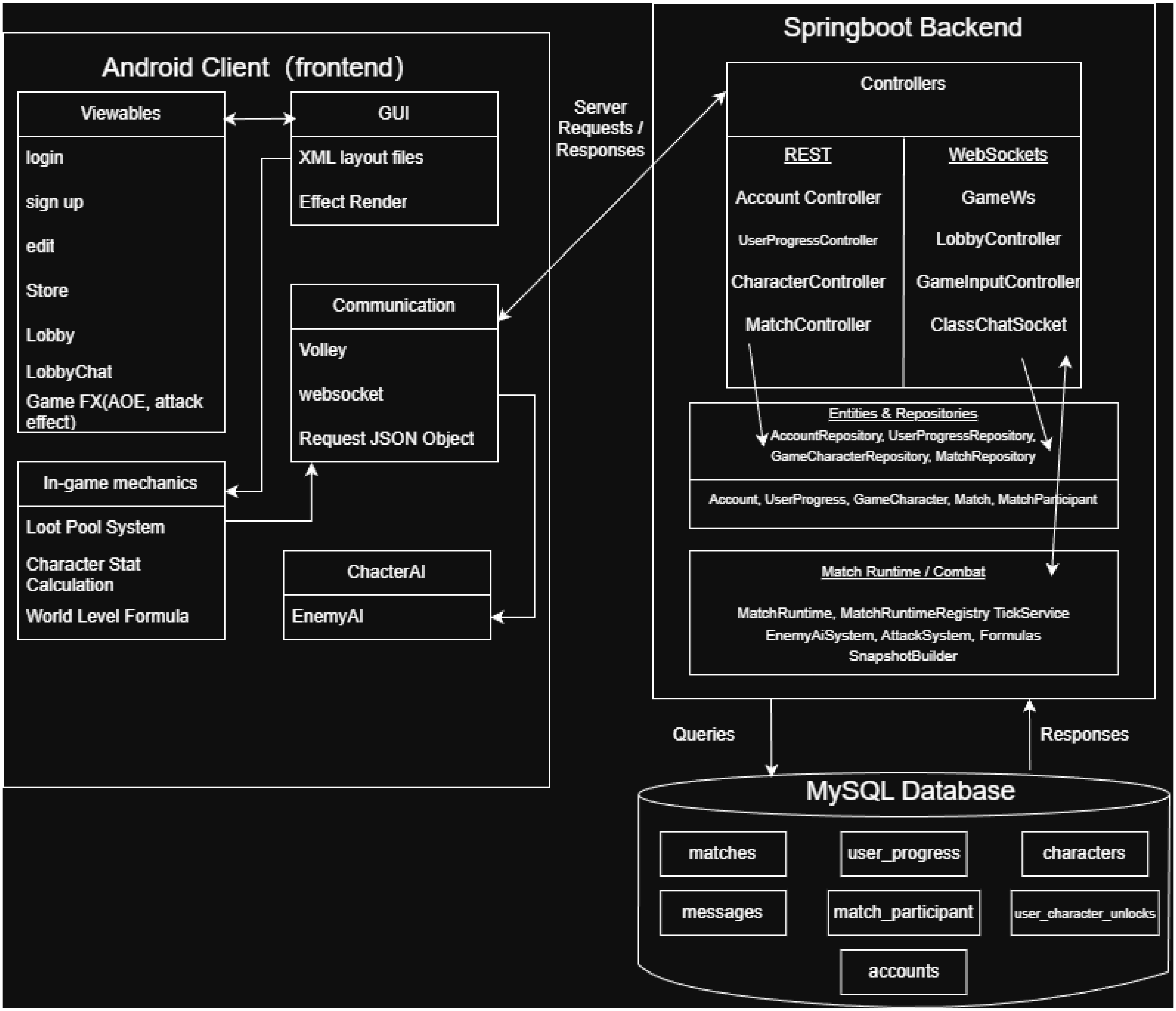
Design Document for Crypt of the Cubes

Group 3_rasel_6

John Draa (backend sections + DB schema)

Yingxuan Ye (frontend sections)

Alexandra Alter (frontend sections)


