《数字工程软件架构》
课程设计报告

学　　院:　　遥感信息工程学院

班　　级:　　20F10

小组名称:　　大家都队

组长学号:　　2020302131229

组长姓名:　　黄晓轩

组员学号:　　2020302131249

组员姓名:　　马文卓

实习地点:　　遥感楼 201

2022 年 **4** 月 **22** 日

目录

一、功能需求及环境限制分析.....3
（一）功能需求分析.....3
（二）环境限制分析.....4
二、 架构设计、架构风格分析与选择、设计模式应用.....5
（一） 整体架构设计图（高清图见附件：IShare@ICare.png）.....5
（二） 架构风格分析与选择.....6
1. 前端风格分析及优缺点分析.....6
2. 后端风格分析及优缺点分析.....7
（三） 设计模式应用.....8
1. 创建型模式——简单工厂模式.....8
2. 结构型模式——适配器模式.....9
3. 行为型模式——观察者模式.....10
三、 应用软件开发实现.....13
（一）云开发部分.....14
（二） 登录部分.....15
（三） 首页部分.....16
（四） 论坛部分.....17
（五） 地图部分.....18
（六） 我的部分.....19
四、 任务分工.....22

一、功能需求及环境限制分析

（一）功能需求分析

iShare@iCare 需要满足的功能需求如下图：

选题 1：iShare@iCare移动应用

基本需求

地理信息平台支持（地图查询、显示、路径计算、导航）

求助者/“爆料者”

拍照后上传照片、拍照位置、事件类型、基本描述信息

可随时查询自己事件处理进展

可在地图上按类别查询显示周边或兴趣区域发生的事件及处理进展

志愿者

浏览平台发布的各类事件

地图查询并导航至事件发生位置

以照片或视频形式修改任务处理进展状态

系统，事件完结后发信息给求助者/“爆料者”

支持讨论，提供处理线索，支招或转专业部门处理(用@专业部门模拟)

我们团队在自主分析并结合上述功能需求的基础上，加附了一些额外的功能，简化合并了一些功能；当然这些都是在最大化用户使用体验的基础上进行的。下面我们将用**面向对象的功能分析法**来分析我们开发的软件所具备的功能。

1、首先对于基类——也就是每一个使用 iShare@iCare 的用户，需要为所有用户都提供：

(1) 可以浏览所有求助者/”爆料者”发布的事件消息；并能够按照**事件类型**查询显示对应的事件消息；这里我们将在地图上显示周边事件的功能分为了两步实现：首先查询到对应的事件消息，再进入消息详情页面，点击所给出的地址链接即可自动跳转至地图服务栏；

(2) 可以在个人信息页面修改自己的个人信息；比如：真实姓名、联系电话、居住城市等，我们认为这些信息有助于贴合 iShare@iCare 的主题——因为需要志愿者帮助到求助者，因此所有人都尽可能提供真实信息，可以方便这一进程；除此之外，我们还设计了个性签名和个性背景供用户选择；

(3) 可以访问地图相关的服务；包括：导航、路线规划及查找附近建筑等服务

(4) 可以咨询客服及相关部门(目前只由我们开发者接收这部分消息并给予回

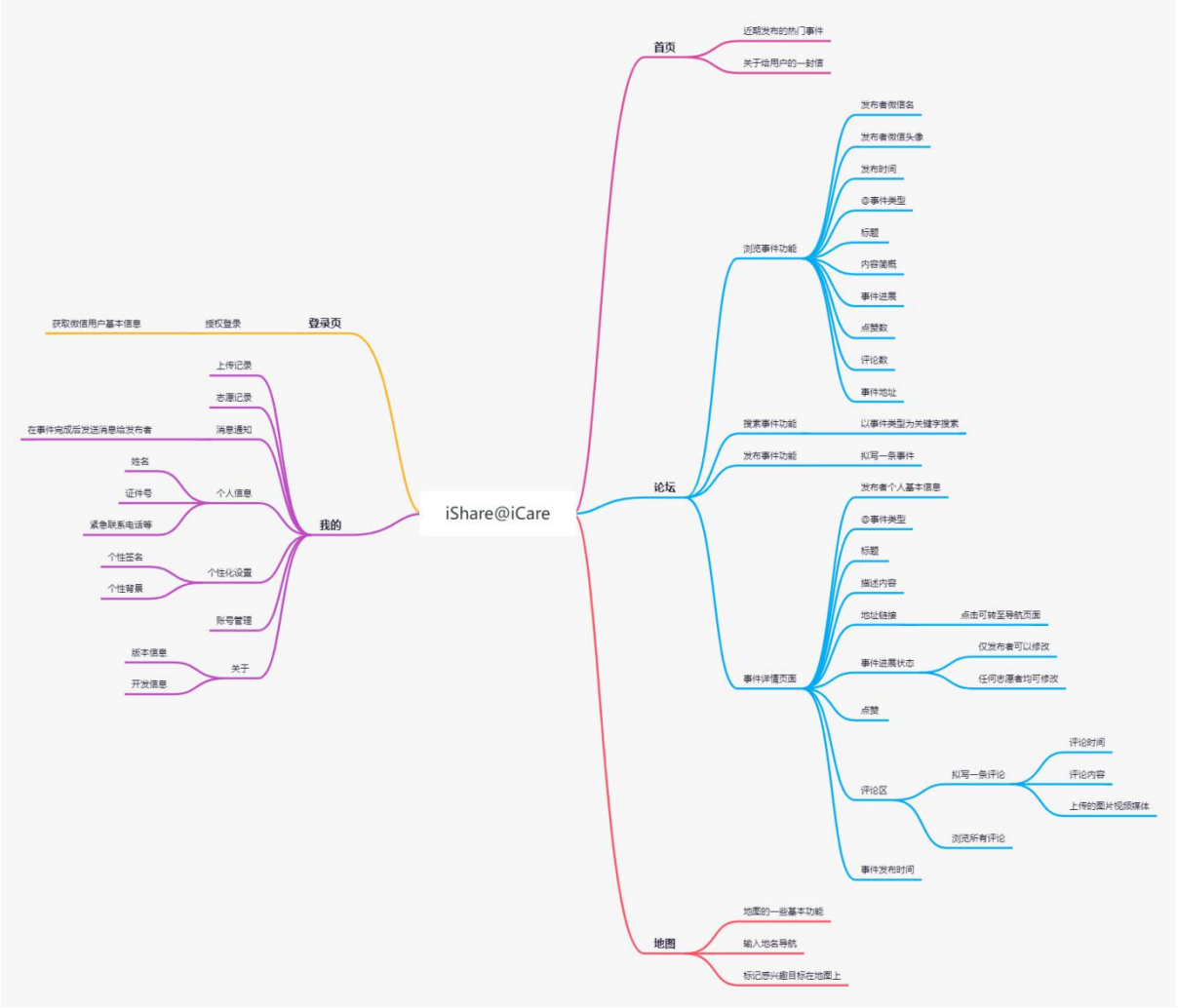
复）
2、然后是对于求助者/”爆料者”，应该提供如下功能：
(1) 可以发布一条事件；具体的包括标题、描述内容、上传图片或视频、选择地址、选择事件类型、选择是否所有人可更改事件进展（事件进展分为“尚无进展”、“进展中”和“已解决”三类，发布时置为“尚无进展”）（若选择的是“任何人可更改事件进展”则所有看见这条事件的用户都可以更改这条事件的进展；否则就只能自己能够修改进展）
(2) 可以在个人页面查看自己已发布过的所有事件（包括已经解决的事件）；在事件解决时，能够收到系统发送的消息通知；
(3) 除此之外，我们还特意实现了每条事件的点赞功能供用户体验使用；以表示某事件的热度；
(4)如果设置了紧急联系人，可以进行紧急呼救
3、再者是对于志愿者，应提供如下功能：
(1) 可以浏览所有的已发布事件，并查看任一条事件的详细页面；在此页面内，志愿者可以看见所有已有的评论，并能写几条新的评论（评论内容仅支持 100 字内的描述以及 9 张图片或视频媒体资源）；除此以外，志愿者还能够给这个事件点赞；
(2) 可以通过直接点击所给出的地址链接自动跳转至地图服务栏，导航至事件发生处，并提供路径规划；
(3) 若发布这条事件的用户设置了“任何人可更改事件进展”，那么志愿者有权更改这条事件的事件进展；

