_group\_member\_exit\_datetime | bigint | 退出时间 |  |
| 6 | p\_group\_member\_status | int | 成员状态 |  |
| 7 | p\_group\_member\_unreadcount | int | 未读数 |  |
| 8 | p\_group\_latest\_msg\_time | bigint | 最新消息时间 |  |

表 14-5 t\_group\_member表

4.3.3 MongoDB消息JSON文档

|  |
| --- |
| {  "\_id" : ObjectId("5eb3d291285d4f609ded9921"),  "msg\_id" : "35ba7544-eaf7-4329-9f56-1fdf4cfb961f",// 消息的id  "msg\_type" : "text",//消息类型，text，image和file三种  "timestamp" : NumberLong("1588843153242129024"),//发送时间  "sender" : "fc092b9f5f7c4bd3b805c85ae9e3cfcb",//发送者用户id  "receiver" : "6214430e56174de98e6a524578871bd5",//接受者用户id  "innerText" : [//文字信息  "1"  ],  "content" : {//文件信息  "src" : "",//文件源URL  "width" : 0,//图片文件宽度  "height" : 0,//图片文件高度  "filename" : "",//文件名  "filesize" : NumberLong(0),//文件大小，单位B  "mtype" : "",//文件的主类型  "type" : ""//文件的详细类型  }  } |

