**Move Piece Scenario:**

* When a player selects a space within range of its robot’s remaining mobility points and selects “Move” from the context menu that appears as a result of selecting the space, the robot is moved to the selected space, the robot’s remaining mobility points are reduced by the cost of the movement, and the history log and what can be seen by the robot are updated as necessary.

Preconditions:

* A player is active.
* The active player has a live robot.
* A robot belonging to the active player is active.
* The robot has mobility points to spend.

Flow of events:

* The use-case begins when it is the active player’s turn.
* The system checks mobility points and shades spaces determined to be within range of the active robot’s remaining mobility points.
* The active player selects a shaded space.
* A context menu appears offering the player the ability to “Move,” “Shoot,” or “Inspect” the space, as well an option to “Cancel” the selection.
* The active player selects the “Move” option from the context menu.
* The piece is rotated if necessary.
* The piece is moved to the selected space.
* The system decreases the mobility points of the active robot by the cost of the move.
* The system updates the game history log.
* The use-case ends when the active player ends their turn or the mobility points of the active robot depletes to zero.

Alternative flow of events:

* The player selects the “Shoot” option from the context menu.
* The player selects the “Inspect” option from the context menu.
* The player selects the “Cancel” option from the context menu.“Move,” “Shoot,” or “Inspect”
  + Another space is selected.
  + The player ends their turn.

Postconditions:

* The active piece is moved to a space within range of its remaining mobility points.
* The active piece’s mobility points are decreased by the cost of the move.
* The game history log is updated.

Errorconditions:

* The active piece is moved to the wrong space.
* The piece is moved to a space outside its range of remaining mobility points.

**Shoot Scenario:**

* When a player selects a space within range of its robot’s range points and selects “Shoot” from the context menu that appears as a result of selecting the space, the selected space is shot at, the remaining health of any robots occupying the space is reduced by the attack points of the player’s robot, and the game history log and the live robots are updated as necessary.

Preconditions:

* A player must be active.
* The active player has a live robot.
* A robot belonging to the active player is active.
* The active robot has not already shot.

Flow of events:

* The use-case begins when it is the active player’s turn.
* The system checks the active robot’s range points and shades spaces determined to be within range of the active robot’s range points.
* The active player selects a shaded space.
* A context menu appears offering the player the ability to “Move,” “Shoot,” or “Inspect” the space, as well an option to “Cancel” the selection.
* The player selects the “Shoot” option from the context menu.
* The piece shoots at the selected space.
* The system decreases the active robot’s attack points from the health points of each robot occupying the selected space.
* The system updates the game history log and the remaining robots as necessary.
* The use-case ends when the active player ends their turn or the active robot has shot.

Alternative flow of events:

* The player selects the “Move” option from the context menu.
* The player selects the “Inspect” option from the context menu.
* The player selects the “Cancel” option from the context menu.
  + Another space is selected.
  + The player ends their turn.

Postconditions:

* The active robot’s attack points are subtracted from the health points of all and only robots occupying the space shot at.
* Any robot whose health depletes to zero is eliminated.
* The robot shoots only once.

Errorconditions:

* The robot has shot for the second time.
* A robot occupying the selected space is not damaged.
* A robot not occupying the selected space loses health points.
* Health points are decreased incorrecly for a robot occupying the selected space.

**End Turn Scenario:**

* When a player clicks the “End Turn” button after some combintation of moving, shooting, and viewing the board or the player’s time limit has expired, the system ends the player’s turn and selects the next player and robot.

Preconditions:

* Either:
  + A player is active and has ended their turn.
  + The time limit to play a turn has expired.

Flow of events:

* The use-case begins when the active player has chosen to end their turn and clicked the “End Turn” button.
* A confirmation box prompts the acitve player for confirmation of their choice to end their turn.
* The active player clicks the “Yes” button.
* The system ends the active player’s turn.
* The system updates the game history log.
* The system determines the next active player and robot and sets the active player and robot accordingly.

Alternative flow of events:

* The player selects the “No” button when prompted for confirmation of their choic eto end their turn.
* The time limit to play a turn expires and the system automatically ends the player’s turn without confirmation.

Postconditions:

* The currenly active player’s turn is ended and the next player and robot become active.

Errorconditions:

* An eliminated player becomes active.
* The active player’s turn after clicking “No” when prompted for confirmation of their choice to end their turn.
* The active player’s turn does not end after clicking “Yes” when prompted for confirmation of their choice to end their turn.
* The active player’s turn does not end after the time limit to play a turn has expired.