（二）环境限制分析

1. 大小限制，整个小程序所有分包大小不超过 8M，单个分包/主包大小不能超过 2M；
2. 嵌套 H5 的跳转限制，小程序跳转的 H5 链接，必须是 https 协议，且所跳转的链接其域名服务器下必须放置校验文件（即所跳链接我们要有其服务器管理修改权限），才可在小程序中进行跳转；
3. 请求接口限制，小程序请求的接口必须是 https 协议（这一点我们深有体会，在请求用户地址信息时需要向官方申请相关接口，通过后才能正常获取）；
4. app 跳到小程序，小程序才有返回 app 的能力，小程序无法单方面主动跳回 app；
5. 嵌套的 H5 无法直接使用小程序的 api，如果 H5 要使用小程序的 api，则需引入微信的函数库

二、架构设计、架构风格分析与选择、设计模式应用

（一）整体架构设计图（高清图见附件：IShare@ICare.png）



总述：

1. 整个微信小程序除登录授权页面外主要包含了四个页面：首页、论坛、地图及我的，他们通过 tabbar 实现页面跳转；

2. 前端采用 MVC 软件架构风格，后端使用云开发；

3. 核心功能均是围绕着事件的相关操作——发布事件、浏览事件及评论事件；而为实现这些数据的同步，采用微信小程序提供的云开发功能；具体使用到了云数据库、云储存及云函数；主要涉及的功能如下：云数据库——储存 iShare@iCare 用户的个人基本信息（包括微信名、微信头像地址等）和储存每一条事件的详细信息（发布者微信 openid、发布时间、描述内容、事件类型、事件地址、设置修改事件进展权限等）；为前端提供这些功能对应的接口；

4. 前端则围绕这些核心功能以及地图提供给用户的相关功能（导航、路线规划等），采用 MVC 架构风格实现接口的可视化——将封装好的功能以较易识别、较好使用的形式呈现给用户，以最大化用户的软件使用体验。

（二）架构风格分析与选择

1．前端风格分析及优缺点分析

（1）风格分析

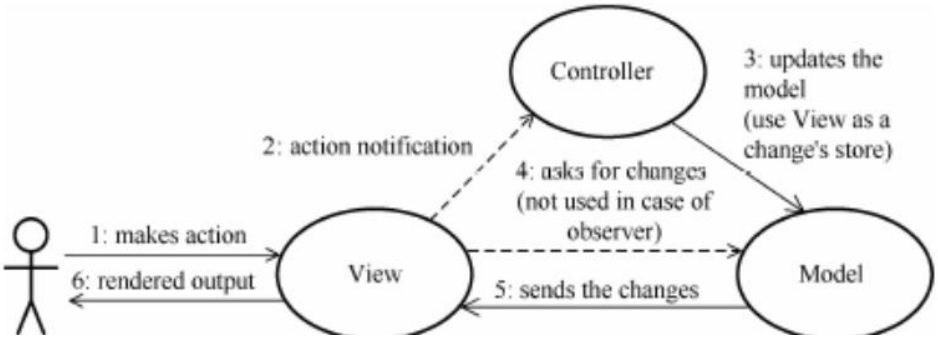
前端采用的软件架构风格是MVC风格：MVC风格可以划分为Model(模式)、View(视图)和Controller(控制器)。这三种组件通过交互来协作。视图创建了控制器之后，控制器根据用户交互调用模式中的服务，而模式会将自身的改变通知视图，视图会读取模式的信息更新自身。MVC模式比较适用与GUI程序。

Model 是对应用状态和业务功能的封装，我们可以将它理解为同时包含数据和行为的领域模型。Model 接受 Controller 的请求并完成相应的业务处理，在状态改变的时候向 View 发出相应的通知。

View 实现可视化界面的呈现并捕捉最终用户的交互操作（例如鼠标和键盘的操作）。

View 捕获到用户交互操作后会直接转发给 Controller，后者完成相应的 UI 逻辑。如果需要涉及业务功能的调用，Controller 会直接调用 Model。在完成 UI 处理后，Controller 会根据需要控制原 View 或者创建新的 View 对用户交互操作予以响应。

MVC模式用于映射传统的输入、处理、输出功能在一个逻辑的图形用户界面的结构中,用一种业务逻辑、数据、界面显示分离的方法组织代码。



在本次开发的微信小程序中，wxml及wxss担任View（视图）的功能，主要接受用户的行为请求操作，并将处理好的数据呈现给用户查看；并将所接受的用户命令参数传递给Controller（控制器）及模式（Model）进行对应的逻辑运算处理和监视；而这部分则是由javascript（js）完成的；例如一些具体的功能：上传事件、搜索特定事件类型的事件、发布评论等功能均是由js进行封装，为使用提供接口，这称为模式（Model）；在js收到视图（View）发来的数据请求时，便会充当控制器（Controller）的用途调用对应的模式作出相对应的回应；

简单来说就是由控制器（Controller）操作模式（Model）对视图（View）发出的命令操作以及数据请求作出响应,并将经过逻辑处理的数据内容返回给视图（View），视图再将其可视化提供给用户查看；

（2）优点

A、MVC模式实现了多个视图与一个模型相对应的共性，大大提高了代码的可重性；

B、MVC模式具有良好的移植性，可以增强代码的使用性；

C、MVC 模式使三个模块相互独立，改变其中一个不会影响其他两个模块；反过来，当功能需求发生变化时，只需要改变其中一个部分就能满足要求；故依据这种模式能够构建良好的松耦合性的组件

（3）缺点

A、MVC 模式会增加系统设计和运行的复杂性；对于简单页面，将三个模块分离可能反而会增加结构的复杂性，降低运行效率；

B、MVC 模式会因为视图与控制器连接过去紧密，而妨碍两者的独立重用

C、MVC 模式会导致视图模块对模型数据的低效率访问；依据不同的接口类型，视图可能需要多次调用才能获得足够的显示数据,同时可能会经常访问一些未变化的数据，从而导致访问的效率比较低下，也将会降低整个系统的性能；

2. 后端风格分析及优缺点分析

（1）风格分析

后端采用的微信小程序提供的云开发服务；主要用到了其提供的云数据库、云存储及云函数等服务；采用的软件架构风格——无服务器架构（Serverless Architectures），准确来说是使用的第一类无服务器架构 BaaS（Backend as a Service，后端应用即服务），即应用的架构是由一大堆第三方 API 来组织的。一切状态和逻辑都由这些服务提供方来管理；前后端的通信以 API 调用为主，而所需的服务不再由服务端应用开发工程师和运维工程师来维护,只需要调用提供服务的第三方 API 就可以完成相应的功能。例如我们使用的云上的数据库服务、微信用户认证服务、地图导航相关功能等服务。

（2）优点

A、低运营成本：微服务架构中的服务需要一直运行，实际上，在高负载情况下每个服务都不止一个实例，这样才能完成高可用性，在 Serverless 架构中则是没有事件发生时不会有服务运行，可以按需执行功能并共享资源，主机平台会只有在需要时才会执行相应的函数，因此可以有效地降低运营成本，并使应用程序运行得更快；

B、高度可扩展性：由于无服务器计算具有高度可扩展性，可以在几秒钟内对应用程序进行缩放和扩展；

C、低开发成本：对于开发者而言，Serverless 架构无疑为其提供了更为高效、快捷的开发方式，可以在短时间内完成软件功能的构建和部署；

（3）缺点

A、启用服务较慢，延迟较高：Serverless 架构具有高度分布式、低耦合的特点,因此延迟问题是不可避免的;在冷启动期间,错误的函数调用以及其他“陷阱”可能潜伏在角落。这些最终可能会破坏开发的预期用例。这一点我们在开发时深有体会；

B、水平可伸缩性收到很大的限制：这使得扩展更容易，但也使计费费用的积累变得更加容易。同样，调用 Serverless 功能可能很便宜，将其输出写入日志实用程序的成本最终可能会比预期更快地产生高成本；