4.4部署流程设计

本系统采用Docker容器化技术部署，使用docker-compose简化部署流程。

对于前端服务，需要将前端代码编译打包，并部署nginx服务器，将编译后的前端文件放在nginx服务器目录底下，再做相应的nginx服务器配置。

对于后端服务，需要依赖redis，MySQL和mongoDB的服务。这些一并使用docker-compose部署，对外提供API服务。

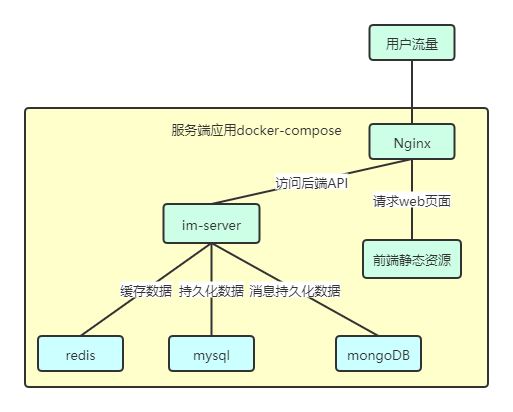


图 4-7 系统部署架构图

1. 即时通信前端的详细设计与具体实现

5.1详细设计简要说明

在本章中，会对系统的核心模块做详细的设计，以及说明前端的主要实现原理。在本章设计中介绍主要的四大模块：用户管理模块，即时通信模块，文件传输模块和群组模块。

5.2项目代码结构

项目前端代码将业务逻辑代码分为展示的组件代码和负责前端状态存储的vuex层。

│ App.vue ---------- 框架根节点

│ g\_websocket.js ---------- WebSocket处理

│ main.js ---------- 主文件

│

├─api ---------- 请求后端配置文件

│ index.js

│

├─components ---------- 页面组件

│ │ account.vue ---------- 注册登录根组件

│ │ discovery.vue ---------- 发现页组件

│ │ home.vue ---------- 首页组件

│ │ infomation.vue---------- 小圈子组件

│ │ login.vue ---------- 登录组件

│ │ master.vue ---------- 聊天首页

│ │ Notfound.vue ---------- 404组件

│ │ register.vue ---------- 注册组件

│ │ setting.vue ---------- 个人设置组件

│ │

│ ├─chat ---------- 聊天组件

│ │ chat.vue ---------- 聊天组件

│ │ emoji.vue ---------- emoji组件

│ │ gif.vue ---------- 表情包组件

│ │ message.vue---------- 消息组件

│ │

│ ├─common ---------- 公共组件

│ │ butif\_loading.vue ---------- 精美的loading组件

│ │ home\_loading.vue ---------- 首页loading组件

│ │ loding.vue ---------- 通用loading组件

│ │ mutliline\_input.vue ---------- 多行输入组件

│ │ mycommon.js ---------- 系统的一些对象

│ │ myhead.vue ---------- 头像组件

│ │ status.vue ---------- 用户状态组件

│ │ user\_info.vue---------- 个人资料组件

│ │

│ └─friend ---------- 好友管理

│ add\_friend.vue ---------- 新增好友组件

│ friend.vue ---------- 好友根组件

│ friend\_all.vue ---------- 全部好友列表组件

│ friend\_online.vue ---------- 在线好友列表组件

│ friend\_shield.vue ---------- 屏蔽好友列表组件

│ friend\_waiting.vue---------- 待定好友列表组件

│ now\_active.vue ---------- 活动好友列表组件

│

├─router ---------- 前端路由组件

│ index.js

│

└─store ---------- 前端状态存储组件

│ index.js

│

└─modules

chat\_list.js ---------- 聊天列表存储组件

chat\_message.js ---------- 聊天信息存储组件

emoji.js ---------- emoji表情存储组件

emojis\_res.js ---------- emoji表情数据源

friend.js ---------- 好友信息存储组件

gif.js ---------- gif表情包存储组件

group\_member.js ---------- 群组成员

myself.js ---------- 个人信息

5.3公用组件的实现

5.3.1 公共头像组件

在系统中，许多地方需要展示用户的头像及其在线离线的状态信息。为了头像的统一展示效果和减少代码量，将头像组件抽离出来。在头像组件的状态信息调用了另一个公共组件，即状态组件。状态组件根据用户不同的状态可表现在线状态，离线状态，请勿打扰状态和闲置状态四种状态。公共头像组件可选择是否展示状态，若with\_status为false则不展示状态，若为true，则根据user\_status的数值展示相应的状态。headimg参数决定用户头像的图片文件，若为空则展示默认的头像图片。实现如下：

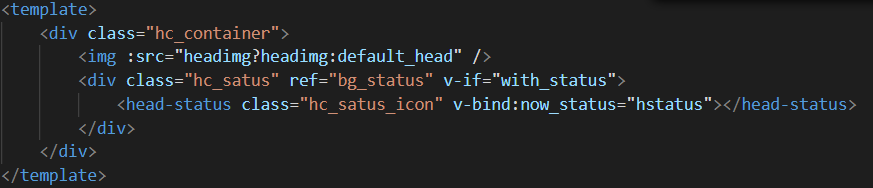


图 5-1 头像html实现图

使用如下：

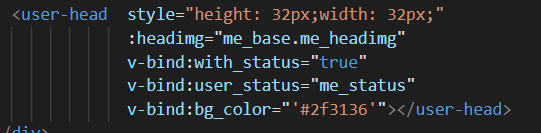


图 5-2 头像组件使用图

效果如下：



图 5-3 头像组件效果图

5.3.2 个人资料组件

个人资料组件主要展示用户的基本信息，如：用户名，头像，电子邮箱地址，性别，年龄，所在城市和电话号码等。用户名和电子邮箱地址是必填的，假如用户注册后没有完善其他进本信息，则显示为空。个人资料组件除了展示信息之外，也提供相关的操作，比如：打开聊天窗口，删除好友，屏蔽好友等。当用户点击某个用户查看个人资料时，只有一部分基本信息，待个人资料弹框弹出，前端会自动发送HTTP请求获取用户全部的详细资料。GetUserInfo主要根据用户的uid获取用户的信息，并及时更新用户资料的展示，关键代码如下：

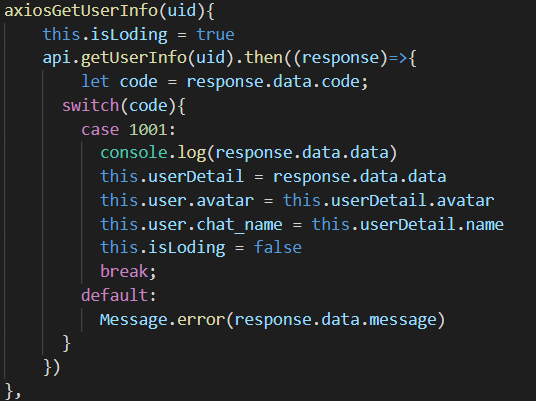


图 5-4 获取用户资料代码

由于这个公共组件，前端只会渲染一次，并根据数据的不同而展示不同的信息。因为部分信息是从服务端请求的，所以数据的变动需要重新想服务器请求。

效果如下：

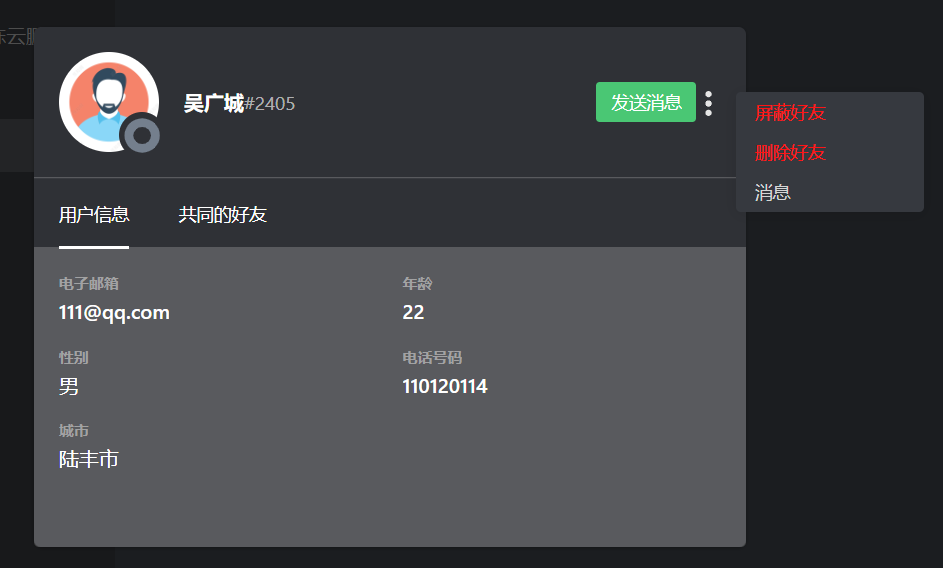


图 5-5 个人资料组件效果图

5.3.3 公共输入框组件

由公共输入框组件主要使用在聊天窗口发送文字消息。输入框使用div标签和contenteditable="true"来实现。传统的input输入框不能实现多行输入，textarea标签可以实现多行输入但不够灵活。公共输入框组件使用mutli\_input\_keydown函数监听Enter键和Ctrl键。若只有enter事件发生，则发送消息。若是Ctrl+Enter键，则表示换行继续输入。Enter的键码是13，代码如下：

