<u>Informal Use Cases:</u>

Queue Up Game

- Enter username and password
- Enter multiplayer matchmaking system
- Wait approximately 30s to 2 minutes while queueing up

In-lobby Customization

- Pick out customize tab while waiting in a lobby
- Choose between different tabs in the customizable section: monsters, colors, etcs.
- Continue further customization in-game (for premium users)

Avatar/Profile Customization

- Go to Profile
- Click on User Profile
- Select Customization
- Select tab with detail you wish to customize
- Click save to save your avatar
- Can also click random to generate random costume

Player Turn

- Player activates favor card effects if applicable
- Player rolls dice and keeps dice they want and rerolls the others. (3 rolls occur unless favor card affect states otherwise)
- Player commences with action phase.
- Player may add VPs to their VP count, may heal LP, and/or may attack other players using fish.
- Player must land on the Island if the Island is empty.
- Player may purchase community favor cards to use.
- Player activates favor cards purchases if applicable.

Formal Use Cases:

Name: Register new player account
Identifier: UC1
Description: For a new user, add to the system a unique username and associated email, with a secure password.
Preconditions: User does not have an account known by the system; cannot enter website beyond login page.
Postconditions: User/player is able to enter the game site using login credentials (username and password); game’s database securely stores the username and password.
Basic Course of Action:

1. Use case begins when the user enters the domain name into their browser search bar.
2. The website displays the splash screen briefly before bringing the user to the login page.
3. The user clicks on the link to create a new player account.
4. The site loads the registration page.
5. User enters their email (for password recovery and game updates) and a desired username that fits the requirements of being between 6 and 64 characters and consisting of only letters, numbers, and special characters.
6. Password is created by the user that is:
    a. At a minimum, 8 characters
    b. Has one of at least 3 of the following 4 categories:
        i. Upper case (A-Z)
        ii. Lower case (a-z)
        iii. Digit (0-9)
        iv. Special character (~`!@#$%^&*()+=_-{}[]\|:;”’?/<>,.)
7. User clicks button to check validity.
8. System checks username and password validity. It then checks availability of username, if valid. Provides green checks next to both text boxes because they are both allowed by the system.
9. User clicks button to finish the registration process and is now a game player.
10. System returns player to login screen.
11. Player enters their username and password, the latter of which is hidden by black dots, to protect their privacy.
12. The use case ends when the system checks against the database, finds and validates the credentials, and takes the player to the home screen.

Alternate Course A: Username doesn’t fit the requirements, but password is fine.
A.8 System checks username validity, but finds that it does not meet the requirements.
A.9 A red X with a message saying “Invalid username” is placed next to the username textbox and a green check mark is put next to the password textbox.
A.10 User enters a new username.
A.11 Return to step 8 and continue.

Alternate Course B: Username is valid, but taken. Password is fine.
B.8 System checks database of current player usernames and finds an entry with the same set of characters as desired username.
B.9 A red X with a message saying "Username unavailable" is placed next to the username textbox. A green check mark is positioned next to the password textbox.
B.10 User enters new username.
B.11 Return to step 8 and continue.

Alternate Course C: Username is valid and available, but the password does not meet requirements.
C.8 System checks the password for validity and finds that it does not fulfill one or more of the requirements.
C.9 A green check mark is put next to the username textbox and a red X with a message saying "Invalid password" is put next to the password textbox.
C.10 The user changes the password and enters it in the textbox.
C.11 Return to step 8 and continue.

Alternate Course D: Player enters login credentials incorrectly.
D.11 Player misspells or misremembers their username and/or password.
D.12 The system checks the database first for the username and doesn't find it, or finds it and sees that the password does not match.
D.13 The system displays a message saying that something was entered incorrectly.
D.14 Player clicks on link "Forgot username or password?"
D.15 Site loads a new page with a textbox to enter an email address to change the password.
D.16 User enters the email address they entered at registration time.
D.17 The system checks the database for the existence of that email address and finds that it is there.
D.18 The system sends an email to the user's email with a link to reset the password.
D.19 User clicks link in email and is taken to the password reset form page.
D.20 User enters a new password and clicks enter button.
D.21 System checks that the password is not the same as the old one and is valid; updates password entry in database.
D.22 Return to step 10 and continue.

