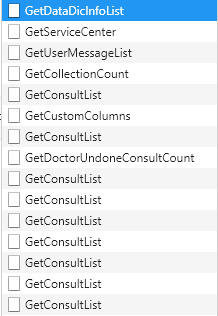
远程会诊探索

是

1.

登陆之后，当点击量远程会诊。



这边每改一次筛选条件，就相当于调用一次GetConsultList接口。



（1）



获取字典数据：

public List<DataDicInfo> GetDataDicInfos(List<string> lookupKeys)

{

var query = from s in \_datadicinfoRepository.Table

where lookupKeys.Contains(s.LookupKey)

select s;

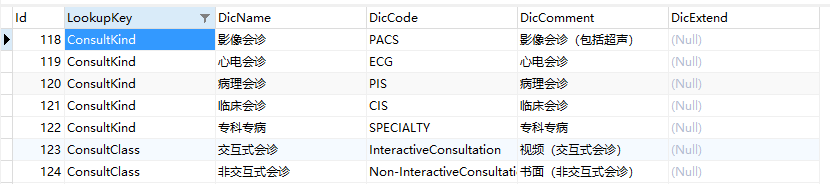
return query.AsNoTracking().ToList();

}

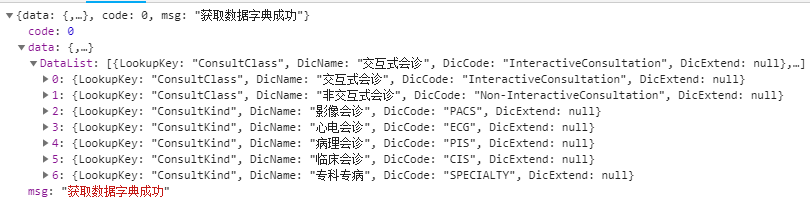
传参：



对应数据库中的结果：



后台返回response的数据:



将结果展示在前台页面：



这里涉及MapList

/// <summary>

/// 数据字典

/// </summary>

public partial class DataDicInfo : BaseEntity

{

/// <summary>

/// 字典类型

/// </summary>

public string LookupKey { get; set; }

/// <summary>

/// 字典名称

/// </summary>

public string DicName { get; set; }

/// <summary>

/// 字典代码

/// </summary>

public string DicCode { get; set; }

/// <summary>

/// 补充说明

/// </summary>

public string DicComment { get; set; }

/// <summary>

/// 扩展

/// </summary>

public string DicExtend { get; set; }

}

从上面的变成下面的

/// <summary>

/// 数字字典模型

/// </summary>

public class DataDicInfoModel

{

/// <summary>

/// 字典类型

/// </summary>

public string LookupKey { get; set; }

/// <summary>

/// 字典名称

/// </summary>

public string DicName { get; set; }

/// <summary>

/// 字典代码

/// </summary>

public string DicCode { get; set; }

/// <summary>

/// 扩展

/// </summary>

public string DicExtend { get; set; }

}

上面还有从BaseEntity获取的，比方Id，就是6个数据变成了4个属性的数据。

result.data = new ApiListModel<DataDicInfoModel>

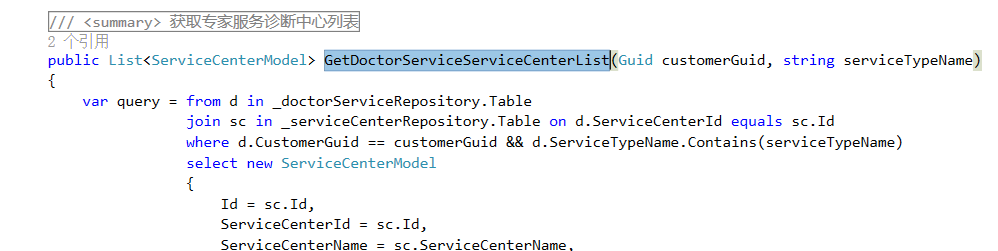
{

DataList = \_dataDicInfoService.GetDataDicInfos(keys).MapToList<DataDicInfoModel>()

};

