스케일러블 DM 시스템 구현을 위한 Documentation

참고: 기존 Documentation에 있던 채팅 관련 API는 전부 Deprecated.

세줄 요약:

1. 채팅 서버가 분리되고, 각 채팅 버블마다 고유 번호가 생긴다.
2. 고유 번호는 순서성이 있기 때문에 이를 기반으로 몇 개를 덜 읽었는지 쉽게 알 수 있다.
3. 대부분의 API가 RESTful 하게 바뀌어서, 이제 진짜 채팅 송수신만 socket으로 하면 된다.

Chat 스키마:  
텍스트, 스크린샷, 소프트웨어, 운영 체제이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다. isBanned\*는 차단 여부를 확인하는 용도이다.

차단 기능의 경우, 양 측에서 모두 차단을 해제해야 비로소 차단 리스트에 보이지 않게 된다.

BubbleSchema는 무엇인가?

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

이렇게 생겼다. 여기서 autoIncrementId가 정말 중요하니까 꼭 기억 해 두길 바란다.  
autoIncrementId를 기반으로 각 기기별로 어디까지 읽었고, 어디서부터 불러와야 하는지, 안 읽은 채팅은 몇 개인지를 계산 할 수 있다.

채팅 서버가 분리되었다. 그래서 “일부” API들은 완전히 다른 주소로 요청해야 하는데, 일단 그것부터 설명 들어간다.

그런데 또 주의해야 할 점!!! 토큰이 만료되었을 때 refresh 요청하는건 원래 주소랑 같다!!!

1. <http://jeongwoo-kim-web.myds.me:5000/mapp/dm/joinChat/:cid>?readedUntil=”someValue”
   1. GET
   2. 채팅에 진입할 때 호출하는 API이다.
   3. 요청 인자: cid자리에 Chat의 \_id 속성을 넣고,  
      readedUntil은 해당 기기에서 가장 마지막에 본 채팅의 autoIncrementId를 넣어주면 된다.
   4. 반환 인자: 해당 기기에서 아직 보지 않은 채팅의 BubbleSchema의 List.
   5. 유의 사항: 아직 보지 않은 채팅이 밀려 들어온 뒤에 새 채팅이 뜨고 발신이 이뤄져야겠죠?  
      또 스트링 하드코딩 하지 말고 최근에 리팩터링 한 것 처럼 리소스에서 불러오도록 설계 부탁드립니다.
   6. 오류 코드
      1. 403: 서버 내부 알고리즘 오류

Socket에서 의사는 /dm\_doctor 네임스페이스를 활용하고  
유저는 /dm\_user 네임스페이스를 활용한다.

1. RESTful API
   1. User용
      1. /mapp/dm/user/curateScreen
         1. POST
         2. 큐레이팅 화면에서 상담 요청 버튼을 클릭했을 때 반응하기 위한 API이다.
         3. 요청 인자: 
         4. 반환 인자: 채팅방이 기존에 없었으면 새로 만들어서, 기존에 있었다면 기존의 Chat의 \_id 속성을 반환한다.
         5. 오류 코드
            1. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 큐레이팅 화면에서 상담 요청 버튼을 클릭했을 때, 해당 의사(또는 상담사)와의 채팅창으로 바로 이동하기 위해 사용된다.
      2. /mapp/dm/user/list
         1. GET
         2. 채팅방 리스트를 보기 위한 API이다.
         3. 요청 인자: 없음(토큰으로 본인인증)
         4. 반환 인자: {date, cid, isBanned, doctorId, doctorName, recentChat} 객체의 배열을 반환한다. 여기서 recentChat 객체는 상기 BubbleSchema를 가지고 있기 때문에, 기기에 저장된, 내가 본 해당 채팅방의 가장 최근 채팅의 autoIncrementId와 recentChat이 반환하는 가장 최근 채팅의 autoIncrementId를 빼기 연산하면 몇 개를 덜 읽었는지 알 수 있다!
         5. 오류 코드
            1. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅 리스트 화면에서 아직 안읽은 채팅의 개수와 최근 메시지, 채팅방 업데이트 시간 순에 따른 정렬, 의사 이름 표시, 그리고 추후 추가될 프로필 사진불러오기에 대비해서 의사의 \_id 속성을 불러오는 용도로 쓰인다.
      3. /mapp/dm/user/delete/:cid
         1. DELETE
         2. 채팅방을 삭제하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 파라미터 인자로 넘겨줘야 한다.
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 채팅이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅을 삭제할 때 사용한다.
      4. /mapp/dm/user/ban
         1. POST
         2. 채팅방을 차단하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 body에 chatId라는 이름으로 넘겨줘야 한다.  
            
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 채팅이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅을 차단할 때 사용한다.
      5. /mapp/dm/user/banList
         1. GET
         2. 차단한 DM 목록을 확인하기 위한 API이다.
         3. 요청 인자: 없음
         4. 반환 인자: 정상 응답(200)시 차단한 DM 리스트를 반환한다. (chat 스키마의 list)
         5. 오류 코드
            1. 403: 서버 내부 알고리즘 오류
      6. /mapp/dm/user/unban
         1. DELETE
         2. 차단한 DM을 차단 해제하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 body에 chatId라는 이름으로 넘겨줘야 한다.  
            
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 \_id를 가진 채팅이 없음
            2. 402: 차단된 채팅이 아님
      7. /mapp/dm/user/appointmentStatus/:cid
         1. GET
         2. 약속의 수락 여부를 확인하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 cid 자리에 파라미터로 넘겨줘야 한다.
         4. 반환 인자: Boolean
         5. 오류 코드
            1. 401: 아직 약속이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. setInterval등으로 비동기적으로 몇 밀리초 마다 서버로 요청을 보내서 약속의 체결 여부를 확인한다.
   2. Doctor용
      1. /mapp/dm/doctor/list
         1. GET
         2. 채팅방 리스트를 보기 위한 API이다.
         3. 요청 인자: 없음(토큰으로 본인인증)
         4. 반환 인자: {date, cid, isBanned, userId, userName, recentChat,} 객체의 배열을 반환한다. 여기서 recentChat 객체는 상기 BubbleSchema를 가지고 있기 때문에, 기기에 저장된, 내가 본 해당 채팅방의 가장 최근 채팅의 autoIncrementId와 recentChat이 반환하는 가장 최근 채팅의 autoIncrementId를 빼기 연산하면 몇 개를 덜 읽었는지 알 수 있다!
         5. 오류 코드
            1. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅 리스트 화면에서 아직 안읽은 채팅의 개수와 최근 메시지, 채팅방 업데이트 시간 순에 따른 정렬, 의사 이름 표시, 그리고 추후 추가될 프로필 사진불러오기에 대비해서 환자의 \_id 속성을 불러오는 용도로 쓰인다.
      2. /mapp/dm/doctor/delete/:cid
         1. DELETE
         2. 채팅방을 삭제하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 파라미터 인자로 넘겨줘야 한다.
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 채팅이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅을 삭제할 때 사용한다.
      3. /mapp/dm/doctor/ban
         1. POST
         2. 채팅방을 차단하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 body에 chatId라는 이름으로 넘겨줘야 한다.  
            
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 채팅이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. 채팅을 차단할 때 사용한다.
      4. /mapp/dm/doctor/banList
         1. GET
         2. 차단한 DM 목록을 확인하기 위한 API이다.
         3. 요청 인자: 없음
         4. 반환 인자: 정상 응답(200)시 차단한 DM 리스트를 반환한다. (chat 스키마의 list)
         5. 오류 코드
            1. 403: 서버 내부 알고리즘 오류
      5. /mapp/dm/doctor/unban
         1. DELETE
         2. 차단한 DM을 차단 해제하기 위한 API이다.
         3. 요청 인자: Chat의 \_id 속성을 body에 chatId라는 이름으로 넘겨줘야 한다.  
            
         4. 반환 인자: 없음
         5. 오류 코드
            1. 401: 그런 \_id를 가진 채팅이 없음
            2. 402: 차단된 채팅이 아님
      6. /mapp/dm/user/appointmentStatus/:appid
         1. GET
         2. 약속의 수락 여부를 확인하기 위한 API이다.
         3. 요청 인자: Appointment의 \_id 속성을 appid 자리에 파라미터로 넘겨줘야 한다.
         4. 반환 인자: Boolean
         5. 오류 코드
            1. 401: 그러한 코드를 가진 약속이 없음
            2. 403: 서버 내부 알고리즘 오류
         6. 사용 방법
            1. setInterval등으로 비동기적으로 몇 밀리초 마다 서버로 요청을 보내서 약속의 체결 여부를 확인한다.
2. Socket API
   1. Socket에서 의사는 /dm\_doctor 네임스페이스를 활용하고  
      유저는 /dm\_user 네임스페이스를 활용한다.
   2.   
      Handshake시에 query에 토큰 뿐 만 아니라 chat의 \_id 속성인 방 번호까지 주어야 한다.
   3. 공통 오류
      1. notYourChatorBannedChat 이벤트
         1. 본인이 속한 채팅이 아니거나 밴 당한 채팅일 때 서버측에서 emit 된다.
      2. tokenExpiredError 이벤트
         1. 토큰이 만료되었을 때 서버측에서 emit 된다.
         2. 이 이벤트를 수신했다면 토큰을 리프레시 하면 된다.
   4. 공통 Receiver
      1. 채팅 시스템이 어떻게 바뀌었냐 하면,  
         도표, 텍스트, 스크린샷, 사각형이(가) 표시된 사진

         AI가 생성한 콘텐츠는 부정확할 수 있습니다.이렇게 바뀌었다. 다시 말해서, 내가 전송하고 땡이 아니라 그것이 서버에 도착해야 비로소 실제로 전송이 이루어지도록 근본적인 설계를 바꾼 것이다.  
         그래서 내가 보낸 메시지든, 상대방이 보낸 메시지든 서버에 도달하고 처리되면 publish되어 해당 채팅방에 접속중인 모든 기기에 뿌려진다.  
         정렬은 위에서 제시한 autoIncrementId를 활용하면 된다.  
         따라서, 이 때 수신받은 채팅이 내 것이면 (message 객체의 role을 통해서 확인 가능)내 쪽에 표시하고, 상대방 측이라면 상대방 측에 표시하면 된다.  
         이 공통 이벤트는 다음과 같다:  
         서버측에서 emit되는 chatReceivedFromServer 이벤트가 바로 그것이다.  
         이걸 수신해서 메시지를 렌더링 하면 된다.
   5. User용
      1. SendChat 이벤트: 유저측에서 emit
         1. socket의 data 영역에 plain text로 바로 메시지를 주면 된다.
         2. 이게 끝이다.
      2. disconnect 이벤트: 유저측에서 emit
         1. 채팅방을 떠날 때나 앱이 백그라운드로 빠질 때 등, 즉 채팅 화면이 비활성화 되면 호출해야 한다.
         2. 앱이 꺼질 때 socket 연결이 같이 종료되므로 앱이 꺼지는 경우는 걱정 할 필요가 없으나, **앱이 꺼지지 않고 채팅방이 비활성화 될 경우**만 관리 해 주면 된다.
   6. Doctor용
      1. SendChat 이벤트: 의사측에서 emit
         1. socket의 data 영역에 plain text로 바로 메시지를 주면 된다.
         2. 이게 끝이다.
      2. disconnect 이벤트: 의사측에서 emit
         1. 채팅방을 떠날 때나 앱이 백그라운드로 빠질 때 등, 즉 채팅 화면이 비활성화 되면 호출해야 한다.
         2. 앱이 꺼질 때 socket 연결이 같이 종료되므로 앱이 꺼지는 경우는 걱정 할 필요가 없으나, **앱이 꺼지지 않고 채팅방이 비활성화 될 경우**만 관리 해 주면 된다.