钉钉项目一些整理

1.

所有的用户数据，都是融创那边已经建立好的，比方群组——浙江医联体，旗下很多个医院，医院下面是所有医生用户。

而我们给他们的，我们数据库的医生资料，他们是单独备份的，因为有可能他们那边有，也可能没有。

同样的，我们从他们那边获取的钉钉用户，和我们的用户表进行对接，可能是有的，那我就得获取他的权限啊，所在机构啊，服务中心啊，拥有哪些可以进行的服务啊。这样。

这应该就是我要做的业务整理。

而如果该用户，我们的用户表里面没有数据，我们还得重新建立？并让他跳到注册平台？但这里，医生的权限，不是应该来自他所在的机构，服务中心吗？

那要让医生重新去和服务中心，机构，去建立注册关系吗？

还是后台管理员去加这个医生信息到用户表中的？

2.

比方医生A点击进入到我们的会诊，没有注册的，是否可以用惯用的，直接使用钉钉登陆，我们调用一个钉钉的接口，而获取用户的头像啊，信息啊，这些所有信息，然后在我们的用户表中加入这些所有的数据，就相当于注册了。

同样的，每次直接使用钉钉信息进行登陆，我们后台都要验证是否表里已经有这个数据，如果有的话，

3.

义乌云的整理：

分析了下为什么我们这边只有这几个调用，比方调用当前登录用户信息，而不是搜索的李智贤的医生信息，因为是对方获取我们登陆的用户guid，调取我们的接口获取当前登录用户的信息，日志在我们这里，因为调的是我们的接口。就相当于我们和融创对接，我们去融创那里调取用户信息UserId，日志肯定在融创那边。

在那边调用接口，info就保存在哪边的日志中。

而我搜索出来的医生信息这些，就是在对方的日志当中，因为是我调用他的接口。

却决于谁调用谁。

4.

5.

开发整理：

1.

最后回复时间，初始值是提问时间，一旦有聊天的时间，就获取该问题的最后回复时间，覆盖提问的初始时间。

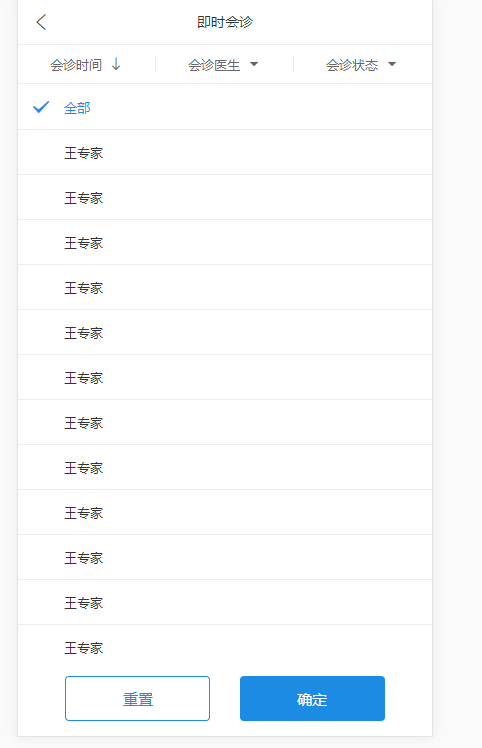
2.

