Use Case: Uploading a Bot

BASIC COURSE:

The user clicks on the “upload a bot” link. The user then selects their player bot from a file navigation menu to upload to the server. The user then enters a name that is associated with their player bot. The user then clicks the “submit” button. The client then uploads the bot to the server. The server then validates the bot. The server sends a message to the client stating “The bot was successfully uploaded.” The client then displays this message to the user.

ALTERNATE COURSES:

**Malicious bot detected:** The server rejects the player bot and displays an error message on the client stating, “The player bot you tried to upload was malicious and rejected.”

**Player bot is not a java file:** The server rejects the player bot and displays an error message on the client stating, “The player bot you tried to upload was rejected because it is not a java file.”

**File does not exist:** The client displays an error message stating, “The file you selected does not exist.”

Use Case: Drawing Game States

BASIC COURSE:

At the beginning of the turn, the server passes the game state to the client. The client then passes the game state to the game display module. The game display module then draws animations, game states, map, characters, and props with sprites on the client. The game display module then returns a message to the client stating that it drew the game state.

ALTERNATE COURSES:

**Turn limit exceeded:** The game has exceeded the maximum number of turns allowed in a game. The server terminates the game and passes a notification to the client stating that the game has ended. The client then passes the notification to the game display module. The game display module then displays the notification to the user.

Use Case: User Input in Testing Mode

BASIC COURSE:

After the user clicks the “User Input Testing” button, the client will first prompt the user for their input. The client shall then send the user’s input to the server. The server will then validate the user’s input, and determine if the input was a game winning move. The client will then prompt the game display module to end the current turn and halt any animations on screen. The client will then wait for a response from the server, and this process will repeat.

ALTERNATE COURSES:

**The game has been already won:** The client will display the message “Game finished” to the user. The client will refuse any further input from the user, and will return the user to the Testing Arena main menu.

**The user input was not valid:** The client will display the error message “The user input was not valid” to the user. The client will refuse any further input from the user, and will return the user to the Testing Arena main menu.

Use Case: Client-side Bot Input in Testing Mode

BASIC COURSE:

After the user clicks the “Client Bot Input” button, the client will first prompt the user to modify their bot. The client shall then run the bot once, obtain the bot’s output, and then send the bot’s output to the server. The server will then validate the bot’s output, and determine if the output was a game winning move. The client will then prompt the game display module to end the current turn and halt any animations on screen. The client will then wait for a response from the server, and this process will repeat.

ALTERNATE COURSES:

**The user does not modify their bot:** The client will run the bot that was originally submitted to the Testing Arena. The client will make no further changes to the Basic Course scenario.

**The game has been already won:** The client will display the message “Game finished” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

**The bot input was not valid:** The client will display the error message “The bot input was not valid” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

**The bot takes too long to produce output:** The client will display the error message “The bot failed to quickly produce output” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

Use Case: Server-side Bot Input in Testing Mode

BASIC COURSE:

After the user clicks the “Server Bot Input” button, the client will prompt the user to proceed to the next turn. The client shall then run the bot once, after which the client will send the bot’s output to the server. The server will then validate the bot’s output, and determine if the output was a game winning move. The client will then prompt the game display module to end the current turn and halt any animations on screen. The client will then wait for a response from the server, and this process will repeat.

ALTERNATE COURSES:

**The game has been already won:** The client will display the message “Game finished” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

**The bot input was not valid:** The client will display the error message “The bot input was not valid” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

**The bot takes too long to produce output:** The client will display the error message “The bot failed to quickly produce output” to the user. The client will refuse any further input from the bot, and will return the user to the Testing Arena main menu.

Use Case: Challenge Mode

BASIC COURSE:

The user selects Challenge mode to play. The client will prompt the user for the challenge they would like to play. The user will select the challenge and the player bot they wish to use. The client will send the bot and challenge to the server. The server will run the challenge mode game with these options by passing commands produced by the bots to the game evaluation module. The game evaluation module will produce a new game state.

The server will send the client the game data (states, commands, and errors). The client will pass game states to the Game Display Module. The Game Display Module will display the game to the user.

ALTERNATE COURSES:

**Bot input produces error:** The server will treat the command as “idle” and pass this command to the game evaluation engine.

**500 turns exceeded:** The server will automatically terminate any games lasting a large preset number of terms (currently 500) as a precaution to an endless stalemate.

**Bot takes too long to produce output:** The server will terminate a bot that takes too long to produce a command and produce an error stating “Server-side bot did not produce command fast enough”.

Use Case: Replay Mode

BASIC COURSE:

The user selects replay mode. The user selects from a list of games that are able to be viewed. The server will send the client the game data to be viewed (states, commands). The client will accept this data and use the Game Display Module to display the game to the user. The user will have the option to rewind, fast-forward, pause, and play the game with buttons provided.

ALTERNATE COURSES:

**Live game in replay mode:** The user will not be able to rewind, fast-forward, pause, and play in the case that the game is a live game.