Name: New player goes through tutorial
Identifier: UC2
Description: After successful registration, tutorial mode begins, which has limited functionality and more messages than normal mode.
Preconditions: User has just finished creating a username and password, and has been taken to the home screen with no knowledge of the interface or how the game works.
Postconditions: Home screen is in normal mode, with all functionalities and buttons enabled and no messages. User knows how the game works, where to go if they forget, need help, or want other relevant information.
Basic Course of Action:

1. The use case begins when the player finishes registering their account and the home screen page loads in tutorial mode.
2. The system puts a translucent black screen over the home page and displays a message explaining that the tutorial stuff can be found in the sidebar if they want to skip to the end and to click anywhere on the screen to move the presentation forward. It also shows a highlighted arrow pointing to the right that allows the player to skip the entire tutorial if desired, a second arrow to skip to the next section of the presentation, and a highlighted border around the tutorial link in the sidebar.
3. User clicks anywhere (except the skip-to-the-end arrow).
4. System first shows an arrow pointing to the current stats link/section of the side tab with a message explaining its purpose (to go to a page with more numbers and visualizations, like the player’s win-loss ratio, among other statistics).
5. Player clicks anywhere on the screen to move forward.
6. A back arrow appears in an easily seen place, and is highlighted to show the player that they can return to the previous piece of information at any time in the tutorial.
7. The player clicks anywhere again and the system removes the highlight to continue.
8. System then goes down to the next section/link on the side tab of the main/home page and points an arrow at it with a message explaining that it’s where the player will find the game store to purchase skins, card decks, etc.
9. Player clicks.
10. System finally points an arrow at the last section: support, which is where the player can find contact information and references like a summary of gameplay options. It is also where the contents of the tutorials can be reviewed.
11. Player clicks.
12. Game shows animation of arrows moving to click the “PLAY” button.
13. Here, the user cannot click anywhere, must click button to advance. User clicks the big “PLAY” button.
14. System loads gameboard screen with pre-chosen values for all the variables (number of players, LPs, VPs, number of items given to each player, etc.).
15. Site is covered with translucent black screen again and the tutorial explains that the game is taken in turns that start with a roll of 6 dice and players deal with a number of items that may change the values of different players’ point values. It explains the

existence of The Island and The Ocean and the backstory. Then details the different ways to win (i.e., being the last one with LPs or obtaining 20 VPs).

16. Player clicks when ready and the tutorial goes through the screen, pointing and explaining each piece, including the counters for LPs, VPs, and pearls, the space for favor cards, the area of the screen dedicated to displaying other players' information as well as the queue of turns, the game tracker (transcribes game updates like "user123 used a favor card to do something" or "user123's turn ended"), and the tab that, when clicked, pops out a sidebar with the same options as on the home screen, plus pause and quit options (stats, store, and help).
17. System simulates clicking on the tab in an animation to show the location of the sidebar and how to access it.
18. At the end, the system thanks the player and wishes them a fun game future.
19. The player clicks to exit.
20. The use case ends when the home screen is loaded once again, now in normal mode, with the "PLAY" button ready to be clicked.

Alternate Course A: Player skips tutorial
A.* At any step, the player clicks the "skip all" arrow/button.
A.2 Return to step 20
Alternate Course B: Player skips one or more sections of the tutorial
B* At any step except the final one, the user can skip to the next section of the tutorial by clicking on the "skip section" arrow/button.
B.2 The system loads the next step of the presentation.
B.3 Return to the next non-"Player clicks" step after the original step.

Name: Begin a new game
Identifier: UC3
Description: User starts a new game by pressing the play button, then makes some choices about game options. The system sets up the game board to the initial state and decides on an order of player turns.
Preconditions: User is at the home page and the play button is not yet clicked.
Postconditions: Opponent-type selected (local, online, or hybrid), character chosen, and gameboard initialized to start state: each player gets 10 life points, 0 victory points, and 0 other items (favor cards, pearls, fish, doubloons, and golden dice); everyone is placed in The Ocean and The Island is unoccupied. Player turn order is determined and players arranged on the screen to visually represent that order.
Basic Course of Action:

1. The use case begins when the player clicks the "PLAY" button on the home screen.
2. The site loads the opponent-type selection screen.
3. User picks local players option, but since the user is playing alone, no other players will join and the user selects 5 out of 5 of the others to be computer players. They then click the next button.
4. The system makes note of the number of players to generate.
5. The site loads the character selection screen.
6. The player chooses a character and presses the next button.
7. The system generates the 5 other characters and sets up the game board to the initial state as described in the postconditions of this use case.
8. The system puts translucent black screen over the game board with the words "Ready? Click anywhere to begin the game."
9. The user clicks and the use case ends when the game board is revealed and the game tracker starts off by printing "Game begun."