这里的会诊查询，

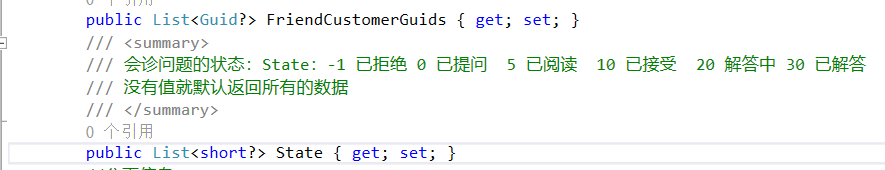


不传参数就是默认的全部选择。

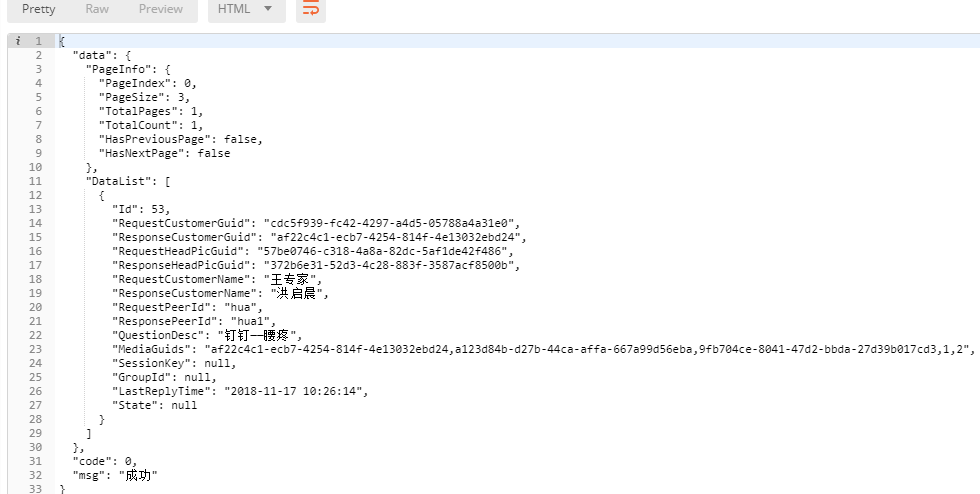
另外的会诊医生和会诊时间也是，不传就默认的显示所有的分页数据，不过滤。



3.



这里是可以查到null值的，我还是把这个bug填了吧，也不用什么预留，方便以后可以查null值了。



4.

提问方才有结束会诊的按钮。

而解答方有拒绝会诊的按钮。

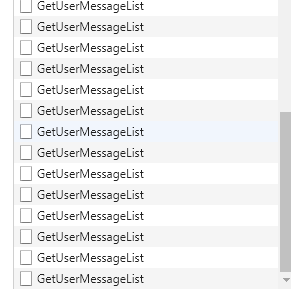
（可根据当前登录用户的guid和问题返回列表的requestGuid，responseGuid进行对比）

5.

5.1 轮询的问题（目前需要设置三个轮询）

（1）

获取消息，web走的是redis中获取，打开该页面的时候一直轮询



那我这里的方案，也是只要打开了聊天的页面，就一直轮询是否有新的消息。

（2）即时资讯问题列表，也就是会诊的问题列表也是需要轮询，一旦最后的提交时间变了，就相当于有新消息来了。就可以给红点了。

轮询的条件，可以去限定State的状态为0，因为其余状态值的聊天即时功能已经被关闭。



（3）问题详情中一对一的会诊列表也是，轮询，一旦最后的提交时间变了，就给红点。



6.

实时聊天这里，是不传头像的，因为在问题列表中，已经把头像传给前端了。

所以这里我就不用传了。

{

"Id": 52,

"RequestCustomerGuid": "cdc5f939-fc42-4297-a4d5-05788a4a31e0",

"ResponseCustomerGuid": "a123d84b-d27b-44ca-affa-667a99d56eba",

"RequestHeadPicGuid": "5eef797b-c006-4531-87f2-2aa63d3d022e",

"ResponseHeadPicGuid": "5eef797b-c006-4531-87f2-2aa63d3d022e",

"RequestCustomerName": "王专家",

"ResponseCustomerName": "沈医生",

"RequestPeerId": "hua",

"ResponsePeerId": "hua2",

"QuestionDesc": "钉钉——头疼",

"MediaGuids": "af22c4c1-ecb7-4254-814f-4e13032ebd24,a123d84b-d27b-44ca-affa-667a99d56eba,9fb704ce-8041-47d2-bbda-27d39b017cd3,1,2",

"SessionKey": null,

"GroupId": null,

"LastReplyTime": "2018-11-18 17:25:21",

"State": 5

},

7.

发送消息这里，需要去鉴别当前问题的状态，当已经结束或者已经拒绝了，就关闭聊天功能。

返回前台的code码是41004

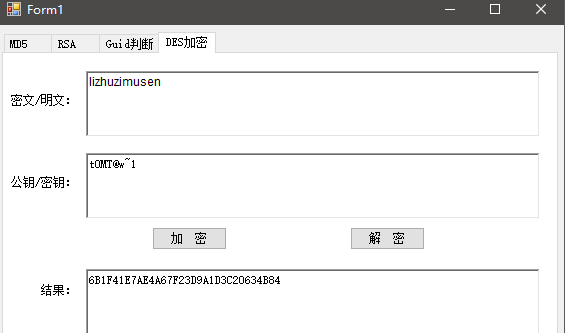
msg：聊天功能已关闭

发消息的时候我会在数据库中默认创建一张内容和图片均为null的数据，交给前端的时候，王毅判断下，这条不要去显示。

也不用，我在GetMsgs的时候，获取不为空的就ok了。

8.

key：6B1F41E7AE4A67F23D9A1D3C20634B84



9.

10.

11.

12.

记录：

1.