Android Client (frontend)
Viewables
login
sign up
edit
Store
Lobby
LobbyChat
Game FX(AOE, attack effect)
GUI
XML layout files
Effect Render
Communication
Volley
websocket
Request JSON Object
In-game mechanics
Loot Pool System
Character Stat Calculation
World Level Formula
ChacterAI
EnemyAI
Server Requests / Responses
Springboot Backend
Controllers
REST
Account Controller
UserProgressController
CharacterController
MatchController
WebSockets
GameWs
LobbyController
GameInputController
ClassChatSocket
Entities & Repositories
AccountRepository, UserProgressRepository, GameCharacterRepository, MatchRepository
Account, UserProgress, GameCharacter, Match, MatchParticipant
Match Runtime / Combat
MatchRuntime, MatchRuntimeRegistry TickService EnemyAiSystem, AttackSystem, Formulas SnapshotBuilder
Queries
Responses
MySQL Database
matches
user_progress
characters
messages
match_participant
user_character_unlocks
accounts

Frontend:

SignUp: The signup generates a page featuring the following elements:

- EditText: Email
- EditText: Username
- EditText: Password
- EditText:Confirm Password
- Button: Sign Up
- Button: Back

Upon clicking the button Sign Up the values for the EditTexts are sent as a POST request to the server.

Login: The login generates a page featuring the following elements:

- EditText Username
- Edittext Password
- Button Login
- Button: Back

Upon clicking the Login button, the user input for the username and password is checked if there is a match in the server's data. If the user input matches the account details stored in the server then the user will be taken to the Main m Menu page.

Main Menu:

- Button: Account
- Button: logout
- Button: Startgame
- Button: Shop
- Button: Exit

Clicking Account allows modification of the currently logged-in account. After confirmation, changes are synchronized to the server. Clicking Start Game retrieves account information from the server and advances to the Game page. Clicking Store enters the Store page, where character-related information is fetched from the server.

Gameplay: Users are taken to a page that features the following elements:

- EditText: Join Code
- Button: Create Lobby
- Button: Character selection
- Button: Game start

Clicking Create Lobby will automatically create an online lobby with the current user as the host and copy the lobby's join code. Enter the copied code in the Join Code field and click Join to enter a lobby with the same code. Clicking the corresponding character selection button will select and lock that character (excluding the default character). Once all members' characters are locked, clicking the Start Game button will begin the game.

In Game: Since in the actually game user only need to move so theres nothing except one joy-con.

Backend:

Communication

The backend uses REST endpoints and WebSockets to exchange data with the android client and update the database. REST is used for the slower data focused operations (accounts, progress, characters), while WebSockets handles the fast operations such as lobby and in-game updates.

POST: Send info to create new records of accounts, progress entries, matches, and chat messages

GET: Request info from accounts, character lists, progress/leaderboard data.

PUT: Modify existing records, like adding coins and score to a player's progress or changing account info.

DELETE: Remove testing data such as old accounts or progress records.

WebSocket STOMP: Sends/receives player inputs, ready status, lobby chat, and match start/stop events; sends/receives visibility filtered game snapshots, level up / damage / death events, and lobby updates in real time.

