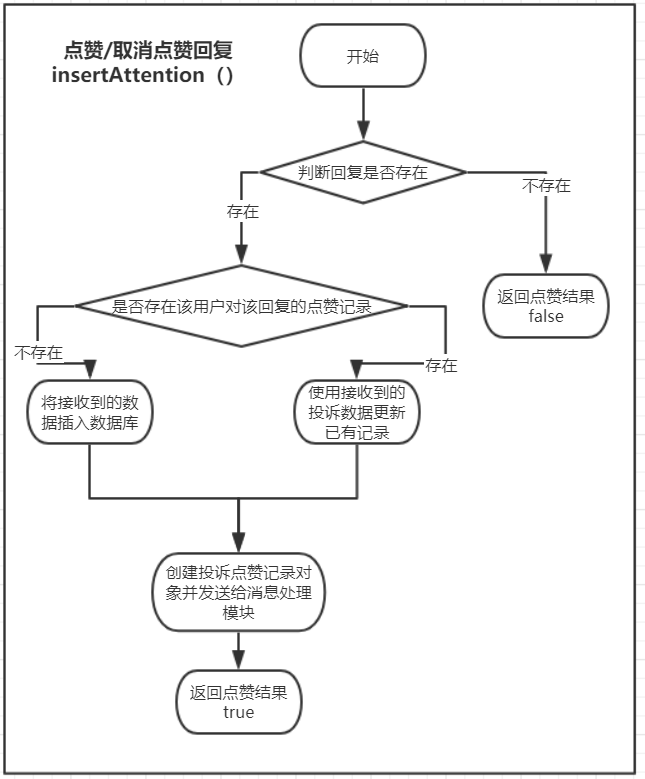
**一、关注与点赞模块**



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加回复所属问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

}

}

---------------------------------------------------------------------------------------------------------------------------------

//调用responseService.insertResposne(ResponseVO responseVO)创建点赞回复

//数据库中没有记录的情况

{

“response”:{

“questionId”://上述testTitle1的id,

“authorId”://testMan1的id

“createTime”:”2020-02-01”

}

}

---------------------------------------------------------------------------------------------------------------------------------

//调用LikeService.insertLikeInfo(Likes like)创建点赞记录

{

“userId”://testMan1的id,

“responseId”://上述回复id,

“flag”:1

}

---------------------------------------------------------------------------------------------------------------------------------

//调用LikeService.insertLikeInfo(Likes like)取消点赞记录

{

“userId”://testMan1的id,

“responseId”://上述回复id,

“flag”:1

}

---------------------------------------------------------------------------------------------------------------------------------

//调用responseService.insertResposne(ResponseVO responseVO)创建点赞回复

//数据库中有记录的情况

{

“response”:{

“questionId”://上述testTitle1的id,

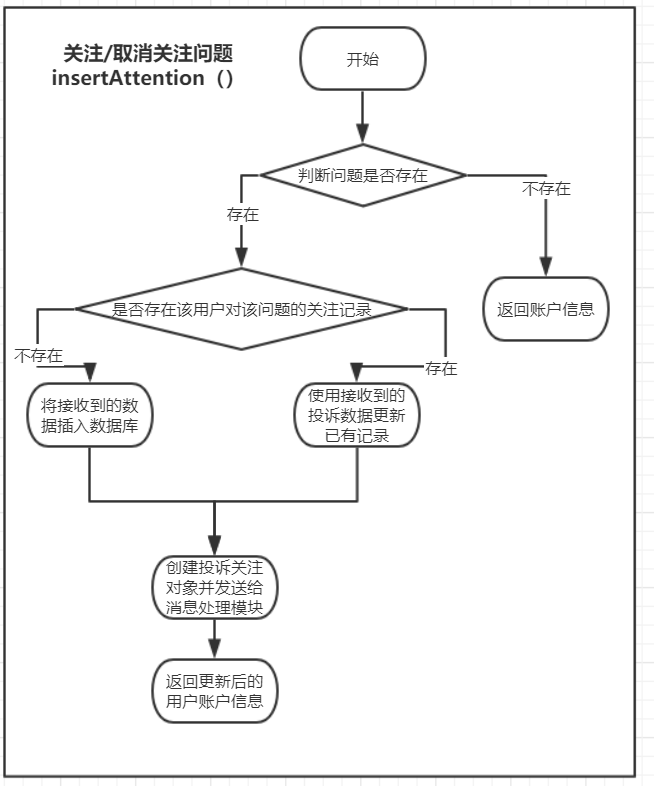
“authorId”://testMan1的id

“createTime”:”2020-02-01”

}

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| insertLikeInfo | 数据库没有测试用户关于该问题的点赞记录时点赞 | 成功添加点赞记录  该回复的点赞数+1  被点赞用户积分增加对应分数  返回结果true | 成功添加点赞记录  该回复的点赞数+1  被点赞用户积分增加对应分数  返回结果true | 正确 |
| 数据库存在测试用户关于该问题的点赞记录时取消点赞 | 点赞记录标志位被设为0  该条回复点赞数-1  回复作者扣除相应积分  返回结果true | 点赞记录标志位被设为0  该条回复点赞数-1  回复作者扣除相应积分  返回结果true | 正确 |
| 数据库没有测试用户关于该问题的点灭记录时点灭 | 成功添加点灭记录  该回复的点灭数+1  返回结果true | 成功添加点灭记录  该回复的点灭数+1  返回结果true | 正确 |
| 数据库存在测试用户关于该问题的点灭记录时取消点灭 | 点灭记录标志位被设为0  该条回复点灭数-1  返回结果true | 点灭记录标志位被设为0  该条回复点灭数-1  返回结果true | 正确 |
| 点赞的回复不存在 | 返回结果false | 返回结果false | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要关注的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

}

}

----------------------------------------------------------------------------------------------------------------------------

