

软件工程团队项目：“归雁”
家庭聚会策划及共享平台
《交付文档》



清华大学软件学院

2021 年 12 月

队长：陈彦扬 2019080117

成员：霍建宇 2019080112

白承桓 2017080338

许霆康 2019080126

魏乐 2019080124

目录

1. 引言	3
1.1 编写目的	3
1.2 项目背景	3
1.3 参考文献	3
2. 产品目标	4
2.1 产品概述	4
2.2 功能目标	4
2.3 性能目标	5
3. 开发组织管理	6
3.1 过程管理	6
3.2 人员分工	6
3.3 开发环境	7
4. 系统设计	7
4.1 前端交互	7
4.2 后端模块	14
4.3 接口规范	15
4.4 数据库设计	16
5. 重点和难点问题及其解决方法	18
5.1 重点和难点汇总	18
5.2 解决方法	18
6. 测试总结	19
6.1 功能测试	19
6.2 性能测试	20
7. 系统部署	21
8. 交付产品	21

1. 引言

1.1 编写目的

本交付文档为“归雁”家庭聚会策划及共享平台项目的交付文档，目的在于总结开发产品的目标、管理、设计、测试等。文档读者对象包括用户和项目负责人（助教和老师）。

1.2 项目背景

“我们这一家”，“幸福的一家”等等家庭微信群在近几年来似乎已经成了每个家庭必备的沟通与联系平台，然而这类微信群在某些情况下的功能受限，比如说当一个家庭成员在群内分享出行照片和影片时，很快就被会其他的信息淹没，让其他家庭成员要找回这些照片慢慢欣赏时略为麻烦。另外，当要策划家庭聚会时，各个家庭成员之间的时间谈不拢，也往往成为一个麻烦之处。因此，我们想开发一款集合聚会策划与照片及影片共享功能的平台，解决这些微信群不能解决的问题。

1.3 参考文献

1. day-span vuetify

<https://github.com/ClickerMonkey/dayspan-vuetify?ref=madewithvuejs.com>

2. 产品目标

2.1 产品概述

我们所开发的产品名为“归雁”，是一个让人数较大且分散的家庭互相分享生活的平台。这个平台功能主要可分为三大类，即“聚会策划”、“共享”与“家谱”。

- “聚会策划”功能可让一个私密家庭群组的成员发起聚会计划，各家庭成员可提名聚会时间与地点，并一起投票决定聚会细节。投票完成后，聚会细节会被记录在群组日历种。同时，当家庭成员间无法确定聚会细节（如地点、活动内容等）时，平台也提供了“随机转盘”功能帮助家庭成员决定聚会细节。
- “共享”功能主要是让私密家庭群组的成员分享生活中的照片。其他家庭成员可在“照片墙”上看到分享的内容，并可点赞和评论。
- “家谱”功能让群组成员建立和修改家谱结构，并保存到数据库中。

2.2 功能目标

1. 注册/登录：用户可以注册账户和通过用户名登录账户。
2. 家庭群组：用户可创建私密家庭群组，并通过邀请码邀请其他用户加入群组种。
3. 群组管理：群主可赋予/移除其他成员的群主权限，也可移除成员、更改群名和删除群组。
4. 家谱：群组成员可建立家谱，将群组成员添加到家谱中或添加自定义成员，也可删除家谱中成员，修改后的家谱可保存到数据库中，且可以在之后提取作修改。
5. 聚会管理：群组成员可发起新聚会提议/投票（单选/多选）。提议期间，群组成员可添加新聚会提议。达到提议截止时间后，系统会自动将该提议转为投票。投票期间，群组成员可对其进行投票。达到投票截止时间后，系统会自动将最高票选项添加到群组日历和各个成员个人日历中。

6. 日历管理：当成员加入群组时，其生日会自动更新到该群组日历和该群组所有成员个人日历中。经过投票阶段的聚会也会自动更新到日历当中。群组成员可对日历进行修改，并保存到数据库中。同时，也提供了个人日历，用户可对自己的个人日历进行修改，并保存到数据库中。
7. 照片墙：群组成员可将图片/动图上传到照片墙中，该群组所有成员都能看到上传到该群组的图片/动图，并可对其进行点赞或评论。
8. 个人信息：用户可点击“个人信息”浏览自己的个人信息（邮箱、手机号码、生日日期和标语），并可对手机号码和标语进行修改。同时，群组成员也可在群组管理页面通过点击“详情”浏览群组中各个成员的个人信息。
9. 随机转盘：用户可输入 1-8 个选项，点击“转”后，随机转盘会旋转并给出结果。

2.3 性能目标

1. 所支持客户端数：~100
2. 所支持并发用户数：~100
3. 文件与记录规模：~1GB
4. 操作响应时间：< 5s
5. 可用性：开发完成时间至 2022 年 1 月（服务器租用期限）
6. 安全性：未登录用户或非群组成员不能访问该群组任何内容
7. 易用性：对用户友善，会浏览网页的用户就能上手

项目开发初期的需求与目标可见附件“需求规格说明书”

