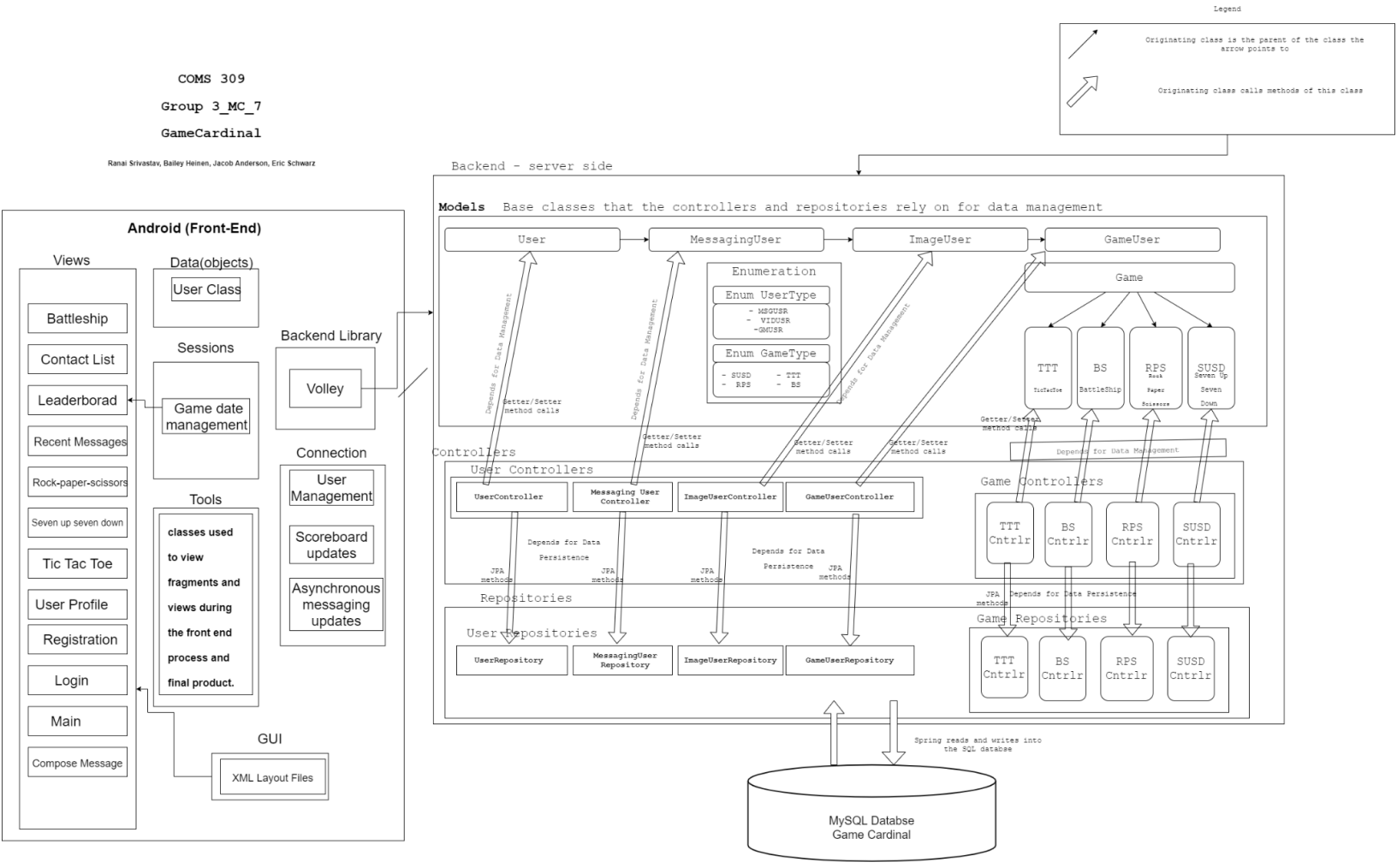
Design Document GameCardinal

Group 3_m_7

Bailey Heinen: 25%
Eric Schwartz: 25%
Jacob Anderson: 25%
Ranai Srivastav: 25%

Block Diagram File



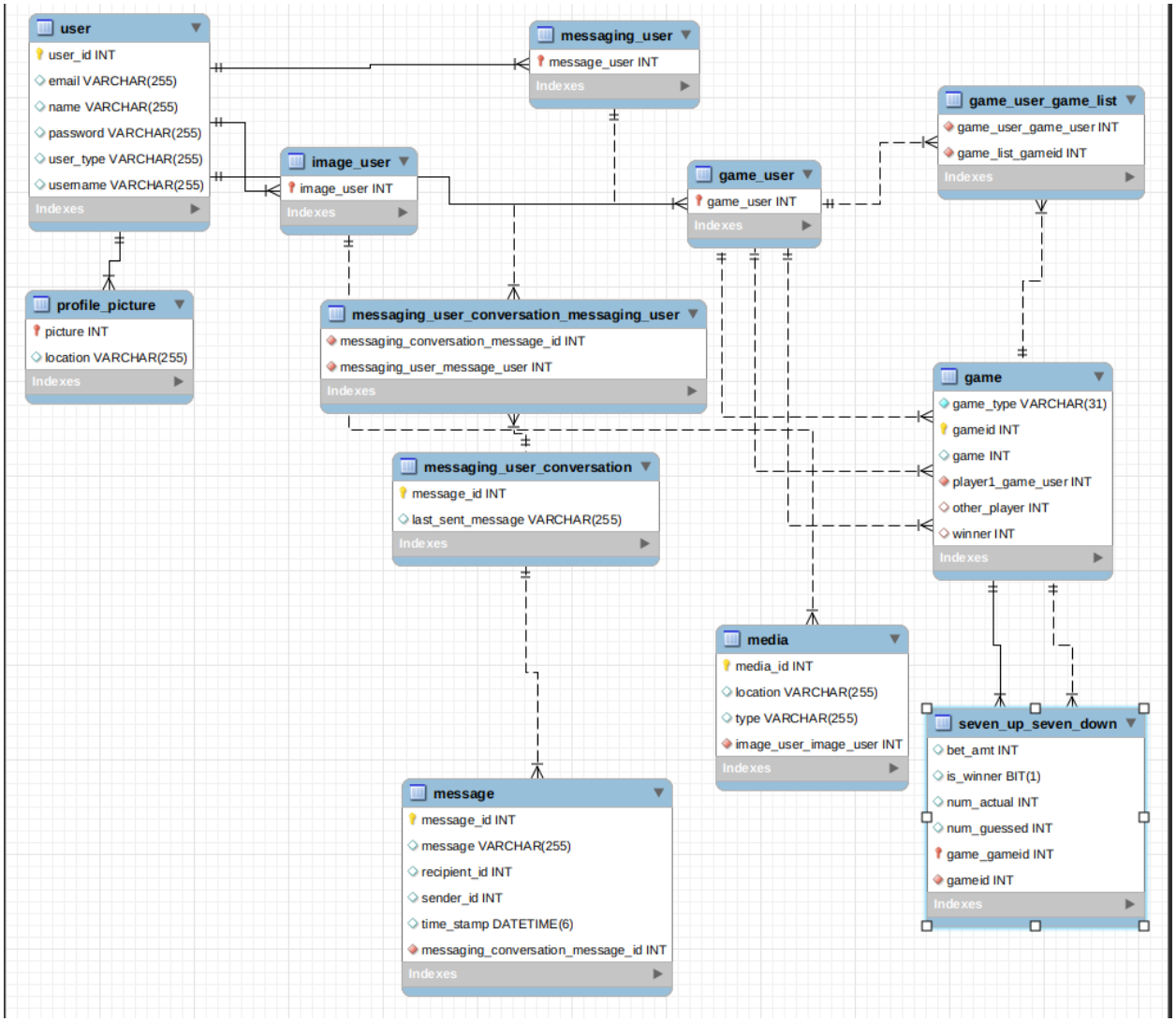
Overview

The app “GameCardinal” the features of this app that are currently in place include, front and backend communication through the login and registration screen, in these screens information is created and matched from frontend to backend to allow access to the main application. The purpose of this app is to have a communication platform where you can text and play games with contacts that are also using the app, all scores from these games will be saved in the leaderboards page to compete with friends. The main application that has been created however may not be finished include; the game Seven-up-Seven-down, settings page, main activity, hub activity, and compose message. The games we plan on having in this app are as follows; Seven-up-seven-down, battleship, tic-tac-toe, and rock-paper-scissors. The GUI that we used to create these applications in XML using java classes as “definitions” for objects in the XML.

How the backend end is constructed is we have our four levels of users, a regular user, messaging user, image user, and game user. The messaging user will just give the user to be able to send messages with other local messaging users in the application. Then the image user extends the messaging user adding the ability to send images. Finally we can upgrade to a game user which will give the user the ability to send games from game user to other game users. The game user will also have the ability to do all the other functionality that other users would have as it is the master.