//创建关注消息（当前数据库没有本测试用户对该问题的关注记录）

{

“userId”://testMan1的id,

“questionId”://上述创建问题的id,

“createTime”:”2020-02-21”,

“flag”:1

}

----------------------------------------------------------------------------------------------------------------------------

//取消关注消息（当前数据库有本测试用户对该问题的关注记录）

{

“userId”://testMan1的id,

“questionId”://上述创建问题的id,

“createTime”:”2020-02-21”,

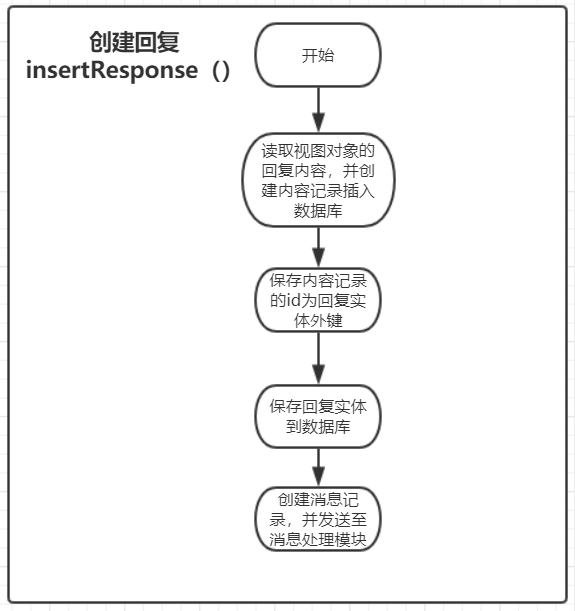
“flag”:0

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| InsertAttention | 创建对某个问题的关注（数据库不存在测试用户对该问题的关注记录） | 数据库创建该关注记录  用户关注数+1 | 数据库成功创建该关注记录  用户关注数+1 | 正确 |
| 取消对某个问题的关注（数据库存在测试用户对该问题的关注记录） | 数据库记录的标志位设为0  用户关注数-1 | 数据库记录的标志位设为0  用户关注数-1 | 正确 |
| 创建对某个问题的关注，数据库不存在该问题 | 用户关注数不变 | 用户关注数不变 | 正确 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getAttentionQuestionList | 获取某个用户所有关注的问题的列表（该用户存在） | 获取的问题ID与上述插入的问题ID一致 | 获取的问题ID与上述插入的问题ID一致 | 正确 |
| 获取某个用户所有关注的问题的列表（该用户不存在） | 获取的问题ID列表长度为0 | 获取的问题ID列表长度为0 | 正确 |

1. **回复模块**



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要回复的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

}

}

---------------------------------------------------------------------------------------------------------------------------

//调用ResponseService.insertResponse(ResponseVO responseVO)创建回复消息

{

“response”:{

“questionId”://上述问题的id,

“authorId”://testMan的id,

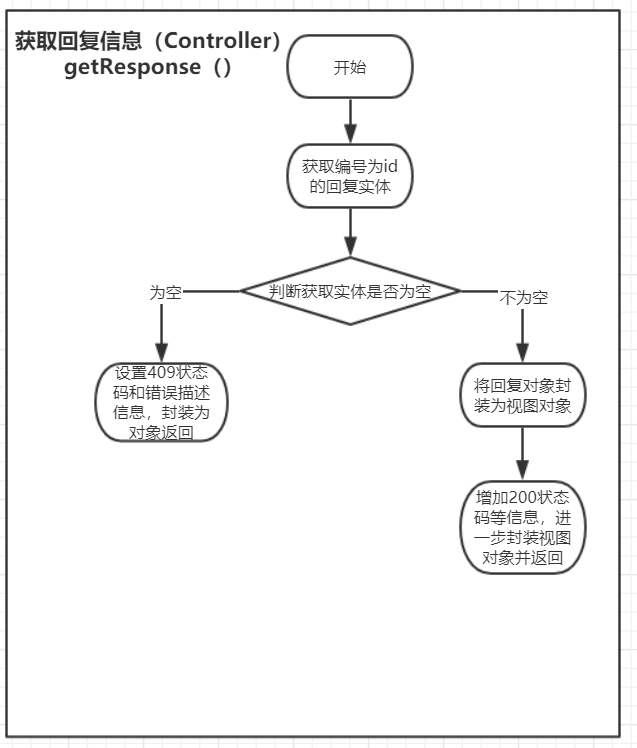
“createTime”:”2020-03-21”,

},

“content”:”responseTestContent1”

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| insertResponse | 添加对某个问题的回复 | 数据库增加该问的回复记录  对应问题的回复数+1  回复者的回复数+1，积分、经验值增加对应分数  返回true | 数据库成功增加该问的回复记录  对应问题的回复数没变  回复者的回复数+1，积分、经验值增加对应分数  返回true | 错误 |
| 回复的问题不存在 | 返回false | 返回false | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要回复的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

}

}

---------------------------------------------------------------------------------------------------------------------------

//调用ResponseService.insertResponse(ResponseVO responseVO)创建回复消息

{

“response”:{

“questionId”://上述问题的id,

“authorId”://testMan的id,

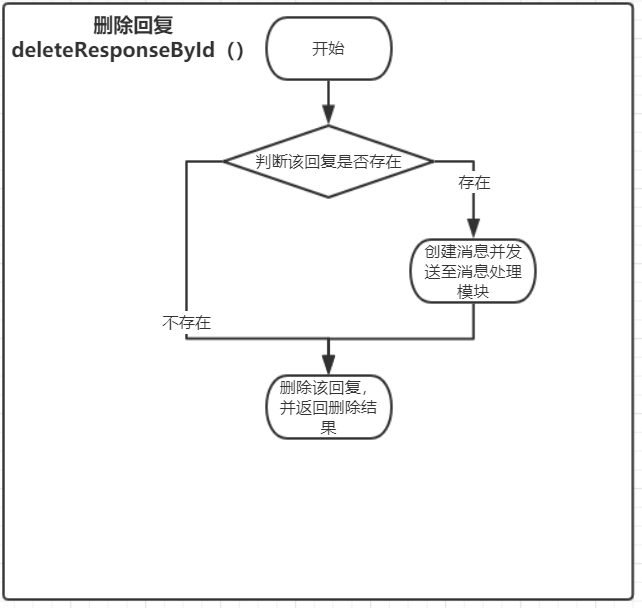
“createTime”:”2020-03-21”,

},

“content”:”responseTestContent1”

