Design Document for Sportsphere

Group 215

Member1 Kaden Wingert: % 25

Member2 Arie Kraayenbrink: % 25

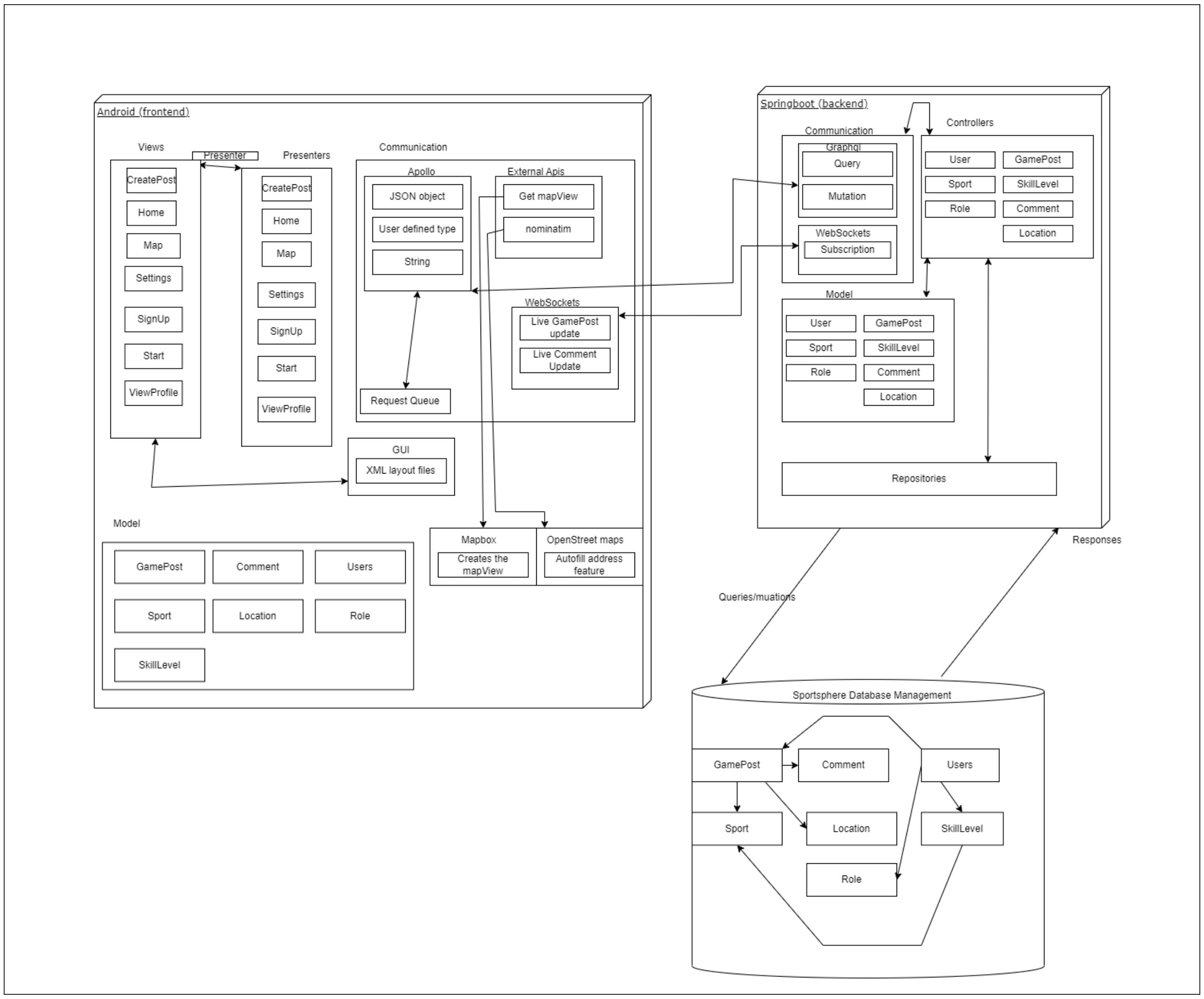
Member3 Sam Frost: % 25

Member4 Kerwn Shaw: % 25

PUT THE BLOCK DIAGRAM PICTURE ON THIS PAGE! (Create the picture using pencil or drawIO)

Link to access drawIO diagram:

https://app.diagrams.net/#G1UMolMZ4ejy4ImkBRNA25yc-aCwtY6mGd



Use this third page to describe complex parts of your design.

A large part of the way that the front end is developed relies on verifying which user is using the app. Several screens require this functionality. The best example of this is, is when the user is viewing a specific game post. If the post that the user is viewing is the user's own post, different actions will be available to them than if they were viewing someone else's game post. To do this, a string id is created and passed between each activity. That way, the ID can be checked and the rest of the screen will populate accordingly.

An important decision we made early on would be the use of GraphQL in the backend. This decision was made with several long term ideas in mind, most importantly, flexibility. Being able to use only one endpoint allows us the option of easily adjusting queries and mutations, to limit the amount of time a change in plans would cost us. This decision also plays a role in the way that WebSockets will be implemented. GraphQL utilizes Subscriptions, which provides a persistent read operation on the app. This will be used to continuously update the game posts, as well as the comments on each post.