（2）



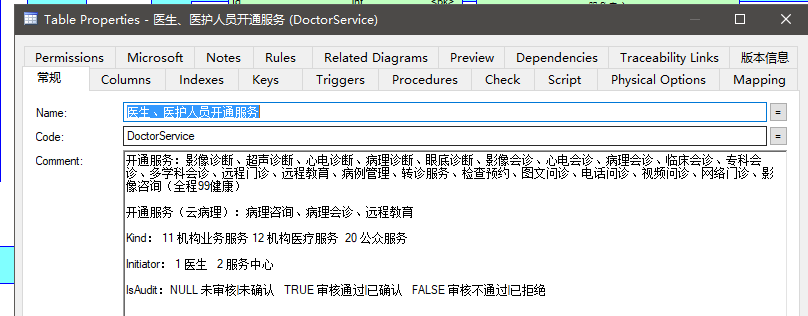


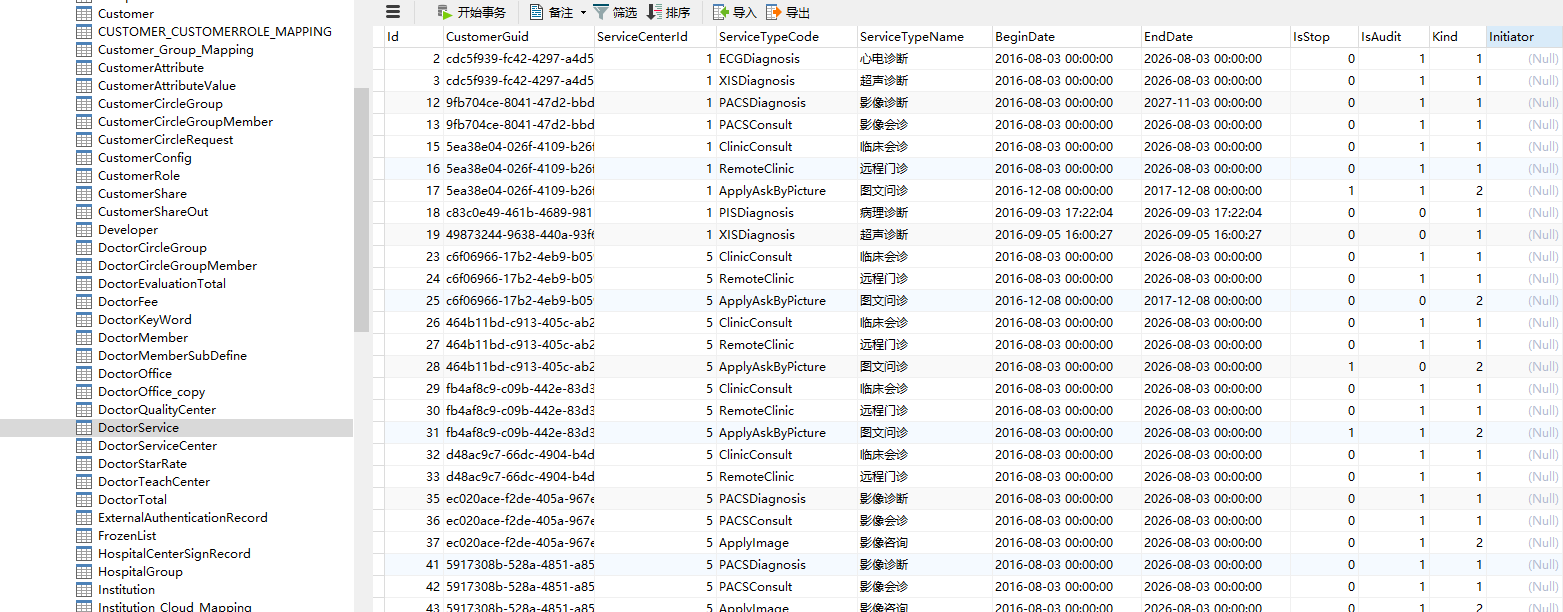
根据当前用户去匹配对应的服务中心列表

两张表：

ServerCenter服务中心表







后台返回：





返回到前台：



（3）



<http://localhost:9500/Doctor/Common/GetUserMessageList>

对应异步方法：

#region 站内消息管理

/// <summary>

/// 站内消息管理

/// </summary>

/// <returns></returns>

[HttpPost]

public async Task<ActionResult> GetUserMessageList()

{

var user = CustomerIdentity;

dynamic apiResult = null;

await Task.Factory.StartNew(() =>

{

apiResult =

\_commonClient.Invoke(m => m.GetUserMessageList(user.CustomerGuid));

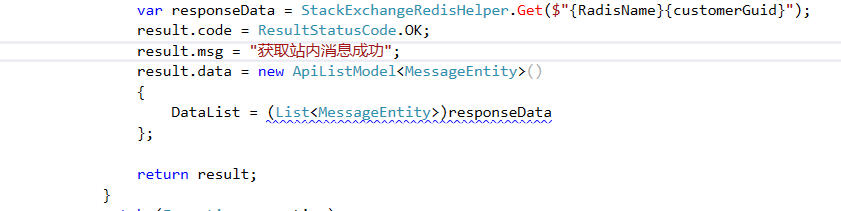
});

return Json(apiResult);

