Gobang

Data Flow Diagram

David Liu 1155216841

Chak Ming Sin, 1155157388

Zizhe Chen, 1155173938

Lana McKay, 11552169070

Serena Poonawalla, 1155216924

Department of Computer Science and Engineering

The Chinese University of Hong Kong

Table of Content

1 High Level Context Diagram

1.1 Description (Level 0)

Please note that within this document and the diagrams, the terms “User” and “Player” will be used interchangeably. Their meaning is always the same.

In this general Data Flow Diagram (DFD) of **Gobang**, the rectangles with the grey background colour represent different operators (can also be interpreted as input device). There are two types of operators, i.e. **Admin** and **User**. The main jobs of admin are managing and modifying the *User Account Data*, supervising the *Chat Record* dataset, receiving feedback from users, and enforcing penalty settings on users who violate chat rules. On the other hand**, User** enjoys the main features of playing the game, chatting with other players, replaying previous games, and modifying their profile and settings.

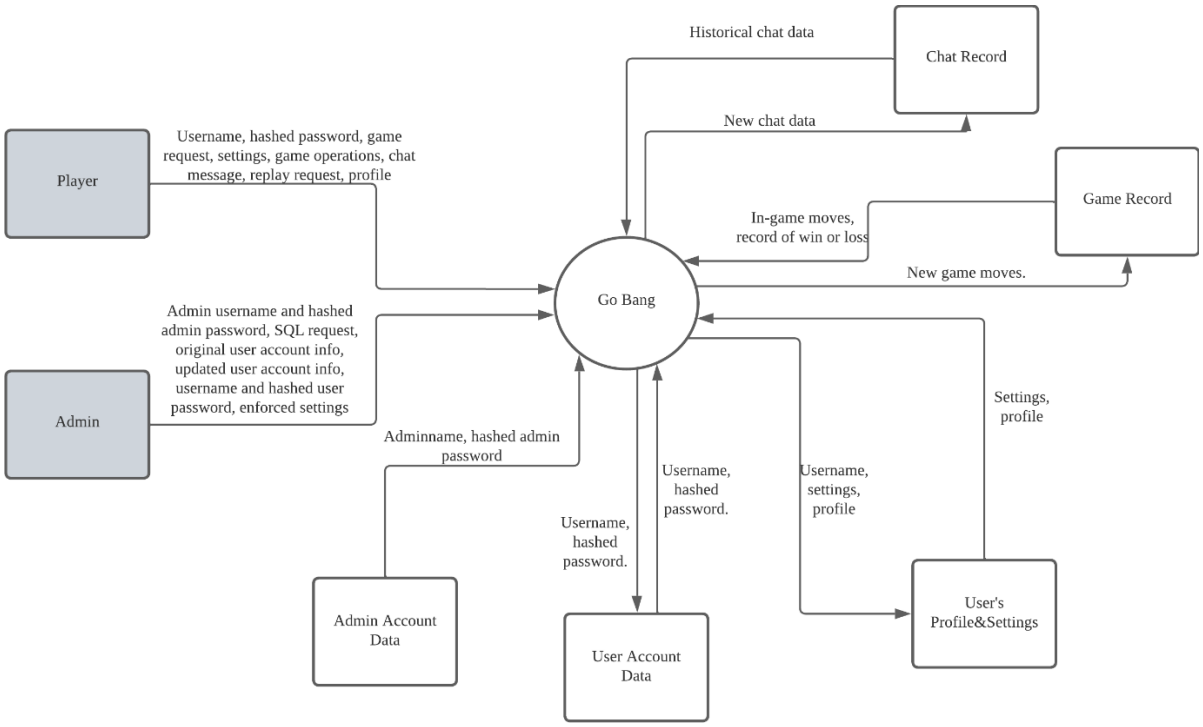
The circle with the white background colour represents function or system. In this DFD, to generally show all components, **Gobang** is treated as one whole system for processing data. **Gobang** will be separated into many subsystems and functions in DFDs below.

The arrows with text labels represent data flows. Only data that will be processed or stored in the target component can be considered as data flow. There are multiple data flows to/from one

component, and the data flow to/from the same components might not be sent/received simultaneously.

The rectangles with the white background colour represent databases. There are in total 5 databases, i.e. *Chat Record, Game Record, User's Profile, User Account Data, Admin Account Data. Chat Record* database stores the game id, username, time, and message content. *Game Record* database stores the game id, username, game moves, time, and winner. *User's Profile&Settings* stores the username, profile, game settings, and UI settings. *User Account Data* and *Admin Account Data* store account information.

1.2 DFD (Level 0)



2 System Diagram

2.1 Description (Level 1)

In the system level DFD, we will describe every system that appears. First is the **Admin Login System**, which receives the admin name and hashed password from **Admin** to compare with the fetched hashed password from *Admin Account Data* database. If they are the same, **Admin Login System** will generate a random token, which will be sent to **Admin Functions** every time admin sends request to **Admin Functions**. Receiving many types of data, which are not required simultaneously, **Admin Functions** contains multiple functions that enable admin to query *Chat Record,* reply to users' feedback, modify *User Account Data* and *User's Profile&Settings*.