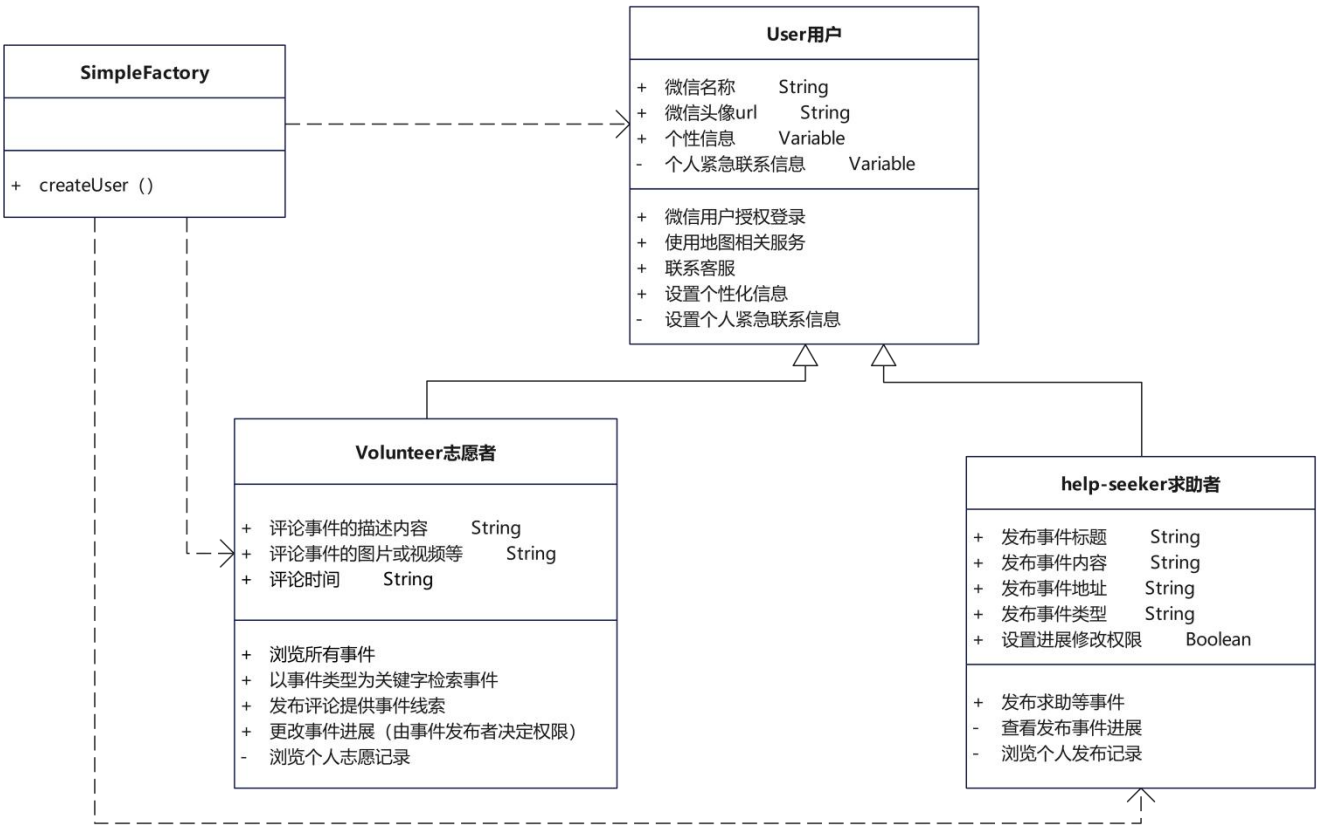
C、复杂性大大增加：在使用 Serverless 架构时，当软件程序越精细、功能越强大时，其就会越复杂；尽管函数的代码可能会变得愈加简单，但整个应用程序将会变得更加复杂；因为原本只需要管理一个应用程序的主程序，现在需要管

理几个甚至上十个不同的子服务。

（三）设计模式应用

1. 创建型模式——简单工厂模式

（1）UML 类图说明：



（2）模式分析

定义一个创建对象的接口，让子类自己选择需要实例化哪一个工厂类，创建实例的任务放到子类里去完成，这样的设计模式即为工厂模式。这满足创建型模式中所要求的“创建与使用相分离”的特点。工厂模式一般分为 3 种实现方式，分别是简单工厂模式、工厂方法模式和抽象工厂模式。本次开发的微信小程序主要使用到了**简单工厂模式**。

简单工厂模式在实例化对象的时候不再使用 new Object()形式，可以根据用户的选择条件来实例化相关的类。对于客户端来说，去除了具体的类的依赖。只需要给出具体实例的描述给工厂，工厂就会自动返回具体的实例对象。

由于本例开发的微信小程序的类主要是构造用户类型，而用户类型从根本上说是允许交叉融合的——也就是说以通用用户 User 为基类的志愿者 Volunteer 类及求助者 help-seeker 类是可以同时出现的，即一个用户既可以是志愿者也可

以是求助者;因此我们考虑使用简单工厂的设计模式即可完成小程序对类的开发需求。

（3）优点

A、工厂类包含必要的逻辑判断,可以决定在什么时候创建哪一个产品的实例。客户端可以免除直接创建产品对象的职责，很方便的创建出相应的产品。工厂和产品的职责区分明确

B、用户端无需知道所创建具体产品的类名，只需知道参数即可

C、也可以引入配置文件,在不修改用户端代码的情况下更换和添加新的具体产品类

（4）缺点

A、简单工厂模式的工厂类单一，负责所有产品的创建，职责过重，一旦异常，整个系统将受影响。且工厂类代码会非常臃肿，违背高聚合原则

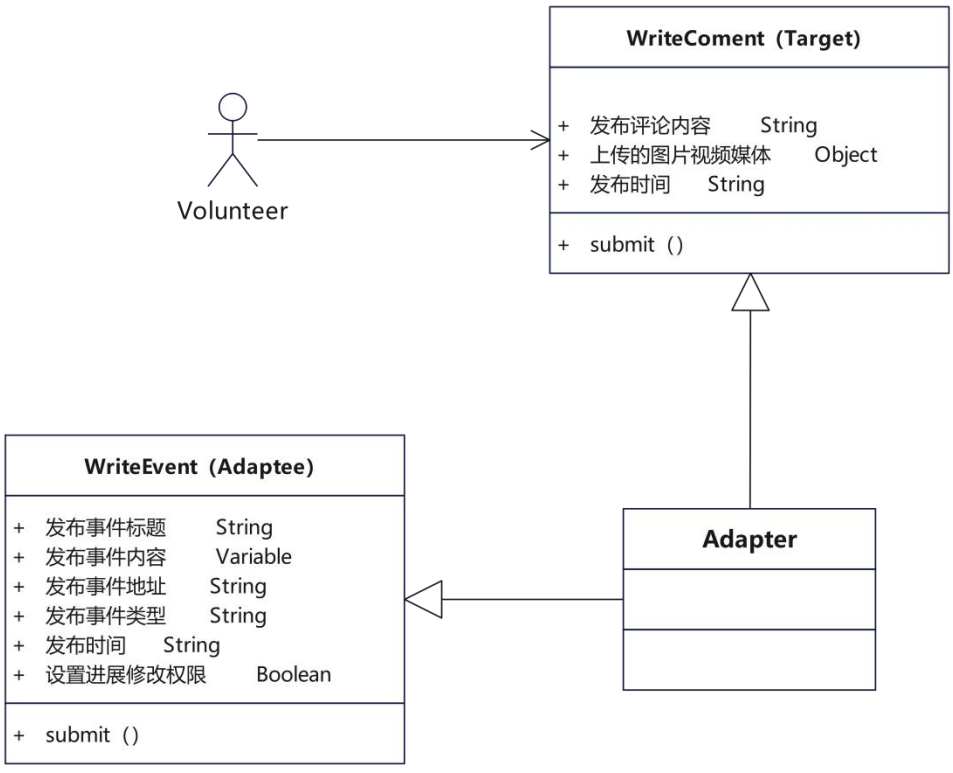
B、使用简单工厂模式会增加系统中类的个数（引入新的工厂类），增加系统的复杂度和理解难度