3. 开发组织管理

3.1 过程管理

- 每周四晚上通过 Discord 举行一次线上例会，例会议程为检查各成员开发进度、确定下周开发目标、整合当个星期各成员代码、前后端接口对接、问题讨论与解决及任务分工。
- 每周 1-2 次通过 Discord 和 Team Viewer 集体编程，2-4 位成员参与，一同开发及解决问题，个别成员作息时间、时间安排不同，独立开发小部分。
- Github 上建立一个主分支（main）和每个成员一个分支，主分支为每个阶段所整合的代码，至少一周更新一次。
- 利用 Github 上 Issues 功能，列出各个功能模块并分配任务，也将自己在每个功能模块中遇到的 bug 整合到该模块下，由模块负责人及时修复。
- 前后端接口文档在项目初期时通过 Google Docs 进行编写及阅读，之后经过助教提议后改到 RunAPI/ShowDoc 上进行，RunAPI 结合了接口测试与接口文档功能，可自行生成接口文档。
- 其余文档与 PPT 的编写通过 Google Docs 和 Google Slides，所有成员一同在线完成。
- 通过微信和 Discord 进行线上沟通，Discord 里还用以记录部分重要信息如会议纪要等。
- 团队会议纪要

团队会议纪要

标号	时间	地点	参加人员	讨论事项	人员分工	会议成果
1	10.14 22:00-0:00	线上	全员	产品名称、竞品分析及产品定义与故事	一起讨论和书写文档	完成“需求调研与用户故事”文档，想了几个项目中可以增

						加的功能和确定产品名称
2	10.28 22:00-0:00	线上	陈彦扬、霍建宇、白承桓、许霆康	交互界面设计	一起讨论、设计页面、书写文档	完成中期报告 PPT 初稿，具体化页面排版与进一步探讨功能实现方式，进行增删
3	11.03 21:00-23:00	线上	全员	期中展示 PPT	一起讨论、设计 ppt 页面	完成中期报告 PPT
4	11.11 20:00-20:30	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
5	11.18 20:00-22:30	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
6	11.25 21:00-23:30	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
7	12.02 21:45-5:00	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
8	12.9 21:00-4:10	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
9	12.16 21:30	线上	全员	下次迭代目标	讨论迭代目标及分工	确定迭代目标及分工
10	12.22 20:00	线上	全员	代码整	一起知足	完成答辩

				合，制作答辩 PPT，录制展示视频	答辩 PPT，录制展示视频	PPT 和展示视频
11	12.23 20:00	线上	全员	最终测试，编写测试报告	完成测试，编写测试报告	完成所有测试，修复 bug，完成测试报告

3.2 人员分工


- 陈彦扬：群组目录、群组管理、聚会管理、家谱、个人信息页面美化、照片墙
- 霍建宇：后端+数据库、日历（前端）、照片墙基本架构（前端）
- 白承桓：群组目录、群组管理、聚会管理、家谱
- 许霆康：随机转盘基本架构、个人信息页面基本架构、照片墙
- 魏乐：登入页面美化、日历美化、随机转盘美化

3.3 开发环境

- 开发语言：后端（Python 3.8），前端（Vue）
- IDE：PyCharm
- 数据库：MySQL
- 部署工具：Docker

4. 系统设计

4.1 前端交互



登录


用户

密码

登录

[还未注册? 注册](#)