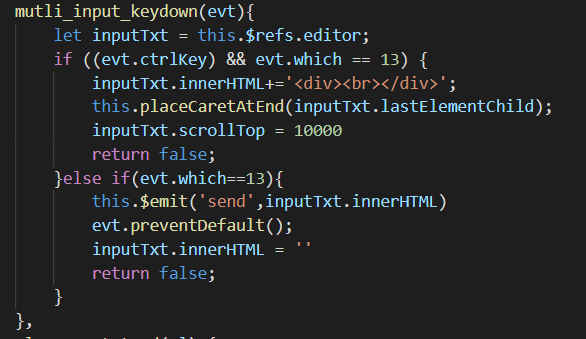


图 5-6 输入监听处理代码

placeCaretAtEnd函数主要处理展示的标签和保持光标在文字的最后。



图 5-7 光标处理代码

实现效果如下：

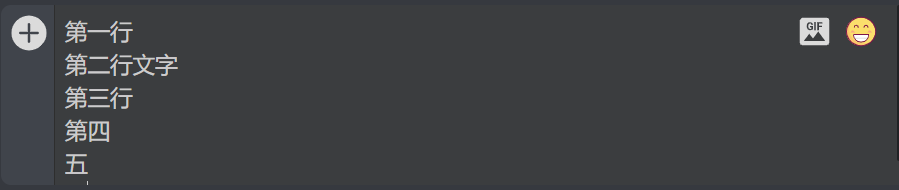


图 5-8 输入组件效果图

5.4模块功能的实现

5.4.1用户管理模块

用户注册：

用户进入注册页面后，输入电子邮箱地址，用户名和密码。向服务端API提交表单，若成功，则跳转到登录页面，否则提示相关错误信息。前端代码如下：



图 5-9 用户注册前端代码

服务端API根据用户输入的信息，判断电子邮箱和用户名是否被创建，若未被创建则创建新的用户并初始化用户的基本信息。用户注册的活动图如下：

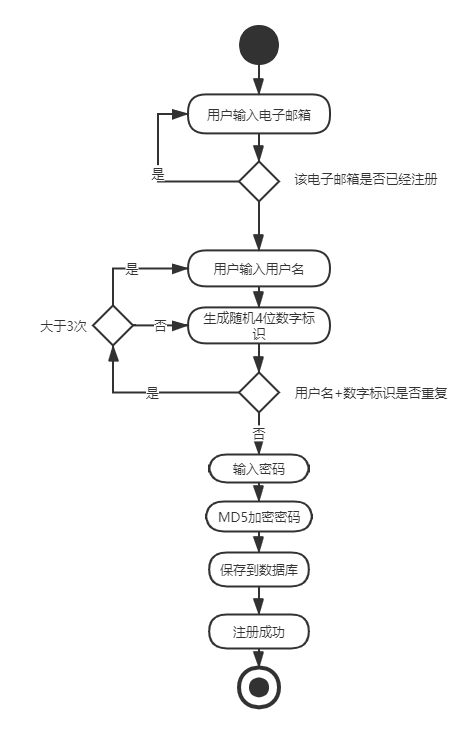


图 5-10 注册活动图

用户注册界面如下：



图 5-11 注册前端效果图

用户登录：

用户进入登录界面后，输入电子邮箱地址和密码，点击登录。前端请求后端登录API，若登录成功则跳转首页，否则提示后端返回来的错误信息。服务端登录API接收到请求，则用MD5算法对比用户密码。用户登录活动图如下：

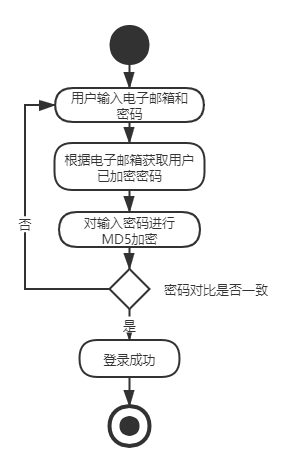


图 5-12 用户登录活动图

用户登录界面如下：



图 5-13 登录效果图

新增好友：

新增好友根据好友的email地址进行添加。用户在输入框输入email地址后，点击发送好友请求按钮。前端自动将输入字符串的多余空格删去。如果字符串为空则提示“输入不能为空”。若不为空则请求后端API接口。前端代码如下：