Next is **Sign Up**, which is only applicable to **User**. After **Sign Up** receives username and password from a user, since username is unique for each user, **Sign Up** queries for the username in *User Account Data* and checks whether there is duplication. If not, **Sign Up** will store the new username and password into the database and send to **Login** to automatically login. **Login** is similar with **Admin Login System**. What is different is that **Login** also fetches the profile to display and fetch this user's previous settings as default settings to send to all other systems, after user successfully logins.

To start the game, **User** sends game request and his/her modified/default settings, which include whether user accepts chat and retraction as well as the accepted move time, to **Matching System**.

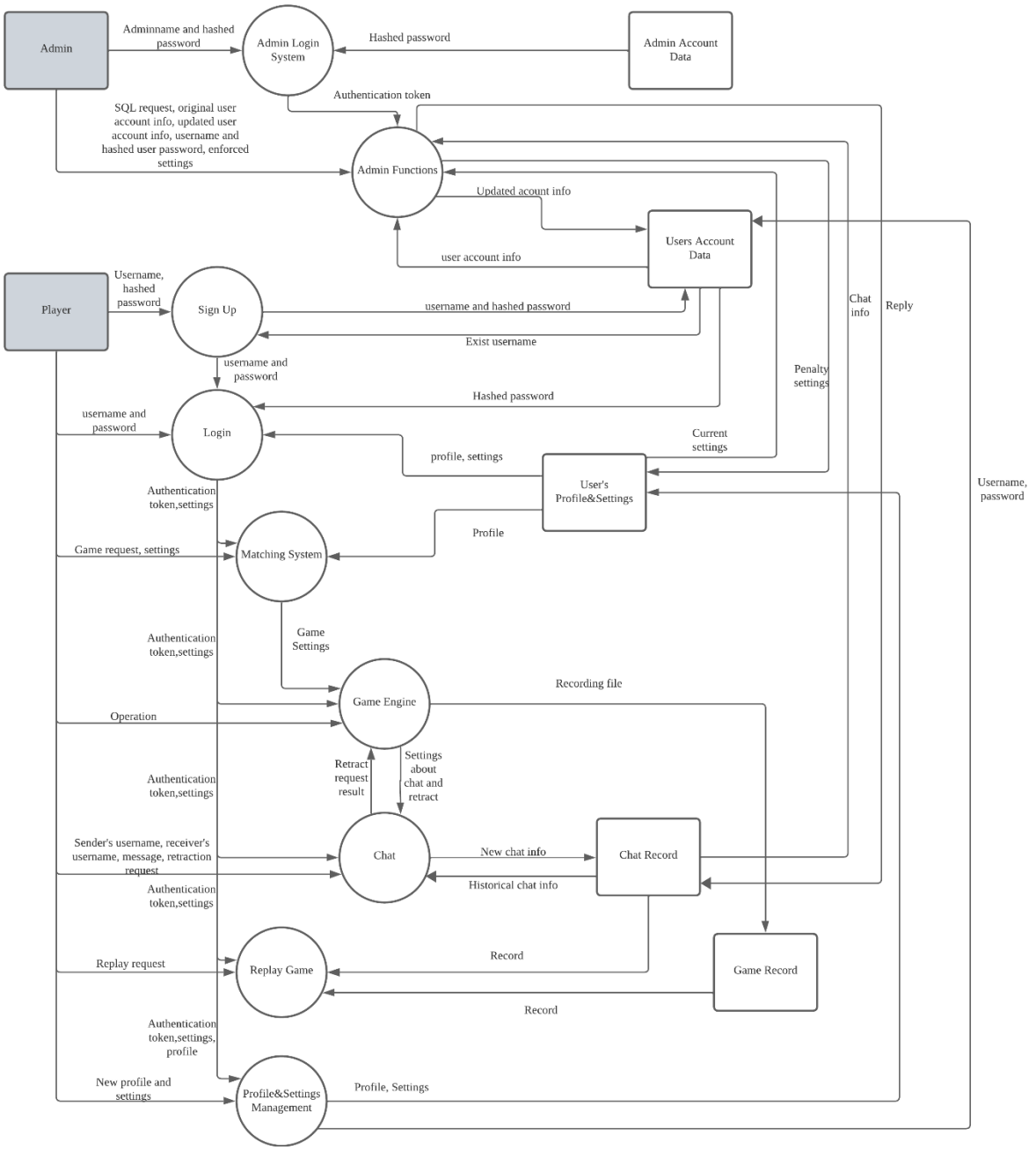
Matching System stores this player in memory and continuously searches for another player that shares similar settings. After finding the players with similar settings, the **Matching System** will match them together and fetch their profile from *User's Profile&Settings* to show to each player. Then, **Matching System** sends the final game settings to initialize **Game Engine**, which processes all the game moves. **Game Engine** receives game moves from players, performs operations such as validating the moves, and stores the moves into *Game Record*. Furthermore, **Game Engine** enables the retraction feature through interacting with **Chat**, since retraction requires agreement on both players. Therefore, **Game Engine** sends settings about chat and retraction to initialize **Chat** after its own initialization. When **Game Engine** receives the retraction agreement from **Chat**, it reads the target chessboard status from memory and performs retraction.

