<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: Take a turn
Identifier: UC4
Description: Player makes a sequence of choices about actions that can be taken during a turn. This provides the key mechanism for advancement of the game.
Preconditions: Dice have not been rolled and are indicated as such somehow.
Postconditions: Counters have been updated, damage to other players has been dealt, used favor cards have been discarded, location changes have been represented visually, the game tracker has been updated with the moves made, and the turn queue is updated to reflect who's next.
Basic Course of Action:

1. The use case begins it is the user's turn in the turn queue.
2. The system brings the six dice up, and somehow indicates that none of them have been rolled yet.
3. The user clicks one of the dice to roll them all.
4. The system randomly generates an array of dice outcomes and presents them visually as upturned faces of the dice.
5. The user clicks on each die they would like to remove from the keep pile and reroll. They click that die again to return it to the keep pile. Clicking the roll button rerolls the discarded dice.
6. The system stores the kept dice values and randomly generates new outcomes for the rest.
7. The player and system repeat steps 5 and 6 three times, or as many times as the player has discarded dice, whichever comes first.
8. After the third/final roll, the user chooses how to use the results.
9. The system updates the game board to reflect these moves.
10. The use case ends when the player chooses to end their turn by clicking a button.

Name: Damage Distribution
Identifier: UC5
Description: System determines where players are on the board and distributes damage according to whether player characters are inside or outside the Island zone.
Preconditions: Game state determines player positions on the board. Player has decided to attack other players with fish from dice rolls.
Postconditions: Damage is distributed to players inside or outside the Island.
Basic Course of Action:

1. Player rolls dice to determine their actions. [See UC4]
2. Game determines player wants to use fish dice.
3. Number of fish dice are shown on-screen and a fish splat is displayed on-screen.
4. Game determines the position of the player.
5. System distributes damage to other players' LPs. [Alternate Course A][Alternate Course B]

Alternate Course A:

1. Player position is identified as outside the Island.
2. Damage is distributed to all players inside the Island.

Alternate Course B:

1. Player position is identified as inside the island.
2. Damage is distributed to all players outside the Island.

Name: Purchasing/Playing Favor Card(s)
Identifier: UC6
Description: Allow players to purchase and play favor cards during their turn
Preconditions: The player must have the appropriate amount of pearls to purchase favor cards.
Postconditions: Favor cards effects are applied to the game based on card text.
Basic Course of Action:

1. Player chooses to purchase a community card. [Alternate Course A]
2. Player character data determines the amount of pearls and a player has.
3. Game checks the community card prices. [Alternate Course B]
4. Player purchases a community card.
5. Card purchased has its effect activated if applicable [Alternate Course C][Alternate Course D][Alternate Course E]
6. Community card bank refills to have 3 cards face up.

Alternate Course A:

1. Player clicks on the next phase button.
2. System moves to the next phase of the game.
3. End use case.

Alternate Course B:

1. Player's pearl count is less than all individual card prices.
2. End use case.

Alternate Course C:

1. Card effect activates immediately and effects players' state accordingly.
2. Continue use case.

Alternate Course D:

1. Card effect activates at EoT. [See UC7]
2. Continue use case.

Alternate Course E:

1. Card effect remains continuous throughout game until removed.
2. Continue use case.

Name: Determine End of Turn game state
Identifier: UC7
Description: Determines the game's state after the end of a player's turn.
Preconditions: Player's turn is over for the round.
Postconditions: Game state is determined and ready for the next player.
Basic Course of Action:

1. The EoT game state checks if player has favor cards with EoT effects and executes the card actions. [Alternative Course A]
2. The EoT game state determines which players are on the Island and which players are outside the Island.

Alternative Course A:

1. The EoT game state determines the player does not have favor cards.
2. Continue use case.

Name: Determine Victory
Identifier: UC8
Description:.Determines if a player has achieved victory
Preconditions: A player must have 20 or above VPs or all other players' health is 0
Postconditions: The game is over and the UI displays the end game victory or defeat banners.
Basic Course of Action:

1. System determines a player does have 20 or more VPs [Alternate Course A]
2. System determines that all other players' LPs are 0 [Alternate Course B]
3. Users click the banner for statistics on the game.

Alternate Course A:

1. Use case determines a player does not have 20 or more VPs.
2. Continue use case.

Alternate Course B:

1. Use case determines that at least two or more players have more than 0 LP.
2. Continue use case.