Name: Display Player In-game Info
Identifier: UC4
Description: Display enhanced view of a player's in-game info.
Preconditions: User clicks on any player's in-game profile/their own.
Postconditions: Player info is visually enhanced to show in-game profile details including number of oysters, pearls, VPs, and favor cards that player owns. (Visual Display)
Basic Course of Action:

1. A player wants to see another player's or their own in-game information.
2. The player clicks on a profile they want to see.
3. Profile info is zoomed in on the display.
4. Player clicks away from the enhanced profile to revert display back to normal.
5. Use case ends.

Name: Display Board/Map
Identifier: UC5
Description: Show the board/map to all players.
Preconditions: Loading screen is on display.
Postconditions: Board/map is visible on display. (Visual Display)
Basic Course of Action:

1. Multiplayer game has started and loading screen is up.
2. While loading screen is up, the map display is set up.
3. Use case ends.

Name: Determine Player Location
Identifier: UC6
Description: The game checks for the player's location.
Preconditions: The Island/The Island Beach/The Island Skull Forest is unoccupied or a fish dice has been rolled.
Postconditions: Determines the player's location is either outside The Island or inside The Island. (System Mechanic)
Basic Course of Action:

1. The player wants to determine player locations before proceeding with either occupying The Island or attacking other players with fish.
2. The game background checks all players' locations.
3. The game determines each player is either inside or outside The Island.
4. All players' locations are used to determine whether the player can enter The Island or which players will receive damage from an attack.
5. Use case ends.

Name: Player Oysters Count
Identifier: UC7
Description: Determine the number of oysters a player has.
Preconditions: A player’s number of oysters is needed to be kept track of for in-game operations.
Postconditions: Gives a numerical value for the amount of oysters a player has. (System Mechanic and Visual Display)
Basic Course of Action:

1. A player that wants to see a profile will need to see that profile’s stats.
2. The game determines the amount of oysters for the in-game profile being accessed for display.
3. The oyster count is displayed along with other in-game stats.

Name: Player Pearls Count
Identifier: UC8
Description: Determine the number of pearls a player has.
Preconditions: A player’s number of pearls is needed to be kept track of for in-game operations.
Postconditions: Gives a numerical value for the amount of pearls a player has. (System Mechanic and Visual Display)
Basic Course of Action:

1. A player that wants to purchase a favor card needs to check if they have enough pearls through their own profile, or wants to see a profile’s stats.
2. The game determines the total amount of pearls the player has for the in-game profile being accessed for display.
3. The pearl count is displayed along with other in-game stats.

Name: Player VPs Count
Identifier: UC9
Description: Determine the number of VPs a player has.
Preconditions: A player’s number of VPs is needed to be kept track of for in-game operations.
Postconditions: Gives a numerical value for the amount of VPs a player has. (System Mechanic and Visual Display)
Basic Course of Action:

1. A player that wants to see a profile will need to see that profile’s stats or the game needs to check how many VPs a player has.
2. The game determines the amount of VPs for the in-game profile being accessed for display or for determining if a victory condition has been met.
3. The VP count is displayed along with other in-game stats.

Name: Rolling Dice
Identifier: UC10
Description: A player will start their turn by rolling 6 dice three times and keeping which dice they want after each roll. Or dice are being rolled to determine an effect or outcome.
Preconditions: A player needs to roll dice.
Postconditions: The dice are rolled for their values and then displayed. (System Mechanic and Visual Display)
Basic Course of Action:

1. The player is rolling dice at the start of their turn or is rolling because of an effect.
2. The player clicks on the roll dice button.
3. The game rolls each of the six dice for the first roll.
4. The player clicks on the dice they want to lock in.
5. The player clicks on the roll dice button. [Alternate Course A]
6. The game rolls the remaining dice for the second roll.
7. The player clicks on the dice they want to lock in and or unlock.
8. The player clicks on the roll dice button. [Alternate Course A]
9. The game rolls the remaining dice for the third roll.
10. The player will move on to resolve the dice they've rolled.
11. Use case ends.