During the game, depend on their settings, two players can chat with each other and retract previous moves through **Chat**. For chatting, player sends message to **Chat**, and then **Chat** sends this message to another player's screen and stores this message into *Chat Record*. For retraction, one player sends a retraction request to **Chat**, and then **Chat** sends the request to the other user and waits for response. If yes, **Chat** will send the agreement to **Game Engine** to execute retraction and store the request result to *Chat Record*. If not, then **Chat** will only store the request result to *Chat Record.* Players can also chat with each other outside the game by sending sender's username and receiver's username to **Chat** to open a chat, which automatically retrieves and display historical chat info from database. What's more, players can provide feedback about this application or send other special request to admin through **Chat**. Admin will receive the feedback and response to player.

After the game, users can review the game through **Replay Game**, which fetches record from *Game Record* and *Chat Record* to replay the game step by step.

Finally, users modify their profile and settings through **Profile&Settings Management**, which receives previous profile&settings from **Login** to display and new profile&settings from **User**. After that, **Profile&Settings Management** stores new profile&settings to *User Profile&Settings*.

2.2 DFD (Level 1)



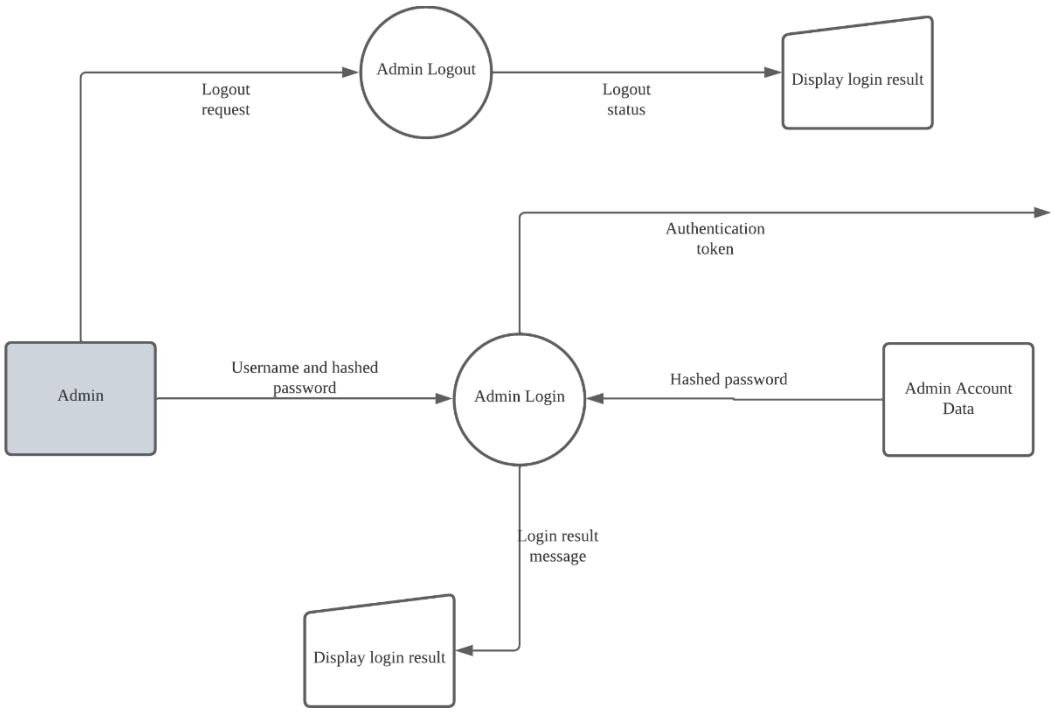
3 Feature Diagrams

3.1 Admin Login System

3.1.1 Description

Admin Login System is relatively simple. The Admin Login function receives the admin name and hashed password from **Admin**. After that, it fetches the hashed password from *Admin Account Data* database by using the username and compares the two hashed passwords. If they are the same, **Admin Login System** will send the successful login message to admin's screen and generate a random authentication token. Otherwise, **Admin Login System** sends login error messages to admin's screen. After login, admin can logout through **Logout** function, which receives logout request and display the logout status on screen.