登录页面：用户输入用户密码进行登录



注册

用户名

密码

邮箱

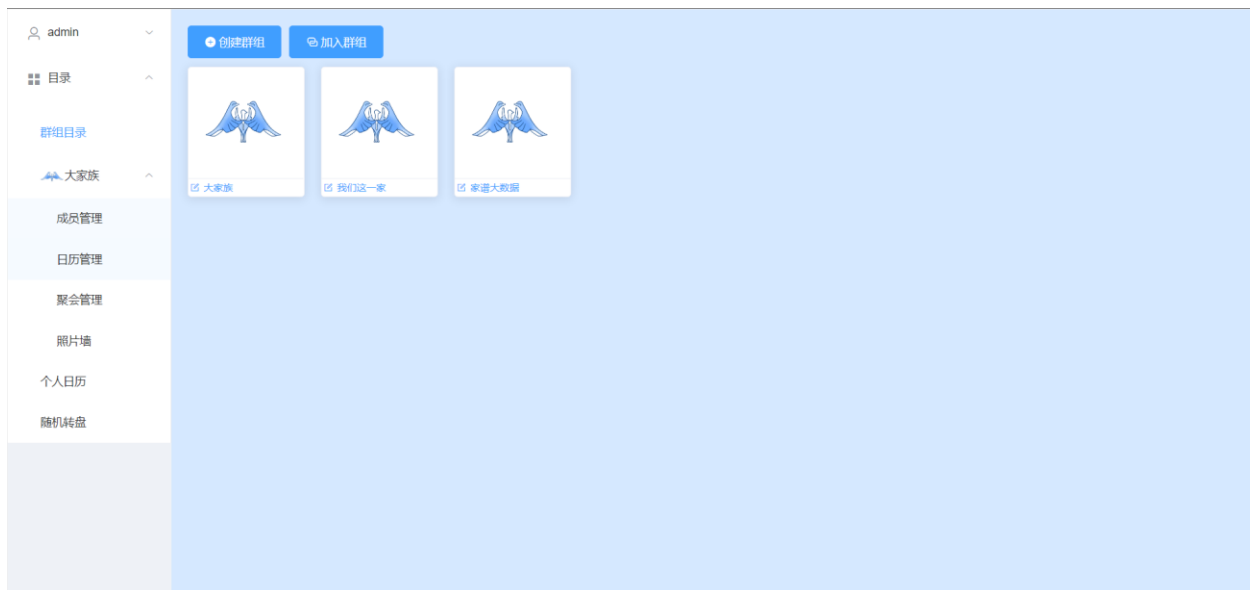
手机号码

生日日期

注册

[已有账户? 登录](#)

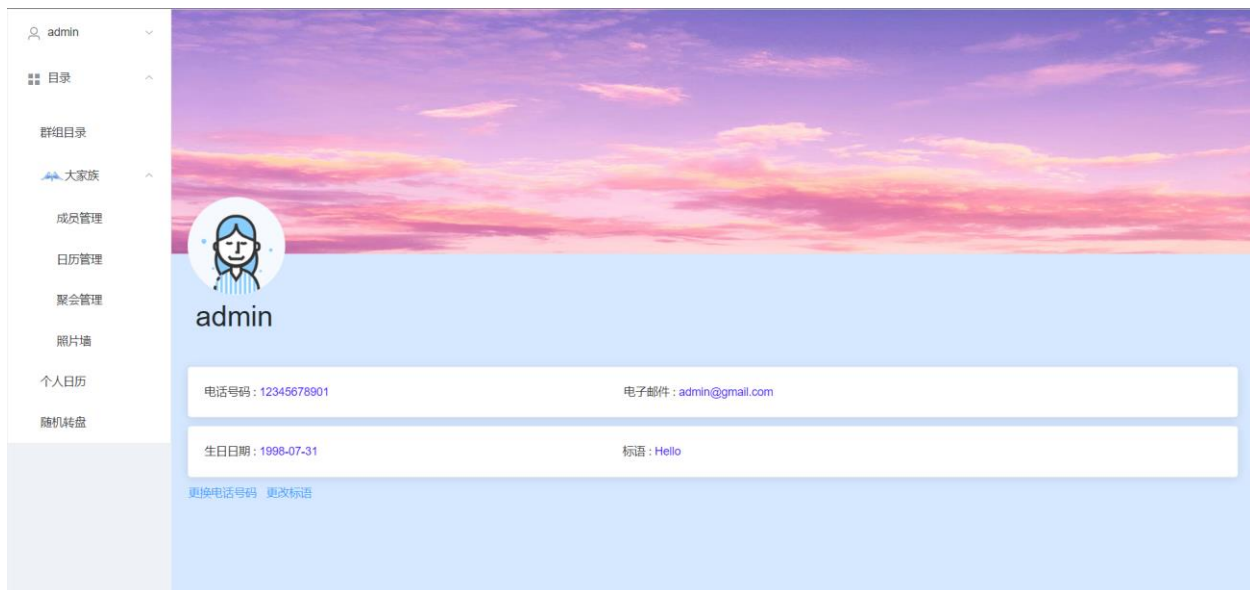
注册页面：用户输入合法的个人信息进行注册



群组目录页面： 点击群卡片进入组，也可以点击“创建群组”“加入群组”进行对应的操作。



成员管理页面： 左侧显示组员信息，右侧显示并可以编辑家谱，右下侧可以获取邀请码或退出群，具有群主权限的用户可以移除或赋予群主权限给成员，也可以删除群。可点击群名旁边按钮修改群名。家谱可编辑。