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getResponseById | 获取存在的回复 | 成功获取对应回复记录 | 成功获取对应回复记录 | 正确 |
| 获取不存在回复 | 返回null | 返回null | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要回复的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

}

}

---------------------------------------------------------------------------------------------------------------------------

//调用ResponseService.insertResponse(ResponseVO responseVO)创建回复消息

{

“response”:{

“questionId”://上述问题的id,

“authorId”://testMan的id,

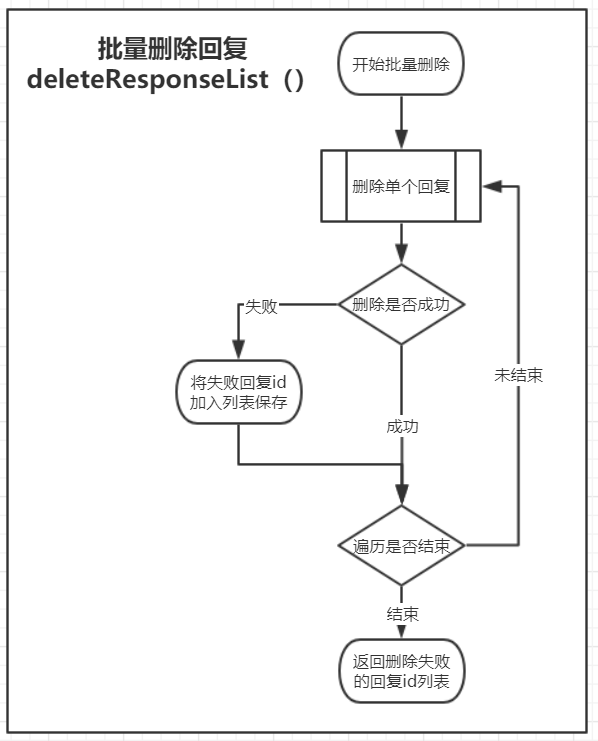
“createTime”:”2020-03-21”,

},

“content”:”responseTestContent1”

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| deleteResponseById | 删除一条存在的回复 | 成功删除记录  问题回复数-1  作者回复数-1  函数返回1 | 成功删除记录  问题回复数没变  作者回复数-1  函数返回1 | 错误 |
| 删除一条不存在的回复 | 函数返回0 | 函数返回0 | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要回复的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

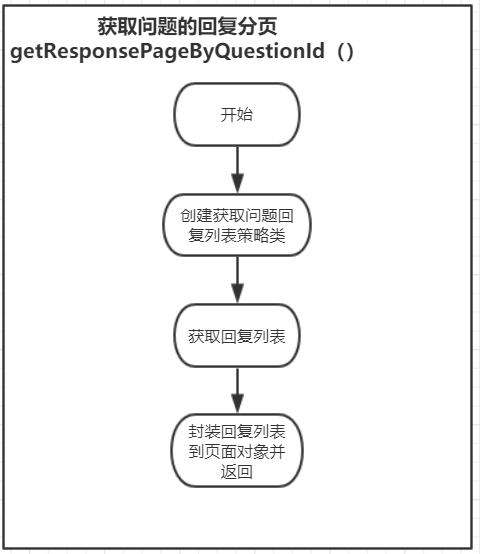
“createTime”:date

}

}

通过insertResponse循环添加10条对上述问题的回复

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试 |
| deleteResponseList | 删除列表中的回复都存在 | 列表中的回复均被删除  测试用户回复数减去对应的删除数  函数返回列表长度为0 | 列表中的回复均被删除  测试用户回复数减去对应的删除数  函数返回列表长度为0 | 正确 |
| 删除列表中含有不存在的回复 | 存在的回复都被删除  测试用户回复-减去对应的删除数  函数返回列表包含不存在的回复的ID | 存在的回复都被删除  测试用户回复-减去对应的删除数  函数返回列表包含不存在的回复的ID | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

