Design Document for *Gamedr*

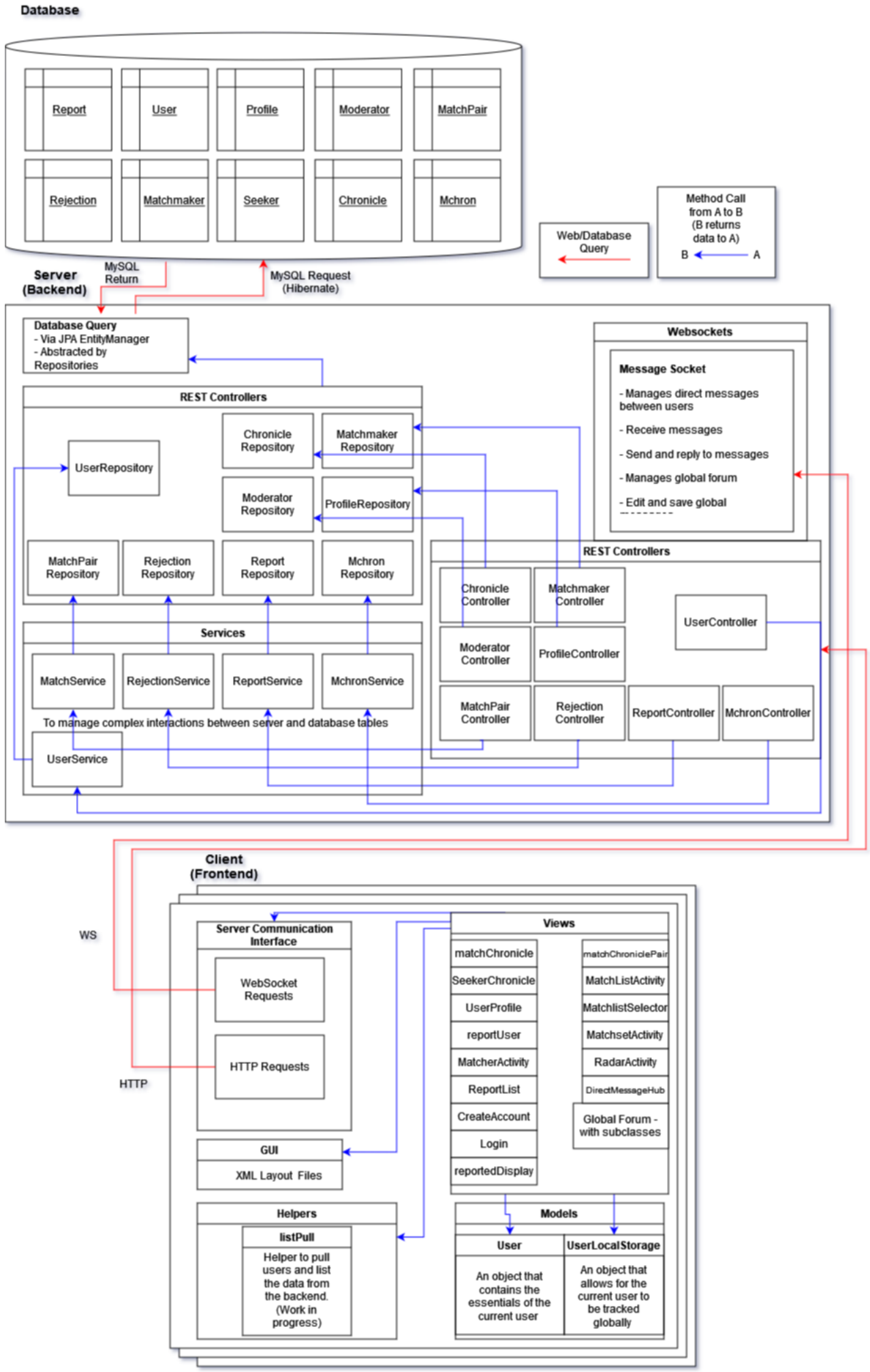
Group 2_yn_8

David Dong: 25 % contribution

Chandler Sihom: 25 % contribution

Jacob Huseman: 25 % contribution

Sibhat Yosef: 25% contribution


Database
Report
User
Profile
Moderator
MatchPair
Rejection
Matchmaker
Seeker
Chronicle
Mchron
Web/Database Query
Method Call from A to B (B returns data to A)
B
A
Server (Backend)
MySQL Return
MySQL Request (Hibernate)
Database Query
- Via JPA EntityManager
- Abstracted by Repositories
Websockets
Message Socket
- Manages direct messages between users
- Receive messages
- Send and reply to messages
- Manages global forum
- Edit and save global
REST Controllers
Chronicle Repository
Matchmaker Repository
UserRepository
Moderator Repository
ProfileRepository
MatchPair Repository
Rejection Repository
Report Repository
Mchron Repository
REST Controllers
Chronicle Controller
Matchmaker Controller
UserController
Moderator Controller
ProfileController
MatchPair Controller
Rejection Controller
ReportController
MchronController
Services
MatchService
RejectionService
ReportService
MchronService
To manage complex interactions between server and database tables
UserService
Client (Frontend)
WS
HTTP
Server Communication Interface
WebSocket Requests
HTTP Requests
Views
matchChronicle
SeekerChronicle
UserProfile
reportUser
MatcherActivity
ReportList
CreateAccount
Login
reportedDisplay
matchChroniclePair
MatchListActivity
MatchlistSelector
MatchsetActivity
RadarActivity
DirectMessageHub
Global Forum - with subclasses
GUI
XML Layout Files
Helpers
listPull
Helper to pull users and list the data from the backend. (Work in progress)
Models
User
An object that contains the essentials of the current user
UserLocalStorage
An object that allows for the current user to be tracked globally

Android GUI:

The Gamedr has a total of 22 activity classes with corresponding .xml files to provide a view for each class. Most of the activity classes handle grabbing and posting info from and to the back-end. But there will be classes to help store necessary information for the frontend to store globally. The RadarActivity is our main screen in which you can use it to create a match and view it in the matchChronicle view. Other views are used to view users(UserProfile), report(reportUser), set specific matches(MatcherActivity), message with global and direct chat(GlobalForum), and register/login.