}

#endregion

从redis中获取到





一共265条数据

反馈到前端页面：



（4）



<http://localhost:9500/Doctor/Collection/GetCollectionCount>

查询收藏总数：

/// <summary>

/// 查询收藏总数

/// </summary>

/// <returns></returns>

[HttpPost]

public async Task<JsonResult> GetCollectionCount()

{

dynamic ret = null;

var user = CustomerIdentity;

await Task.Factory.StartNew(() =>

{

ret = \_crmClient.Invoke(s => s.GetCollectionCount(user.CustomerGuid, collectionSystem));

});

return Json(ret);

}

dal：

/// <summary>

/// 查询用户收藏总数

/// </summary>

/// <param name="customerGuid">用户唯一号</param>

/// <param name="CollectionSystem">系统来源</param>

/// <returns></returns>

public int GetCollectionCount(Guid customerGuid, int? collectionSystem)

{

var query = from s in \_myCollectionService.Table

where s.CustomerGuid == customerGuid

select s;

if (collectionSystem != null)

{

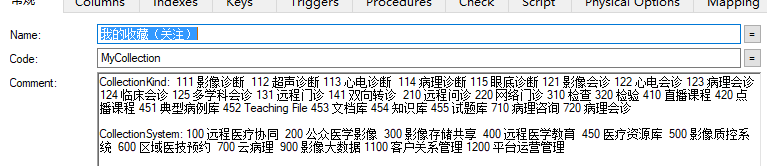
query = query.Where(s => s.CollectionSystem == collectionSystem);

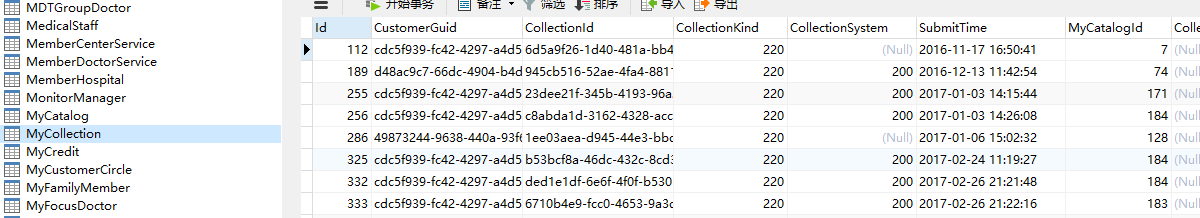
}

return query.Count();

}

表：





反馈到前端页面：



（5）



<http://localhost:9500/Doctor/ConsultV2/GetCustomColumns>

入参：



获取列表自定义列

#region 获取列表自定义列

/// <summary>

/// 获取列表自定义列

/// </summary>

/// <returns></returns>

public JsonResult GetCustomColumns(DoctorCustomColumnsParam param)

{

var result = new BaseApiModel<CustomerConfigModel>();

param.CustomerGuid = CustomerIdentity.CustomerGuid;

var apiResult = \_crmClient.Invoke(m => m.GetCustomerConfigModel(new CustomerConfigParamsModel

{

CustomerGuid = param.CustomerGuid,

ConfigItemName = param.ConfigItemName

}));

if (apiResult.data?.DataList != null && apiResult.data.DataList.Count > 0)

result.data = apiResult.data.DataList[0];

return Json(result);

}

#endregion

dal层：

/// <summary>

/// 获取配置项实体

/// </summary>

/// <param name="param"></param>

/// <returns></returns>

public List<CustomerConfig> GetModel(CustomerConfigParamsModel param)

{

var query = from cf in \_customerConfigRepository.Table select cf;

if (param.CustomerGuid != Guid.Empty)

{

query = query.Where(p => p.CustomerGuid == param.CustomerGuid);

}

if (param.ServiceCenterId > 0)

{

query = query.Where(p => p.ServiceCenterId == param.ServiceCenterId);

}

if (!string.IsNullOrWhiteSpace(param.ConfigItemName))

{

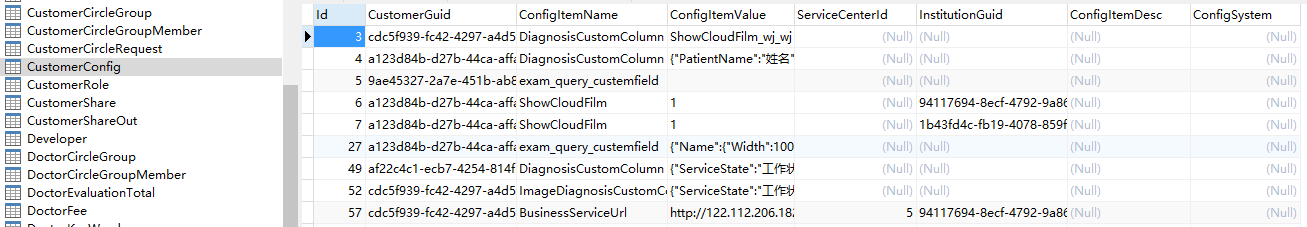
query = query.Where(p => p.ConfigItemName == param.ConfigItemName);