3.1.2 DFD

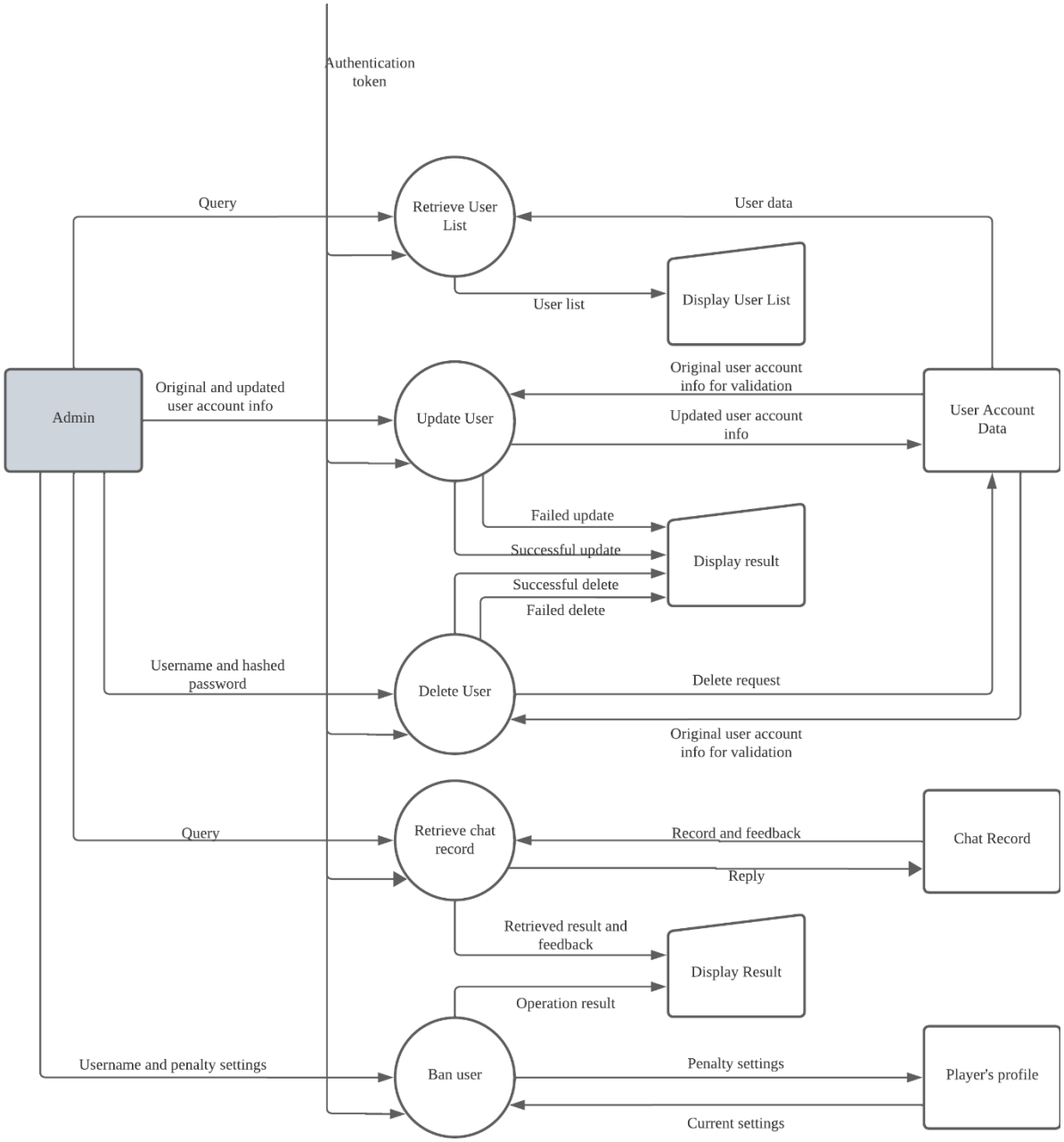


3.2 Admin Functions

3.1.1 Description

Admin Functions contain five functions for admin. The first one is Retrieve User List. It receives query from admin, retrieves eligible users from *User Account Data,* and display the retrieved user list to the screen. Second, admin can update the user account info through *Update User*. It fetches original user account info from *User Account Data* for validation and store the updated user account info to the database. Depending on whether the validation is passed or not, it will display different update result on the screen. The third one, *Delete User*, shares similar workflow with Update User, which is receiving account, validating, deleting, and displaying result. The fourth function is *Retrieve Chat Record*, of which the retrieve part is similar with Retrieve user list. It receives query, fetches the target record, and displays it on screen. What's different is that Retrieve chat record will immediately show the most recent feedback from users when the admin interacts with this interface. Admin can reply to the feedback through writing reply into the database. This function enables admins to supervise the chat to prevent malicious chat messages disrupting the community environment and receive players' feedback of the application. If malicious chat messages are detected, admins can enforce penalty settings on the malicious user depend on his/her current settings through *Ban User*, whose result is also displayed on screen.

