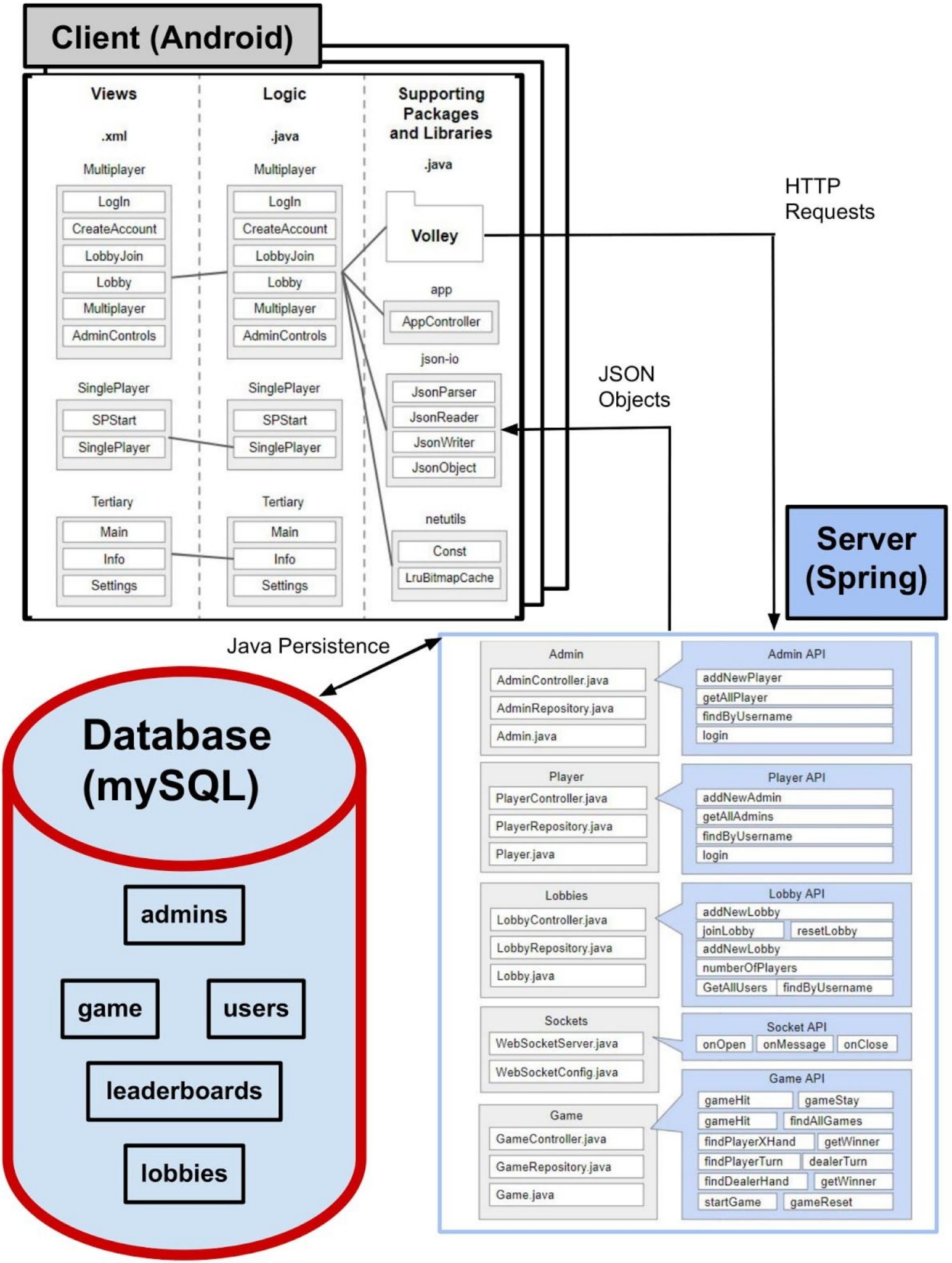
Block Diagram

TwentyOne

SB_4

James Taylor, Keaton Johnson, Thomas Haddy, & Tyler Fuchs


Client (Android)
Views
.xml
Logic
.java
Supporting Packages and Libraries
.java
Multiplayer
LogIn
CreateAccount
LobbyJoin
Lobby
Multiplayer
AdminControls
SinglePlayer
SPStart
SinglePlayer
Tertiary
Main
Info
Settings
Volley
app
AppController
json-io
JsonParser
JsonReader
JsonWriter
JsonObject
netutils
Const
LruBitmapCache
HTTP Requests
JSON Objects
Server (Spring)
Java Persistence
Database (mySQL)
admins
game
users
leaderboards
lobbies
Admin
AdminController.java
AdminRepository.java
Admin.java
Admin API
addNewPlayer
getAllPlayer
findByUsername
login
Player
PlayerController.java
PlayerRepository.java
Player.java
Player API
addNewAdmin
getAllAdmins
findByUsername
login
Lobbies
LobbyController.java
LobbyRepository.java
Lobby.java
Lobby API
addNewLobby
joinLobby
resetLobby
addNewLobby
numberOfPlayers
GetAllUsers
findByUsername
Sockets
WebSocketServer.java
WebSocketConfig.java
Socket API
onOpen
onMessage
onClose
Game
GameController.java
GameRepository.java
Game.java
Game API
gameHit
gameStay
gameHit
findAllGames
findPlayerXHand
getWinner
findPlayerTurn
dealerTurn
findDealerHand
getWinner
startGame
gameReset

Design Description

Client

Our client is individual instances of an Android app created in android studios. Views are created using XML and are all connected to their instance of a java file to create the logic for that page. From there, only multiplayer required more direct support. Multiplayer uses constants and a Bitmap Cache, as well as a json library to support the http requests used in Volley.

Client -- Server

We used Volley library to send HTTP requests (GET, PUT, and POST) to the server. The server receives the http requests, gathers the necessary data for processing, and returns info to the client as JSON objects or Strings.

Server

The Spring Boot server uses built-in libraries such as: autowire, controller, and several others for the database connections. Each table in the database uses three classes to handle the construction, management, and use of these entities. Repositories are used to handle the Query functions. Controllers are used to the handle any updates which need to requested. Then a class is created to represent each row called an entity. Entities are apart of the persistence library.

Server -- Database

For the Spring Boot server to connect to the MySQL database a library called persistence is used. This library is used to manage inputs into the different tables in our database. We also use a repository to assist in the management of these inputs called crude Repository. In Spring Boot a crude repository connects to a given table and can save and update columns with a simple function “save”.

Database

Our MySQL database is used to hold most information from our application. With a total of 5 tables, two being used for types of users (admins, users). A leaderboard table that is used for statistics for users, and a dynamic lobbies table that is used to store the four players in a lobby so they can be loaded together in a game. The tables are accessed by Spring Boot using persistence libraries and

Tables & Fields

When using a 1-1 relationship between tables, the information could all be used added to a single table but is instead split into more. This can be useful in the form of polymorphism. In our case this is used in the admins to users tables. Admins and users are different types of players, but either one can join a lobby and play the game. In our application we do not have any many - many relationships, but the construction would be that same as 1-1, just expanded.

- Admins
    - id
    - email
    - username
    - Password
- Game
    - playerOne, Two, Three, Four
        - playerXHand, playerXValue
    - dealerHand
    - dealerValue
    - playerTurn
- Users
    - id
    - email
    - username
    - password
- Leaderboards
    - id
    - username
    - won
    - games
    - winrate
- Lobbies
    - id
    - playerOne, Two Three, Four