# WebService API

Listed below are the methods provided by the webservice, their purpose, the inputs they require and the outputs they generate.

MethodName: login

Purpose: Used to check if a user is on the system. The user provides their username and password. If successful, the method returns the users id on the system (the autokey field from the users table). If unsuccessful, it returns 0.

Parameters: String username, String password

Return Successful: int // (users.autokey)

Return Unsuccessful: 0

MethodName: register

Purpose: Used to add a user to the system. The user provides their username, password, name and surname. If successful, the method returns the users id on the system (the autokey field from the users table). If unsuccessful, it returns either “ERROR-REPEAT” if a user with that username already exists, “ERROR-INSERT” if the method couldn’t add the data to the users table, “ERROR-RETRIEVE “ if the method cannot retrieve the newly inserted data from the users table, or “ERROR-DB” if the method cannot find the DB.

Parameters: String username, String password, String name, String surname

Return successful: int // (users.autokey)

Return Unsuccessful: ERROR-REPEAT, ERROR-INSERT, ERROR-RETRIEVE, ERROR-DB

MethodName: newGame

Purpose: Used to add a new game to the system. The user provides their id (users.autokey). If successful, the method returns the id of the new game on the system (games.autokey). If unsuccessful, it returns either “ERROR-NOTFOUND” if it cannot find the id of the game it just added, “ERROR-RETRIEVE” if it cannot access the games table, “ERROR-INSERT” if it cannot add a new game to the system, “ERROR-DB” if it cannot access the DBMS.

Parameters: int uid

Return successful: int // (games.autokey)

Return Unsuccessful: ERROR-NOTFOUND, ERROR-RETRIEVE, ERROR-INSERT, ERROR-DB

MethodName: joinGame

Purpose: Used to add a second player to a game. The user provides their user id (users.autokey) and the game id (games.autokey). If successful, the method return 1. If unsuccessful, the method returns either 0 if unable to join the game or “ERROR-DB” if it cannot access the DBMS.

Parameters: int uid, int gid

Return successful: 1

Return Unsuccessful: 0, ERROR-DB

MethodName: getBoard

Purpose: Used to return all moves taken to date for a particular game. If successful, the method returns the moves taken as a string as follows {pId, x, y} for each row, each row separated by \n. The moves are returned in order of play (i.e. first move will be first row returned etc.). Using this we can evaluate how many moves have been taken (i.e. the number of rows), the last player to take a move (pId of the final row), plus all the moves taken to date. If unsuccessful, the method returns either “ERROR-NOMOVES” if no moves have yet been taken, or “ERROR-DB” if it cannot access the DBMS.

Parameters: int gid

Return Successful: String // all moves taken for this game to date.

Return Unsuccessful: ERROR-NOMOVES, ERROR-DB

MethodName: getGameState

Purpose: Used to retrieve the state of game. If successful, the method will return “-1” if waiting for a second player to join, “0” if the game is in progress, “1” if player 1 has won, “2” if player 2 has won, “3” if a draw. If unsuccessful, the method will return “ERROR-NOGAME” if it cannot find a game with the supplied gid (games.autokey), or “ERROR-DB” if unable to access the DBMS.

Parameters: int gid

Return Successful: -1, 0, 1, 2, 3

Return Unsuccessful: ERROR-NOGAME, ERROR-DB

MethodName: setGameState

Purpose: Used to update the state of a game. If successful, the method will return 1. If unsuccessful, the method will return “ERROR-NOGAME” if unable to find a game with the supplied gid (games.autokey), or “ERROR-DB” if unable to access the DBMS.

Parameters: int gid, int gstate

Return Successful: 1

Return Unsuccessful: ERROR-NOGAME, ERROR-DB

MethodName: checkSquare

Purpose: Used to see if a square has been taken. If successful, the method returns “1” if the square has been taken, “0” if the square is available. If unsuccessful, the method returns “ERROR-DB” if unable to access the DBMS.

Parameters: int x, int y, int gid

Return Successful: 1, 0

Return Unsuccessful: ERROR-DB

MethodName: takeSquare

Purpose: Used to take a square in the game. If successful, the method returns 1. If unsuccessful, the method returns 0 if there is a problem adding the square to the moves table, “ERROR-TAKEN” if the square is already taken, “ERROR-DB” if there is a problem accessing the DBMS, “ERROR” if there is a general failure.

Parameters: int x, int y, int gid, int pid

Return Successful: 1

Return Unsuccessful: 0, ERROR-TAKEN, ERROR-DB, ERROR

MethodName: checkWin

Purpose: Used to see if the game has been won. If successful, the method will return “0” if the game hasn’t been won but can continue to be played, “1” if player 1 has won, “2” if player 2 has won, or “3” if the game is a draw. If unsuccessful, the method will return “ERROR-RETRIEVE” if there is an issue getting details about the game, “ERROR-NOGAME” if no game can be found matching the gid, (games.autokey), “ERROR-DB” if there is a problem accessing the DBMS.

Inputs: int gid

Return Successful: 0, 1, 2, 3

Return Unsuccessful: ERROR-RETRIEVE, ERROR-NOGAME, ERROR-DB

MethodName: deleteGame

Purpose: Used to remove a non-started game from the system. User supplies gid (games.autokey) and their uid (users.autokey). If the game has already started or the user is not the creator of the game, the method returns “ERROR-GAMESTARTED”, otherwise it returns “1”. If unsuccessful the method returns “ERROR-DB” if it cannot access the DBMS.

Inputs: int gid, int uid

Return Successful: 1, ERROR-GAMESTARTED

Return Unsuccessful: ERROR-DB

MethodName: showMyOpenGames

Purpose: Used to retrieve all the games a particular user has started that a second player has not yet joined. If the method is successful it returns a string as follows {{g.autokey, u.username, g.started}\n}\*. If unsuccessful, the method returns “ERROR-NOGAMES” if it can find no games, or “ERROR-DB” if it cannot access the DBMS.

Inputs: int uid

Return Successful: String as described above.

Return Unsuccessful: ERROR-NOGAMES, ERROR-DB

MethodName: showAllMyGames

Purpose: Used to retrieve all the games a particular user has played/is playing. If successful it returns a string as follows {{g.autokey, h.username, o.username, g.started}\n}\*. h.username and o.username are the usernames of player 1 and player 2 respectively. If unsuccessful, the method returns “ERROR-NOGAMES” if it can find no games, or “ERROR-DB” if it cannot access the DBMS.

Inputs: int uid

Return Successful: String as described above

Return Unsuccessful: ERROR-NOGAMES, ERROR-DB

MethodName: leagueTable

Purpose: Used to display all the games that have been played. If successful it returns a string as follows {{g.autokey, h.username, o.username, g.gamestate, g.started}\n}\*. h.username and o.username are the usernames of player 1 and player 2 respectively. If unsuccessful, the method returns “ERROR-NOGAMES” if it can find no games, or “ERROR-DB” if it cannot access the DBMS.

Inputs: No inputs needed.

Return Successful: String as described above.

Return Unsuccessful: ERROR-NOGAMES, ERROR-DB

MethodName: showOpenGames

Purpose: Used to display a list of games that have started and are waiting for a second player to join. If successful it returns a string as follows {{g.autokey, h.username, g.started}\n}\*. h.username is the usernames of the player who started the game. If unsuccessful, the method returns “ERROR-NOGAMES” if it can find no games, or “ERROR-DB” if it cannot access the DBMS.

Inputs: No inputs required.

Return Successful: String as described above.

Return Unsuccessful: ERROR-NOGAMES, ERROR-DB