3.1.2 DFD

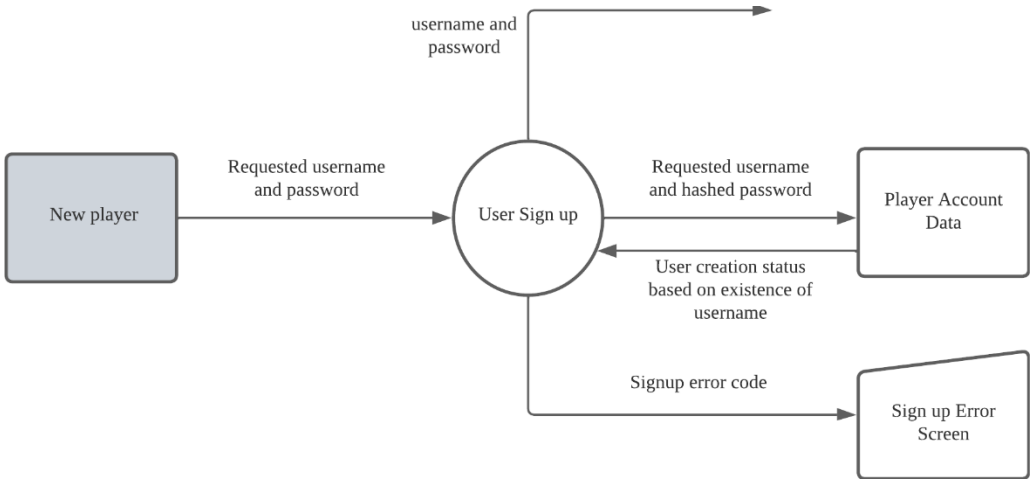


3.2 User Sign Up

3.2.1 Description

New users will first input their requested username and password to the User Sign Up system. After submitting the requested username and password, the user sign up system will check in the player account database if the username already exists. If the username already exists within the database, this signifies that another player is using the requested username and so the new user will be shown an error screen, asking the user to choose a different username. If the username does not already exist within the player account database, the requested username and hashed password will be stored in the player account database. The new user is now created, and they are provided an authentication token to play the game. If at any point in the user sign up process there is an error, the error description will be displayed in an error screen.

3.2.2 DFD



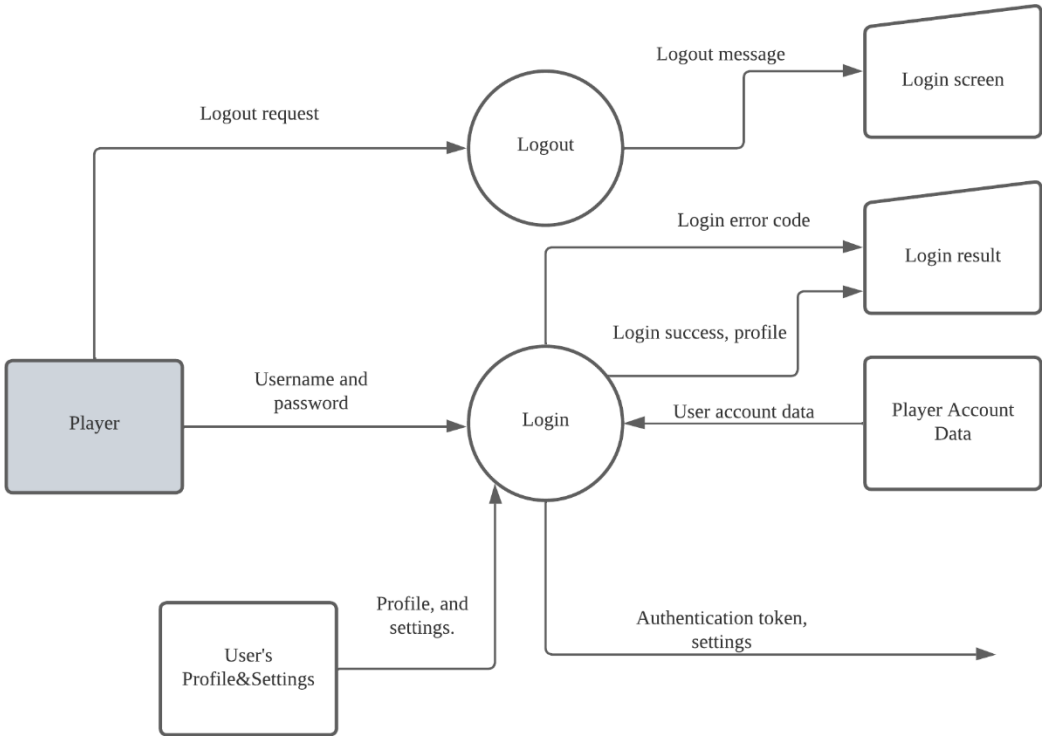
3.3 User Login

3.3.1 Description

Existing users will input their account username and password to the user login system. After submitting their username and password, the user login system will check if the username and password combination exist in the player account database. If the credentials match, the player account database will return the data associated with the user account such as game settings, game records and user profile information. The user will then be provided an authentication token which will grant the user access to the game. If there is an error in the login process, the user will be redirected to an error screen which will display the error code and error description.

When users are logged in, they will be able to log out. If a user would like to logout, they will submit a logout request, which will log the user out of their account, bring them back to the original login screen.