//调用questionService.addQuestion(QuestionVO questionVO)添加要回复的问题

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”://testMan1的id

“createTime”:date

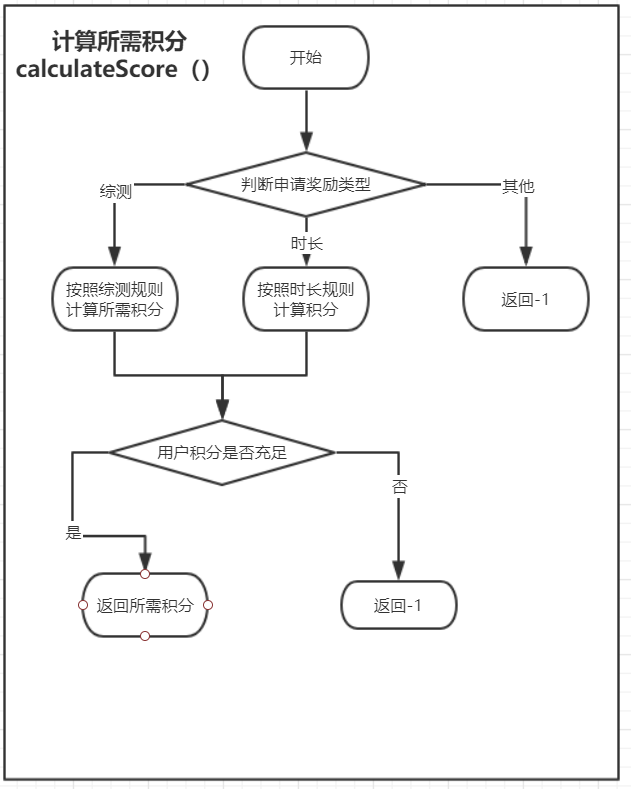
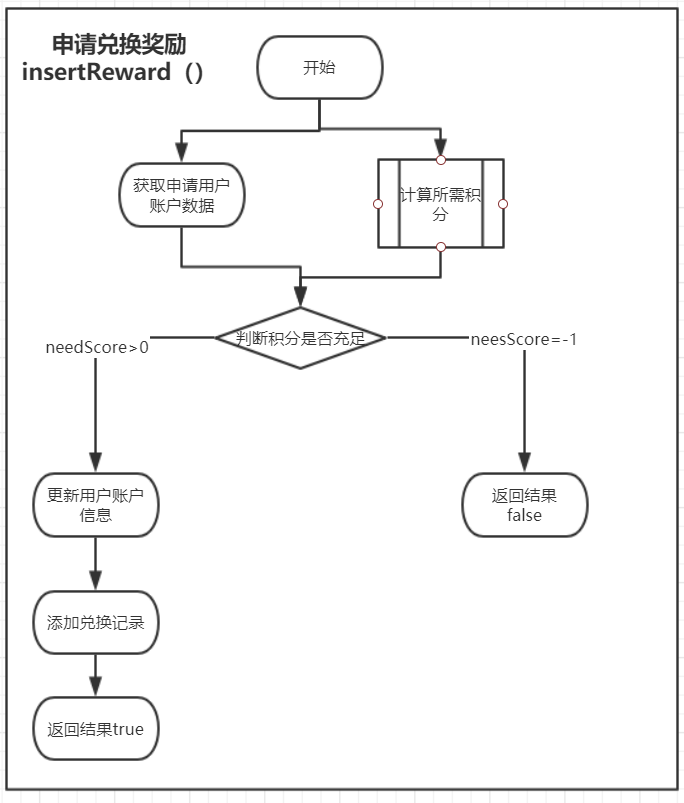
}

}

通过insertResponse循环添加10条对上述问题的回复

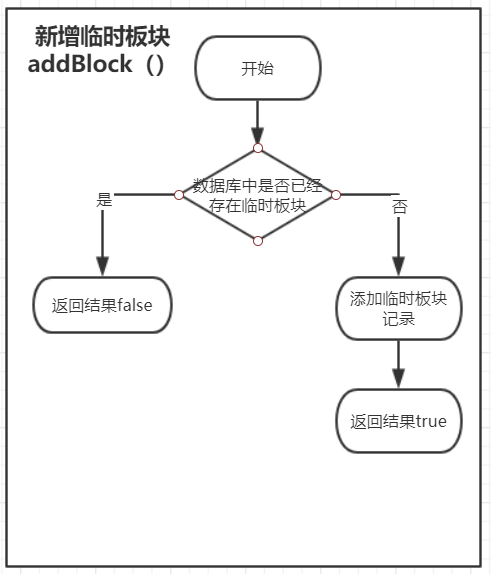
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getResponsePageByQuestionId() | 获取上述问题的回复列表 | 获取回复列表并按点赞数排序 | 成功获取回复列表并按点赞数排序 | 正确 |

1. **奖励模块**

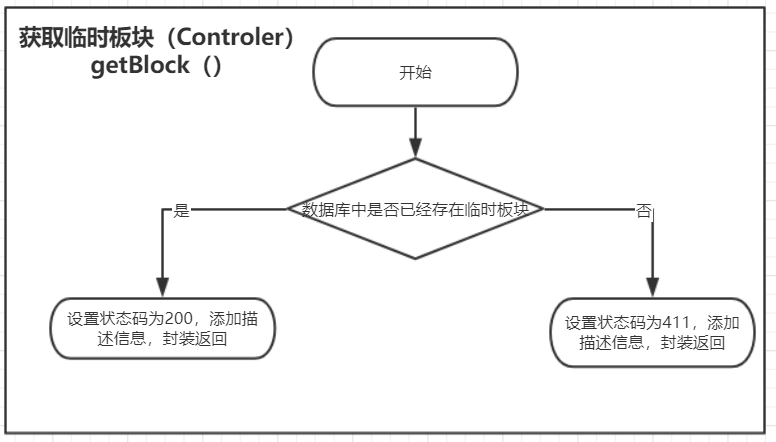


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| insertReward | 申请积分范围内的奖励 | 函数返回true  数据库插入相应记录  用户积分被扣除 | 函数返回true，数据库成功插入，用户积分扣除 | 结果正确 |
|  | 申请超出积分范围的奖励 | 函数返回false | 失败返回false | 结果正确 |

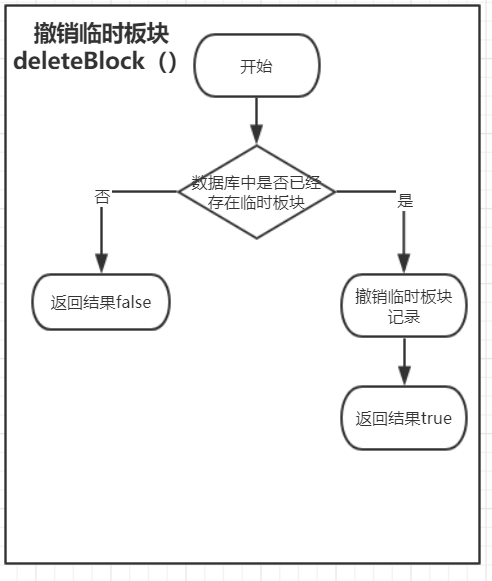
1. **临时板块模块**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| addBlock | 当数据库中不存在临时板块时添加 | 函数返回true  数据库中添加一条记录 | 成功添加记录，并且返回true | 结果正确 |
| 当数据库中存在临时板块时添加 | 函数返回false | 返回false | 结果正确 |