**Pan Screen Scenario:**

* After a player clicks the “Pan Board” button, the player enters the “Pan Board” mode, presses the mouse up against one of the edges of the screen to indicate the direction to move their view, and the system moves the player’s view in the indicated direction. The arrow keys may also be used to indicate the direction of movement.

Preconditions:

* No other action is currenly in progress and the player has clicked the “Pan Board” button.

Flow of events:

* The use-case begins when the user clicks the “Pan Board” button.
* The player enters the “Pan Board” mode.
* The player presses the mouse against the edge of the screen.
* The system moves the player’s view in the direction indicated by the edge the mouse is currently touching.
* The player moves the mouse away from the edge of the screen.
* The system stops the player’s view from moving.
* The use-case ends when the player selects the “Exit Pan Board Mode” button.

Alternative flow of events:

* The player indicates the direction to move the screen with the arrow keys.
* The player uses the escape key to exit the “Pan Board” mode.

Postconditions:

* The player’s view is temporarily moved to a different position on the game board.

Errorconditions:

* The player’s view does not pan.
* The player’s view does not stop panning.
* The player’s view does not pan in the correct direction.
* The player is unable to enter the “Pan Board” mode.
* The player is unable to exit the “Pan Board” mode.

**Zoom Screen Scenario:**

* After the player clicks the “Zoom View” button, the player enters the “Zoom View” mode, the player adjusts the zoom of their view to their liking using the “+” button to indicate the view is to be zoomed in and “-” button to indicate the view is to be zoomed out. The system saves the player’s zoom level after the player clicks the “Exit Zoom Mode” button.

Preconditions:

* No other action is currently in progress and the player has clicked the “Zoom Screen” button.

Flow of events:

* The use-case begins when the user clicks the “Zoom View” button.
* The player enters the “Zoom View” mode.
* The player clicks the “+” button to indicate the view is to be zommed in or the “-” button to indicate the view is to be zoomed out.
* The system zooms the player’s view as the player has indicated.
* The system saves the player’s zoom level.
* The use-case ends when the player clicks the “Exit Zoom Mode” button.

Alternative flow of events:

* The player moves the scroll-wheel forward to indicate the view is to be zoomed in or backwards to indicate the view is to be zoomed out.

Postconditions:

* The system adjusts the player’s view to the indicated zoom level.

Errorconditions:

* The system does not adjust the player’s view to the indicated zoom level.
* The system does not stop zooming.
* The system does not zoom.

**Inspect Space Scenario:**

* When the player selects a space and selects “Inspect” from the context menu that appears as a result of selecting the space, the system will display information and statistics regarding units occupying the selected space in another window.

Preconditions:

* A player is active.
* The active player has a live robot.
* A robot belonging to the active player is active.

Flow of events:

* The use-case begins when it is the active player’s turn.
* The system checks the active robot’s range points and shades spaces determined to be within range of the active robot’s range points.
* The active player selects a shaded space.
* A context menu appears offering the player the ability to “Move,” “Shoot,” or “Inspect” the space, as well an option to “Cancel” the selection.
* The player selects the “Inspect” option from the context menu.
* The system displays information and statistics regarding the units occupying the selected space in another window.
* The system updates the game history log.
* The use-case ends when the active player ends their turn.

Alternative flow of events:

* The use-case begins when not other action is currently in progress.
  + The system shades
  + The system checks the active robot’s range points and shades spaces determined to be within range of the active robot’s range points.
  + The active player selects a shaded space.
  + The system displays information and statistics regarding the units occupying the selected space in another window.
  + The system updates the game history log.
  + The use-case ends when the user begins a different action.
* The player selects “Cancel” from the context menu.

Postconditions:

* Information regarding the units occupying the selected space is displayed in another window.

Errorconditions:

* Information regarding units occupying the selected space is not displayed in another window.
* The information displayed in another window is not correct information regarding units occupying the selected space.

**View Game History Log:**

* Once the player toggles the game history log on, a small window in the corner of the main game screen is displayed containing all actions for all player throughout the current game. The window can be scrolled to view older or new actions.

Preconditions:

* No other action is currently in progress and the player has toggled the game history log on.

Flow of events:

* The use-case begins when no other action is currenly in progress.
* The player toggles the game history log.
* A small window appears in the corner of the main game window.
* The small window displays the sequence of actions taken by all players the player can be aware of.
* The player can see older or more recent actions by scrolling the small window using a scrollbar at the side of the small window.
* The use-case ends when the player toggles the game history log off.