3.3.2 DFD

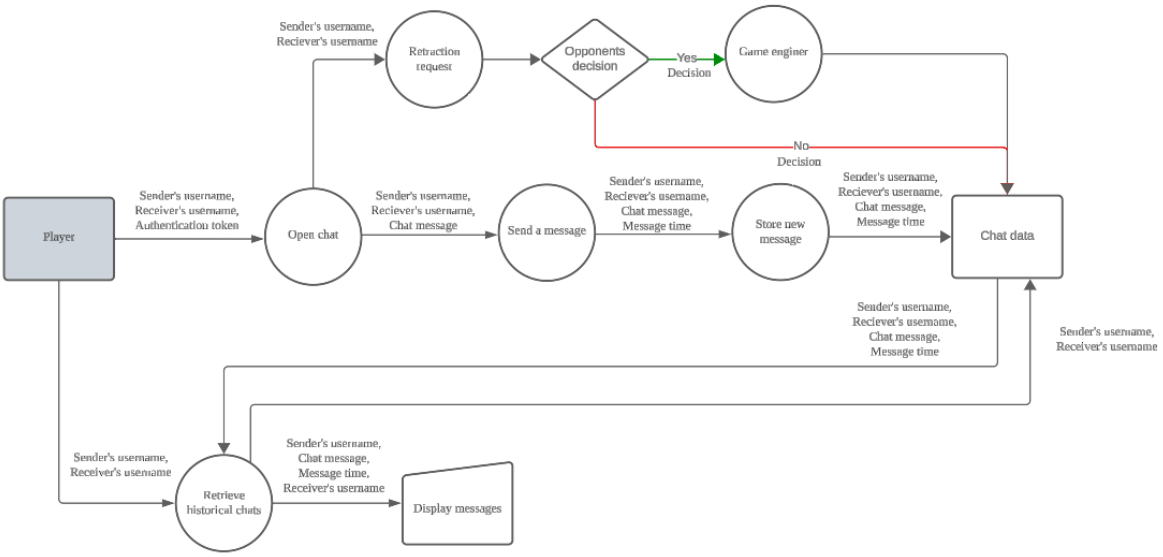


3.4 Chat

3.4.1 Description

In order for players to privately chat with one another, a user can use the open chatroom function and input their username and authentication token and their partner's username. Once open, the retrieve historical chat's function will automatically recover and display past messages from the chat database, including the respective sent time, messages, and usernames involved with each message. Users can send messages to each other within the chatroom using the send message. When new messages are sent, they will be displayed, and their data (time, message, associated usernames) will be stored in the chat database to be retrieved at a later time. As mentioned in section 2.1, the game's retraction feature is conducted through the chat. If a player indicates that they would like to retract a move, they may send a request through the chat. If their opponent agrees to the request, the move can be retracted.

3.4.2 DFD

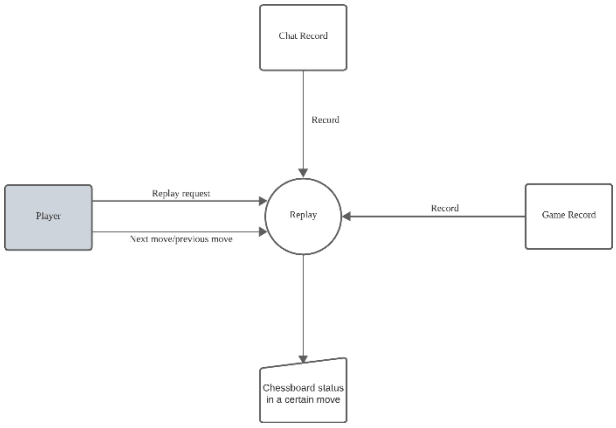


3.5 Replay

3.5.1 Description

Replay is relatively simple. User starts replay by sending replay request containing game ID to **Replay**, and then **Replay** shows the initial chessboard to the user's screen. Different from the normal game interface, in **Replay**, there are only two buttons for user: one is next move, and the other is previous move. **Replay** retrieves record from the database and displays chessboard status according to the next move/previous move that user sends.