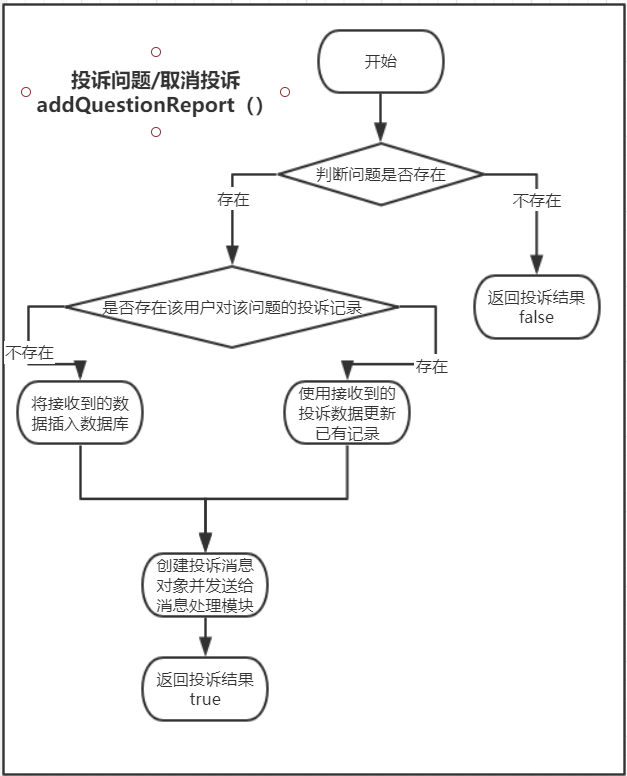


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getBlock | 当数据库有记录时获取临时板块 | 返回临时板块 | 成功返回 | 结果正确 |
| 当数据库没有记录时获取临时板块 | 返回null | 返回null | 结果正确 |



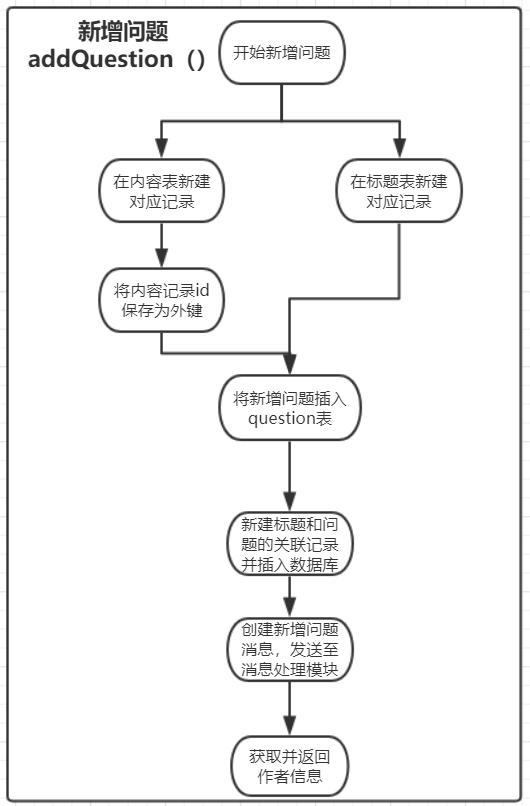
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| deleteBlock | 当数据库存在记录时删除 | 返回true  数据库删除记录 | 数据库成功删除deleteBlock | 结果正确 |
| 当数据库不存在记录时删除 | 返回false | 返回false | 结果正确 |

1. **投诉模块**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| addQuestionReport | 投诉不存在的问题 | 返回false | 返回false | 结果正确 |
| 投诉一条没投诉过的问题 | 数据库新增一条投诉记录  返回true | 返回true并且数据库增加一条report记录 | 结果正确 |
| 取消上面投诉的问题 | 数据库的记录数没增加，对应记录标志位改为0  放回true | 更新数据库对应的report并且返回true | 结果正确 |

