Design Document for Chinese Checkers

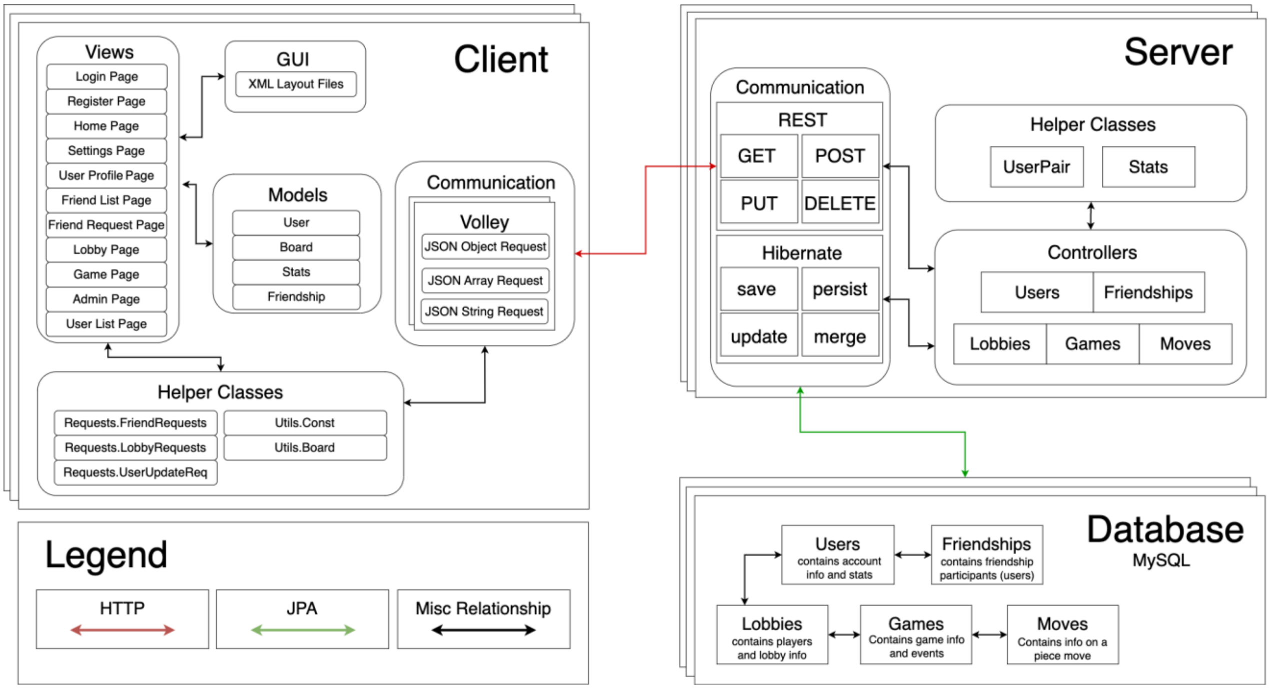
Group 1_HB_5

Evan Schlarmann: 25% contribution

Daniel Sprout: 25% contribution

Bryce Rega: 25% contribution

Marek Jablonski: 25% contribution


Client
Views
Login Page
Register Page
Home Page
Settings Page
User Profile Page
Friend List Page
Friend Request Page
Lobby Page
Game Page
Admin Page
User List Page
GUI
XML Layout Files
Models
User
Board
Stats
Friendship
Communication
Volley
JSON Object Request
JSON Array Request
JSON String Request
Helper Classes
Requests.FriendRequests
Requests.LobbyRequests
Requests.UserUpdateReq
Utils.Const
Utils.Board
Server
Communication
REST
GET
POST
PUT
DELETE
Hibernate
save
persist
update
merge
Helper Classes
UserPair
Stats
Controllers
Users
Friendships
Lobbies
Games
Moves
Database
MySQL
Users
contains account info and stats
Friendships
contains friendship participants (users)
Lobbies
contains players and lobby info
Games
Contains game info and events
Moves
Contains info on a piece move
Legend
HTTP
JPA
Misc Relationship

Client

GUI

The GUI is the XML files representing the visual aspects of the Views that the users interact with. This includes the placements and appearance of all aspects on the screen when interacting with the app.

Views

The View contains which individual activity page is needed for the app. Each View can move to other specified views navigating via user inputs. For example, after a user sign's in, they are moved to the home page, where they can navigate to the to a variety of other pages utilizing the View activities code linked to the GUI's xml elements.

Models

The models are background objects, or a collection of related information used to aid Views in displaying user or instance specific information. Some examples of this are the user model and the board model. The user model contains information about the user starting from when they sign in. This information is then passed between activities to ensure the application can identify the information pertaining to its current user. The board model serves as a local simulated board that is used to drive the GUI so that the user can see the current state of the board. It also contains methods to identify specifics about the current board state, like what possible locations a piece can move to from its current location.

Helpers

Helper classes are unique implementations of the Java Volley classes specific the the unique requests of each view method.

Communication

Java's Volley classes are utilized to communicate with the server via HTTP requests.

Server

Communication

Rest

Rest Methods are used to receive communications from users utilizing GET, POST, PUT, or DELETE requests using the HTTP communication protocol. These requests then access the Controllers for related information and return information pertaining to the HTTP request.