There are many relationships that combine and make our code function correctly. Messaging User, Image User, and Game user will all have a one-to-one mapping with the user because only one user can be one of the respected users. Messaging users will then have a one-to-many relationship with messaging conversation because each messaging user is capable of having many messaging conversations and each conversation can have many messages in it. The reason we did this is because we needed many messaging conversations to store many messages between each user. Building off of this we decided to have a many-to-many relationship with media since each image user should have the ability to send many images between different image users. Finally the game user, how this works is it will have a one-to-many relationship with a class called game. We needed it to be a one-to-many relationship because we have a total of 4 games and needed to be able to access all of these games. Then the game list will have a many-to-many relationship with the game list that was needed to store all of the data that was being collected from each game.

SQL Diagram


user
user_id INT
email VARCHAR(255)
name VARCHAR(255)
password VARCHAR(255)
user_type VARCHAR(255)
username VARCHAR(255)
Indexes
messaging_user
message_user INT
Indexes
image_user
image_user INT
Indexes
game_user
game_user INT
Indexes
game_user_game_list
game_user_game_user INT
game_list_gameid INT
Indexes
profile_picture
picture INT
location VARCHAR(255)
Indexes
messaging_user_conversation_messaging_user
messaging_conversation_message_id INT
messaging_user_message_user INT
Indexes
game
game_type VARCHAR(31)
gameid INT
game INT
player1_game_user INT
other_player INT
winner INT
Indexes
messaging_user_conversation
message_id INT
last_sent_message VARCHAR(255)
Indexes
media
media_id INT
location VARCHAR(255)
type VARCHAR(255)
image_user_image_user INT
Indexes
seven_up_seven_down
bet_amt INT
is_winner BIT(1)
num_actual INT
num_guessed INT
game_gameid INT
gameid INT
Indexes
message
message_id INT
message VARCHAR(255)
recipient_id INT
sender_id INT
time_stamp DATETIME(6)
messaging_conversation_message_id INT
Indexes