1. **问题模块**



测试输入数据://测试结束后请调用删除函数删除脏数据

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

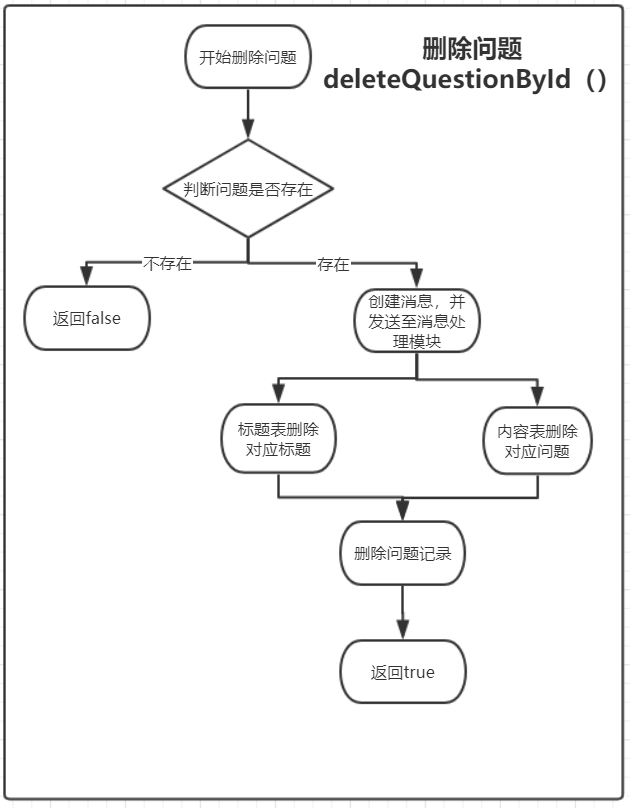
“authorId”://testMan1的id

“createTime”:date

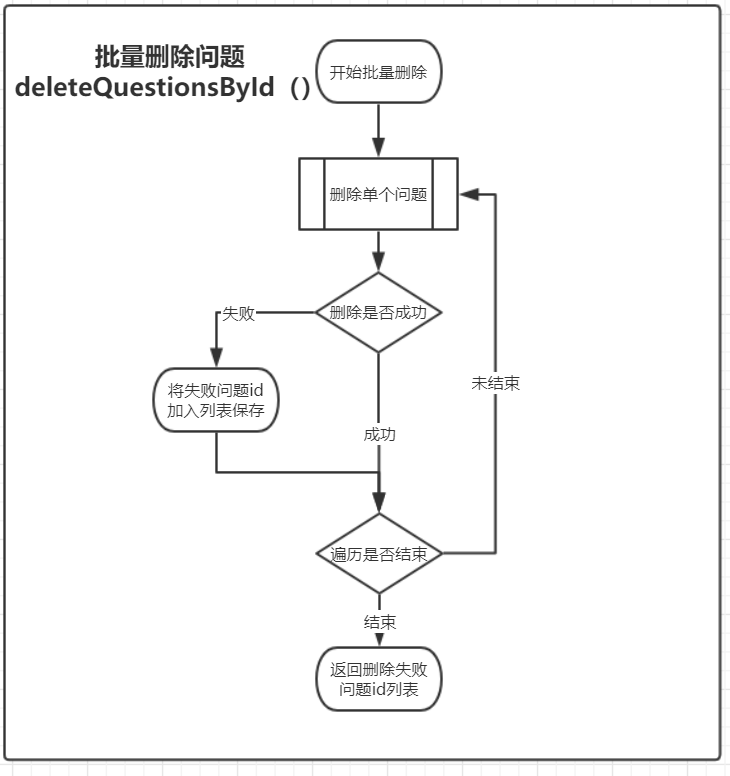
}

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| addQuestion | 输入新增问题 | 数据库新增该问题的记录  创建作者提问数+1，积分获得增加  创作作者的消息数+1 | 与预期相符 | 正确 |



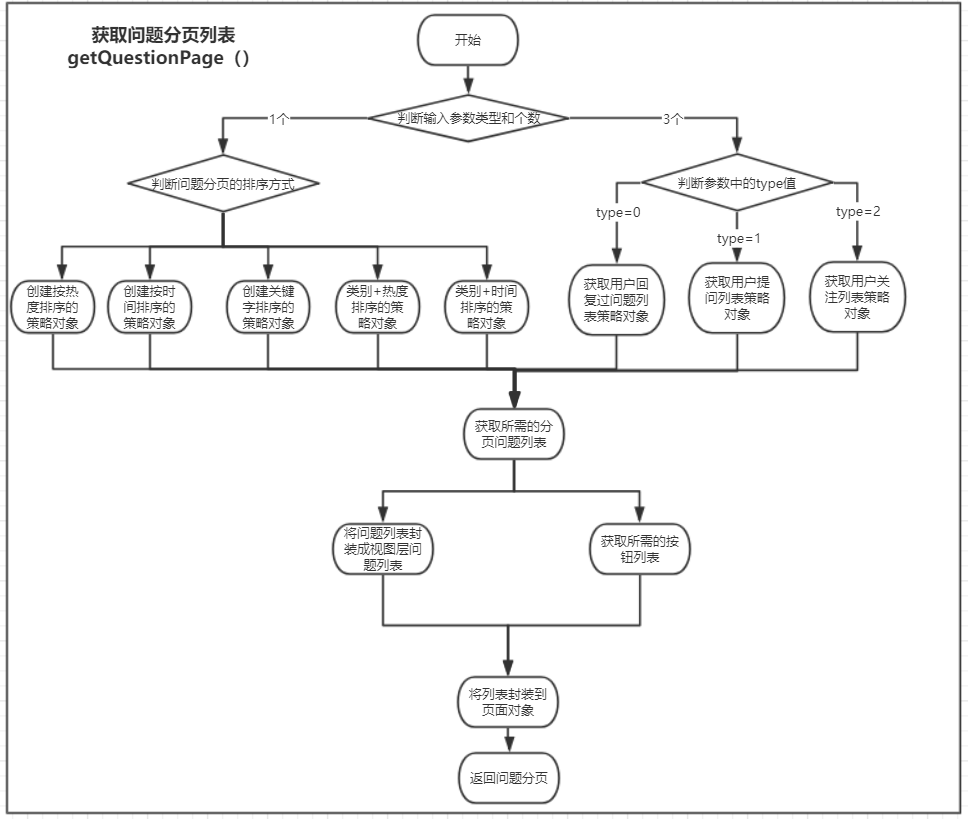
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| DeleteQuestionById | 删除一条已存在的问题记录 | 数据库中该条记录被删除  作者提问数-1  作者的消息数+1  函数返回true | 数据库对应记录被删除  作者提问数-1  作者消息数+1  函数返回true | 正确 |
| 删除一条不存在的问题记录 | 函数返回false | 函数返回false | 正确 |



测试前数据库填充数据：

调用addQuestion方法添加10条作者为test1用户的问题记录，并记录id序列

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| deleteQuestionById | 删除新增问题id序列的问题  （全对） | 数据库中id序列的问被删除;作者问题数-10;返回失败列表长度为0  消息数+10 | 符合预期 | 正确 |
| 正确删除问题后重新再删除一遍上述的id列表（全错） | 返回失败列表长度为10，列表元素为传入的10个元素  用户问题数不变  消息数不变 | 符合预期 | 正确 |
|  | 新增10个问题，将id加入删除序列，再填加5个不存在问题的id到序列（部分对） | 正确的id问题被删除  失败列表长度为5，且为刚添加的不存在的id列表  作者问题数-10  作者消息数+10 | 符合预期 | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”:10,

“createTime”:”当前日期”

“kindId”:0,

“anoymous”:0,

“responseNum”:700

}

}，

{

“title”:”testTitle2”,

“content”:”testContent1”

“question”:{

“authorId”:10,

“createTime”:”明天”,

“kindId”:1,

“anoymous”:0,

“responseNum”:100

}

}，

{

“title”:”test3”,

“content”:”testContent1”

