**布局：**

单独页面。

**左侧**

menu，2部分：我的朋友，我的圈子。

**右侧**

**我的朋友：**

查找朋友：最上方显示input和button。 结果显示在modal中，是一个带有checkbox的list，modal的button为添加和取消。

显示所有朋友分组：下面为朋友分组列表，带有删除（删除组，并把其中所有用户移动到“其他”）和“新建”按钮。每个用户带有删除按钮，直接删除好友。

**我的圈子**

查找群：查找结果显示在modal，此modal是一个带有checkbox的list，按钮为“加入群”和“取消”

显示所有加入的群，群带有退出按钮。

**Db：**

**UserFriends：**用来记录用户所有朋友的coll

**UserFriendsGroup**：用来记录用户所有朋友分组的coll

**publicGroup**：用来记录所有圈子（圈聊）的coll

**usersPublicGroup：**用户加入的群（冗余coll，方便查找）

UserFriends:{

userId: user\_objectId required 用户id

FriendId:user\_objectId required; default [] 用户的朋友

someOtherSetting: ??? Optional 将来可能使用的属性设置

cDate: date required; s 创建用户的时间

mDate:date optional; 增删朋友时修改

dDate:date optional; 删除用户是修改

}

//创建完用户后，直接内部创建默认记录

**User\_friends\_group**:{

name: string required;1~49 用户的朋友分组

userId: user\_objectId required 当前用户

friendsInGroup:[

userid\_objectId

] required; default [] 用户的朋友

cDate: date required; s 创建用户的时间

mDate:date optional; 增删朋友时修改

dDate:date optional; 删除用户是修改

}

**Public\_Group**:{

name: string required;1~50 群名

CreatorId: user\_objectId required 群的创建者

MembersId: [user\_objectId] required 群成员

maxmemberNumber: 100 required 群成员最大数量

AdminsId: [user\_objecId] required 群管理员（初始默认creator，之后可以自行修改）

JoinInRule: string required（default；allowanyone） 新成员加入规则（anyoneAllow/permitAllow，任何人可以加入/管理员允许才能加入）

Histroy: [ required（server自动处理，无需用户输入） 群事件

{

Event: 字段详细解释见下

sourceObjectId:

targetObjectId:

Status; enum.onGoning/done/deny

eventDate:

}

],

cDate: date required; s 创建群的时间

mDate:date optional; 增删群成员时修改

dDate:date optional; 删除群的时间

}

**群事件：**记录群发生的所有事件（独立表而不是内嵌，处理方便）。没有mDate和dDate

**Public\_Group\_Event**:{

publicGroupId: object\_id required

eventType: int required enum

sourceId: user\_objectId required

targedId: user\_objectId optinal

status: int required enum(ongoing/done/deny)

cDate: date required

}

PupublicGroup->history->event

记录圈子事件，使用枚举表示;

**enum.CREATE:** 群建立，无需source/targetObjectId。 Status：done

**NEW\_USER\_APPLY\_TO\_JOIN\_IN**;

新用户申请加入。sourcdId：申请的用户id。 Status： ongoing(permitAloow)/done(anyoneAllow)

如果addInRule为enum.permitAllow:

新用户申请加入批准。sourceId：群管理员Id，targetId；申请的用户id。 Status： done

新用户申请加入拒绝。sourceId：群管理员Id，targetId；申请的用户id。 Status： deny

**GROUP\_MEMBER\_INVITE\_NWE\_USER**：

群成员邀请新用户加入

Source：groupMember\_objectId

Target\_id:newInvitedUserId.

Status: ongoing(permitAloow)/done(anyoneAllow).

如果addInRule为enum.permitAllow:

批准被邀请成员：

sourceId：群管理员Id，targetId；被邀请的用户id。 Status： done

拒绝被邀请成员

sourceId：群管理员Id，targetId；被邀请的用户id。 Status： deny

**CHANGE\_ADMINISTRATOR**;

sourceId：原管理员Id，targetId；管理员用户id。 Status： done

**DELETE\_MEMBER**

sourceId：原管理员Id，targetId；被删除的用户id。 Status： done

**MEMBER\_QUIT\_GROUP**：用户主动退出群

sourceId：用户id。 Status： done

如果用户被管理员处理，记录在此

member\_Penalize:{

creatorId： object\_id required 谁做出的处理

publicGroupId: Object\_id, required 对哪个群（用户可能加入多个群）

memberId: object\_id required

penalizeType: enum required 处理类型，例如禁止发言（NoArticle/NoComment）

duration: int required; 1～30，其他：**天**（更加适合人类的习惯）

cDate:

dDate: 处理是否提前结束

}

Status: Forbit：不许交互

**Public\_Group\_Interaction**:{

publicGroupId: Object\_id, require

content: string require: 15~1000

creatorId: user\_object\_id required

deleteById: user\_object\_id 此信息被谁删除，如果是admin，需要在页面显示“被管理员删除”。

cDate: require

dDate: require

}

记录用户加入，退出群的历史记录，用于数据分析？？

**userPublicGroupHistory**:{

}

**usersPublicGroup：{**

userId: user\_objectId required

currentJoinGroup:[publicgroup\_objectId1...publicgroup\_objectId49] required 当前加入的群

History:[

groupId:

Event:

status

]

**}**