3.5.2 Description



3.6 Matching System

3.6.1 Description

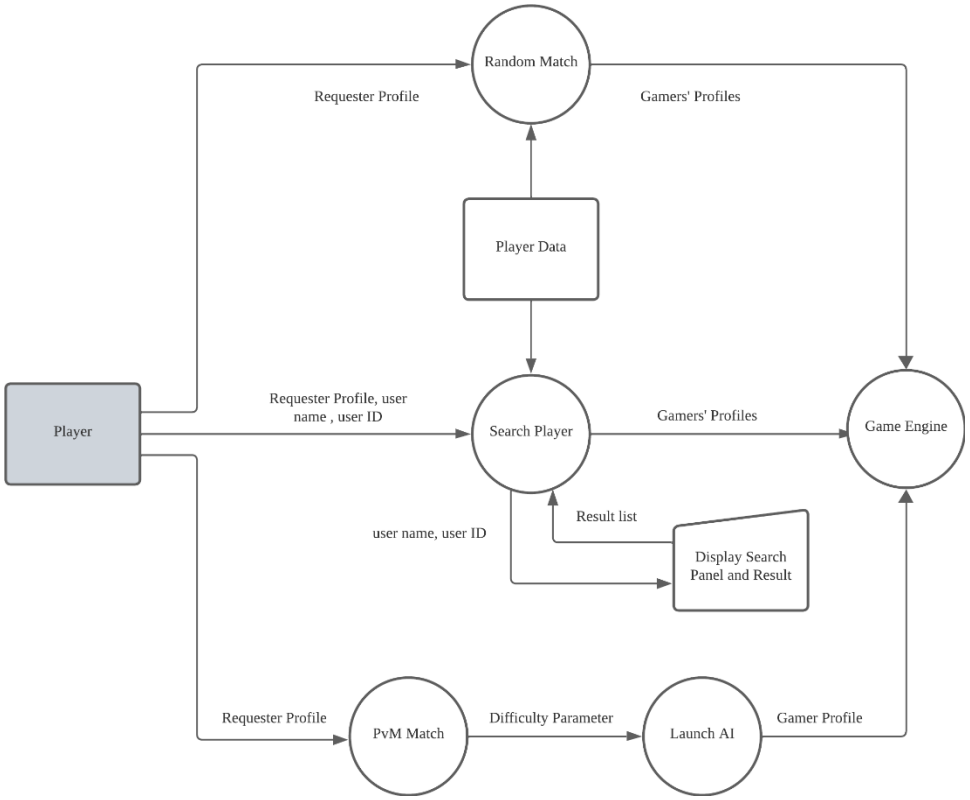
The Player Matching System determines the competing players in a game. The initiating player has three game modes to choose from: competing against a randomly selected player, playing against a friend, or playing against an AI player.

For the 'Random Player Mode', the initiating player's profile is sent to the Random Match Function. The player's profile is placed in a queue and paired with the first available player who also joins the queue. Both players' profiles and game settings are then sent to the Game Engine.

In the 'Play-With-Friend Mode', the initiating player searches for a friend's username or user ID using the Search Player Function. This function displays the search results, enabling the initiating player to find their friend and send a game invitation. Upon acceptance of the invitation, both the initiating player and their friend's profiles are sent to the Game Engine.

In the 'Play-Against-AI Mode', the initiating player's profile is sent to the PvM Match Function. The function determines the appropriate difficulty setting for the AI player based on the initiating player's historical performance. The AI player is then activated, and the initiating player's profile is sent to the Game Engine.

3.6.2 Description



3.7 Game Engine

3.7.1 Description

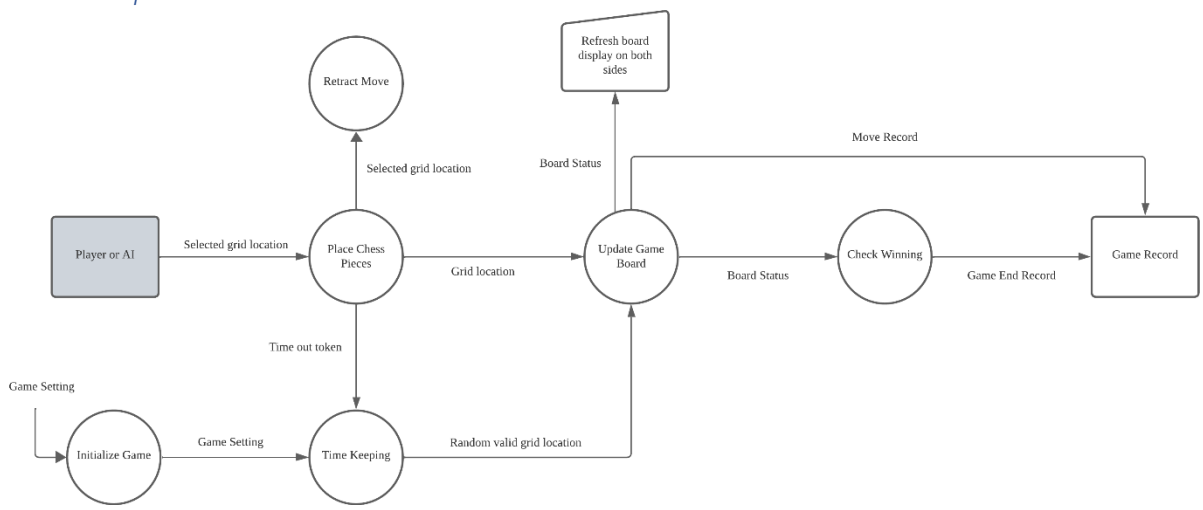
The Game Engine is responsible for all game operations. When it receives the game settings and player profiles from the Player Matching System, the Initialize Game Function is activated. This function pulls both players into the same virtual game room and activates the Time Keeping Function.

