Implementing a Real-time Chat System

Mitch Tabian

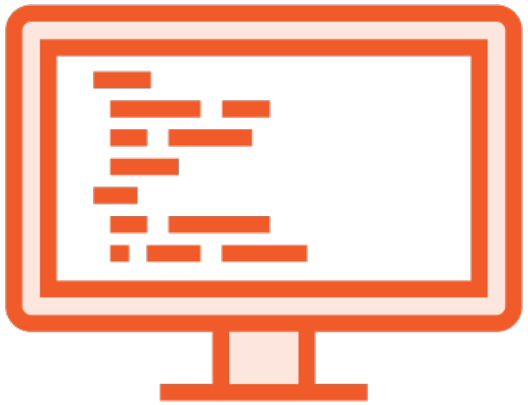
Twitter: @mitch_tabian
Website: codingwithmitch.com
Youtube: youtube.com/c/mitchtabian
Instagram: mitch.tabian

Source Code

“Module_6\start\TabianConsulting”

Save and edit data

Upload and display image from storage



Create new chatrooms

Add security restrictions to chatrooms

Send messages in real-time

Up Next

Creating a new chatroom

Creating a New Chatroom

Up Next

Retrieving a list of chatrooms

Retrieving a List of Chatrooms

Retrieving a List of Chatrooms

Retrieving the objects from the database

Sending the objects to a ListAdapter

Up Next

Deleting chatrooms

Deleting Chatrooms

Up Next

Inserting chatroom messages

Inserting Chatroom Messages

Up Next

Retrieving Chatroom Messages

Retrieving Chatroom Messages

Up Next

Reviewing the chatroom system

Reviewing the Chatroom System

Creating Chatrooms

```
DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
//get the new chatroom unique id
String chatroomId = reference
        .child("chatrooms")
        .push().getKey();

//create the chatroom
Chatroom chatroom = new Chatroom();
chatroom.setSecurity_level(String.valueOf(mSeekBar.getProgress()));
chatroom.setChatroom_name(mChatroomName.getText().toString());
chatroom.setCreator_id(FirebaseAuth.getInstance().getCurrentUser().getUid());
chatroom.setChatroom_id(chatroomId);


//insert the new chatroom into the database
reference
        .child("chatrooms")
        .child(chatroomId)
        .setValue(chatroom);
```

Retrieving a List of Chatrooms

```
mChatrooms = new ArrayList<>();

DatabaseReference reference = FirebaseDatabase.getInstance().getReference();

Query query = reference.child("chatrooms");
query.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        for(DataSnapshot singleSnapshot:  dataSnapshot.getChildren()){
            Log.d(TAG, "onDataChange: found chatroom: "
                    + singleSnapshot.getValue());

            Chatroom chatroom = new Chatroom();
            Map<String, Object> objectMap = (HashMap<String, Object>) singleSnapshot.getValue();

            chatroom.setChatroom_id(objectMap.get("chatroom_id").toString());
            chatroom.setChatroom_name(objectMap.get("chatroom_name").toString());
            chatroom.setCreator_id(objectMap.get("creator_id").toString());
            chatroom.setSecurity_level(objectMap.get("security_level").toString());


            chatroom.setChatroom_id(singleSnapshot.getValue(Chatroom.class).getChatroom_id());
            chatroom.setSecurity_level(singleSnapshot.getValue(Chatroom.class).getSecurity_level());
            chatroom.setCreator_id(singleSnapshot.getValue(Chatroom.class).getCreator_id());
            chatroom.setChatroom_name(singleSnapshot.getValue(Chatroom.class).getChatroom_name());

            //get the chatrooms messages
            ArrayList<ChatMessage> messagesList = new ArrayList<~>();
            for(DataSnapshot snapshot: singleSnapshot
                    .child("chatroom_messages").getChildren()){
                ChatMessage message = new ChatMessage();
                message.setTimestamp(snapshot.getValue(ChatMessage.class).getTimestamp());
                message.setUser_id(snapshot.getValue(ChatMessage.class).getUser_id());
                message.setMessage(snapshot.getValue(ChatMessage.class).getMessage());
                messagesList.add(message);
            }
            chatroom.setChatroom_messages(messagesList);
            mChatrooms.add(chatroom);
        }
        setupChatroomList();
    }
```