图 5-14 新增好友请求代码

后端API接口收到请求需要判断该email是否有存在的用户，以及该请求好友和用户的好友关系。新增好友主要活动图如下：

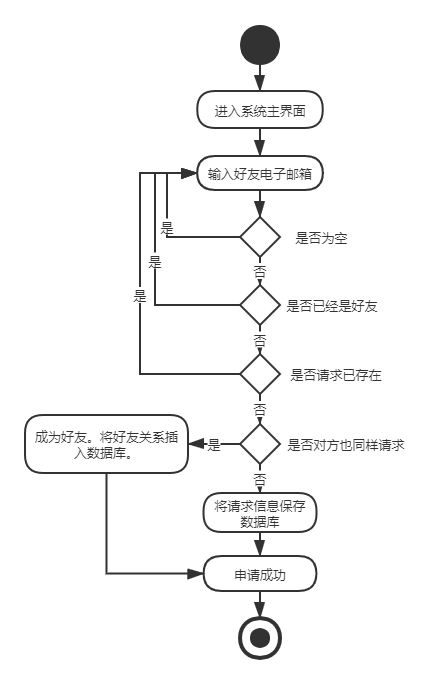


图 5-15新增好友活动图

新增好友页面如下：



图 5-16新增好友效果图

删除好友：

用户登录后，在好友列表点击更多出现下拉选项，点击删除好友。前端发送API请求。若服务器处理成功，则进行下一步，否则提示错误信息。删除成功后，前端使用vuex的commit函数，提交状态变更信息。在好友列表删除该好友，如果在聊天列表或在屏蔽列表存在该好友则删除。前端实现如下：

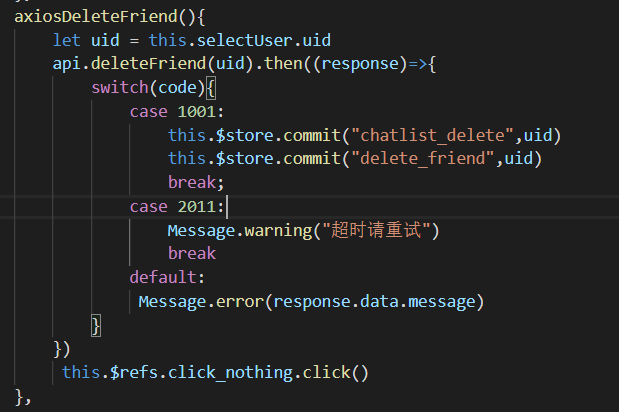


图 5-17删除好友请求代码



图 5-18删除好友前端处理代码

屏蔽好友：

屏蔽好友即将好友加入黑名单。用户登录后，在好友列表点击更多出现下拉选项，点击屏蔽好友。前端发送屏蔽好友请求。若服务器返回成功，则将好友从好友列表和聊天列表删除，并将该好友加入屏蔽列表。前端实现如下：