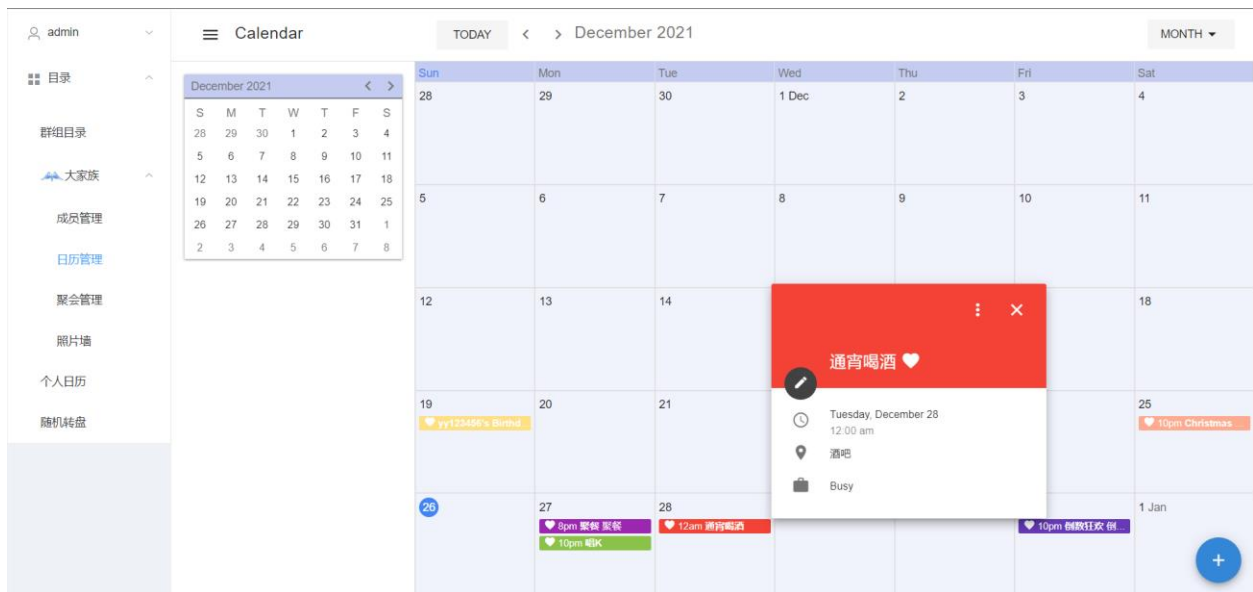


个人信息页面：可以查看用户个人信息也可以修改电话号码和标语。

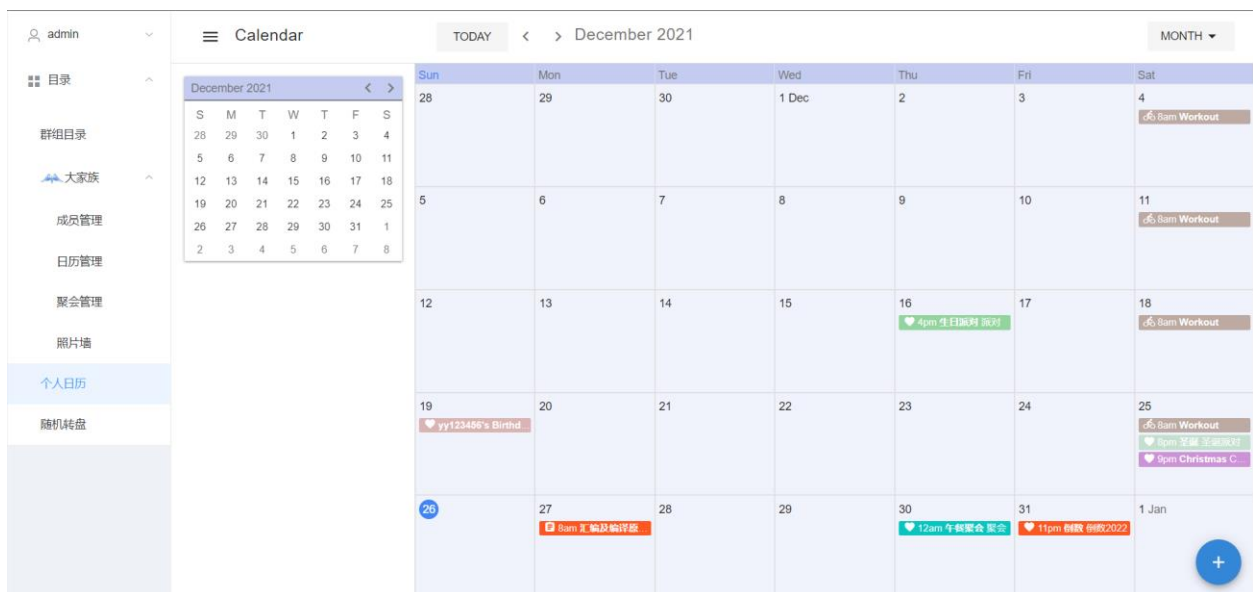


聚会管理页面：用户可以发起聚会，类型可以是投票（单选，多选）或提议，也可以编辑自己发出的聚会（修改截止日期或删除聚会）。可用文本搜索功能搜索聚会名，也可以标签 filter 功能区分投票与提议。