Hibernate

The Hibernate methods, save, persist, update and merge are used by the controllers to connect the MySQL Database using the Java Persistence API, or JPA. The controllers utilize the Hibernate methods to read or write data within the database.

Helper

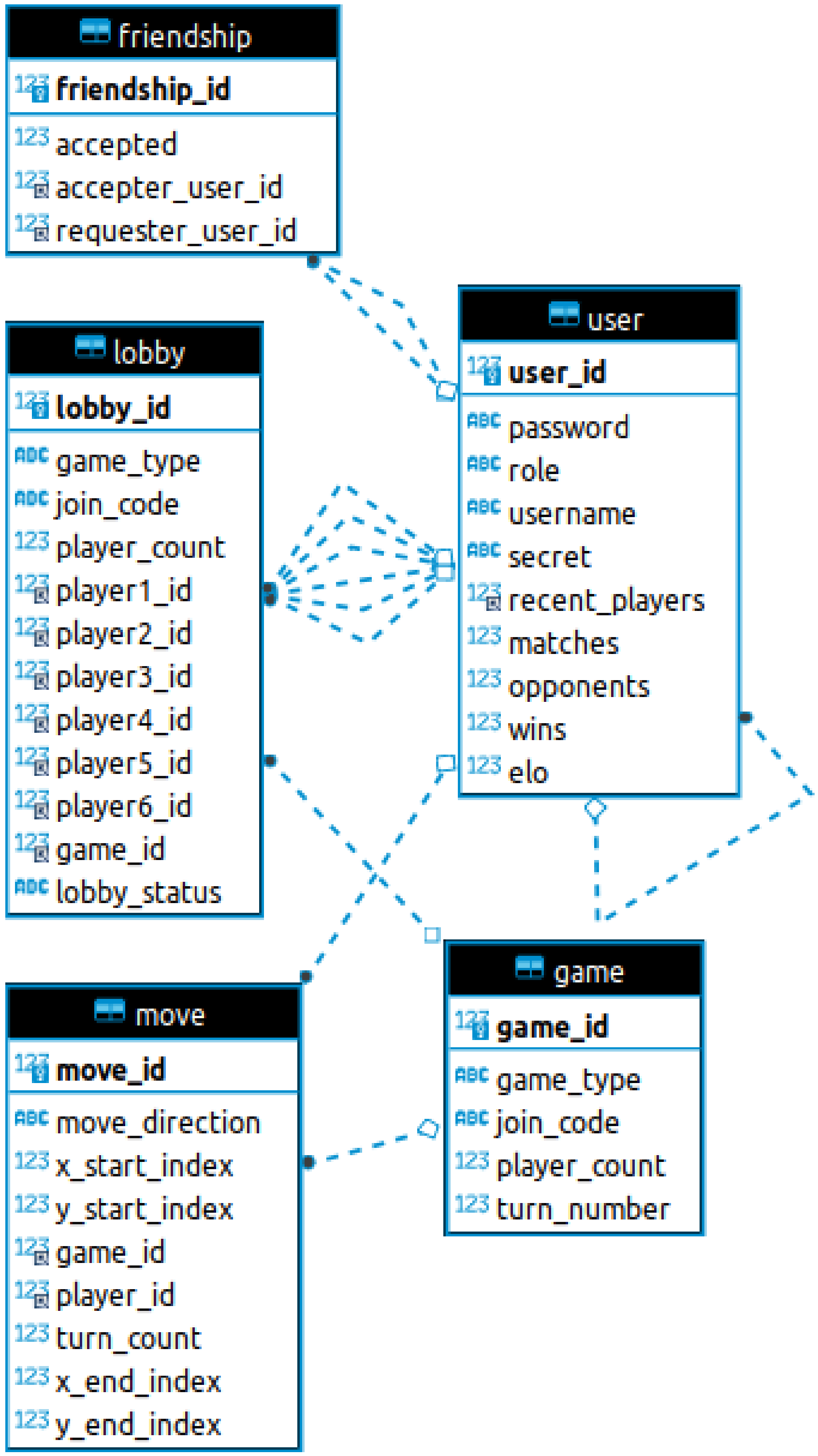
Helpers consists of classes used to manage data transfer, are not explicitly stored separately from the other tables, and in code are treated as a subset of their parent to hide detail unrelated to many other actions.

Controller

The controllers are mirrored specifications of the database tables that utilize the information stored within the database and can also update information within the database if need be. They also designate what information is needed to pass through communication requests, such as what is returned to the Client when they use a GET request.

Database

The database hosts the collection of information surrounding the application that does not need to be or cannot be locally hosted. The database utilized MySQL which stores information within tables. Each table represents a different entity or entity relation. For example, a user is an entity with a set of defined characteristics. Such as a username and password. The friendship between the users is the relation between them, which must be stored in a separate table as users can have more than one friend at a time. The combinations of user ids in the friendship are used as the ‘primary key’ of the relation, which is a unique identifier used to ensure there are no duplicate friendships.


friendship
friendship_id
accepted
accepter_user_id
requester_user_id
lobby
lobby_id
game_type
join_code
player_count
player1_id
player2_id
player3_id
player4_id
player5_id
player6_id
game_id
lobby_status
user
user_id
password
role
username
secret
recent_players
matches
opponents
wins
elo
move
move_id
move_direction
x_start_index
y_start_index
game_id
player_id
turn_count
x_end_index
y_end_index
game
game_id
game_type
join_code
player_count
turn_number