“question”:{

“authorId”:10,

“createTime”:”后天”,

“kindId”:0,

“anoymous”:1,

“responseNum”:500

}

}，

---------------------------------------------------------------------------------------------------------------------------------

//调用ResponseService.insertResponse(ResponseVO responseVO)添加回复

{

“response”:{

“questionId”://问题testTitle1的id

“authorId”://testMan1的id

“createTime”:”2020-10-02”

},

“content”:”responseContent1”

},

{

“response”:{

“questionId”://问题testTitle2的id

“authorId”://testMan1的id

“createTime”:”2020-10-03”,

},

“content”:”responseContent2”

}

---------------------------------------------------------------------------------------------------------------------------------

//调用AttentionService.insertAttention(Attention attention)插入关注记录

{

“userId”://testMan1的id,

“question”://testTiltle2的id,

“createTime”:”2020-03-12”,

“flag”:1

}

{

“userId”://testMan1的id,

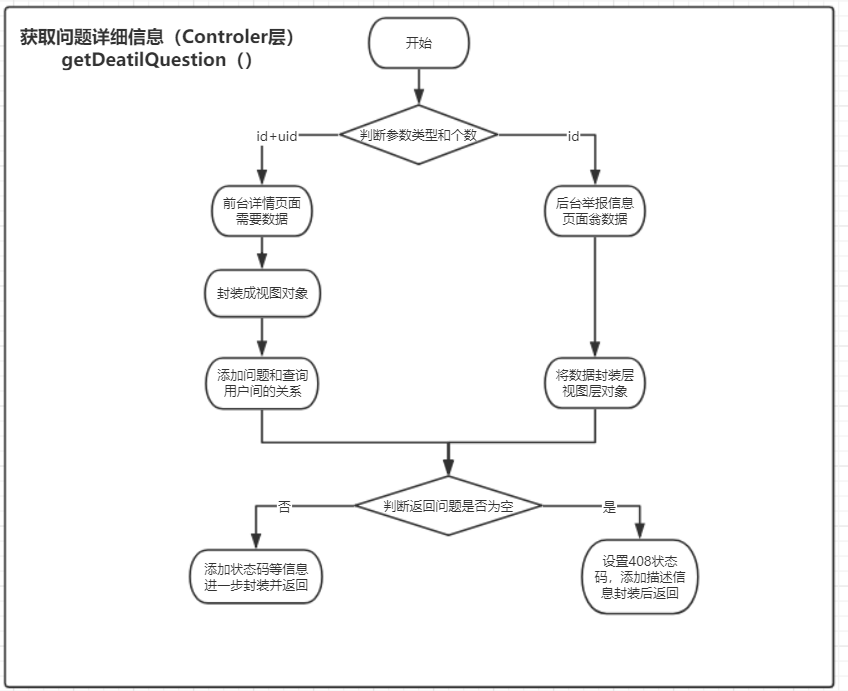
“question”://testTiltle3的id,

“createTime”:”2020-03-12”,

“flag”:1

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getQuestionPage  (QuestionPage) | 按热度排序 | 获取列表的回复数按递减排列 | 符合预期 | 正确 |
| 按时间排序 | 返回列表顺序按时间由近到远排序 | 符合预期 | 正确 |
| 按关键字查询 | 返回问题序列的标题包含查询关键字 | 符合预期 | 正确 |
| 按类别0+热度排序 | 返回类别为0的问题序列并按热度排序 | 符合预期 | 正确 |
| 按类别1+时间排序 | 返回类别为1的问题序列并按时间由近到远排序 | 符合预期 | 正确 |
| getQuestionPage  (Integer userId, QuestionPage page, int type) | 查看用户回复过的问题列表(type=0) | 返回问题序列，数据库存在用户对列表中包含问题的回复记录 | 符合预期 | 正确 |
| 查看用户提问列表(type=1) | 返回问题序列，序列中包含问题作者为测试用户 | 符合预期 | 正确 |
| 获取用户关注问题列表(type=2) | 返回问题序列存在对应的关注记录，同时记录标志位为1 | 符合预期 | 正确 |



测试输入数据://测试结束后请调用删除函数删除脏数据

{

“title”:”testTitle1”,

“content”:”testContent1”

“question”:{

“authorId”:10,

“createTime”:”2020-01-01”

“kindId”:0,

“anoymous”:0,

“responseNum”:10

}

}，

---------------------------------------------------------------------------------------------------------------------------------//调用AttentionService.insertAttention(Attention attention)插入关注记录

{

“userId”://testMan1的id,

“question”://testTiltle2的id,

“createTime”:”2020-03-12”,

“flag”:1

}

---------------------------------------------------------------------------------------------------------------------------------