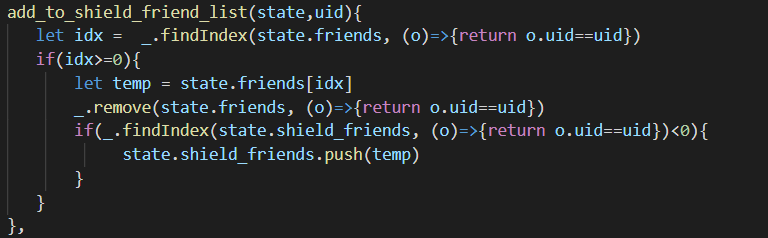


图 5-19屏蔽好友前端处理代码

个人资料设置：

用户登录进入系统后，可选择完善用户的基本信息。用户可变更用户名，手机号码性别所在城市等基本信息。前端页面如下：

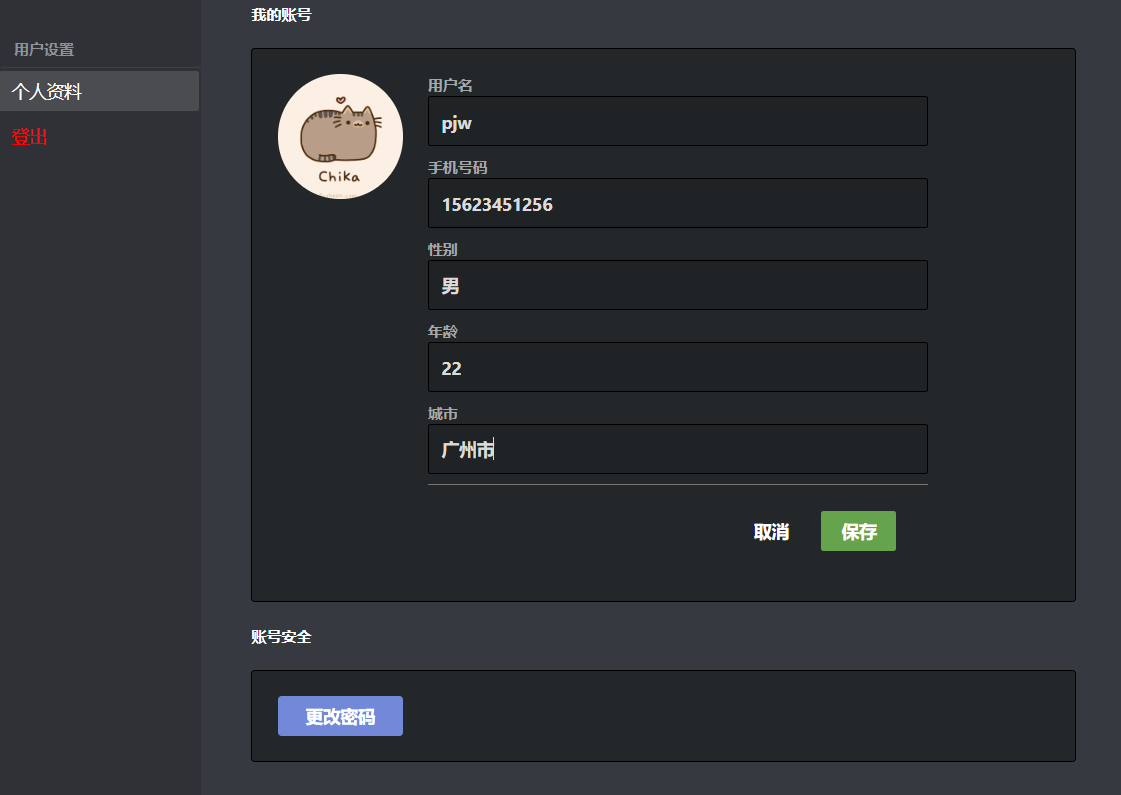


图 5-20个人资料效果图

5.4.2即时通信模块

初始化连接：

用户登录系统后，前端首先向服务端获取WebSocket连接URL，URL的格式为wss://<ip>:<port>/ws。并设置10次的超时重试请求，若10次后任然无法获取则向用户提示错误信息。获得WebSocket的连接URL后，前端新建WebSocket对象并设置相应的处理函数。

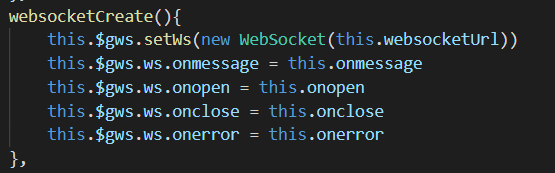


图 5-21websocket初始化代码

其中onclose和onerror函数被调用，若用户并没有登出，则进行断线重连，重新创建新的WebSocket对象。直至onopen函数被调用，说明WebSocket连接已经建立起来，可以进入主页操作。onmessage函数最为关键，服务器有新的消息推送，则会调用onmessage函数，函数内会根据消息的格式来对不同信息做不同的处理。

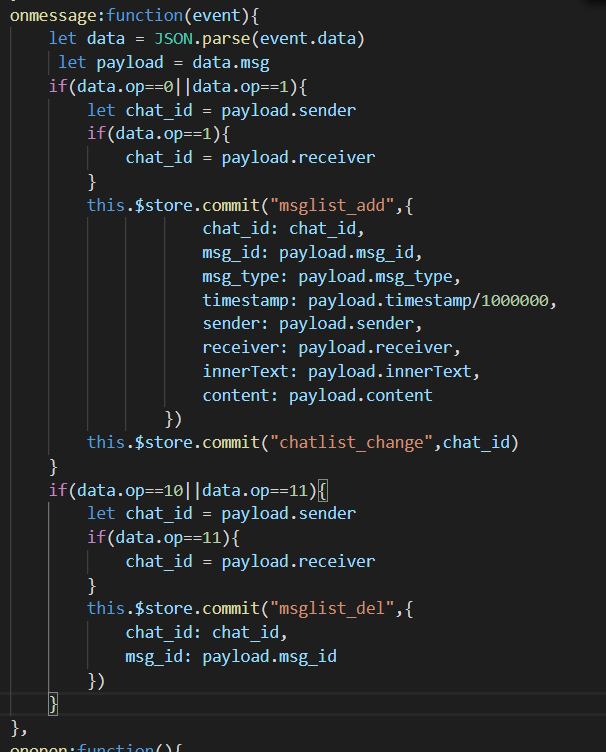


图 5-22 websocket接收消息函数

发送文字消息：

在公共输入框组件输入文字消息或者选择表情插入到文字中，按下回车，前端构造消息体，使用JSON序列化消息体并使用WebSocket的send函数发送给服务端。界面如下：