Alternative flow of events:

* None.

Postconditions:

* A small window displaying the sequence of actions known to the player is shown.

Errorconditions:

* The history game log cannot be toggled on.
* The history game log cannot be toggled off.
* Actions not known to the player are displayed in the history game log.

**End Game Scenario:**

* When the player clicks the “End Game” button in the top-left corner of the main game window, a confirmation window appears prompting the player to confirm their decision to end the game. The game is ended when the user clicks the “Yes” button in the confirmation window. The game continues if the player clicks the “Cancel” button in the confirmation window.

Preconditions:

* A game is currently in progress.
* No other action is currently in progress.

Flow of events:

* The user-case begins when no other action is in progress.
* The player clicks the “End Game” button in the top-left corner of the main game window.
* A confirmation window appears asking the player to confirm their decision to end the game.
* The player clicks the “Yes” button in the confirmation window.
* The system ends the game for all players.
* The system returns the player to the main menu window.
* The use-case ends when the player has returned to the main menu screen.

Alternative flow of events:

* The player clicks the “Cancel” button to cancel their decision to end the game.
* Another player decides to end the game.

Errorconditions:

* The confirmation window does not appear prompting the player to confirm their decision to end the game.
* The current game does not end when the player clicks the “Yes” button in the confirmation window.
* The current game does end when the player clicks the “Cancel” button in the confirmation window.

**View Game Statistics Scenario:**

* When the player clicks the “View Stats” button, another window appears displaying the current statistics of the player’s robots. The statistics consist of a robot’s remaining health, remaining mobility points, range points, attack points, damage dealt, number of kills, and distance moved.

Preconditions:

* No other action is currently in progress.

Flow of events:

* The use-case begins when the are no other actions in progress.
* The player toggles the game statistics window.
* A small window displaying the current game statistics for the player’s robots is shown.
* The game statistics displayed consist of remaining health, remaining mobility points, attack points, range points, damage dealt, number of kills, and distance moved.
* The use-case ends when the player toggles the game statistics window off.

Alternative flow of events:

* None.

Postconditions:

* A small window displaying the current game statistics of the player’s robots is shown.

Erroconditions:

* The current game statistics cannot be toggled on.
* The current game statistics cannot be toggled off.
* The current game statistics displayed are inaccurate.
* The statistics shown do not belong the player’s robots.

**Start New Game Scenario:**

* When the application starts up and the user clicks the “Start Game” button from the main menu, the user is redirected to the “Player Select” window, in which the user selects the number of players, whether each player is human or AI, enters the name of each player and clicks the “Start” button, upon which a new game begins with the specified players. From the “Player Select” screen, the user has the option of clicking the “Options” button and adjusting the game options before beginning the game.

Preconditions:

* The user is running the system and is currently in the main menu window.

Flow of events:

* The use-case begins when the user begins running the system.
* The user clicks the “Start Game” button.
* The user is redirected to the “Player Select” window.
* The user selects the number of players, whether each player is human controlled or AI, and enters a name for each player.
* The user adjusts other options, such as turn clock time.
* The user clicks the “Start Game” button.
* The system begins a new game with the given game parameters.
* The use-case ends when the system begins a new game.

Alternative flow of events:

* The user clicks the “Options” button from the “Player Select” screen.
  + The user is redirected to the “Options” window, which has a list of game options.
  + The user changes some, all, or none of the game options.
  + The user clicks the “Save Options” button to save the options as they appear in the “Options” window or the “Cancel” button to cancel all changes to the options.
* The player clicks the “Back” button to return to the main menu.

Errorconditions:

* The game begins when necessary parameters have not been specified.
* The game begins with the wrong parameters.
* The “Options” window is not shown when the player clicks the “Options” button.
* The “Back” button does not return the user to the main menu.
* The game does not begin when the user clicks the “Start Game” button.

**View Options Scenario:**

* When the user clicks the “Options” button in the “Player Select” window, the user is redirected to the “Options” window. The “Options” window contains all the game options the user is allowed to change. Changing the options is as simple as selecting an alternative to the current option setting. The changes will only be applied to the game if the user clicks the “Save Options” button. Otherwise, if the user clicks the “Cancel” button, all changes will be discarded.

Preconditions:

* The user is running the application and is in the “Player Select” window.

Flow of events:

* The use-case begins after the player has clicked the “Start Game” button in the main menu window and has been redirected to the “Player Select” window.
* The user clicks the “Options” button.
* The user is redirected to the “Options” window.
* The user can change game certain game options.
* The user clicks the “Save Options” button.
* The system saves the options as they are in the “Options” window.
* The user is returned to the “Player Select” window.
* The use-case ends when the user is returned to the “Player Select” window.