}

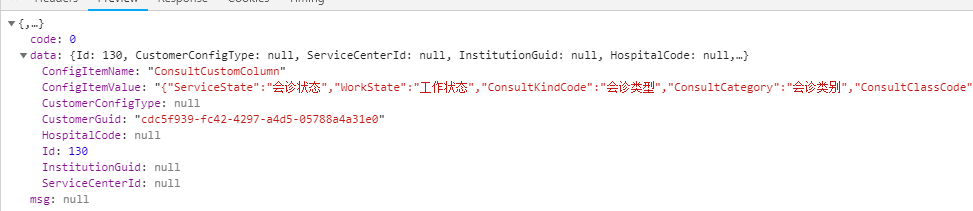
return query.ToList();

}

CustomerConfig表：



后台返回：

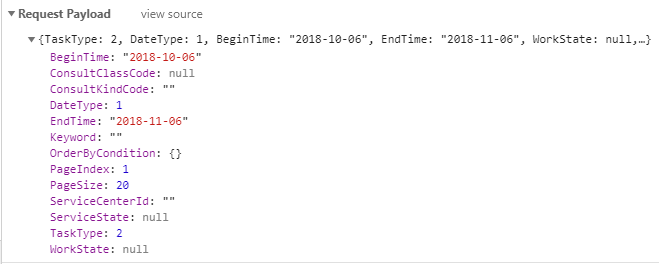


（6）



<http://localhost:9500/Doctor/ConsultV2/GetDoctorUndoneConsultCount>

入参：



处理：

获取未完成任务的数量：

#region 获取未完成任务的数量

/// <summary>

/// 获取未完成任务的数量

/// </summary>

/// <param name="param"></param>

/// <returns></returns>

public int GetDoctorUndoneConsultCount(ConsultListParam param)

{

var query = from ac in \_applyConsultRepository.Table

join dr in \_applyDiagnosisReplyRepository.Table on ac.ServiceId equals dr.ServiceId into t

from lt in t.DefaultIfEmpty()

where lt.CustomerGuid == param.CustomerGuid && ((ac.ServiceState < 40 && ac.ServiceState >= 20) || (ac.ConsultClassCode == "Non-InteractiveConsultation" && ac.ServiceState == 0))

select new

{

ac.Id,

ac.ServiceId,

ac.ServiceCenterId,

ac.CustomerGuid,

ac.CustomerName,

ac.RequestHospitalCode,

ac.ConsultClassCode,

ac.BusinessId,

ac.ServiceState,

ac.IsUrgency,

ac.RequestDate,

ac.ConsultKindCode,

ac.ConsultDate,

ac.FeeMode,

ac.PatientName,

ac.ServiceCenterName,

ac.IsCall,

ac.RequestHospitalName,

ac.IsNoName,

ac.ExpertGuids

};

if (param.RequestBeginTime.HasValue)

{

query = query.Where(p => p.RequestDate >= param.RequestBeginTime);

}

if (param.RequestEndTime.HasValue)

{

query = query.Where(p => p.RequestDate <= param.RequestEndTime);

}

if (param.ConsultBeginTime.HasValue)

{

query = query.Where(p => p.ConsultDate >= param.ConsultBeginTime);

}

if (param.ConsultEndTime.HasValue)

{

query = query.Where(p => p.ConsultDate <= param.ConsultEndTime);

}

if (param.ServiceCenterId > 0)

{

query = query.Where(p => p.ServiceCenterId == param.ServiceCenterId);

}

if (!string.IsNullOrWhiteSpace(param.ConsultKindCode))

{

query = query.Where(p => p.ConsultKindCode == param.ConsultKindCode);

}

if (!string.IsNullOrWhiteSpace(param.ConsultClassCode))

{

query = query.Where(p => p.ConsultClassCode == param.ConsultClassCode);

}

return query.Count();

}

#endregion

后台返回：

{

"data": 0,

"code": 0,

"msg": "获取医生未完成统计量成功"