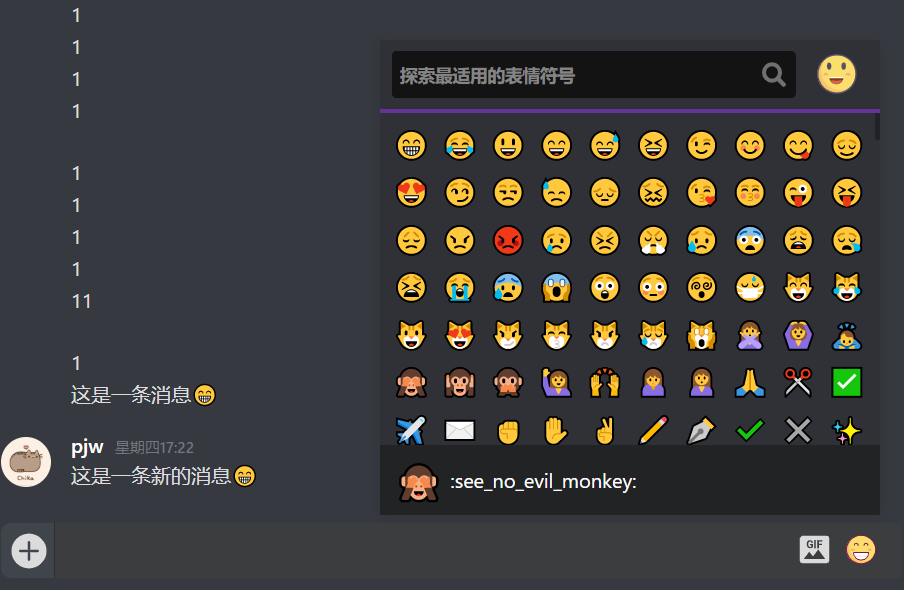


图 5-23发送文字消息效果图

前端根据该聊天是否属于群聊构建不同的消息类型。在本系统中，私聊的消息类型为0，群聊的消息类型为1。服务端则根据消息类型的不同而做不同的处理。op则表示消息类型，msg则表示消息体。消息的唯一id在前端生成，并设置消息体类型msg\_type为text，image或file其中一种。InnerText是一个字符串数据，表示发送的多行文字。

发送图片或文件消息：

当用户想发送图片或文件信息时，必须先上传文件之后才能发送信息。用户可以点击公共输入框左边的加号选择文件。每个文件都有文件名，文件大小和文件类型等信息。

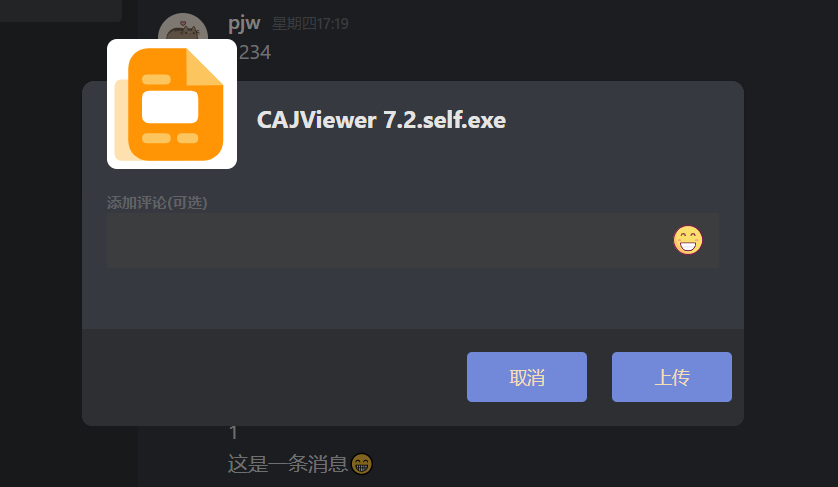


图 5-24文件上传效果图

如果上传的文件是图片则获取图片的宽度和高度。使用FileReader读取文件信息，构建Image对象，则可以获取图片的高度和宽度信息。并且在上传时显示图片的预览。实现如下：