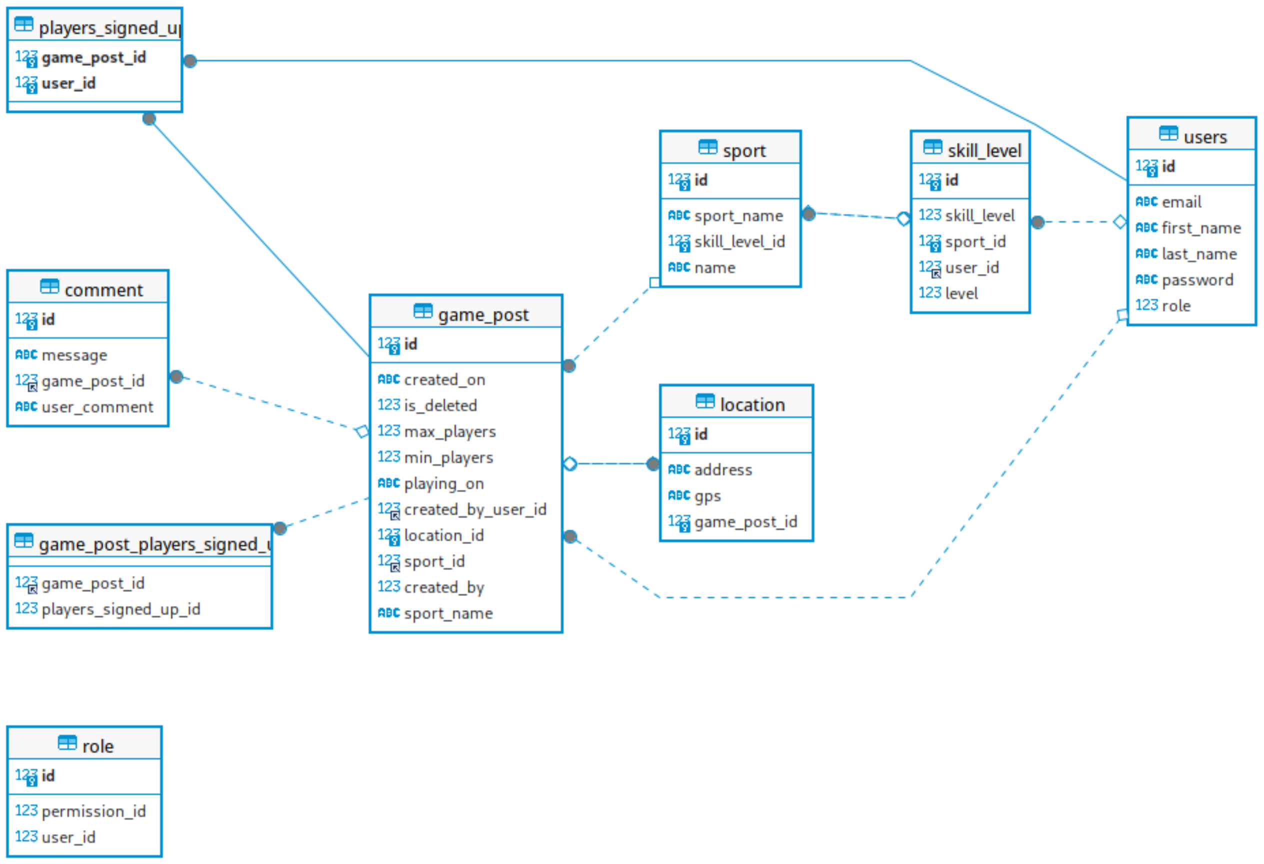
Game posts are the primary function of the app, as they are what facilitates the gathering of people for a game. At its core, the GamePost type represents events or matches, detailing the sport, participants, and location. The Location type not only stores address details but also GPS coordinates. The Comment type allows users to engage in discussions about specific game posts, fostering community interaction.

Two external APIs are used to implement the map functionality in the app. As shown in the diagram, MapBox provides the map view, and OpenStreet allows street addresses to be searched. An important feature that has been added due to the addition of these APIs, is prompting the user to consent to location services. Users are given the option of sharing once, sharing while using the app, and not sharing. Though not currently implemented, these APIs will be linked to the database, to store a set amount of key addresses.

User accounts are dependencies for many major aspects of the app. Most notably, the game posts. Game posts are created by a single user, and are "owned" by this user. This means if the user deletes the post, or the user's account is deleted, the game post will no longer be visible to other users. This remains the case even if user's have already joined the game. A user's comments to posts will also be deleted upon termination of the account.

Our app utilizes the Toast library in android studio, to provide on screen prompts to the user that are separate from the app itself. These prompts are used most frequently to assist users with login errors, and verifying network connection. A similar library, DialogueInterface, is used to prompt users when entering information. This is present when the user attempts to delete their account, as the user is required to enter their password.

PUT THE TABLE RELATIONSHIPS DIAGRAM on this fourth page! (Create the picture using MySQLWorkbench)


players_signed_u
game_post_id
user_id
sport
id
sport_name
skill_level_id
name
skill_level
id
skill_level
sport_id
user_id
level
users
id
email
first_name
last_name
password
role
comment
id
message
game_post_id
user_comment
game_post
id
created_on
is_deleted
max_players
min_players
playing_on
created_by_user_id
location_id
sport_id
created_by
sport_name
location
id
address
gps
game_post_id
game_post_players_signed_u
game_post_id
players_signed_up_id
role
id
permission_id
user_id