Controllers

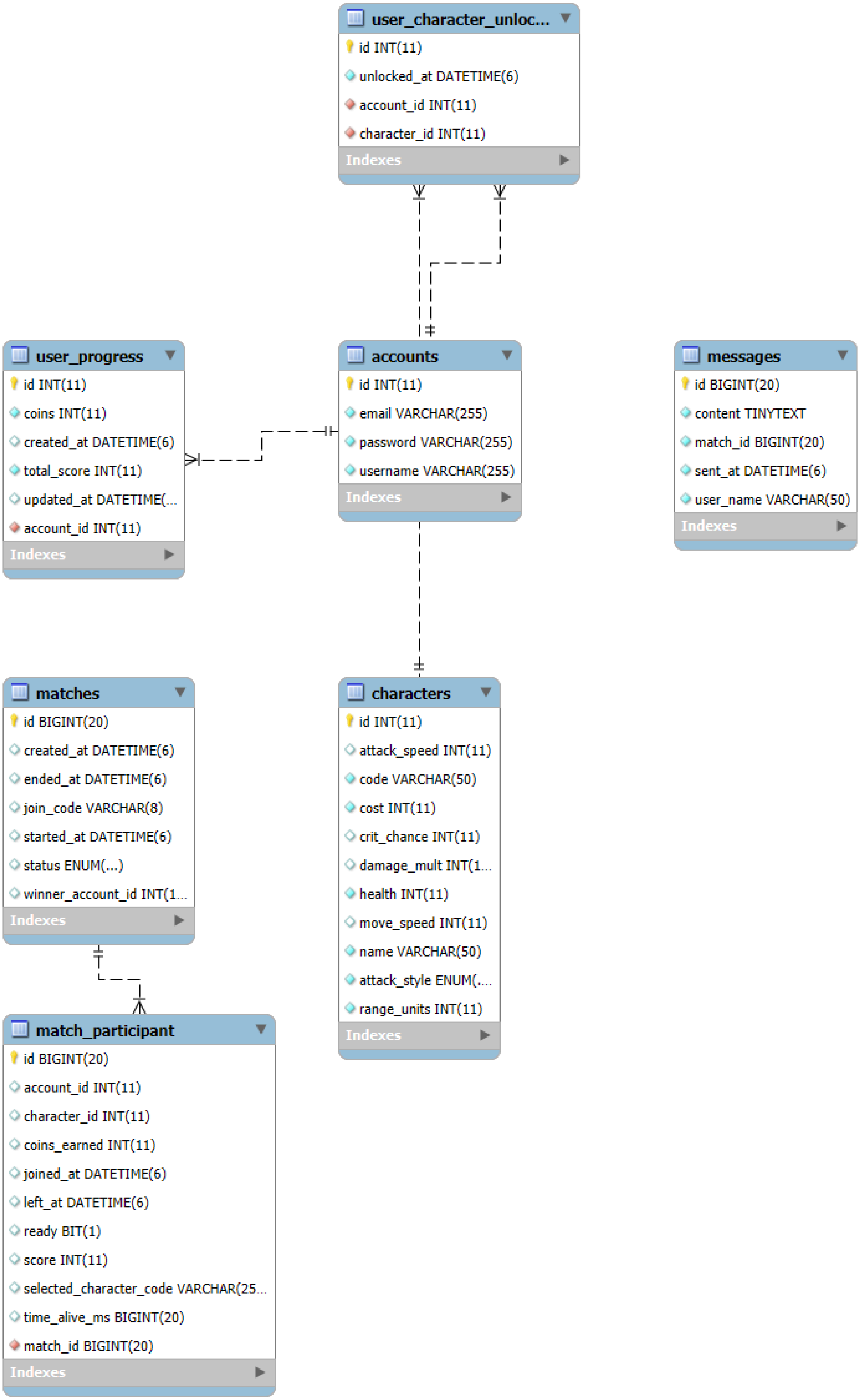
The controllers contain the mapping for communication between the frontend and the backend services/database. They translate HTTP or WebSocket messages into domain operations.

Account Controller handles signup, login, and account updates; each account has exactly one progress record and many character unlocks.

Progress Controller tracks coins and total score, and exposes leaderboard queries.

Character Controller serves character stats and unlock info.

Match/WebSocket Controllers manage lobbies, active matches, and chat; they receive inputs from players, run the server-side match logic, and push filtered game state back to each client in real time.



user_character_unloc...
id INT(11)
unlocked_at DATETIME(6)
account_id INT(11)
character_id INT(11)

user_progress
id INT(11)
coins INT(11)
created_at DATETIME(6)
total_score INT(11)
updated_at DATETIME(...
account_id INT(11)

accounts
id INT(11)
email VARCHAR(255)
password VARCHAR(255)
username VARCHAR(255)

messages
id BIGINT(20)
content TINYTEXT
match_id BIGINT(20)
sent_at DATETIME(6)
user_name VARCHAR(50)

matches
id BIGINT(20)
created_at DATETIME(6)
ended_at DATETIME(6)
join_code VARCHAR(8)
started_at DATETIME(6)
status ENUM(...)
winner_account_id INT(1...

characters
id INT(11)
attack_speed INT(11)
code VARCHAR(50)
cost INT(11)
crit_chance INT(11)
damage_mult INT(1...
health INT(11)
move_speed INT(11)
name VARCHAR(50)
attack_style ENUM(...
range_units INT(11)

match_participant
id BIGINT(20)
account_id INT(11)
character_id INT(11)
coins_earned INT(11)
joined_at DATETIME(6)
left_at DATETIME(6)
ready BIT(1)
score INT(11)
selected_character_code VARCHAR(25...
time_alive_ms BIGINT(20)
match_id BIGINT(20)