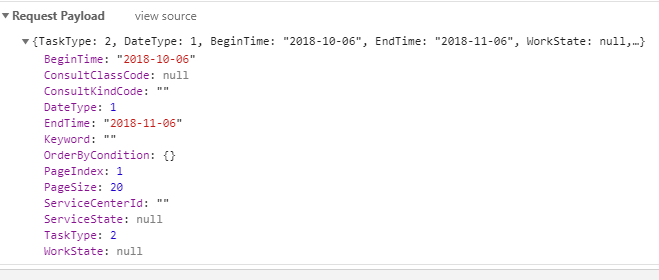
}

（7）



每点击一次就会触发而返回一次结果。

<http://localhost:9500/Doctor/ConsultV2/GetConsultList>



#region 获取工作任务列表

/// <summary>

/// 获取工作任务列表

/// </summary>

/// <returns></returns>

public JsonResult GetConsultList(ConsultParam param)

{

var result = new BaseApiModel<ApiPageListModel<ConsultListModel>>();

涉及bll层的各种拼接：

#region 获取工作列表

/// <summary>

/// 获取工作列表

/// </summary>

/// <param name="param"></param>

/// <returns></returns>

public BaseApiModel<ApiPageListModel<ConsultListModel>> GetConsultList(ConsultListParam param)

{

var result = new BaseApiModel<ApiPageListModel<ConsultListModel>>();

try

{

#region 参数校验

var checkResult = new ValidateParaExtend<ConsultListParam>(param).IsNull("搜索条件不能为空")

.Check(p => param.CustomerGuid == Guid.Empty, "用户唯一号不能为空");

if (checkResult.IsStop)

{

result.code = ResultStatusCode.LACK\_PARA;

result.msg = checkResult.Msg;

}

#endregion

//获取医生签约的服务中心

var doctorServiceCenter = \_doctorServiceCenterService.GetServiceCenterIdList(param.CustomerGuid);

//获取医生签约的服务中心信息

List<ServiceCenterModel> serviceCenters = \_institutionService.GetServiceCenterByIds(doctorServiceCenter);

//获取医生开通的服务

DoctorServiceListParam doctorServiceListParam = new DoctorServiceListParam()

{

CustomerGuid = param.CustomerGuid

};

List<DoctorService> doctorServices = \_doctorServiceService.GetDoctorServiceList(doctorServiceListParam);

//获取开通非交互式自领取的服务中心

List<int> isAutoConsultServiceCenterId =

serviceCenters.Where(p => p.IsAutoConsult.GetValueOrDefault(true))

.Select(p => p.ServiceCenterId)

.Distinct()

.ToList();

//赋值医生签约的服务

foreach (var serviceCenterId in isAutoConsultServiceCenterId)

{

List<string> openService =

doctorServices.Select(p => ServiceTypeConvert.ConsultKindCode(p.ServiceTypeCode))

.Distinct()

.Where(s => !string.IsNullOrEmpty(s))

.ToList();

if (openService.Count < 1)

{

continue;

}

param.DoctorOpenService.Add(new DoctorOpenServiceModel()

{

ServiceCenterId = serviceCenterId,

ServiceTypeCode = openService

});

}

#region 返回结果

if (param.PageIndex > 0)

{

param.PageIndex = param.PageIndex - 1;

}

if (param.PageSize < 1 || param.PageSize > 20)

{

param.PageSize = 20;

}

var consultList = \_applyConsultService.GetConsultList(param);

result.code = ResultStatusCode.OK;

result.msg = "查询会诊列表成功";

result.data = new ApiPageListModel<ConsultListModel>()

{

DataList = consultList,

PageInfo = new PageInfo

{

HasNextPage = consultList.PageIndex + 1 < consultList.TotalPages,

HasPreviousPage = consultList.PageIndex > 0,

PageIndex = consultList.PageIndex + 1,

PageSize = consultList.PageSize,

TotalCount = consultList.TotalCount,

TotalPages = consultList.TotalPages

}

};

return result;

#endregion

}

catch (Exception exception)

{

SystemLogHelper.LogInsert(new AddLogParam { ShortMessage = "协同-[会诊]-获取会诊列表", FullMessage = $"传入参数：{JsonConvert.SerializeObject(param)}，详细错误：{exception}" });

result.code = ResultStatusCode.SYSTEM\_BUSY;

result.msg = "获取会诊列表失败";

return null;

}

}

后台返回：

{

"data": {

"PageInfo": {

"PageIndex": 1,

"PageSize": 20,

"TotalPages": 0,

"TotalCount": 0,

"HasPreviousPage": false,

"HasNextPage": false

},

"DataList": []

},

"code": 0,

"msg": "查询会诊列表成功"

}

2.

回家看下maplist

3.

4.

5.