新的掌握的，去重方法。

#region 获取钉钉远程会诊问题列表(第二种方式，可以优化分页，但是要现在QuestionProcess中加入初始的时间，否则连不了这张表，没有数据）

/// <summary>

/// 获取钉钉远程会诊问题列表

/// </summary>

/// <returns>获取钉钉远程会诊问题列表的返回结果</returns>

public PagedList<DingQuestionsListModel> GetDingQuestionsList(GetDingQuestionsModelParam param)

{

var query = from qt in \_questionTargetRepository.Table

join q in \_questionRepository.Table on qt.QuestionId equals q.Id

join qp in \_questionProcessRepository.Table on qt.QuestionId equals qp.QuestionId

//join c in \_customerRepository.Table on q.CreateCustomerGuid equals c.CustomerGuid

where qt.TargetCustomerGuid == param.CustomerGuid || q.CreateCustomerGuid == param.CustomerGuid

//默认是IM来调用

where q.Kind == param.Kind

//按最新的回复时间来

//orderby qp.SubmTime descending

select new DingQuestionsListModel()

{

Id = q.Id,

RequestCustomerGuid = q.CreateCustomerGuid,

RequestCustomerName = q.CreateCustomerName,

//请求方头像

//（2018年11月13日17:10:30）数据库中的int类型变成了string类型

RequestPeerId = q.CreatePeerId,

//回答方

ResponseCustomerGuid = qt.TargetCustomerGuid,

ResponseCustomerName = qt.TargetCustomerName,

ResponsePeerId = qt.TargetPeerId,

//HeadPicGuid = c.HeadPicGuid,

QuestionDesc = q.QuestionDesc,

LastReplyTime = qp.SubmTime ?? q.CreateTime,

//钉钉这里，SessionKey和 GroupId暂时不要

//SessionKey = q.SessionKey,

//GroupId = q.GroupId,

MediaGuids = q.MediaGuids,

State = q.State

};

//当有具体的会诊问题对象

if (param.FriendCustomerGuids.Count > 0)

{

//传进来的guid对应提问方，也就是包含提问方Guid

//query = query.Where(q => param.FriendCustomerGuids.Contains(q.RequestCustomerGuid.Value));

//改为传进来的数据包含选中成员的结果

query = query.Where(q => param.FriendCustomerGuids.Contains(q.RequestCustomerGuid.Value) || param.FriendCustomerGuids.Contains(q.ResponseCustomerGuid.Value));

}

//当有钉钉远程会诊状态值传过来

if (param.State != null && param.State.Count > 0)

{

query = query.Where(q => param.State.Contains(q.State.Value));

}

//当有开始，截止时间传进来

//申请开始时间

if (param.BeginTime.HasValue && param.BeginTime.Value != DateTime.MinValue)

{

query = query.Where(q => q.LastReplyTime >= param.BeginTime.Value);

}

//申请结束时间

if (param.EndTime.HasValue && param.EndTime.Value != DateTime.MinValue)

{

query = query.Where(q => q.LastReplyTime < param.EndTime.Value);

}

//query.Distinct()

//var questionsList = query.ToList();

//这里都要提到BLL层去

//foreach (DingQuestionsListModel questionsListMoel in query)

//{

// var requestCustomer =

// \_customerRepository.Table.Where(c => c.CustomerGuid == questionsListMoel.RequestCustomerGuid).FirstOrDefault();

// var responseCustomer =

// \_customerRepository.Table.Where(c => c.CustomerGuid == param.CustomerGuid).FirstOrDefault();

// if (requestCustomer != null)

// {

// questionsListMoel.RequestHeadPicGuid = requestCustomer.HeadPicGuid;

// }

// if (responseCustomer != null)

// {

// questionsListMoel.ResponseHeadPicGuid = responseCustomer.HeadPicGuid;

// }

//}

query = query.OrderByDescending(q => q.LastReplyTime).DistinctBy(q => q.Id).AsQueryable();

PagedList<DingQuestionsListModel> page = new PagedList<DingQuestionsListModel>(query, param.PageIndex, param.PageSize);

return page;

}

#endregion

分组方式：

#region 第二版本测试，要删除：获取钉钉远程会诊问题列表

/// <summary>

/// 获取钉钉远程会诊问题列表

/// </summary>

/// <returns>获取钉钉远程会诊问题列表的返回结果</returns>

public PagedList<DingQuestionsListModel> GetDingQuestionsList(GetDingQuestionsModelParam param)