Alternative flow of events:

* The user clicks the “Cancel” button.
* The system discards all changes made to the options.
* The user is returned to the “Player Select” window.

Post Conditions:

* The game options the player is allowed to change are set as indicated in the “Options” window.

Errorconditions:

* The user is not redirected to the “Options” window upon clicking the “Options” button in the “Player Select” window.
* The changes the user makes to the options are not saved when the user clicks the “Save Options” button.
* The changes the user make to the options are saved when the user clicks the “Cancel” button.
* The user is not returned to the “Player Select” window.
* The system does not save the options correctly when the user clicks the “Save Options” button.

**View Game Results Scenario:**

* When the game ends, the winner of the game is displayed in a small window. The player clicks the “Okay” button and is redirected to another window in which the game results are displayed. Results are displayed for each robot of each player.

Preconditions:

* A game has just ended.

Flow of events:

* The use-case begins when the game ends.
* The player is shown a small window indicating which player has won the game.
* The player clicks the “Okay” button.
* The player is redirected to another window in which the game results of each robot for each player is shown.
* The player observes the game results.
* The player clicks the “Back to Title” button.
* The player is returned to the “Title” window.
* The use-case ends when the player is redirected to the “Title” screen.

Alternative flow of events:

* The player clicks the “View Log” button and is shown the entire game history log.

Postconditions:

* The player is shown the game results and then redirected to the “Title” screen.

Errorconditions:

* The game results are not shown.
* The incorrect game results are shown.
* The player is not redirected back to the “Title” window upon clicking the “Back to Title” button.

**Monitor Turn Length:**

* Once the player begins their turn, they have a limited amount of time to complete their turn. The system monitors the time a player has consumed and ends their turn once the maximum time limit has been reached. The time allowed for a player to take their turn can be adjusted in the game settings.

Preconditions:

* A player must be active.
* A robot belonging to the active player must be active.

Flow of events:

* The use-case begins when the active player’s turn begins.
* The time consumed by the player is monitored by the system.
* The player’s turn is ended by the system if the player should consume the maximum amount of allowed time.
* The use-case ends when the player’s turn ends.

Alternative flow of events:

* The player chooses to end their turn before the time limit expires by clicking the “End Turn” button.

Postconditions:

* The player’s turn does not exceed the allowed time limit.

Errorconditions:

* The system does not end the player’s turn after the allowed time limit has expired.
* The system ends the player’s turn before the allowed time limit has expired and the player has not chosen to end their turn by clicking the “End Turn” button.

**Register Scenario:**

* After clicking “Start Game” and selecting at least on player to be AI, the user clicks the “Register Robot” button. A small window appears prompting the user to select a robot file from their file system. The user locates and selects their desired file, clicks the “Register” button, and the system registers the robot into the game.

Preconditions:

* The “Select Player” window must be shown and the user must click the “Register Robot” button.
* The registered robot program must be written in forth and be syntactically correct.

Flow of events:

* The use-case begins when the user clicks the “Start Game” button in the “Title” window.
* The user clicks the “Register Robot” button next to an AI player.
* Another window appears prompting the user to select a robot file to register.
* The user locates and selects their desired robot file.
* The user clicks the “Register” button.
* The robot file is registered to the player.
* The use-case ends when the robot file is registered.

Alternative flow of events:

* The user clicks the “Cancel” button when prompted for the robot file to register.

Postconditions:

* A robot file is registered into the database of robots.

Errorconditions:

* The user cannot select and register a robot file.
* The robot file is not correctly registered.

**Retire Scenario:**

* While the user is browsing the robots, after clicking the “View Robots” button from the “Title” screen, the user can choose to retire a robot by clicking the “Retire” button next to each robot file. When the user clicks the “Retire” button, the robot can no longer be selected to play in the game.

Preconditions:

* The user must be viewing all the robots and click the “Retire” button.

Flow of events:

* The use-case begins when the user clicks the “View Robots” from the “Title” window.
* Another window is shown displaying all the current robots that can be used.
* The user clicks the “Retire” button associated with the robot the user desires to retire.
* The system retires the robot and the robot can no longer be used in the game.
* The use-case ends once a robot has been retired.

Alternative flow of events:

* The user exits the “View Robot” window without retiring a robot.

Postconditions:

* Only the desired robot is retired.

