# 业务逻辑

**1. 主题系统**

* 用户可在某一主题下发帖（如NBA、CBA、影视、数码）。
* 每个帖子可关联一个或多个主题。

**2. 帖子**

* 帖子标题。
* 帖子可被用户收藏。
* 帖子可关联多个被评分对象（如球员、电影、手机等）。

**3. 用户系统**

* 用户拥有唯一身份，可发帖、评分、评论。

用户行为会被记录：发帖、评论、打分、点赞、收藏等。

**4. 评分系统**

* 用户可以对评分对象打分（0-10分）。
* 支持查看平均分、打分人数、分布图等统计信息。

**5. 评分对象**

* 可以是球员、球队、电影、数码产品等。
* 每种对象类型属性: 名称，简介。

**6. 评论系统**

* 评论可以关联一个评分。
* 评论可嵌套（楼中楼结构）。
* 评论可点赞。

|  |  |  |
| --- | --- | --- |
|  |  |  |

| **实体** | **描述** | **主要属性** |
| --- | --- | --- |
| User | 用户 | id, 昵称, 等级, 注册时间 |
| Post | 帖子 | id, user\_id, 标题, 发布时间 |
| Topic | 主题/分类 | id, 名称 |
| PostTag | 帖子-主题关联 | post\_id, topic\_id (一对多) |
| Comment | 评论 | id, post\_id, user\_id, content, parent\_comment\_id， 时间戳，target\_id |
| ScoreTarget | 被评分对象 | id, 名称, 简介, post\_id |
| ScoreRecord | 用户评分记录 | user\_id, target\_id, post\_id ,score, 时间戳 |
| UserActionLog | 行为记录 | user\_id, 动作类型, 目标id, 时间戳 |
| FavoritePost | 收藏记录 | user\_id, post\_id |
| Like | 点赞 | user\_id, comment\_id |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# API 第一个前向后，第二个后向前

后端会有一个函数接受前端传来的字典，通过req\_type判断类型，返回结果的字典

Def solve\_request(request) -> result

“req\_type”: 所有传输加上这个域表示类型

1. 用户发布帖子 post\_post

{

“user\_id”: string,

“topic\_id”: [string],

“title”: string  
}  
{}

2. 用户发布帖子关联的评分对象 post\_tartget

{

“post\_id”: string

“name”: string

“description” : string

}

{}

3. 用户对帖子的评分对象评论，并评分 post\_comment\_score

{

“post\_id”: string

“target\_id”: string

“user\_id”: string

“content”: string

“score”: int

}

{}

4. 用户对帖子的评分对象的评论评论 post\_comment\_comment

{

“post\_id”: string

“target\_id”: string

“user\_id”: string

“content”: string

“parent” : string

}

{}

5. 查询主题下的帖子 req\_topic\_posts

{

“topic\_id”: string

}

{

[

{

Post\_id, string

user\_id, string

Title string

Post\_time: string

}

]

}

6. 查询帖子评分对象的平均分 req\_target\_avgscore

{

“post\_id”:

“target\_id”:

}

{

“score”: float

}

7. 查询帖子评分对象下的评论 req\_target\_comments

{

“post\_id”:

“target\_id”

}

{

[

{

“commet\_id”:

“user\_id”

“content”

“parent\_id”

“commet\_time”

}

]

}

8. 查询用户记录 req\_user\_actions

{

“user\_id”

}

{

[

{

“action\_type” : string

“action\_id”: string

“action\_time”:string

}

]

}

# 分锅

1. 前端 汪之立 DDL: 代码周日12：00 debug周日17：00

2. 表设计、文档 : 陈麟丰 DDL: 设计周五12：00 文档周日尽快

3. 后端 马天麟: DDL: 代码周日12：00 debug周日17：00