先总结出有哪些场景

1. 内部交流通道：

如AD\_1 、AU\_1 、CS\_1 交流

1、其中一个人登录了，后端通过websocket将内部交流通道未读的消息以json格式一次性发送

如下，客服CS\_1 上线首先接收这个json格式的对象数组，显示小红点。

{

“type” : “chat/unread”

“msg” : [

{

“type” : “chat” ,//内部交流通道

“id” : 1,

“senderSessionId” : “AD\_1” ,

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : false

}，

{

“type” : “chat” ,

“id”: 2,

“senderSessionId” : “AU\_1” ,

“content” : “管理员好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 11234326000 ,

“isRead” : false

}

]

}

2、客服CS\_1点入内部交流通道，立即通过websocket 发送以下消息

{

“token” : xxxxxxx ,

“type” : “chatIsRead”,

“id” : [1,2]

}

后端针对CS\_1修改消息状态isread

1. 客服进入内部交流通道后，获取历史信息，请求/chat/history (暂定)，get方式请求

请求带token字段即可

响应体如下

{

“type” : “chat/history”

“msg” : [

{

“type” : “chat”,

“id” : 0,

“senderSessionId” : “AD\_1” ,

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

{

“type” : “chat” ,//内部交流通道

“id” : 1 ,

“senderSessionId” : “AD\_1” ,

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

{

“type” : “chat” ,

“id”: 2,

“senderSessionId” : “AU\_1” ,

“content” : “管理员好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 11234326000 ,

“isRead” : true

}

后两个，因为之前已经向后端发送确认读取的信息，现在已被列为历史消息，需要前端自行去重

]

}

1. CS\_1想发一条消息，通过websocket

{

“type” : “chat/send”

“msg” :

{

“type” : “chat” ,//内部交流通道

“senderSessionId” : “CS\_1” , //这个个人信息前端自行可以从token解析出来

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

//目前全为”text”

“timestamp” : 112343242343

//去除isRead字段，自己发送的肯定自己已读

}

}

1. 后端转发给其他人,通过websocket

{

“type”:”chat/receive”

“msg”:

{

“type” : “chat” ,//内部交流通道

“id” : 3,

“senderSessionId” : “CS\_1” , //这个个人信息前端自行可以从token解析出来

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

//目前全为”text”

“timestamp” : 112343242343 ,

//恢复isRead字段

“isRead” : false

}

}

1. 读取后重回到2

即发送{

“token” : xxxxxxx ,

“type” : “chatIsRead”,

“id” : [3]

}

1. 用户和客服的反馈：  
   如CS\_1 、CU\_1交流

1、其中一个人登录了，后端通过websocket将内部交流通道未读的消息以json格式一次性发送

如下，客服CS\_1 上线首先接收这个json格式的对象数组，显示小红点。

{

“type” : “feedback/unread”

“msg” : [

{

“type” : “feedback” ,//反馈

“id” : 1,

“senderSessionId” : “CU\_1” ,

“content” : “你好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : false

}，

{

“type” : “feedback” ,

“id”: 2,

“senderSessionId” : “CU\_1” ,

“content” : “我有一个问题” ,(加密过)

“contentType” : “text” ,

“timestamp” : 11234326000 ,

“isRead” : false

}

]

}

2、客服CS\_1点入内部交流通道，立即通过websocket 发送以下消息

{

“token” : xxxxxxx ,

“type” : “feedbackIsRead”,

“id” : [1,2]

}

后端针对CS\_1修改消息状态isread

1. 客服进入反馈界面后，获取历史信息，请求/feedback/history (暂定)，get方式请求

请求带token、csSessionId 、 cuSessionId

响应体如下

{

“type” : “feedback/history”

“msg” : [

{

“type” : “feedback”,

“id” : 0,

“senderSessionId” : “CS\_1” ,

“content” : “你好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

{

“type” : “feedback” ,

“id” : 1 ,

“senderSessionId” : “CU\_1” ,

“content” : “你好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

{

“type” : “feedback” ,

“id”: 2,

“senderSessionId” : “CU\_1” ,

“content” : “我有一个问题” ,(加密过)

“contentType” : “text” ,

“timestamp” : 11234326000 ,

“isRead” : true

}

后两个，因为之前已经向后端发送确认读取的信息，现在已被列为历史消息，需要前端自行去重

]

}

1. CS\_1想发一条消息，通过websocket

{

“type” : “feedback/send”

“msg” :

{

“type” : “feedback” ,

“senderSessionId” : “CS\_1” , //这个个人信息前端自行可以从token解析出来

“receiverSessionId” : “CU\_1”