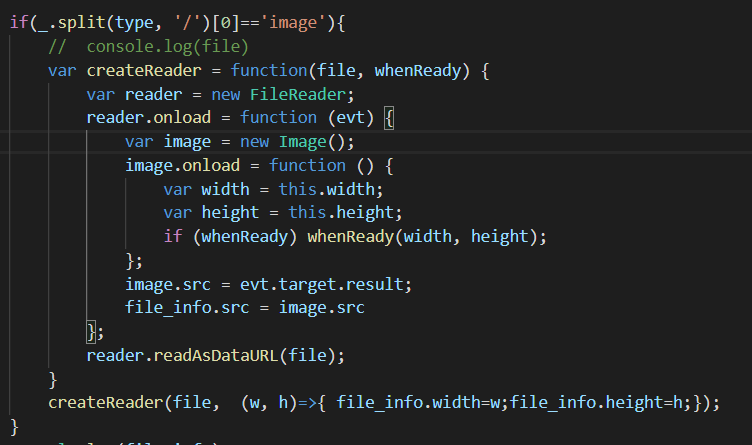


图 5-25文件上传获取文件信息

图片或文件上传的过程中，提供用户体验显示良好的进度条。onUploadProgress获取文件上传的数据量和文件大小，经过百分比计算可得出进度信息。

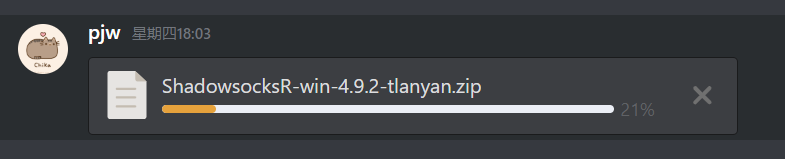


图 5-26进度条效果图

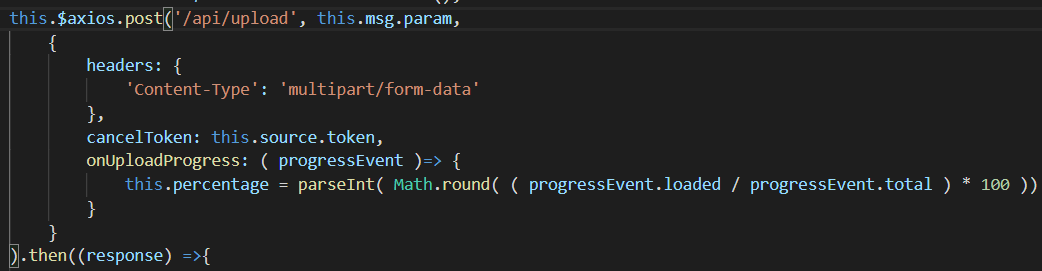


图 5-27进度数据处理

图片或文件上传成功后，服务端返回上传文件在服务器中的URL地址，并构建消息体。在msg消息体中的content则表示文件的信息，并设置msg\_type为image或file。content包含文件资源地址url,文件名称，文件大小，若是图片文件则还包含高度和宽度信息。

消息的展示：

消息的类型不同，展示也不同。消息组件根据不同的消息类型进行展示。文字消息最为简单，直接展示即可。若是图片消息，根据图片消息里的content属性，获取图片高度和宽度信息，并设置展示大小。若图片未加载完成之前，则展示同样宽高的背景颜色块表示图片即将加载。若是文件消息，则展示文件名和文件大小，并提供一个下载按钮给用户下载文件。

若同一个人连续发送多条消息，并且消息的时间间隔不超过一分钟，则除了第一条展示用户头像和用户名外，其余消息都不显示并且消息紧紧靠在一起，就像一次性发送多行信息一样。



图 5-28不同类型消息展现形式

5.4.3群组模块

创建群聊：

创建群聊点击聊天列表的加号弹出好友选择框，选择相应的好友，点击创建群组私信。如果只选择一位好友，则打开与该好友的私信窗口，若大于等于2个好友则新建群组。前端界面如下：

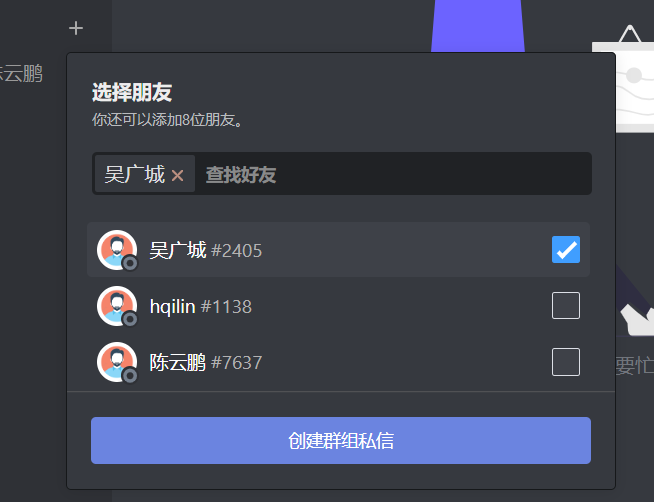


图 5-29创建群组效果图

前端实现如下：

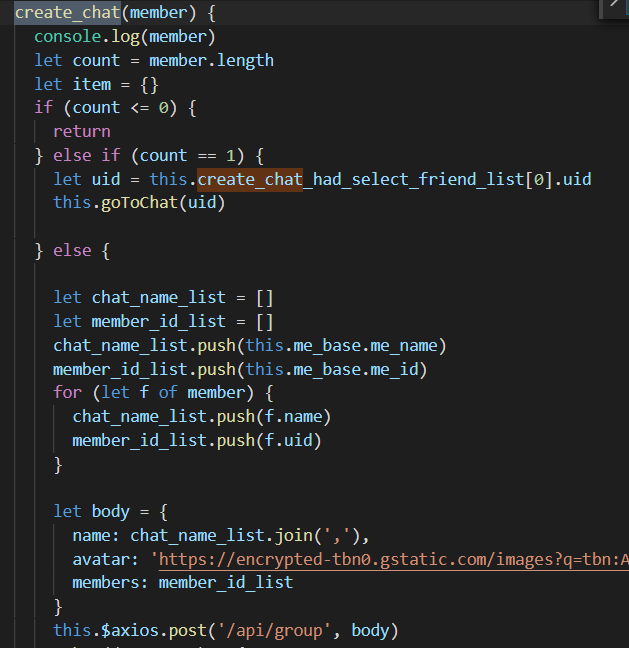


图 5-30创建群组前端实现代码

总结

传统的即时通信应用需要安装客户端且要经常更新的这一缺点在不断地放大而慢慢不能满足用户的需求。如果在别的电脑需要用到即时通信应用，则需要等待下载和安装，使用完后可能还需要卸载。基于Web的即时通信应用可以避免这些麻烦，无需下载，无需安装，打开网站即可使用，并且保留传统即时通信应用的基本功能。前端良好的页面设计，丰富的表情文字和友好的图片文件展示功能，提升用户的使用体验。

通过这次毕业设计实践，终于将这套系统实现落地，体会了软件工程设计中的关键步骤，从需求分析到软件设计，代码实现，再到最后的部署上线。由于时间的关系，实现了主要的功能，系统中还有许多用户体验优化的功能还没来得及实现，这个需要在后续继续开发。通过这一次实践，充分熟悉前端开发和后端开发，为自己的全栈开发之路打开崭新的大门。这次也充分认识到计算机基础知识的重要性。自己的软件的设计架构能力在这次实践中也得到了提升，让自己有开发一款应用的能力。

参考文献

[1]谢希仁.计算机网络（第六版)[M].北京:电子工业出版社,2013.

[2]李代立,陈榕. WebSocket在Web实时通信领域的研究[J]. 电脑知识与技术,2010,6(28):7923-7925+7935.

[3]龙奇. 下一代Web通信技术HTML5 Web Socket的研究[J]. 科技信息,2011(36):273.

[4]白帽子讲Web安全[M]. 电子工业出版社 , 吴翰清, 2012.

[5]赖晓京.基于Html5 WebSocket的即时通信系统[J].电子技术与软件工程,2013(17):61-62.

[6]吴晶.探讨网页设计中CSS技术的应用[J].科学技术创新,2019(35):67-68.

[7]江涛,谢世芳.Web前端开发技术HTML5与CSS3的融合及未来发展趋势[J].电脑编程技巧与维护,2019(04):170-172.

[8]威廉·肯尼迪 (William Kennedy). Go语言实战（第六版)[M].人民邮电出版社,2017.

[9]Sau Sheong Chang.Go Web Programming[M].Manning,2015:325.

[10]Jingyang Wang. The Design of Instant Messaging System Based on Web[C]. The Institute of Electrical and Electronics Engineers、IEEE Beijing Section.Proceedings of 2017 IEEE 8th International Conference on Software Engineering and Service Science.The Institute of Electrical and Electronics Engineers、IEEE Beijing Section:IEEE BEIJING SECTION(跨国电气电子工程师学会北京分会),2017:308-311.

[11]Yash Gupta,Himanshu Dewan,Alka Leekha. Real-time monitoring using AJAX and WebSockets[J]. Journal of Statistics and Management Systems,2020,23(1).

[12]CHATTERJEE S,ABHICHANDANIT,LIH, et al. InstantMessaging and Presence Technologies for College Campuses[J]. IEEE Network . 2005.

[13]Paul DuBois.MySQL Cookbook[M].O'Reilly Media,2007.

[14]Baron Schwartz, Vadim Tkachenko, Peter Zaitsev, Derek J. Balling.High Performance MySQL: Optimization, Backups, and Replication[M].O'Reilly Media,2008.

[15]Andrew Lombardi.WebSocket: Lightweight Client-Server Communications[M].O'Reilly Media,2015.

[16]Adrian Mouat.Using Docker: Developing and Deploying Software with Containers[M].O'Reilly Media,2015.

致谢

大学四年即将过去，毕业论文的结束就意味着大学的学习生活画上了句号。精心回顾在本科阶段四年的经历，心中思绪万千。特别是在大学的最后一学期，感受到了大学生活的离去，心中如此怀念当时的点点滴滴。作为一个即将踏入社会的年轻人，我感到迷茫又仿徨，该如何面对社会的风吹雨打？或许自己会给自己答案：有家人，朋友，同学和老师的支持，自己可以勇敢地起航。

首先，我要感谢母校广州大学对我的培养！这个承载了我人生最美好，最拼搏，最有活力的本科大学。在此后的道路我会秉承着“博学笃行，与时俱进”的校训，不断拼搏，不断前进。

其次，我要感谢我的指导老师金政哲老师对我的指导和培养。我记得最开始CST选拔的时候，金老师的高要求促使我不断突破自己，让我明白努力与坚持过后一定会有成果。金老师总会让我们从实践中去学习，在做项目的过程中提供实现思路，这样不断地锻炼我们的独立思考能力和解决问题的能力。

感谢CST的小伙伴们，我们共同构建一个良好的学习环境，在一起合作，学习和成长，这段经历我会一直记得。当我看到每个人都在努力拼搏时，我知道我也不能落下，不断跟上你们的脚步。

感谢我可爱的舍友们，在四年的本科生涯里，陪我嬉笑怒骂，插科打诨，精彩地度过每一天。谢谢你们在我情绪低落时，给予我安慰和理解。谢谢你们给我留下精彩丰富的成长之旅。

最后，感谢父母给我一个温暖热情美好的学习成长环境。谢谢你们给我一个坚实的后盾，让我有勇气去追求美好的事物。

希望将来，我们都能遇见更好的我们！