C、系统扩展困难，一旦增加新产品不得不修改工厂逻辑，在产品类型较多时，可能造成逻辑过于复杂

D、简单工厂模式使用了 static 工厂方法，造成工厂角色无法形成基于继承的等级结构

2. 结构型模式——适配器模式

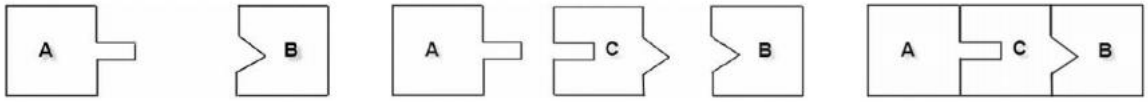
（1）UML 图说明：



以发布事件和发布评论（线索）为例的适配器 UML 图

（2）模式分析

适配器模式（Adapter Pattern）：将一个类的接口变换成客户端所期待的另一种接口，从而使原本因接口不匹配而无法在一起工作的两个类能够在一起工作；其又叫做变压器模式，它也是包装模式（Wrapper）的一种，另一种是装饰模式。适配器模式就是把一个接口或类转换成其他的接口或类。



如上图，如果需要将两个已经成型的模块 A 和 B 联系在一起，可以通过引入一个模块 C 分别适配 A 和 B，然后将 ABC 拼接到一起，C 就是适配器；这种设计模式就是适配器模式；

适配器模式中有三个主要角色：

- Target 目标角色：该角色定义把其他类转换为何种接口，也就是期望接口
- Adaptee 源角色：要转换成目标角色的角色，它是已经存在的、运行良好的类或对象，经过适配器角色的包装，它会成为一个崭新的角色
- Adapter 适配器角色：适配器模式的核心角色，是需要新建立的，它的职责非常简单：通过继承或类关联的方式，把源角色转换为目标角色

在本次开发微信小程序中，以发布评论（提供线索）和发布求助事件为例，我们已经拟写好了关于发布求助事件的功能接口；当在拟写关于发布评论（提供线索）的接口时，便可以知道二者具有很大的关联性以及冗余度，便可以引入一个适配器，将二者拼接起来，可以让两个接口同时起作用。

（3）优点

A、适配器模式可以让两个没有任何关系的类在一起运行，只要适配器这个角色能够将他们连接起来；

B、增加了类的透明性：访问的是 Target 目标角色，但是具体的实现都委托给了源角色，而这些对高层次模块是透明的，也是它不需要关心的；

C、提高了类的复用度：源角色在原有的系统中还是可以正常使用，而在目标角色中也可以充当新的演员，充分提高了效率和利用率；

D、灵活性非常好：不想要适配器，删除掉这个适配器就可以，其他的代码都不用修改，基本上就类似一个灵活的构件，想用就用，不想就卸载

（4）缺点

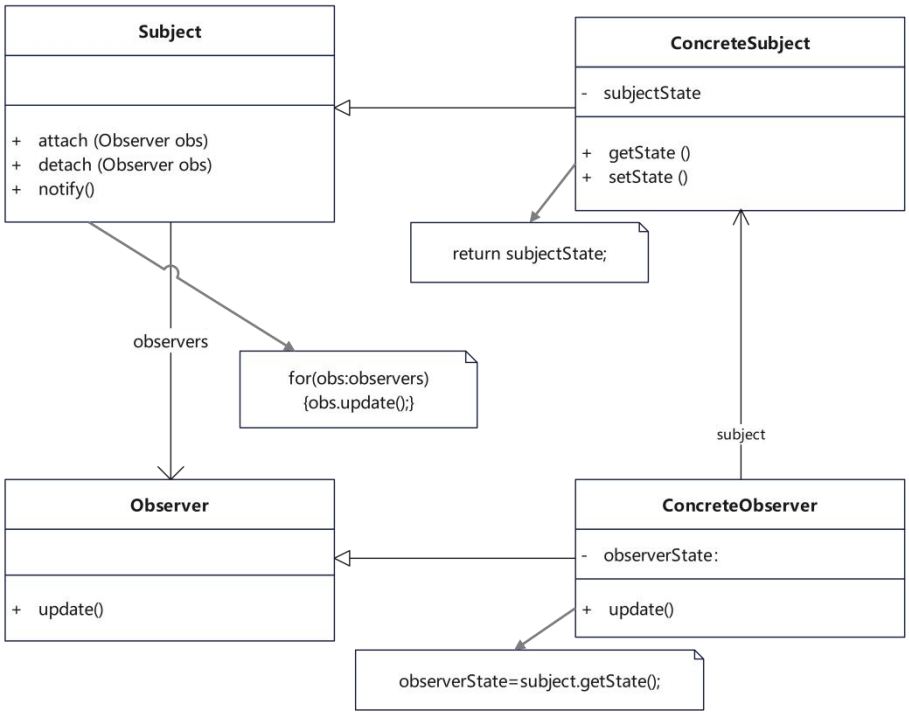
A、适配器模式使用的主要场景是应用扩展，不符合原有设计的时候才考虑通过适配器模式减少代码修改带来的风险；

B、适配器模式要求项目一定要遵守依赖倒置原则和里氏替换原则，否则即使在适合使用适配器的场合下，也会带来非常大的改造成本代价；

C、使用适配器模式的注意事项颇多：适配器模式不是为了解决还处在开发阶段的问题，而是解决正在服役的项目问题

3. 行为型模式——观察者模式

（1）UML 图说明



（2）模式分析

当对象间存在一对多关系时，则使用观察者模式（Observer Pattern）。比如，当一个对象被修改时，则会自动通知依赖它的对象。观察者模式属于行为型模式。观察者模式由下面四个部分组成：

(1) 抽象目标角色（Subject）：目标角色知道它的观察者，可以有任意多个观察者观察同一个目标。并且提供注册和删除观察者对象的接口。目标角色往往由抽象类或者接口来实现。

(2)抽象观察者角色（Observer）：为那些在目标发生改变时需要获得通知的对象定义一个更新接口。抽象观察者角色主要由抽象类或者接口来实现。

(3)具体目标角色(Concrete Subject)：将有关状态存入各个 Concrete Observer 对象。当它的状态发生改变时，向它的各个观察者发出通知。

(4)具体观察者角色（Concrete Observer）：存储有关状态，这些状态应与目标的状态保持一致。实现 Observer 的更新接口以使自身状态与目标的状态保持一致。在本角色内也可以维护一个指向 Concrete Subject 对象的引用。

在本次微信小程序中，服务器端作为观察者，可以存在多个的用户端则作为被观察者(即目标角色)；当被观察者(用户端)发出某些请求或其他要求，就可以视为其状态发生变化，进一步观察者就会察觉到这一变化，并由其具体的变化状态作出具体的响应操作，并将返回值呈递给用户，从而满足客户的要求并完成一系列的功能。