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

//目前全为”text”

“timestamp” : 112343242343

//去除isRead字段，自己发送的肯定自己已读

}

}

此处CU\_1发送**第一条**消息前应该向/feedback/getId 请求

携带token

后端返回

{

sessionId : “CS\_1”

}

再向CS\_1发送请求，格式如上，反之客服暂时设计成不能主动和用户联系

1. 后端转发给另一个人,通过websocket

如CU\_1接收到

{

“type”:”feedback/receive”

“msg”:

{

“type” : “feedback” ,

“id” : 3

“senderSessionId” : “CS\_1” , //这个个人信息前端自行可以从token解析出来

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

//目前全为”text”

“timestamp” : 112343242343 ,

//恢复isRead字段

“isRead” : false

}

}

1. 读取后重回到2 websocket

即发送{

“token” : xxxxxxx ,

“type” : “feedbackIsRead”,

“id” : [3] //这里的id是数组形式

}

1. 任务发布方和接受方的信息交流：

如CU\_1 在任务1 与 CU\_2交流，在任务2 与 CU\_3 交流 ，这个交流不分任务发送者和接受者，两方平等

1、其中一个人登录了，后端通过websocket将内部交流通道未读的消息以json格式一次性发送

如下,CU\_1 上线首先接收这个json格式的对象数组，显示小红点。

{

“type” : “talk/unread”

“msg” : [

{

“type” : “talk” ,

“id” : 1,

“taskId” : 1 , //增加taskId ,唯一确定是哪个任务的对话，用receiverId则不行

“senderSessionId” : “CU\_2” ,

“content” : “我是CU\_2” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : false

}，

{

“type” : “talk” ,

“id”: 2,

“taskId” : 2 , //增加taskId ,唯一确定是哪个任务的对话，用receiverId则不行

“senderSessionId” : “CU\_3” ,

“content” : “你好，为什么还没完成任务” ,(加密过)

“contentType” : “text” ,

“timestamp” : 11234326000 ,

“isRead” : false

}

]

}

2 用户 CU\_1点入任务编号为1的对话，立即通过websocket 发送以下消息

{

“token” : xxxxxxx ,

“type” : “talkIsRead”,

“id” : [1]

}//这里不需要taskId, 因为对话编号是唯一确定的，后端知道

后端针对CS\_1修改消息状态isread

1. 用户进入任务编号为2的对话后，获取历史信息，请求/talk/history (暂定)，get方式请求

请求带token、taskId

响应体如下

{

“type” : “talk/history”

“msg” : [

{

“type” : “talk”,

“id” : 0,

“taskId” : 2,

“senderSessionId” : “CU\_3” ,

“content” : “你好” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

{

“type” : “talk” ,

“id” : 2 ,

“taskId” ：2，

“senderSessionId” : “CU\_3” ,

“content” : “你好，为什么还没完成任务” ,(加密过)

“contentType” : “text” ,

“timestamp” : 112343242343 ,

“isRead” : true

}，

后一个，因为之前已经向后端发送确认读取的信息，现在已被列为历史消息，需要前端自行去重

]

}

1. CU\_1想发一条消息，通过websocket

{

“type” : “talk/send”

“msg” :

{

“type” : “talk” ,

“senderSessionId” : “CU\_1” ,

“receiverSessionId” ：”CU\_2”,//这个个人信息前端自行可以从token解析出来

“taskId” : 1,

“content” : “大家好” ,(加密过)

“contentType” : “text” ,//目前全为”text”

“timestamp” : 112343242343

//去除isRead字段，自己发送的肯定自己已读

}

}

1. 后端转发给其他人,通过websocket

CU\_2 此时收到消息

{

“type”:”talk/receive”

“msg”:

{

“type” : “talk” ,

“id” : 3,

“senderSessionId” : “CU\_1” , “receiverSessionId” ：”CU\_2”,

“taskId” : 1,

“content” : “大家好” ,(加密过)

“contentType” : “text” ,

//目前全为”text”

“timestamp” : 112343242343 ,

//恢复isRead字段

“isRead” : false

}

}

1. 读取后重回到2

即发送{

“token” : xxxxxxx ,

“type” : “talkIsRead”,

“id” : [3]

}

要注意的是一上线的websocket推送未读信息，后端暂时设计成分三种依次推送，而不是全部混合一并推送

还有任务的实时更新websocket

CU\_1 、CU\_2都是一个接收逻辑

{

“type” : “task/insert”//有新的待领取的任务发布

其余字段



}

{

“type” : “task/update”//状态码变化,

“task\_id” : xx,

“task\_staus” : “xxxx-xx”,

}