UNIVERSITY OF PUERTO RICO
RECINTO UNIVERSITARIO DE MAYAGÜEZ
MAYAGÜEZ, PUERTO RICO
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Project Phase One: Photo Messaging Application

ICOM 5016 - 116 : Database Systems
Prof. Manuel Rodriguez

Bernardo Sein
Jean C Merced
Noel Valentín

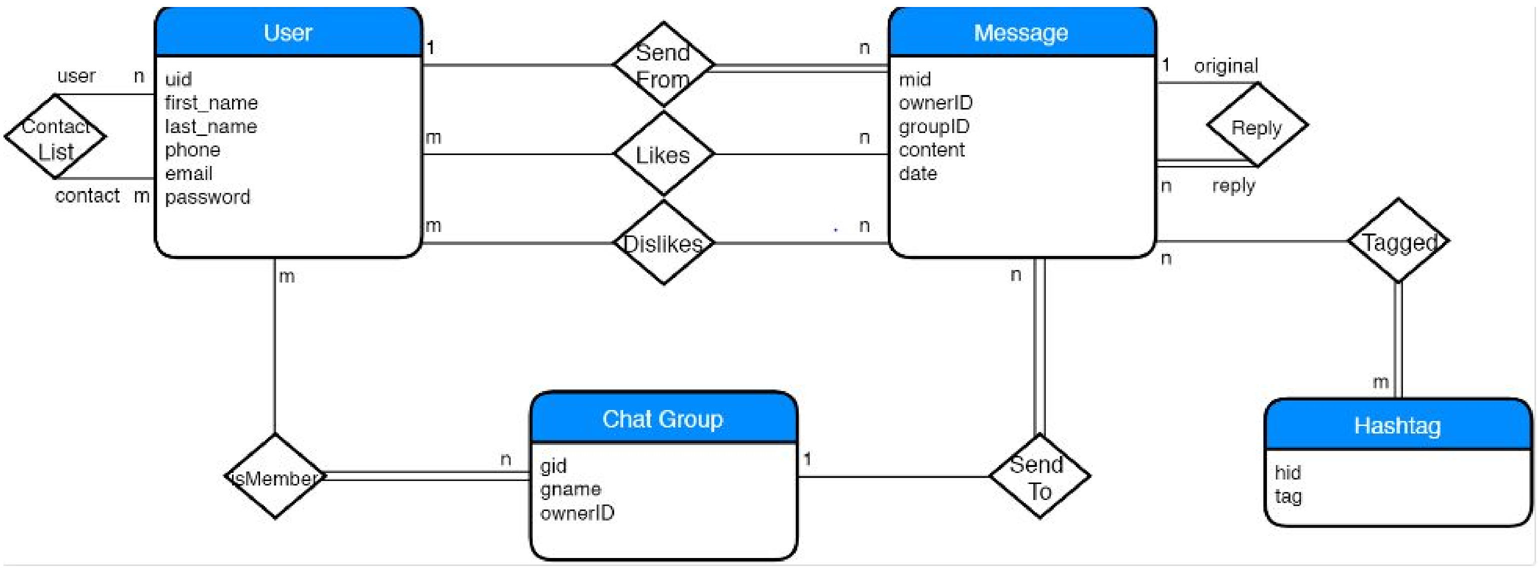

Figure 1 - ER Diagram

The E-R Diagram above has the following entities:

- *User* Entity, with attributes for User ID, First name, Last name, phone, e-mail and password
- *Message*, with attributes for the message ID, the ID of the user that posted it, the group chat that the message was posted to, the date of publication, and the contents of the message (usually a message and/or a picture)
- *Chat Group*, with attributes for Group Chat ID, Group chat name, and the ID of the user that created the group chat
- *Hashtag*, with an ID for each hashtag and the tag that it represents.

The user will have a list of contacts that will be based on their respective User IDs, which is why the relationship stays within the same entity. The User entity also maintains a relationship with the chat group entity, *isMember*. It requires total participation from the chat group side, given the fact that there cannot be a chat group without any users in it, but there can be users without certain group chats.

The user also has three relationships with the Message entity: *SendFrom, Likes & Dislikes.* The user can send messages, and the messages will contain the ID of the user that posted the message within its attributes. The same happens with *Likes and Dislikes*: each like element will have the ID information of the user that performed the "like", along with the ID of the message that the "like" was intended for. The messages also have a relationship to themselves, *Replies*: this states that replies are also messages. Given the fact that there can be messages without replies but no replies without original messages, it is a one-sided total participation relationship.

Most relationships in the diagram portray many-to-many relationships between the entities (which suggests that both entities in the relationship can have many relationships among themselves), with some exceptions. For example, the *Reply* relationship is one-to-many, based on the observation that one message can have many replies, but no reply has more than one original message. On the other hand, one user can send many messages, but no unique message can be sent by more than one user, which justifies the one-to-many relationship in the *Send From* relationship. Lastly, the *Send To* relationship is one-to-many as well because the same message can not belong to two different chat groups, but one chat group can contain many unique messages

The replies have very similar parameters to the message entity, the key difference being that the reply has an attribute that contains the ID of the message that the reply is "replying" to. Therefore, because the replies have message IDs of their own, it could be possible to reply to a reply with this setup. The messages are also linked to the Chat Group in which they were sent: the message entity has an attribute to obtain the chat group that the message is located in. Lastly, each message could have one or more Hashtag within its contents, which should be tagged as trending topics. Thus, the hashtag will have an ID and the respective string that shows the Hashtag that was sent in the message. Since there can be messages with no hashtag, yet there can't be hashtags without messages, there is an addition of total participation on the Hashtag side of the relationship *Tagged*.