//调用AttentionService.insertAttention(Attention attention)插入关注记录

{

“reportorId”://举报人testMan1的id,

“questionId”://testTiltle1的id,

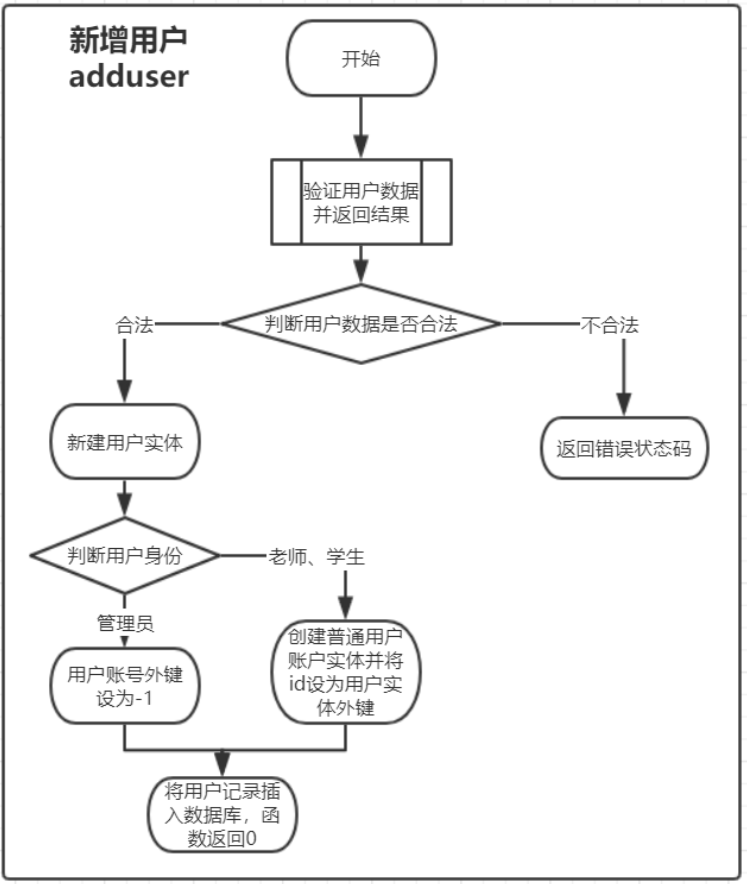
“createTime”:”2020-03-12”,

“flag”:1

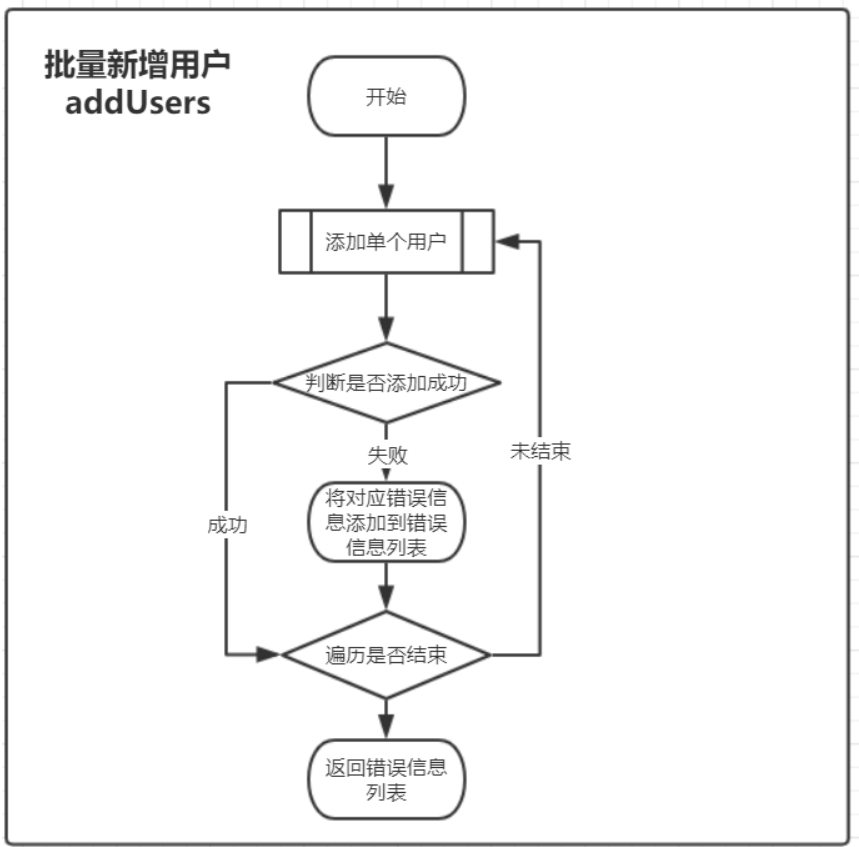
}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| getQuestionById | 查询存在的问题 | 返回问题的数据库实体数据 | 符合预期 | 正确 |
| 查询不存在的问题 | 返回null | 符合预期 | 正确 |
| addRelationToUId | 添加查询问题与查者之间的关系（没关注，有投诉） | questionVO.doesAttention=false  questionVO.doesReported=true | 符合预期 | 之前 |
| 添加查询问题与查者之间的关系（有关注，没投诉） | questionVO.doesAttention=true  questionVO.doesReported=false | 符合预期 | 正确 |

1. **用户模块**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| addUser() | 添加正确的学生用户 | 数据库成功添加用学生的用户记录，并为其创建账户 | 符合预期 | 正确 |
| 添加正确的老师用 | 数据库成功添加用学生的用户记录，并为其创建账户 | 符合预期 | 正确 |
| 添加正确的管理员用户 | 数据库成功添加用学生的用户记录，账户外键为-1 | 符合预期 | 正确 |
| 添加账号已经存在的用户 | 没有添加对应记录  返回状态码412 | 符合预期 | 正确 |
| 添加身份证已经存在的用户 | 没有添加对应记录  返回状态码413 | 符合预期 | 正确 |
| 添加账号错误的账号 | 没有添加对应记录  返回状态码414 | 符合预期 | 正确 |
| 添加身份证错误的用户 | 没有添加对应记录  返回状态码415 | 符合预期 | 正确 |
| 添加身份错误的用户 | 没有添加对应记录  返回状态码421 | 符合预期 | 正确 |
| 添加账号为空的用户 | 没有添加对应记录  返回状态码417 | 符合预期 | 正确 |
| 添加身份证为空的用户 | 没有添加对应记录  返回状态码419 | 符合预期 | 正确 |
| 添加身份为空的用户 | 没有添加对应记录  返回状态码420 | 符合预期 | 正确 |
| 添加姓名为空的用户 | 没有添加对应记录  返回状态码418 | 符合预期 | 正确 |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
| addUsers | 添加一批正确的用户 | 返回错误信息列表个数为0，  数据库成功创建列表中包含用户 | 符合预期 | 正确 |
| 添加一批全错的用户 | 返回错误信息列表个数和传入长度相同  错误用户信息没有被添加到数据库 | 符合预期 | 正确 |
| 添加部分正确部分错误用户列表 | 返回错误信息列表长度为错误个数  正确用户成功添加 | 符合预期 | 正确 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试函数 | 测试描述 | 预期结果 | 实际结果 | 测试评价 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |