# Firebase Firestore working

SnapShot Listener

private fun loadMessages() {  
 val messagesRef = firestore.collection("chatRooms")  
 .document(chatRoomId)  
 .collection("messages")  
 .orderBy("messageTime", Query.Direction.*ASCENDING*)  
  
 messagesRef.addSnapshotListener **{** snapshots, e **->** if (e != null) {  
 // Handle error  
 return@addSnapshotListener  
 }  
 messagesList.clear()  
  
 snapshots?.*let* **{** for (doc in **it**.*documents*) {  
 val message = doc.toObject(Message::class.*java*)  
 message?.*let* **{** messagesList.add(**it**) **}** }  
 **}** messageAdapter.notifyDataSetChanged()  
 **}**}

Documents Loading Simple

private fun loadUsersData() {  
  
  
 Variables.db.collection("users") // firestore collection name is users  
 .get()  
 .addOnSuccessListener **{** documents **->** for (document in documents) {  
 // document jo json m ha usko convert kru User model class m  
 val user = document.toObject(User::class.*java*)  
 if (user.uid != (Variables.auth.*currentUser*?.*uid* ?: "") ){  
 userList.add(user)  
 }  
  
 }  
 userApapter.updateUsers(userList) // Update adapter with job list  
 shimmerFrameLayout.stopShimmer() // Stop shimmer animation  
 shimmerFrameLayout.*visibility* = View.*GONE* // Hide shimmer layout  
 binding.rvUsers.*visibility* = View.*VISIBLE* // Show RecyclerView  
 **}** .addOnFailureListener **{** e **->** Variables.displayErrorMessage("Failed to load users: ${e.message}", requireContext())  
 shimmerFrameLayout.stopShimmer()  
 shimmerFrameLayout.*visibility* = View.*GONE* **}**}

Sending Data

private fun sendMessage(messageText: String) {  
 // Generate unique messageId  
 val messageId = firestore.collection("messages").document().*id* val message = Message(  
 messageId = messageId,  
 messageText = messageText,  
 messageTime = Timestamp.now(),  
 messageType = "text",  
 receiverId = receiverId,  
 senderId = currentUserId  
 )  
  
 // Add the message to Firestore  
 val chatRoomRef = firestore.collection("chatRooms").document(chatRoomId)  
  
 chatRoomRef.collection("messages").document(messageId).set(message)  
 .addOnSuccessListener **{** // Clear input field  
 binding.editTextMessage.*text*.clear()  
  
 // Update ChatRoom with the last message and timestamp  
 chatRoomRef.set(  
 *mapOf*(  
 "lastMessage" *to* messageText,  
 "lastMessageTime" *to* Timestamp.now(),  
 "participants" *to listOf*(currentUserId, receiverId)  
 ),  
 SetOptions.merge() // Merge to keep other fields intact  
 )  
 **}** .addOnFailureListener **{** e **->  
  
 }**}