Alternate Course A:

1. The player is satisfied with their roll.
2. The player clicks on the end roll button.
3. Use case ends.

Name: Dice Resolution (Victory Points)
Identifier: UC11
Description: The player wants to resolve dice that give potential victory points, after their roll.
Preconditions: The player has finished rolling dice and has decided which dice to keep.
Postconditions: The player gains the correct amount of victory points according to the roll.
Basic Course of Action:

1. The player has finished rolling their dice.
2. The game starts dice resolution automatically without prompt.
3. The game checks that three-of-a-kind was rolled. [Alternate Course A]
4. The number whose face, on the dice, shows up at least three times for the six dice is added to the player's VP total.
5. Any additional dice of the same number as the first three dice will be treated as one VP and then added to the player's VP total.
6. Use case ends.

Alternate Course A:

1. A three-of-a-kind of a number was not rolled.
2. Use case ends.

Name: Dice Resolution (Oysters)
Identifier: UC12
Description: The player wants to resolve dice that give potential oysters, after their roll.
Preconditions: The player has finished rolling dice and has decided which dice to keep.
Postconditions: The player gains the correct amount of oysters according to the roll.
Basic Course of Action:

1. The player has finished rolling their dice.
2. The game starts dice resolution automatically without prompt.
3. The game checks for the number of oyster dice rolled out of the six dice.
4. The amount of the dice that have oysters on them is then added to the oyster count of the player if they have less than 10 oysters. [Alternate Course A]
5. Use case ends.

Alternate Course A:

1. The player's oyster count goes above ten when added to their oyster count..
2. The game subtracts any extra oysters until oyster count is at ten.
3. Use case ends.

Name: Dice Resolution (Pearls)
Identifier: UC13
Description: The player wants to resolve dice that give potential pearls, after their roll.
Preconditions: The player has finished rolling dice and has decided which dice to keep.
Postconditions: The player gains the correct amount of pearls according to the roll.
Basic Course of Action:

1. The player has finished rolling their dice.
2. The game starts dice resolution automatically without prompt.
3. The game checks for the number of pearl dice rolled out of the six dice.
4. The game adds that number of pearls to the player's pearl count.
5. Use case ends.

Name: Dice Resolution (Fish)
Identifier: UC14
Description: The player wants to resolve dice that have fish, after their roll.
Preconditions: The player has finished rolling dice and has decided which dice to keep.
Postconditions: The player deals damage to other players according to the roll.
Basic Course of Action:

1. The player has finished rolling their dice.
2. The game starts dice resolution automatically without prompt.
3. The game checks for the number of pearl dice rolled out of the six dice.
4. The game checks the player's location and deals damage according to the number of fish rolled.
5. The game takes the oyster count from each of the players damaged, and subtracts it from the number of fish rolled.
6. Use case ends.

Name: Moving into The Island (4 or fewer players)
Identifier: UC15
Description: Sets the location of the player as inside The Island.
Preconditions: The player is outside The Island and The Island is empty.
Postconditions: The player is inside The Island.
Basic Course of Action:

1. The Island is currently empty on the player's turn. [Alternate Course A]
2. The game checks the player's location and determines that the player is outside The Island.
3. The player must move into The Island.
4. The game changes the player's location to inside The Island.
5. The game changes The Island to occupied.
6. Use case ends.

Alternate Course A:

1. The game finds The Island is occupied.
2. Use case ends.

Name: Move out of The Island (4 or fewer players)
Identifier: UC16
Description: Sets the location of the player as outside The Island.
Preconditions: The player is inside The Island and has taken damage from another player's fish roll.
Postconditions: The player is outside The Island.
Basic Course of Action:

1. The game checks the status of The Island's occupation and finds that it is occupied. [Alternate Course A]
2. Another player rolls a fish and damages the player in The Island.
3. The player inside The Island chooses to leave The Island. [Alternate Course B]
4. The game sets the player location as outside The Island.
5. The game sets The Island as unoccupied.
6. Use case ends.