（3）优点

A、可以实现在观察目标和观察者之间建立一个抽象的耦合

B、可以实现表示层和数据逻辑层的分离

C、可以支持广播通信，简化了一对多系统设计的难度

D、其符合开闭原则

（4）缺点

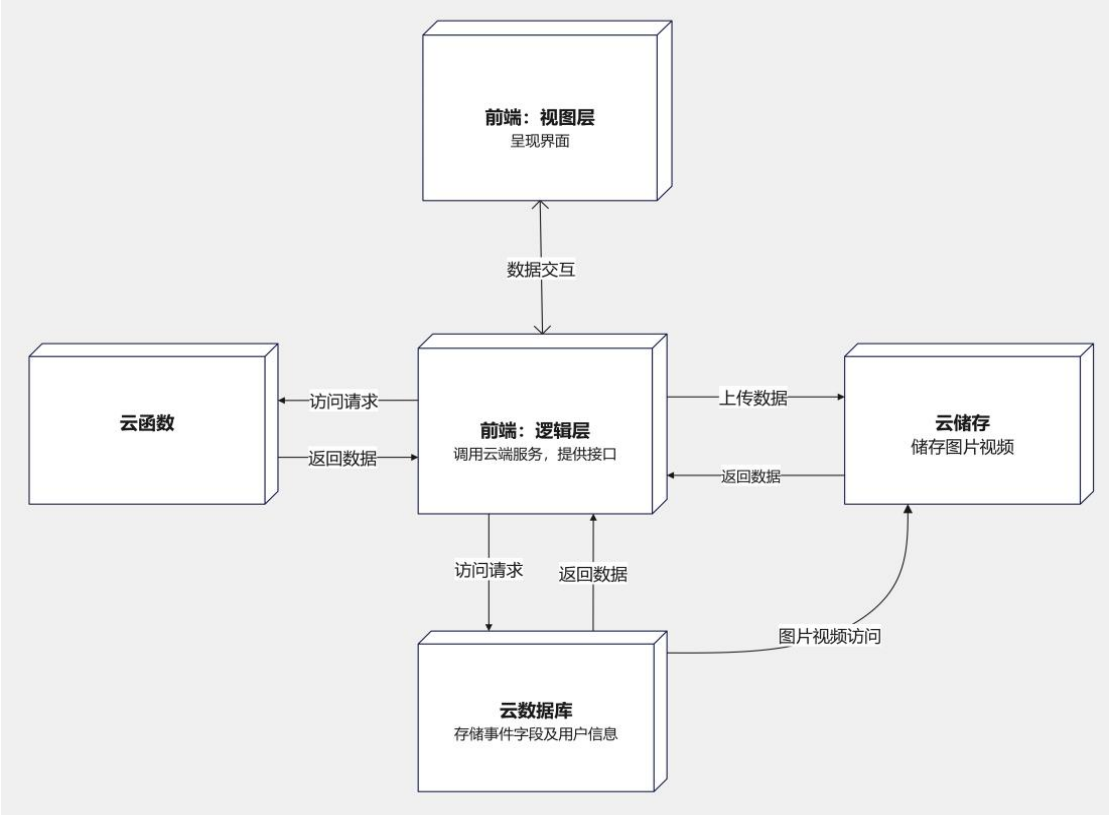
A、目标状态变化导致通知所有观察者会花费很多不必要的时间

B、如果存在循环依赖关系时可能会导致系统崩溃

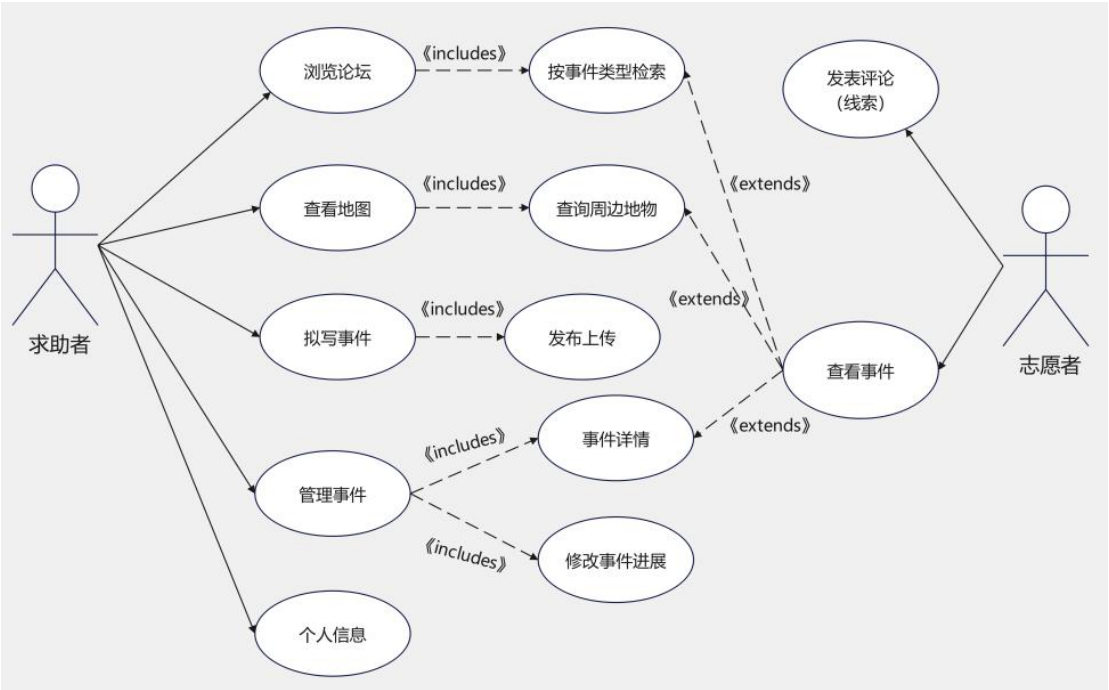
C、只知道观察目标状态发生了变化，而没有相应的机制让观察者知道所观察的目标对象是如何发生什么样的变化的

三、应用软件开发实现

对于整个程序的开发，我们大致分为这六个部分：云开发、登录、首页、论坛、地图、我的。除开登录界面为最开始的授权之外，其他四个部分使用 tabBar 串联成底部菜单，构成整个软件的框架。整体的部署图和用例图如下。



（部署图）



（用例图）

（一）云开发部分

1.云数据库

我们开设了两个数据库（UserInfo、INFO），分别存储用户信息和事件信息。

UserInfo:


```
"_id": "1fee1e97625bd7140065f9f84d329129"
"_openid": "oo7bN5NUeZB9BvASSh0JtOSU34Wg"
"avatarUrl": "https://thirdwx.qlogo.cn/mmopen/vi_32/A1ayNUqfveAgVkW1V8i..."
"backimg": "wxfile://tmp_4f16553d94ad3053533be93bbbd63958876df0dc5afe..." (string)
"city": ""
"country": ""
"gender": ""
"id": ""
"job": ""
"name": ""
"nickname": "Decite"
"openid": "oo7bN5NUeZB9BvASSh0JtOSU34Wg"
"personalsign": "正义人"
"phone": ""
"province": ""
"vitalphone": "15272104528"
```


INFO:


```
"_id": "5464a294625bca000086e63e0d9ac60e"
▸ "Comment": [{"content":"哈哈","date":"2022/04/17 16:54:26","imgsrc":[...]
"Content": "bhbi"
"Date": "2022/04/17 16:04:14" (string)
"EventType": "寻人启事"
"Latitude": 30.59276
"Likes": 3
"LocationName": "江岸区武汉市人民政府(一元路北)"
"Lontitude": 114.30525
"Openid": "oo7bN5O9S9h9jJv7itB08pxcVNyw"
"Progressing": "已解决"
"Title": "bibui"
"_openid": "oo7bN5O9S9h9jJv7itB08pxcVNyw"
"checkType": 0
▸ "imgpath": []
▸ "mp4path": []
```

2. 云存储

主要用于存储用户在发布事件、发布评论中上传的照片和视频。

文件名称	File ID	文件大小	最后更新时间
1649519601545.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649519601545.png	305.77 KB	2022-04-09 23:53:22
1649519864903.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649519864903.png	436.10 KB	2022-04-09 23:57:46
1649560229483.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649560229483.png	813.11 KB	2022-04-10 11:10:31
1649560229487.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649560229487.png	1.05 MB	2022-04-10 11:10:31
1649560257805.mp4	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649560257805.mp4	1.27 MB	2022-04-10 11:11:00
1649560436035.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649560436035.png	159.00 KB	2022-04-10 11:13:56
1649560443437.mp4	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649560443437.mp4	222.27 KB	2022-04-10 11:14:03
1649564189857.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649564189857.png	27.23 KB	2022-04-10 12:16:29
1649564856585.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649564856585.png	108.98 KB	2022-04-10 12:27:36
1649564856589.png	cloud://data-1gplk8k9d9fd263f.6461-data-1gplk8k9d9fd263f-1310983671/1649564856589.png	178.28 KB	2022-04-10 12:27:36

3. 云函数

主要编写了两个云函数,分别为 getid(用于获取登录用户的 openid)和 msg_send（用于事件状态改变时给用户发送通知）

```
// getid云函数入口函数
exports.main = async (event, context) => {
  const wxContext = cloud.getWXContext()

  return {
    event,
    openid: wxContext.OPENID,
    appid: wxContext.APPID,
    unionid: wxContext.UNIONID,
  }
}
```

```
//msg_send函数入口
exports.main = async (event, context) => {
  try {
    const result = await cloud.openapi.subscribeMessage.send({
      "touser": event.openid,
      "page": 'login',
      "lang": 'zh_CN',
      "data": {
        thing1: {value: event.theme},
        date2: {value: event.startTime},
        date3: {value: event.endTime},
        thing4: {value: event.address}
      },
      "templateId": 'B43Xy25SJfhebP1MNaxnbqxy0opbYULjxhAlrMAAkww',
      "miniprogramState": 'developer'
    })
    return result
  } catch (err) {return err}
}
```