During each player's turn, they can place chess pieces and subsequently retract moves. The grid location of their moves is sent to the Update Game Board Function. If a player does not provide a valid grid location within the time limit due to prolonged decision-making or network delays, the Time Keeping Function generates a random valid grid location and ends the player's turn.

The Update Game Board Function updates the board with the moves made by players or the AI, and records the current board status. It also refreshes the board display for both players and sends the board status to the Check Winning Function. Simultaneously, it sends the move record to the Game Record Database, which records each move of the game for future replays.

The Check Winning Function checks after each move whether a player has won. If a player has won, the game ends and a game end record is sent to the Game Record Database. If the board is filled without any player winning, the game is considered a draw, and a game end record is sent to the Game Record Database to signal the end of the game.

3.7.2 Description



3.8 Profile Management

3.8.1 Description

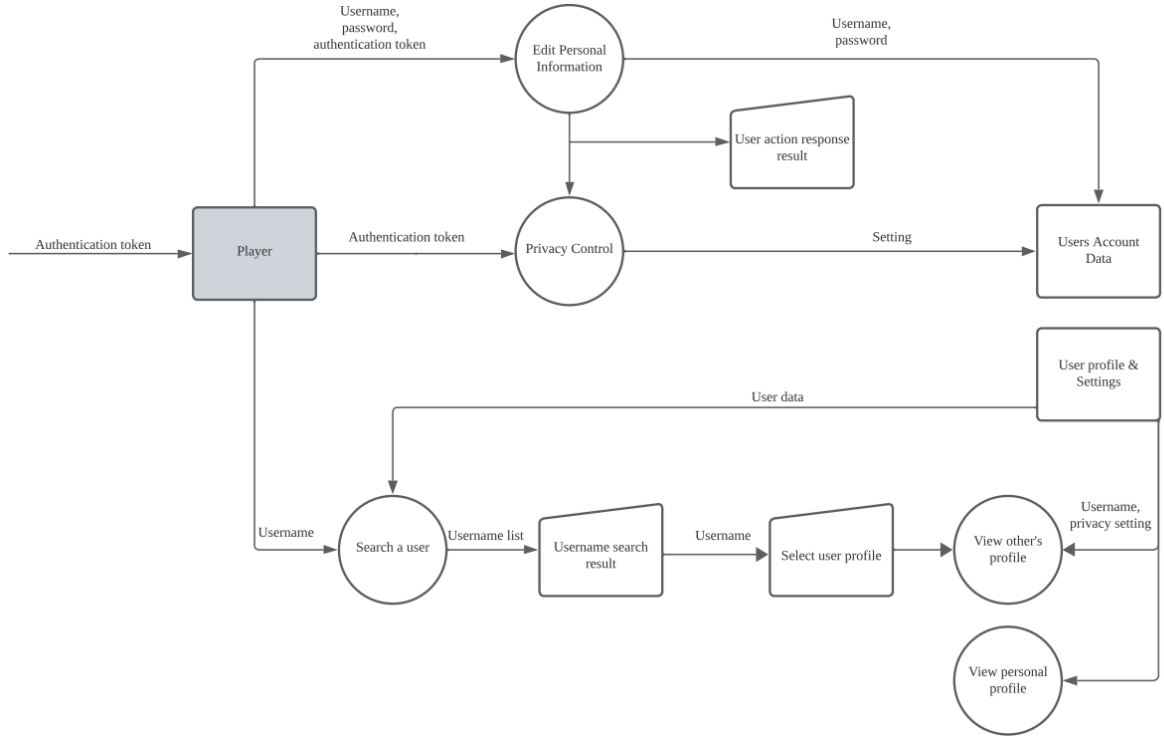
The **Profile Management System** facilitates the management of user profiles and privacy settings. When a user provides their username, password, and authentication token, the *Edit Personal Information* function is invoked, which interacts with the *User Data* database to update user details based on the user's actions and preferences. The updated information is reflected in the *User Action* response result.

The system includes a privacy control function that utilizes the authentication token to safeguard user privacy. This function allows users to set their privacy preferences, which are stored in the *User Data* database and applied to their profiles.

The system features a search a user function, which uses the provided username to search for other users within the system. This function queries the *User Data* database and returns a list of usernames that match the search criteria. The results are displayed in the username search result, which can then be used to view other users' profiles, subject to their privacy settings.

The player interacts with the system using the authentication token, which ensures secure access to profile management features. The system supports viewing both personal profiles and others' profiles, giving users the ability to control the visibility of their own information through the setting interface, while also respecting the privacy settings of other users when attempting to view their profiles. The comprehensive design of the **Profile Management System** aims to provide a secure, user-friendly interface for managing personal information and privacy within the platform.

3.8.2 DFD


Username,
password,
authentication token
Edit Personal
Information
Username,
password
User action response
result
Authentication token
Player
Authentication token
Privacy Control
Setting
Users Account
Data
User profile &
Settings
User data
Username
Search a user
Username list
Username search
result
Username
Select user profile
View other's
profile
Username,
privacy setting
View personal
profile