Alternate Course A:

1. The game finds that The Island is unoccupied.
2. Use case ends.

Alternate Course B:

1. The player inside The Island chooses to stay inside The Island.
2. Use case ends.

Name: Move into The Island Beach/The Island Skull Forest (5-6 Players)
Identifier: UC17
Description: Sets the location of the player as inside either The Island Beach or The Island Skull Forest.
Preconditions: The player is outside The Island Beach and The Island Skull Forest, and either one are empty.
Postconditions: The player is inside either The Island Beach or The Island Skull Forest.
Basic Course of Action:

1. Either The Island Beach or The Island Skull Forest's status is unoccupied during that player's turn. [Alternate Course A]
2. The player's location is checked to determined they are outside The Island Beach/The Island Skull Forest.
3. The player must move into either The Island Beach or The Island Skull Forest
4. The game changes the player's location to inside either The Island Beach or The Island Skull Forest.
5. The game changes either The Island Beach or The Island Skull Forest to occupied.
6. Use case ends.

Alternate Course A:

1. The game finds that both The Island Beach and The Island Skull Forest are occupied.
2. Use case ends.

Name: Move out of The Island Beach/The Island Skull Forest (5-6 Players)
Identifier: UC18
Description: Sets the location of the player as outside The Island Beach and The Island Skull Forest.
Preconditions: The player is inside The Island Beach or The Island Skull Forest, and has taken damage from another player's fish roll.
Postconditions: The player is outside The Island Beach and The Island Skull Forest
Basic Course of Action:

1. The game checks the status of The Island's occupation and finds that it is occupied. [Alternate Course A]
2. Another player rolls a fish and damages the player inside The Island.
3. The player inside The Island chooses to leave The Island. [Alternate Course B]
4. The game sets the player location as outside The Island.
5. The game sets location the player was at, either The Island Beach or The Island Skull Forest, as unoccupied.
6. Use case ends.

Alternate Course A:

1. The game finds that The Island is unoccupied.
2. Use case ends.

Alternate Course B:

1. The player inside The Island chooses to stay inside The Island.
2. Use case ends.

Name: Community Favor Card Purchase
Identifier: UC19
Description: The player wants to purchase a favor card from the available face-up community cards.
Preconditions: The player has enough pearls to purchase the favor card they want.
Postconditions: The pearls of the player are exchanged for the favor card they wanted to buy. The card is now associated with the player and is given to them.
Basic Course of Action:

1. The player views the available face-up community cards.
2. The player clicks on which card(s) they want to purchase.
3. The player's pearl count determines that they have enough pearls to purchase the card(s). [Alternate Course A]
4. The card(s) selected and bought are given to the player and are now associated with them.
5. Use case ends.

Alternate Course A:

1. The player doesn't have enough pearls to purchase the favor card(s).
2. The game notifies the player that they do not have enough pearls.
3. Use case ends.

Name: Community Favor Card Reveal/Refresh After Purchase
Identifier: UC20
Description: Reveals the next card(s), up to 3, on the top of the deck.
Preconditions: A community card(s) has been purchased by the player.
Postconditions: A new card(s) will take the place of the card(s) removed from the community card area..
Basic Course of Action:

1. The player purchases a card(s) from the three available face-up community cards.
2. The game checks how many cards are being replaced.
3. The game draws that same number of cards and reveals them in the shop area.
4. Use case ends.

Name: Community Favor Card Reset
Identifier: UC21
Description: Replace all face-up cards in the shop area with 3 new cards from the top of the deck.
Preconditions: The player requests for the community cards available to be reset and replaced with new cards.
Postconditions: New cards will take the place of the cards removed from the community card area..
Basic Course of Action:

1. The player chooses to use their pearls to reset the available face-up community cards.
2. The game sends all face-up community cards to the discard pile.

3. The game draws 3 new cards from the deck.
4. The game places the 3 new cards face-up as community cards.
5. Use case ends.

Name: Favor Card Details/Enlarge
Identifier: UC22
Description: Allows the player a closer inspection of a favor card to view details on the favor card.
Preconditions: The player wants to know something about a favor card and clicks on it.
Postconditions: The favor card is displayed in a larger size on-screen for that player. The player can then click anywhere to zoom back out again.
Basic Course of Action:
1. The player clicks on a card that interests them or that they want to see more details about.
2. The card is enlarged on-screen for the player to view.
3. The player clicks on any part of the screen again to zoom out.
4. Use case ends.

Name: Card Resolution Immediate/End-of-Turn
Identifier: UC23
Description: Resolves the effect of a favor card that must happen immediately or at the end-of-turn.
Preconditions: The player has finished buying favor card(s) and purchased a favor card whose card effect type is immediate/end-of-turn.
Postconditions: The effect of the favor card is implemented by the game and the favor card is sent to the discard pile.
Basic Course of Action:
1. The player purchases a favor card.
2. The player clicks on the favor card to activate its effect and determine card effect order.
3. The game finds the favor card to be of type immediate.
4. The effect of the card is then applied to the game, changing values. [Alternate Course A]
5. Use case ends.

Alternate Course A:
1. If the effect of the card is negated or ineffective due to an effect from another favor card, then the effect is canceled for whichever player controls that opposing favor card.
2. Use case ends.

Name: Card Resolution Continuous
Identifier: UC24
Description: Applies a continuous effect from a favor card to the game until the favor card is discarded.
Preconditions: Resolves and applies the effect of a favor card that is classified as type continuous.
Postconditions: The effect of the favor card is implemented by the game throughout the game, until the favor card is discarded.
Basic Course of Action:

1. The player purchases a favor card.
2. The player clicks on the favor card to activate its effect and determine card effect order.
3. The game finds the favor card to be of type continuous.
4. The effect of the card is then applied to the game, changing values.
5. The game checks for the effect for the rest of the game. [Alternate Course A]
6. Use case ends.

Alternate Course A:

1. The favor card is removed from play by an effect or by the player.
2. The game removes the effect from the list of continuous effects to check.
3. Use case ends.

Name: Card Effect Player Targeting
Identifier: UC25
Description: Determines the target(s) of an effect of a favor card.
Preconditions: A favor card(s) effect is active or activated and requires a target.
Postconditions: The effect of the favor card(s) is applied to the targeted player(s).
Basic Course of Action:

1. A favor card(s) effect is active or activated.
2. The player clicks on the profiles of the players to be selected as targets of the effect. [Alternate Course A]
3. The game applies the favor card(s) effect to the players selected.
4. Use case ends.

Alternate Course A:

1. Favor card(s) that specify made targets do not give the player the option to choose targets.
2. Use case continues at 3.

Name: Determine Next Player Turn
Identifier: UC26
Description: Determine the next player's turn to start.
Preconditions: The last player's is at end-of-turn.
Postconditions: The next player's turn is started and they are given the option to roll.
Basic Course of Action:

1. The last player's turn has ended.
2. The last player has their turn status set to ended.
3. The game checks the turn order.
4. The next player has their turn status set to start.
5. The next player's turn starts and they are given the option to roll.
6. Use case ends.

Name: Shuffle Community Card Deck
Identifier: UC27
Description: Community card deck needs to be reshuffled after discard pile is added back into the deck.
Preconditions: Discard pile is being added back into the deck/all cards in the deck are exhausted.
Postconditions: Deck is shuffled and three new community cards are displayed
Basic Course of Action:

1. Discard pile is added back into the deck, along with the three face-up community cards.
2. The game shuffles the cards.
3. The game reveals three new face-up community cards.
4. Use case ends.

Name: Display Recently Discarded Cards
Identifier: UC28
Description: The last 4 recently discarded cards are on display on the screen to show players what cards were played.
Preconditions: There must be at least 1 discarded card in the discard pile.
Postconditions: The game displays the discarded card(s) in the bottom right section above the player's profile.
Basic Course of Action:

1. The game checks if the discard pile size is greater than zero. [Alternate Course A]
2. The game displays the at most the last 4 card(s) discarded, otherwise displays blank face-down cards to the right of the discarded cards.
3. Use case ends.

Alternate Course A:

1. The discard pile is empty.
2. The game displays four face-down blank cards where the discarded cards would be displayed.
3. Use case ends.

Name: Add Discard Pile to Deck
Identifier: UC29
Description: Adds the discard pile to the deck to shuffle.
Preconditions: Deck is empty.
Postconditions: Discard pile is empty.
Basic Course of Action:

1. The game checks that the deck is empty.
2. The discard pile is then added back to the deck.
3. The game sets the discard pile back to empty status.
4. Use case ends.

Name: Adding/Subtracting Stats (Oyster, Pearl, VP)
Identifier: UC30
Description: Does the calculation behind adding and subtracting stats in the game.
Preconditions: A task that requires calculation is needed.
Postconditions: The calculation is performed and applied to the game.
Basic Course of Action:

1. The game is given which stat is being calculated.
2. The game performs the calculations for that stat. [Alternate Course A][Alternate Course B]
3. The game sets the values for the stat after the calculation.
4. Use case ends.

Alternate Course A:

1. The amount of oysters does not go below 0 or above 10.
2. Use case continues at 3.

Alternate Course B:

1. The amount of VPs or pearls does not go below 0.
2. Use case continues at 3.

Name: Determine Victory
Identifier: UC31
Description: Determines when a player wins and ends the game.
Preconditions: A player’s VPs reaches 20 or all other player’s oyster counts are 0.
Postconditions: Player is shown a victory banner and is given the option to view statistics for the game.
Basic Course of Action:

1. The game checks all other player’s oyster counts. [Alternate Course A][Alternate Course B][Alternate Course C]
2. The game checks that the player has 20 or more VPs. [Alternate Course D]
3. The game displays the victory banner to the player.
4. The game displays a button for the player to click on to view statistics for the game.

5. Use case ends.

Alternate Course A:

1. The game finds at least one other player other than the current player has at least one or more oysters left.
2. Use continues at 2.

Alternate Course B:

1. The game finds all other players have zero oysters left.
2. Use case ends.

Alternate Course C:

1. The game finds that the current player no longer has any oysters left, even with 20 VPs.
2. Use case ends.

Alternate Course D:

1. The game finds that the current player does not have 20 VPs.
2. Use case ends.

Name: Determine Defeat
Identifier: UC32
Description: Determines when a player is defeated in the game.
Preconditions: A player’s oyster count reaches 0 or another player reaches 20 VPs.
Postconditions: Player is shown a defeat banner and is given the option to view statistics for the game.
Basic Course of Action:

1. The game checks that a player’s oyster count is 0 or another player has 20 VPs.
2. The game displays the defeat banner to the player.
3. The game displays a button for the player to click on to view statistics for the game.
4. Use case ends.

Name: Determine Draw
Identifier: UC33
Description: Determines if the game results in a draw.
Preconditions: All players have zero oysters left.
Postconditions: All players are shown the draw banner and are given the option to view statistics for the game.
Basic Course of Action:

1. The game finds all other players have zero oysters left.
2. The game displays the draw banner to all players.
3. The game displays a button for all players to click on to view statistics for the game.
4. Use case ends.

Name: Accessing/Changing Settings
Identifier: UC34
Description: The user wants to access the settings to modify them.
Preconditions: The user clicks on the settings button.
Postconditions: The system changes the settings as the user requests.
Basic Course of Action:

1. The user clicks on the menu button in the lower right hand corner.
2. The user clicks on the settings button which is the first button that pops up from the menu.
3. The user changes the settings as they see fit.
4. The user saves their changes by clicking on the save button or cancel their changes by clicking on the cancel button.
5. The user exits the settings by clicking on the x in the upper right hand corner.
6. Use case ends.

Name: Accessing/Changing Profile
Identifier: UC35
Description: The user wants to access their profile and modify it.
Preconditions: The user clicks on the profile button.
Postconditions: The system changes their profile as the user requests.
Basic Course of Action:

1. The user clicks on the user profile button in the upper right hand corner.
2. The user modifies the settings in the profile as they see fit.
3. The user saves their changes by clicking on the save button or cancel their changes by clicking on the cancel button.
4. The user exists their profile by clicking on the x in the upper right hand corner.
5. Use case ends.