（二）登录部分

登录部分主要是完成用户授权的功能，主要目的是获取用户的信息。

使用微信自带的 getUserProfile 接口获取用户的微信昵称、微信头像；然后使用云函数 getid 获取用户 openid；然后在数据库 UserInfo 中查询，若有此用户，则更新头像、昵称等信息；若没有此用户，则将 openid、头像、昵称等信息作为一条新的数据项写入数据库 UserInfo。授权完成后跳转到首页 index。


```
// getUserProfile
wx.getUserProfile({
    desc: '展示用户信息',    //不能为空
    success:res=>{
      console.log("获取用户信息",res)
      that.setData({
        wechat_name:res.userInfo.nickName,
        headimgurl:res.userInfo.avatarUrl,
        province:res.userInfo.province,
        country:res.userInfo.country,
        gender:res.userInfo.gender,
        city:res.userInfo.city,
      })
```

```
WriteIn:function()//查询并且写入或者更新
  wx.cloud.database().collection("UserInfo").where({
    openid:app.globalData.loger.openid
  }).get({
    success:res=>{
      if(res.data.length == 0){ // 新建
        wx.cloud.database().collection("UserInfo").add({
          data:{
            openid:app.globalData.loger.openid,
            nickname:app.globalData.loger.wxnn,
            avatarUrl:app.globalData.loger.wxsrc,
```

```
//存入全局变量
app.globalData.loger.wxnn=that.data.wechat_name
app.globalData.loger.wxsrc=that.data.headimgurl
wx.cloud.callFunction({
  name:"getid",
  success:res=>{
    console.log(res)
    app.globalData.loger.openid=res.result.openid
    this.WriteIn()
  },
  fail:res=>{
    console.log(res)
  },
})
```

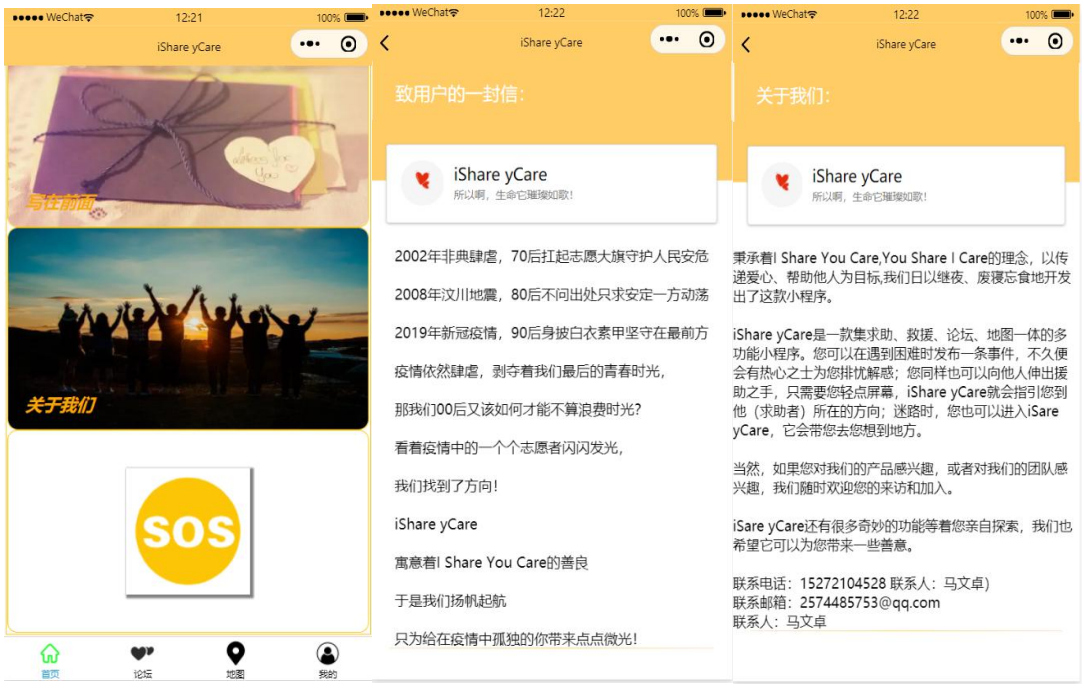
（三）首页部分

首页主要是一些介绍性的功能和紧急求救功能(为了服务于项目的初衷和用户的需求，我们特意增加了紧急求救这个功能）。

利用轮播图 swiper 进行图片展示，在搭配一个滚动界面 scroll-view 进行介绍性内容的放置，在滚动界面的最底部有一个 SOS 按钮，长按它可以直接拨打预存的紧急联系人的号码，进行求救。滚动界面中包括了“写在前面”（点击进入可以浏览致用户的一封信）、“关于我们”(点击进去可以了解项目相关内容)。

这个部分没有涉及到太多的功能模块，主要是一个用户体验的增加。其中主要是使用 wx.makePhoneCall 接口拨打用户设置过的紧急联系人（如果没有设置紧急联系人，则会给出提示）。


```
success: res=> {
  console.log("查询数据成功", res)
  app.globalData.loger.vitalphone=res.data[0].vitalphone
  console.log(app.globalData.loger.vitalphone)
  if(app.globalData.loger.vitalphone==""){
    wx.showToast({
      title: '未设置紧急联系人',
      icon:'error',
      duration:800,
    })
  }else{
    wx.makePhoneCall({
      phoneNumber: app.globalData.loger.vitalphone,
      success(){},
      fail(){}
    })
```