{

//根据当前用户的guid获取提问对象表中的问题id

//根据问题id获取 提问人guid，提问人姓名，提问人PeerId，sessionKey，GroupId，

//根据提问人guid获取其头像

//连三个表一次性查出来吧

//当前登陆用户为回答方

var query = from qt in \_questionTargetRepository.Table

join q in \_questionRepository.Table on qt.QuestionId equals q.Id

join qp in \_questionProcessRepository.Table on qt.QuestionId equals qp.QuestionId

//join c in \_customerRepository.Table on q.CreateCustomerGuid equals c.CustomerGuid

where qt.TargetCustomerGuid == param.CustomerGuid || q.CreateCustomerGuid == param.CustomerGuid

//默认是IM来调用

where q.Kind == param.Kind

//按最新的回复时间来

//orderby qp.SubmTime descending

select new DingQuestionsListModel()

{

Id = q.Id,

RequestCustomerGuid = q.CreateCustomerGuid,

RequestCustomerName = q.CreateCustomerName,

//请求方头像

//（2018年11月13日17:10:30）数据库中的int类型变成了string类型

RequestPeerId = q.CreatePeerId,

//回答方

ResponseCustomerGuid = qt.TargetCustomerGuid,

ResponseCustomerName = qt.TargetCustomerName,

ResponsePeerId = qt.TargetPeerId,

//HeadPicGuid = c.HeadPicGuid,

QuestionDesc = q.QuestionDesc,

LastReplyTime = qp.SubmTime ?? q.CreateTime,

//钉钉这里，SessionKey和 GroupId暂时不要

//SessionKey = q.SessionKey,

//GroupId = q.GroupId,

MediaGuids = q.MediaGuids,

State = q.State

};

//当有具体的会诊问题对象

if (param.FriendCustomerGuids.Count > 0)

{

//传进来的guid对应提问方，也就是包含提问方Guid

//query = query.Where(q => param.FriendCustomerGuids.Contains(q.RequestCustomerGuid.Value));

//改为传进来的数据包含选中成员的结果

query = query.Where(q => param.FriendCustomerGuids.Contains(q.RequestCustomerGuid.Value) || param.FriendCustomerGuids.Contains(q.ResponseCustomerGuid.Value));

}

//当有钉钉远程会诊状态值传过来

if (param.State != null && param.State.Count > 0)

{

query = query.Where(q => param.State.Contains(q.State.Value));

}

//当有开始，截止时间传进来

//申请开始时间

if (param.BeginTime.HasValue && param.BeginTime.Value != DateTime.MinValue)

{

query = query.Where(q => q.LastReplyTime >= param.BeginTime.Value);

}

//申请结束时间

if (param.EndTime.HasValue && param.EndTime.Value != DateTime.MinValue)

{

query = query.Where(q => q.LastReplyTime < param.EndTime.Value);

}

//query.Distinct()

//var questionsList = query.ToList();

//这里都要提到BLL层去

//foreach (DingQuestionsListModel questionsListMoel in query)

//{

// var requestCustomer =

// \_customerRepository.Table.Where(c => c.CustomerGuid == questionsListMoel.RequestCustomerGuid).FirstOrDefault();

// var responseCustomer =

// \_customerRepository.Table.Where(c => c.CustomerGuid == param.CustomerGuid).FirstOrDefault();

// if (requestCustomer != null)

// {

// questionsListMoel.RequestHeadPicGuid = requestCustomer.HeadPicGuid;

// }

// if (responseCustomer != null)

// {

// questionsListMoel.ResponseHeadPicGuid = responseCustomer.HeadPicGuid;

// }

//}

//去重

//list=list.GroupBy(t=>t.issuercode).Select(t=>t.First()).ToList();

var query1 = query.ToList();

var query2 = query1.GroupBy(q => q.Id).Select(q => q.FirstOrDefault()).ToList();

//这里是IM的，默认按照最后的提交时间去排序

// questionsList = questionsList.OrderByDescending(q => q.LastReplyTime).ToList();

//钉钉的记录先是按状态从小到大去排列，然后再按提交时间先后倒序去排，在bll层再根据咨询表中的数据重新再排

var query3 =

query2.OrderByDescending(q => q.LastReplyTime)

.ThenBy(q => q.State).ToList();

//query = query.OrderByDescending(q => q.LastReplyTime).ThenByDescending(q => q.State).Distinct(new DistinctExtensions<DingQuestionsListModel>("Id"));

PagedList<DingQuestionsListModel> page = new PagedList<DingQuestionsListModel>(query3, param.PageIndex, param.PageSize);

return page;

//return questionsList;

}

#endregion

2.

3.

4.