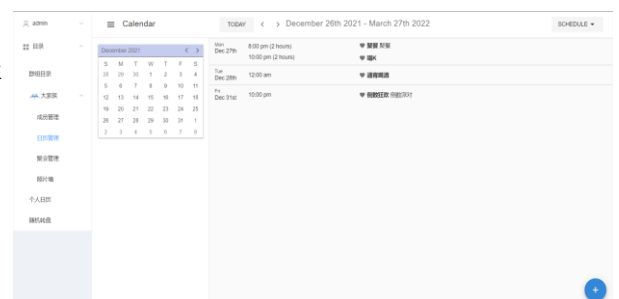


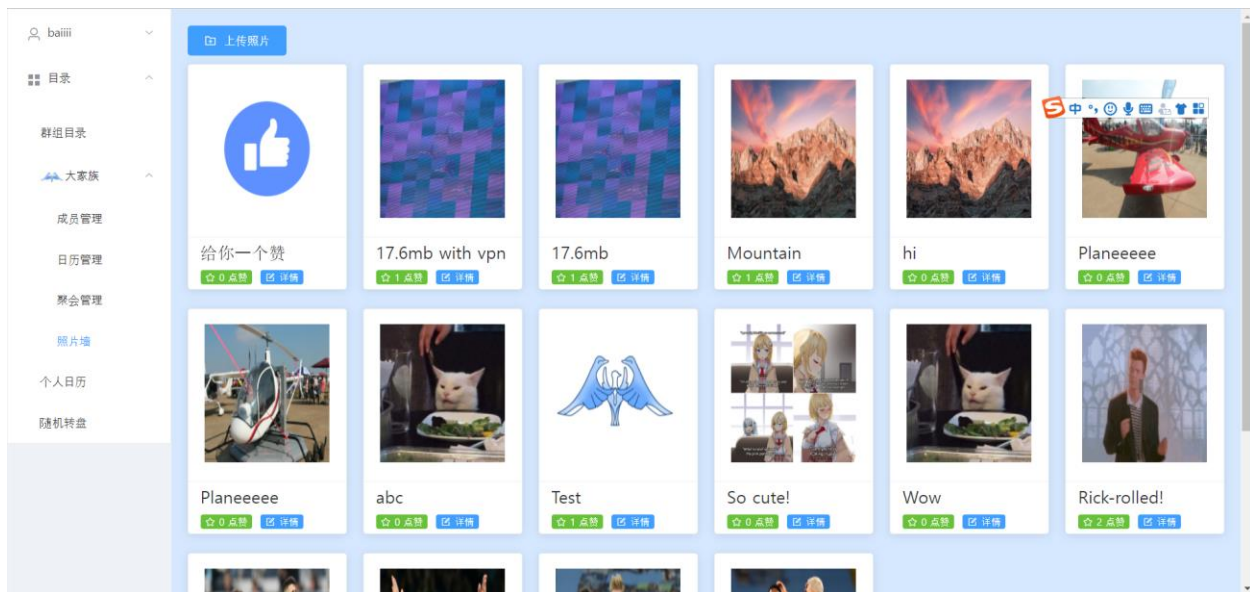
日历管理页面：用户可以通过日历直观的查看给群组的聚会日程（聚会管理中发起的聚会结果会显示到此处），也可以通过点击日期对应的日历栏增删行程



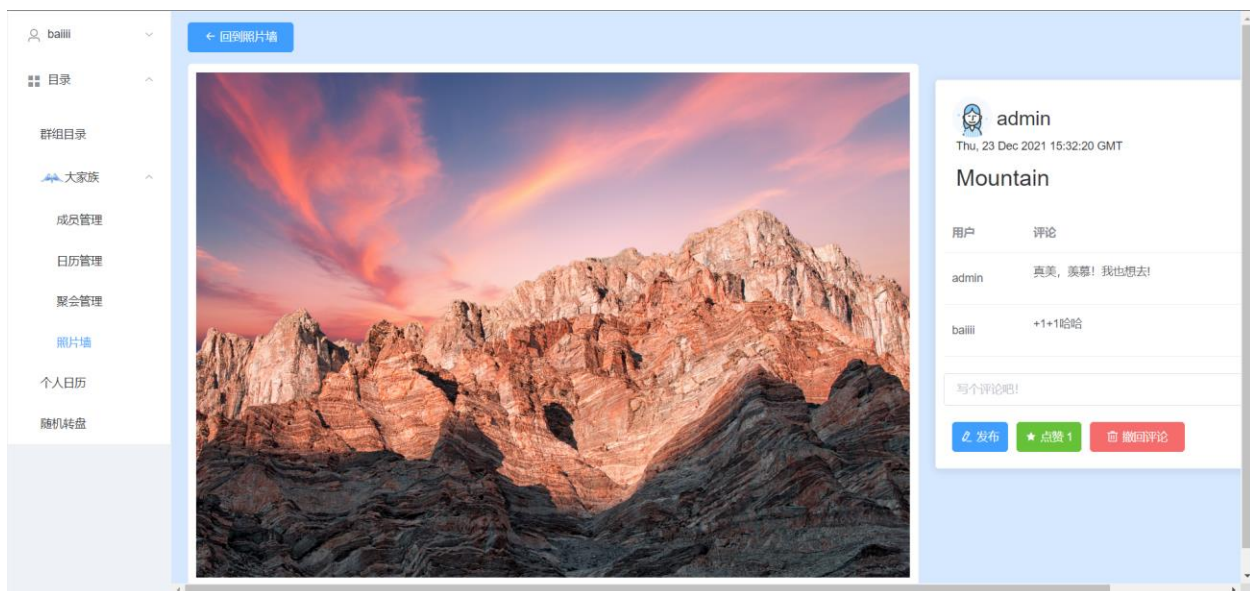
个人日历页面：用户可以通过日历直观的查看给个人的聚会日程，通过点击日期对应的日历栏可以增删行程，右上角切换到 schedule 页面。

日历 Schedule 页面：用户可以一眼查看近期的聚会日程

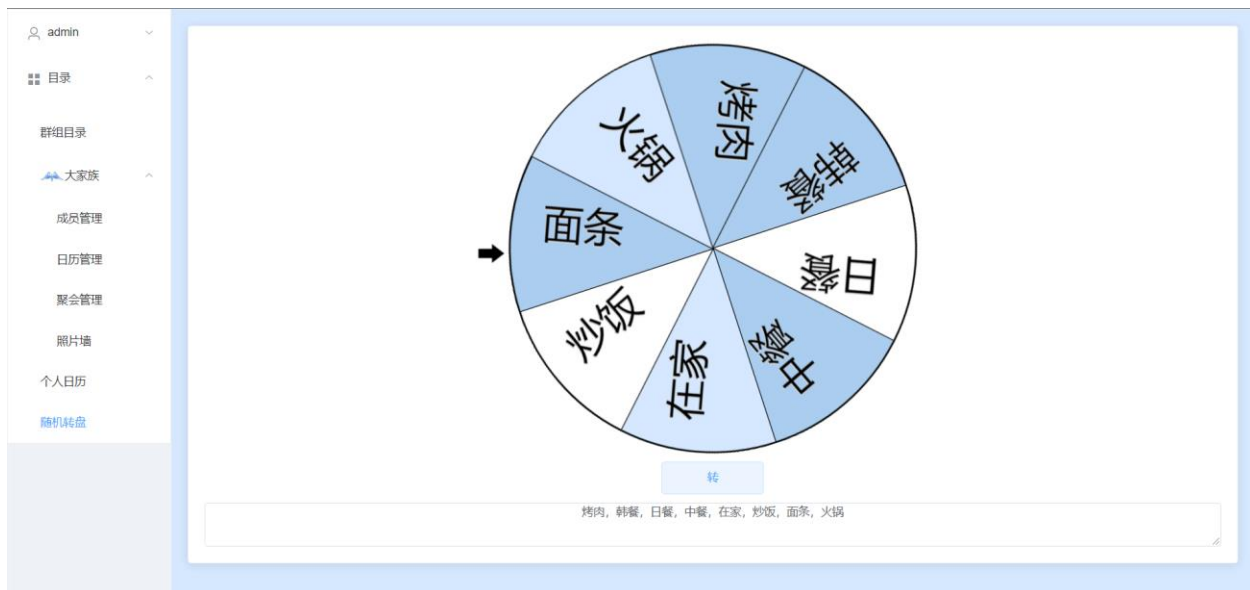




照片墙页面： 点击上传图片发布图片，可浏览已发布照片，可直接点击图片大图浏览，可进行点赞，也可点击详情，进入以下页面。



照片墙详情页面： 可浏览照片、照片发布信息以及评论，可直接点击图片大图浏览，可进行点赞，也可发布评论，也可删除自己已经发布的评论。



随机轮盘页面：可以输入想要随机的东西，用空格、逗号或换行隔开，点击转。

4.2 后端模块

用了 Flask 框架和 flask_sqlalchemy 处理数据库内容。

4.2.1 模块划分

主要可划分为 5 个模块，分别为 auth（注册、登录相关），group（群组相关），gathering（聚会相关），photowall（照片墙相关）和 genealogy（家谱相关）。各个模块中实现与该模块相关的接口。utils 则包含了所有与接口无关的函数。

4.2.2 数据库类

用了 flask_sqlalchemy 将数据库中各个表以类的方式调用，models 中定义了各个类。

4.2.3 安全相关

用户密码通过单向加密方法进行加密。

用户注册时，其密码被传输到后端后用哈希加密之后才会保存到数据库中。用户登录时，所输入的密码将在后端用同样的加密方法加密之后再和数据库中密码比对。这样一来，确保了即使得到了加密后的密码也不能够解密得到明文。

同时，还用了 Token 验证机制，用户登录时，后端将通过 jwt 将用户 user_id 和一个时限（一天）加密生成 token 返回到前端保存到 localStorage 中。之后每次调用接口时都需将该 token 放入 header 中传至后端用 jwt 解密，若解密成功得到 user_id 或 token 尚未过期则验证成功，正常调用接口，否则直接返回错误信息。若想直接通过 URL 绕过登录页面进入网页，也会返回错误并直接跳转回到登录页面。

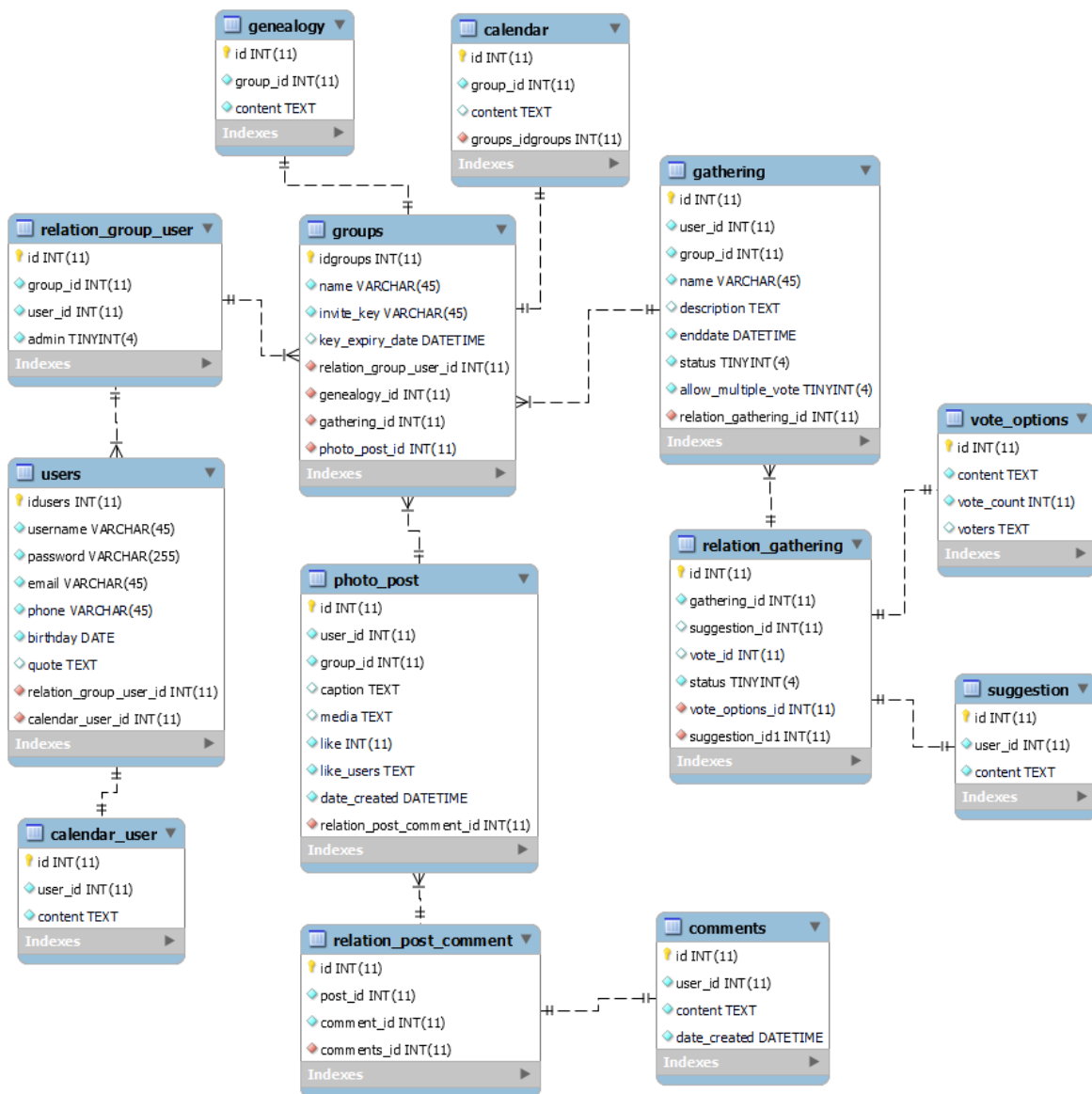
4.3 接口规范

大部分接口都用 POST 的方式将参数以 JSON 形式传输，只有获取图片的接口用 GET 的方式获取。

详细的前后端接口描述可见“前后端接口文

档” (<https://www.showdoc.com.cn/1746664993548682/8187442428186379>)

4.4 数据库设计



- 用户相关: users
- 群组相关: groups, relation_group_user
- 聚会相关: gathering, suggestion, vote_options, relation_gathering
- 日历相关: calendar, calendar_user
- 照片墙相关: photo_post, comments, relation_post_comment
- 家谱相关: genealogy

特点:

- 群组相关数据库定义了 relation_group_user 用以记录 user-group 的对应关系
- 聚会相关数据库定义了 relation_gathering 用以记录 gathering-suggestion 和 gathering-vote_options 的对应关系
- 照片墙相关数据库定义了 relation_post_comment 用以记录 photo_post-comment 的对应关系

5. 重点和难点问题及其解决方法

5.1 重点和难点汇总

5.1.1 后端

1. 自动将聚会提议更新为投票，投票更新到日历中
2. 图片保存
3. 投票/生日自动保存到日历中

5.1.2 前端

1. 寻找合适的家谱库以及其数据结构的了解及删添节点功能的实现
2. 版本兼容等外部库安装问题
3. 组件间变量传递
4. 布局美化

5.2 解决方法

5.2.1 后端

1. 用 flask_apscheduler 设置一个定时器，每隔一个小时检查是否已达到各个聚会截止日期，若是则将该提议转为投票或处理投票保存到日历中。之所以选择每隔一个小时是因为，大多数截止日期都会设置为整点，因此设置为一个小时能够在不延迟太多的情况下处理以上操作。
2. 关于如何保存图片信息，我们做了一些调研，发现直接将整张图片的二进制信息保存到数据库中是不合理的。因此最终选择的策略是上传时将整张图片传到后端，后端将随机给图片生成一个文件名，保存在后端服务器中（按群组划分），并在数据库中保存该随机生成的文件名。加载图片时则通过 GET 方式获取，通过 URL（/show/<int:group_id>/<string:filename>）获取该群组的该图片。

3. 日历前端我们使用了 `dayspan-vuetify` 组件，前期对该组件保存日历的方法不太熟悉，导致在日历部分卡了一段时间。最后经过阅读该组件代码，知道了它的保存方式，在后端模仿该前端保存的方法生成 `events` 后加入到日历中。

5.2.2 前端

1. 每次保存当前选中的节点的层与父节点层，增加新节点则在选中的当前节点层上 `push` 一个新的节点，删除则找到父节点层，找到其中选中的子树，进行数组的 `splice` 操作完全删除其空间（`delete` 并不会完全删除，导致树会保存一个数据为 `undefined` 的节点）
2. 不断进行查找新的库，或者手动修改 `package.json` 中的版本安装兼容当前 `vue2.x` 的库。
3. 使用在 `url` 中写入变量，通过 `router` 转发变量值的方法。或者如果组件间有父子的关系，父组件传给子组件使用 `props` 传递变量，子组件传给父组件使用 `emit` 事件函数传递变量。

6. 测试总结

6.1 功能测试

我们进行的功能测试主要可分为单元测试和对前端的功能测试两部分。

其中单元测试仅针对用户需要手动输入参数的接口，对长度限制的参数进行边界值测试，对非法字符限制的参数进行等价类划分测试。

测试结果：全部通过

前端的功能测试则采用了全局探索式测试将项目的几个模块区分为破旧区、旅馆区、娱乐区、商业区、历史区和旅游区，针对不同区使用不同的测试策略。

- i. 破旧区：本项目基本无破旧区

- ii. 旅馆区：本项目中唯一一个休息时仍在运行的功能是聚会达到截止时间时自动转换为投票或添加到日历当中
 - a. 采用懒汉测试法：对于聚会提议->投票，发起日期和地点都为空的聚会，修改数据库中截止日期，强制系统更新该聚会；对于聚会投票->日历，选择各选项都没有投票数的聚会投票，修改数据库中截止日期，强制系统更新该聚会
- iii. 娱乐区：本项目中的“个人信息”和“随机转盘”功能可被视为娱乐区
 - a. 针对“个人信息”页面，更改标语时不输入任何标语，更改电话号码时输入几个不合法的和合法的电话号码；针对“随机转盘”，输入超长的选项
- iv. 商业区：群组、家谱、聚会和照片墙
 - a. 采用地标测试法：提前确定关键特性（地标），逐一执行，直到访问了列表中所有地标
- v. 历史区：本项目没有历史区
- vi. 旅游区：日历
 - a. 采用超模测试法：反复操作，观察页面是否正确被绘制出来

测试结果：三项测试失败，但都已成功解决

6.2 性能测试

对于性能测试，我们实行了压力测试和大数据量测试。

针对压力测试，我们通过工具模拟多个并发用户（50，100，500）从后端/数据库获取信息、向后端/数据库保存信息和从后端获取图片。

针对大数据量测试，我们试图构造大型家谱，测试其保存和输出的能力；我们也试图上传大小达到 17.6MB 的图片，测试其上传和传输的用时。

测试结果:

压力测试方面查询和修改在 100 个并发用户时响应时间都达到我们的性能目标($< 5s$)。图片传输则较慢些。由于压力测试在中国境外(韩国)进行,由于防火墙等因素可能会有所影响,但在 100 个并发用户的情景中都成功达到我们最开始预估的相应时间($\leq 5s$)。至于图片传输较慢的情况,我们猜测这可能也和我们在境外进行测试有关。

大数据量测试方面,大型家谱需要几秒来加载和渲染,但在普遍情况下延时仍在可接受范围内。至于图片上传和下载,图片上传在中国境内测试时非常快,加载则用时长一些。

安全性和易用性都成功达到目标。

详细测试结果和分析可见附件“测试报告”

7. 系统部署

数据库,前端和后端都通过 Docker 部署,其中数据库占用了 8888 号端口,前端占用 8024 号端口,后端占用 8000 号端口。前后端部署通过 Dockerfile 实现,前端只需要运行

```
docker build -t family_client .
```

```
docker run -p 8024:8024 -d family_client
```

后端除了需要运行

```
docker build -t family_server .
```

docker run -p 8000:8000 -d family_server 之外,如果不是第一次上线还需要在将上一个 container 撤下之前,先从上一个 container 拷贝用户上传的所有图片到本地,再在上线后将这些图片拷贝到新的 container,具体通过这两个命令实现:

```
docker cp /home/ubuntu/upload/. container_id:/tmp/upload
```

```
docker cp container_id:/tmp/upload/. /home/ubuntu/upload
```

8. 交付产品

服务器 IP: 82.156.102.67 (账号: ubuntu, 密码: hjy@20210922)

项目网址: 82.156.102.67:8024

可用账号: admin

密码: 123456