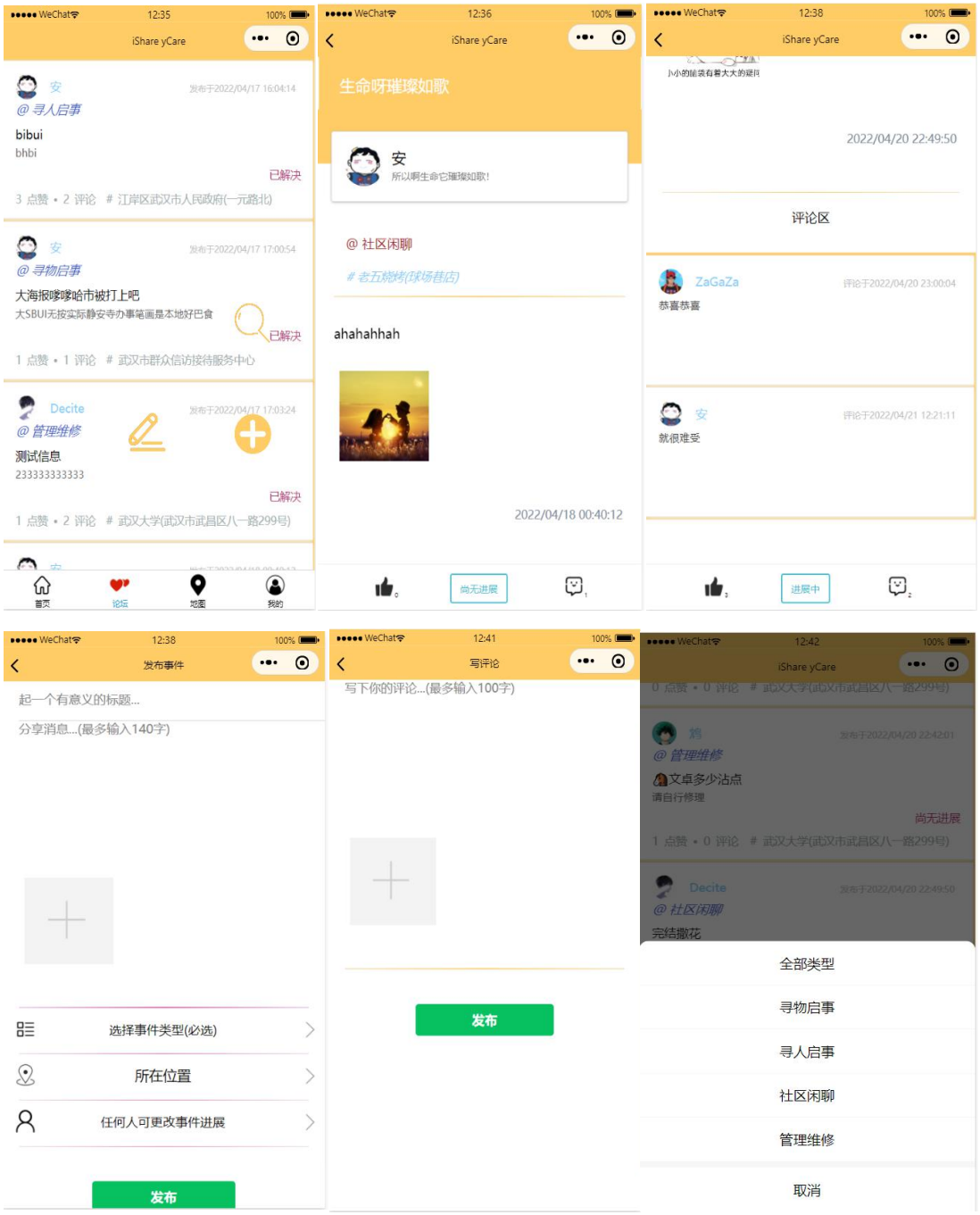
（四）论坛部分

论坛是项目核心功能的聚集地，承载了发布事件、浏览事件、评论事件、修改事件、点赞事件、搜索事件等一系列的功能。

这个部分以 forum 界面为主体，每个事件竖直排列其中，点击一个事件跳转到 Detail 界面查看事件详细信息，里面呈现了关于事件的发布者、事件全文、事件照片、视频、事件的评论等，当然用户可以在这个界面进行点赞、修改状态、评论。在 forum 界面点击悬浮的按钮，可以选择搜寻事件（可按照事件类型搜索）和发布事件（跳转到 Write 界面进行事件的拟写）。当然，为了让用户有更好的

使用体验，我们设计了点击事件位置即可跳转到导航界面，导航去这里。

具体代码实现基本基于从数据库拉取数据、渲染界面、响应事件函数这样一个流程。在开发实现过程中也使用了较多的 API 接口，具体程序太过分散，就不在此过多赘述，具体代码可见附件。



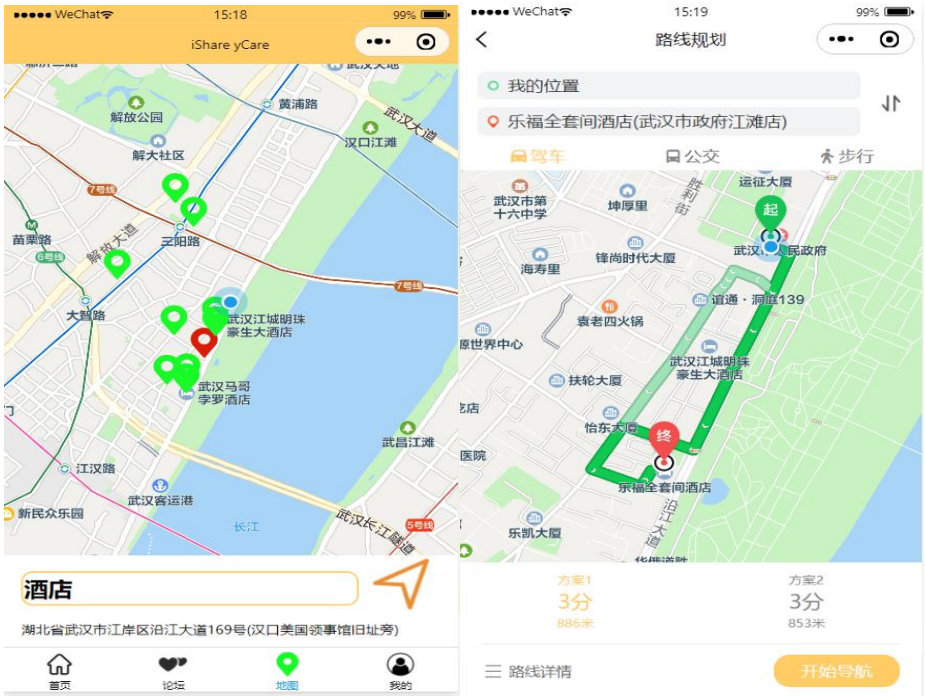
（五）地图部分

这个部分主要是负责地图相关的功能，如地图导航、路径规划、兴趣目标搜索、方案选择等。

地图部门依托于 map 界面，此界面显示出地图定位到用户所在的位置，在这个界面可以输入感兴趣的目标，搜索 10 个距离用户最近的目标供用户进行选择。

选择完毕之后，点击右下角的导航按钮可以跳转到导航界面，进行导航、路径计算、方案规划等。

这个部分的实现主要依托于腾讯位置服务（路径规划、城市选择）第三方插件。使用其提供的接口，可以比较方便的实现地图相关的功能。对于兴趣目标的搜索，我们使用插件的搜寻功能得到十个感兴趣目标（按照最小距离的原则选择），标注到地图上，供用户点击选择。




```
"plugins": {
  "routePlan": {
  "version": "1.0.18",
  "provider": "wx50b5593e81dd937a"
  }
},

"permission": {
  "scope.userLocation": {
  "desc": "你的位置信息将用于小程序定位"
  }
}
```

```
//导航按钮
nav: function (e) {
  let plugin = requirePlugin('routePlan');
  let key = 'I5UBZ-VEAK4-5EPUZ-DBRNI-2YSOH-RGFOO';  //使用在腾讯位置服务申请的key
  let referer = 'iShare@iCare';   //调用插件的app的名称
  let endPoint = JSON.stringify({  //终点
    'name': this.data.address[this.data.index].title,
    'latitude': this.data.markers[this.data.index].latitude,
    'longitude': this.data.markers[this.data.index].longitude
  });
  wx.navigateTo({
    url: 'plugin://routePlan/index?key=' + key + '&referer=' + referer + '&endPoint=' + endPoint+'&themeColor=#ffcc66'+'&navigation='+1
  });
},
```

（六）我的部分

这个部分主要是为了便于用户管理自己的信息、优化用户程序体验而设置的。朗括了管理上传或志愿记录、接受事件更改信息、咨询客服、修改个人信息、设置个性化、账号管理、关于等功能。

1. **上传记录、志愿记录：**点击上传（志愿）记录图标，进入 submit（volunteer）界面，在这个界面里我们通过云数据库的查询，只展示登录用户自己上传（评论）的事件，在这个页面中一样可以点击事件进入其中查看详情（这里实际上是重用了 forum 和 Detail 的代码，体现了重用性）。

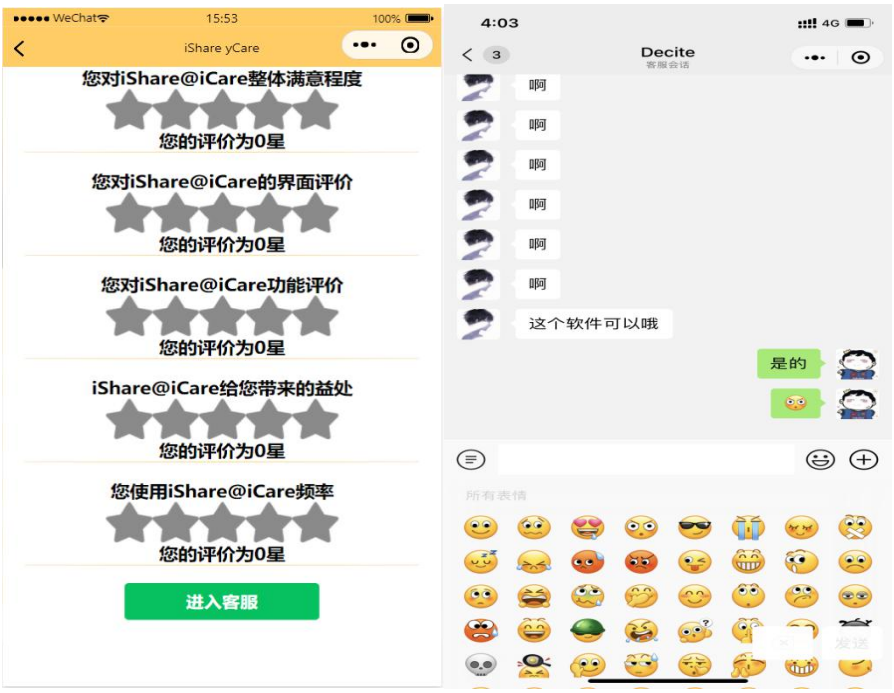

```
})
console.log(app.globalData.loger.openid)
wx.cloud.database().collection("INFO").where({//查询登录者发布的
  Openid:app.globalData.loger.openid
}).get({
  success: res=> {
    console.log("查询数据成功", res)
    for(var idx in res.data){
      this.setData({
        content:this.data.content.concat(res.data[idx].Content),
        eventype:this.data.eventype.concat(res.data[idx].EventType),
        locationname:this.data.locationname.concat(res.data[idx].Loc
        title:this.data.title.concat(res.data[idx].Title),
```