Inserting New Messages

```
//create the new message object for inserting
ChatMessage newMessage = new ChatMessage();
newMessage.setMessage(message);
newMessage.setTimestamp(getTimestamp());
newMessage.setUser_id(FirebaseAuth.getInstance().getCurrentUser().getUid());

//get a database reference
DatabaseReference reference = FirebaseDatabase.getInstance().getReference()
        .child("chatrooms")
        .child(mChatroom.getChatroom_id())
        .child("chatroom_messages");

//create the new messages id
String newMessageId = reference.push().getKey();

//insert the new message into the chatroom
reference
        .child(newMessageId)
        .setValue(newMessage);
```

Retrieving Chat Messages

```
DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
Query query = reference.child("chatrooms")
        .child(mChatroom.getChatroom_id())
        .child("chatroom_messages");
query.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        for(DataSnapshot singleSnapshot: dataSnapshot.getChildren()) {
            //create a final snapshot because we need to use this in the query below
            final DataSnapshot snapshot = singleSnapshot;
            Log.d(TAG, "onDataChange: found chatroom message: "
                    + singleSnapshot.getValue());
            try {//need to catch null pointer here because the initial welcome message to the
                //chatroom has no user id
                ChatMessage message = new ChatMessage();
                String userId = snapshot.getValue(ChatMessage.class).getUser_id();
                if(userId != null){ //check and make sure it's not the first message (has no user id)
                    message.setMessage(snapshot.getValue(ChatMessage.class).getMessage());
                    message.setUser_id(snapshot.getValue(ChatMessage.class).getUser_id());
                    message.setTimestamp(snapshot.getValue(ChatMessage.class).getTimestamp());
                    mMessagesList.add(message);
                }else{
                    message.setMessage(snapshot.getValue(ChatMessage.class).getMessage());
                    message.setTimestamp(snapshot.getValue(ChatMessage.class).getTimestamp());
                    mMessagesList.add(message);
                }

            } catch (NullPointerException e) {
                Log.e(TAG, "onDataChange: NullPointerException: " + e.getMessage());
            }
        }
        //query the users node to get the profile images and names
        getUserDetails();
        initMessagesList();
    }
```

getUserDetails()

```
private void getUserDetails(){
    DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
    for(int i = 0; i < mMessagesList.size(); i++) {
        Log.d(TAG, "onDataChange: searching for userId: " + mMessagesList.get(i).getUser_id());
        final int j = i;
        if(mMessagesList.get(i).getUser_id() != null){
            Query query = reference.child("users")
                    .orderByKey()
                    .equalTo(mMessagesList.get(i).getUser_id());
            query.addListenerForSingleValueEvent(new ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {
                    DataSnapshot singleSnapshot = dataSnapshot.getChildren().iterator().next();
                    Log.d(TAG, "onDataChange: found user id: "
                            + singleSnapshot.getValue(User.class).getUser_id());
                    mMessagesList.get(j).setProfile_image(singleSnapshot.getValue(User.class)
                            .getProfile_image());
                    mMessagesList.get(j).setName(singleSnapshot.getValue(User.class).getName());
                    mAdapter.notifyDataSetChanged();
                }

                @Override
```

Up-To-Date Messages

```
@Override
protected void onDestroy() {
    super.onDestroy();
    mMessagesReference.removeEventListener(mValueEventListener);
}
```

```
ValueEventListener mValueEventListener = new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) { getChatroomMessages(); }

    @Override
    public void onCancelled(DatabaseError databaseError) {

    }
};
```

```
private void enableChatroomListener(){
     /*
        ---------- Listener that will watch the 'chatroom_messages' node ----------
     */

    mMessagesReference = FirebaseDatabase.getInstance().getReference().child("chatrooms")
            .child(mChatroom.getChatroom_id())
            .child("chatroom_messages");

    mMessagesReference.addValueEventListener(mValueEventListener);
}
```

Overview

- Create chatrooms
- Retrieve chatrooms
- Create messages
- Retrieve messages

Up Next

Tying it All Together