Android Models:

There are 2 models in the Gamedr project, User and UserLocalStorage. These are to help store all information of the current user in the front-end so it can be used to grab other information later from the back-end. User helps set 6 fields; description, email, fullname, id, password, and username. These reduce the amount of times needed to pull users from the back-end and can be used to pull other data later. Then UserLocalStorage stores the new User object into a Shared Preference object to allow for global storage.

Android Helpers:

We intend to make extra helpers that can be used to pull a user from the back-end with JSON requests. Since we tend to pull user and list data from the back-end, it would reduce the amount of code by making a helper called “List Pull”. Placing the JSON request code inside these will help reduce the amount of times we write long JSON requests in our views.

REST Controllers:

Note that *Mchron* indicates a Matchmaker’s Chronicle, as described in ScreenSketches. Many interactions are required between clients, and controllers dictate the HTTP URL which facilitate these interactions. All controllers will eventually implement complete basic REST API functionality, but more specific functions are given below:

- User: List and order Users by using specific tags, actor roles, characteristics such as creation date, and other like terms that relate to the User.
- Profile: List by preferences, primarily for use in global forum in searching and finding Profiles similar to the one searching.
- Seeker: List by rating and writing date, used to be able to search for another Seeker and a Matchmaker who pairs the two based on similar interests.
- Matchmaker: List by rating and writing date, a Matchmaker is able to create a new Match for a pair of Seekers based on similar interests. The Matchmaker will then rate both Seekers after the match is concluded.
- Report: List by report severity and report date. Report is used to get a moderators attention to a User who may be breaking rules or posting inappropriate material.
- MatchPair: Generate random MatchPair and order by rating, primarily used to find a random pair of matches.

Services:

Intermediary interfaces between controllers and classes, used only for the JPA Entities of User, MatchPair, Rejection, Report, and Mchron.

Gamedr Table Relationships Diagram