2. 消息通知：点击消息通知进行接收消息授权，授权之后如果自己发布的事件状态发生变化则小程序会发送推送消息给该发布者用户的微信，提醒状态改变。我们是通过 wx.requestSubscribeMessage 接口实验用户订阅消息授权，然后调用自己编写的 msg_send 云函数，给用户微信推送消息。

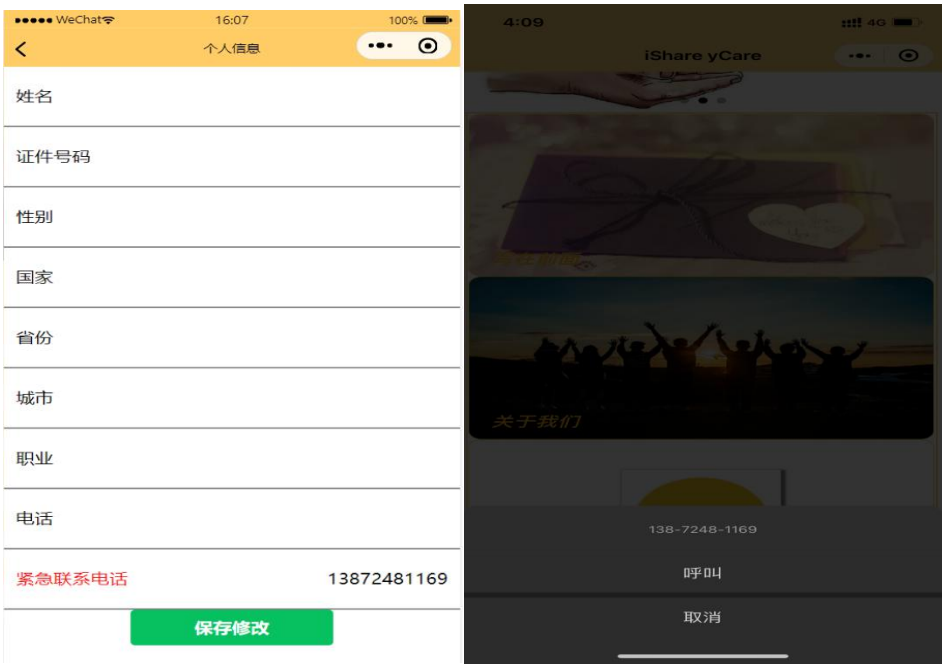

```
wx.cloud.callFunction({//发送消息
  name:"msg_send",
  data: {
    openid: this.data.openid,
    theme:"事件状态修改",
    startTime:"2022-04-22",
    endTime:"2022-04-22",
    address:userinfo.locationname
  },
  success:res=>{
    console.log("success",res)
  },
  fail:res=>{
    console.log(res)
  },
})
```

3. 咨询客服：点击咨询客服，进入评分界面可以为小程序评分，点击进入客服可以和客服沟通交流，在后台我们可以接受到用户的咨询并且进行回复。这里我们借助微信自带的开放样式按钮“contact”，进行客服功能。

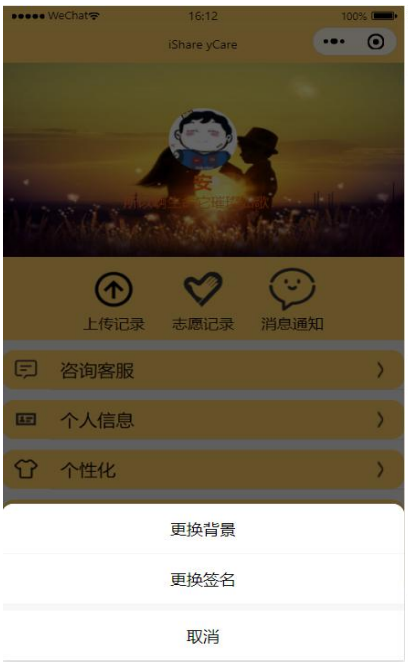
```
<button open-type="contact" type="primary">进入客服</button>
```



4. **个人信息：**主要是各个数据项的编写（只用用户设置了紧急联系人才可以使用首页的 SOS 按钮）。编辑完成之后，点击保存修改，就可保存到云数据库的 UserInfo 当中，在下一次相同用户进入时，会自动读取相应信息，显示在其中。

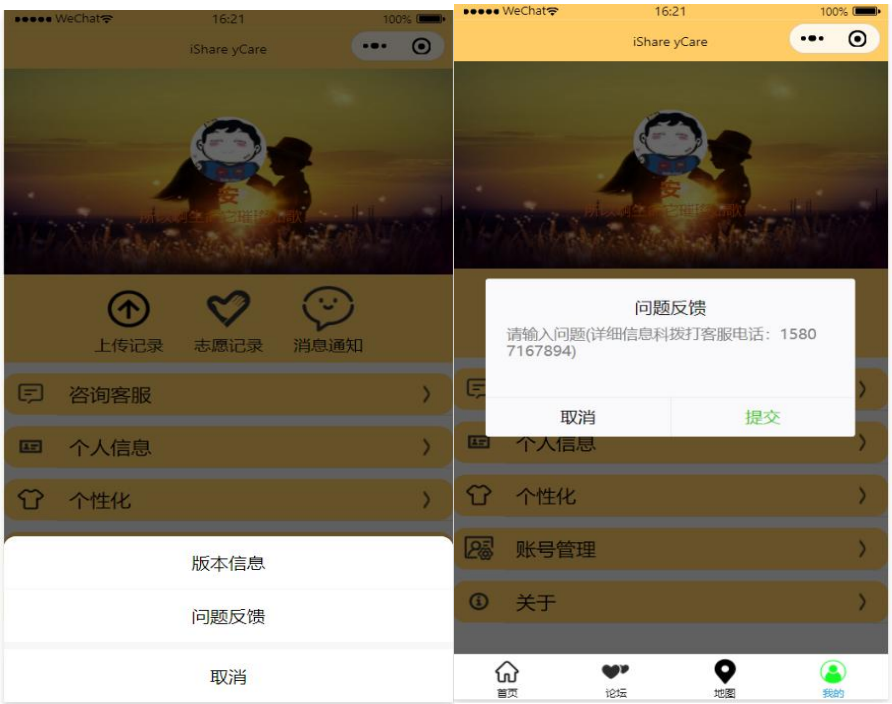


5. **个性化：**点击个性化可以选择更换背景图片和个性签名（使用 wx.showActionSheet），更改之后背景图以 url 的形式、签名以字符串的形式存储在数据库 UserInfo 中，用户发布事件、用户进入我的界面均会展示出来。




```
wx.showActionSheet({
itemList: ['更换背景', '更换签名'],
success:res=>{
  console.log(res.tapIndex)
  if(res.tapIndex==0){//选择更换背景
    wx.chooseMedia({//选择图片
      count: 1,//只能选择1张
      mediaType: ['image'],//图片类型
      sourceType: ['album', 'camera'],//拍照或者相册
      camera:'back',//后置
      success:res=>{
        this.setData({
          backimg:res.tempFiles[0].tempFilePath//存储路径
        })
        wx.cloud.database().collection("UserInfo").where({//上传到数据库
          openid:app.globalData.loger.openid
        }).get({
          success:res=>{
            wx.cloud.database().collection("UserInfo").doc(res.data[0]._id).update(
              data:{
                backimg:this.data.backimg
              },
              success:res=>{
                console.log("修改成功")
              }
```

6. **账号管理和关于：**主要是实现账号注销（使用云数据库的 remove 功能进行用户信息的删除清空）、问题反馈（使用隐藏输入框实现）、版本信息的功能。



四、任务分工

马文卓：主要负责前端工作（页面样式布局、数据渲染、逻辑层的中前端等）
黄晓轩：主要负责后端工作（云数据库开发、云函数设计，逻辑层的中后端等）
注：由于是两人团队，所以在很多事情上都有交叉合作