Errorconditions:

* The desired robot is not retired after its “Retire” button is clicked.
* The wrong robot is retired.

**Download Scenario:**

* After clicking the “Download Robot” button from the “Title” screen, a window appears prompting the user to select a robot file to download from their file system. The user locates and selects their desired robot file and clicks the “Download” button. The system then downloads the robot file.

Preconditions:

* The system must be running and the user must have clicked the “Download Robot” button from the “Title” window.

Flow of events:

* The use-case begins when the user starts the system.
* The user clicks the “Download Robot” button from the “Title” window.
* Another window appears prompting the user to select their desired robot file from their file system.
* The user locates and selects their desired robot file.
* The user clicks the “Download” button.
* The system checks the robot file and downloads the robot file if it is syntactically correct.
* The use-case ends when the system downloads the robot file.

Alternative flow of events:

* The user clicks the “Cancel” button when prompted to select a robot file from their file system.
* The system rejects the robot file after checking the file to be syntactically correct.

Postconditions:

* A syntactically correct robot file is downloaded into the system.

Errorconditions:

* The user cannot select and download a robot file from their file system.
* A syntactically incorrect robot file is accepted.
* A syntactically correct robot file is rejected.
* The system does not correctly update the collection of robots.

**Revise Scenario:**

* After clicking the “View Robots” button from the “Title” window, the user is shown the collection of robots, each with an associated “Revise” button. The revise button allows the robot to be revised if it has not been performing well.

Preconditions:

* The user is shown the collection of robots and clicks the “Revise” button for a robot file.

Flow of events:

* The use-case begins when the user clicks the “View Robots” button in the “Title” window.
* The user locates and clicks the “Revise” button associated with their desired robot file.
* The robot file is opened using the default text editor.
* The user revises the robot file, saves, and exits the text editor.
* The system parses and checks the file is still syntactically correct.
* The system saves the changes made to the robot file.
* The use-case ends when the user is finished revising the robot file and any changes have been saved by the system.

Alternative flow of events:

* The user exits the default text editor without saving the robot file.
  + Any changes made to the robot file are not saved.
* The changes made are rejected by the system
  + The user is given the chance to correct the mistakes via the default text editor.

Postconditions:

* Any acceptable changes made to the robot file are saved by the system.

Errorconditions:

* The user cannot revise the robot file.
* Acceptable changes made to the robot file are not saved by the system.
* Unacceptable changes made to the robot file are saved by the system.
* The system does not update the robot file with the correct revisions.

**Update Statistics Scenario:**

* After the user starts the system and clicks the “View Robots” button, the user is shown a collection of robots, each having an associated “Update Stats” button. When the user clicks the “Update Stats” button, another small window is shown with fields for adding new statistics to the robot file which will be used to update the robot file.

Preconditions:

* The user is shown the collection of robots and clicks the “Update Stats” button for a robot file.

Flow of events:

* The use-case begins when the user clicks the “View Robots” in the “Title” window.
* The user locates and clicks the “Update Stats” button associated with their desired robot file.
* The user is shown another window containing fields for adding new statistics to the robot file which will be used to update the robot file.
* The user enters the data into the fields and clicks the “Update” button.
* The system checks the data for validity and accepts the data.
* The system adds the updates to the statistics of the robot file.
* The use-case ends when the updates are applied to the robot file.

Alternative flow of events:

* The user clicks the “Cancel” button.
  + The statistics of the robot file are not updated.
* The system rejects the data.
  + The user is given another opportunity to update the data.

Postconditions:

* Valid updates are made to the robot file statistics as specified by the user.

Errorconditions:

* The user cannot enter data to be used to update the robot file statistics.
* The system rejects valid data.
* The system accepts invalid data.
* The system does not correcly update the robot file statistics.

**Enumerate Scenario:**

* After the user has started the system, the user clicks the “View Robots” button and the robots are enumerated. Another window is shown in which the robot name is displayed along with a picture of the robot and all of its statistics. A brief snippet of the program associated with the robot is displayed as well.

Preconditions:

* The user is running the system and the user has clicked the “View Robots” button.

Flow of events:

* The use case begins when the user starts the system.
* The user clicks the “View Robots” button.
* Another window is shown enumerating the robot files.
* The robot name is accompanied by a picture, the robot file statistics, a brief snippet of the program, and some buttons for interacting with the robot file.
* The user examines the robot files as they are presented and may click some of the buttons associated with the robots to perform some actions on the robots.
* The use-case ends when the user returns to the “Title” window.

Alternative flow of events:

* None.

Postconditions:

* The robot files are enumerated correctly.

Errorconditions:

* One or more robot files are enumerated incorrectly.
* One or more robot files are not enumerated.

**Exit Application:**

* While the system is running and after the Host has been returned to the Title window by navigating back, the user can click the “Exit Application” button to exit and kill the system.

Preconditions:

* The system is running and the Host is currently in the Title window.

Flow of events:

* The use-case begins when the user is navigated to the Title window.
* The user clicks the “Exit Application” button.
* The system kills any processes belonging to the system and stops execution and releases all its resources.
* The use-case ends when the system stops execution.

Alternative flow of events:

* None.

Postconditions:

* The system stops execution and releases all its resources.

Errorconditions:

* The system does not stop execution.
* The system does not release all its resources.

**Navigate Board:**

* When the Player wishes to see a part of the game board that is not currenly visible in their main game window, the Player will execute a combintation of pan and zoom actions to bring the desired region of the game board into their view.

Preconditions:

* A game is currently in progress.
* No other action is currenly in progress.

Flow of events:

* The use-case begins when no other actions are currently in progress.
* The Player decides to bring in to view a region of the board not currently visible.
* The Player may perform a zoom action to alter the zoom level followed by a pan action to bring into view another region of the board.
* The Player may perform a pan action to bring into view another region of the board followed by a zoom action to alter the zoom level.
* The Player continues to perform a combination of pan and zoom actions until the desired region is visible in their view of the screen.
* The use-case ends when the Player brings into view the desired region of the boad.

Alternative flow of events:

* The user decides to only perform zoom actions.
* The user decides to only perform pan actions.

Postconditions:

* The Player is able to navigate the board such that the whole board can be viewed or any particular region of the board can be viewed.

Errorconditions:

* The Player is unable to pan the board.
* The Player is unable to zoom the board.
* There is a region of the board that cannot be viewed by the Player.

**Take Turn (Player):**

* Once the Player’s turn has begun, the Player has the option to perform a certain combination of move piece, shoot, and inspect space actions. While it is the player’s turn, the Player can perform the navigate board action, view game history log action, and monitor their own statistics. Alternatively, the player may choose to not perform any actions and simply to perform an end turn action.

Preconditions:

* A Player is active and must be human.
* The active Player has at least one remaining robot.
* A robot belonging to the active Player is active.

Flow of events:

* The use-case begins when it is the Player’s turn.
* The Player may or may not perform a navigate board, view game history log, or inspect space action.
* The player may or may not perform a move piece action.
* The Player may or may not perform a navigate board, view game history log, or inspect space action.
* The player may or may not perform a shoot action.
* The Player may or may not perform a navigate board, view game history log, or inspect space action.
* The player may or may not perform another move piece action provided the robot has remaining mobility points.
* The Player may or may not perform a navigate board, view game history log, or inspect space action.
* The Player clicks the “End Turn” button to perform an end turn action.
* The use-case ends when the Player’s turn ends.

Alternative flow of events:

* The Player’s allowed time to take their turn expires.
  + The system ends the Player’s turn.

Postconditions:

* The Player has played their turn within the allowed time limit.

Errorconditions:

* The Player’s turn exceeds the time limit.
* The Player cannot perform one or more of the move piece, shoot, navigate board, or end turn actions.
* An action performed by the Player during the Player’s turn is not correctly executed.

**Take Turn (AI):**

* Once the AI’s turn has begun, the system will read the AI’s program and determine which actions to perform based on the program provided to the system by the robot file. The AI can perform the move piece, shoot, and inspect space actions.

Preconditions:

* A Player is active and is AI.
* The active player has at least one remaining robot.
* A robot is active and must belong to the active Player.

Flow of events:

* The use-case begins when it is the AI’s turn.
* The AI may or may not perform a move piece action depending on the AI’s program.
* The AI may or may not perform a navigate board, view game history log, or inspect space action depending on the AI’s program.
* The AI may or may not perform a shoot action depending on the AI’s program.
* The AI may or may not perform a navigate board, view game history log, or inspect space action depending on the AI’s program.
* The AI may or may not perform a move piece action depending on the AI’s program and if the AI has enough remaining mobility points.
* The AI may or may not perform a navigate board, view game history log, or inspect space action.
* The AI chooses to end their turn.

Alternative flow of events:

* None because the AI program cannot change during gameplay.

Postconditions:

* The AI has taken their turn according to how the AI program was specified.

Errorconditions:

* The AI does not adhere to the program.
* The AI does not play the game by the rules.