Name: Gabriel Arizmendi Mark _____/50

[**Instructions**: Remove everything that is not a heading below and fill in with your own diagrams, etc.]

1. Brief introduction __/3

My feature for the Justy Fishin' game will be implementing the fish and a portion of their mechanics.

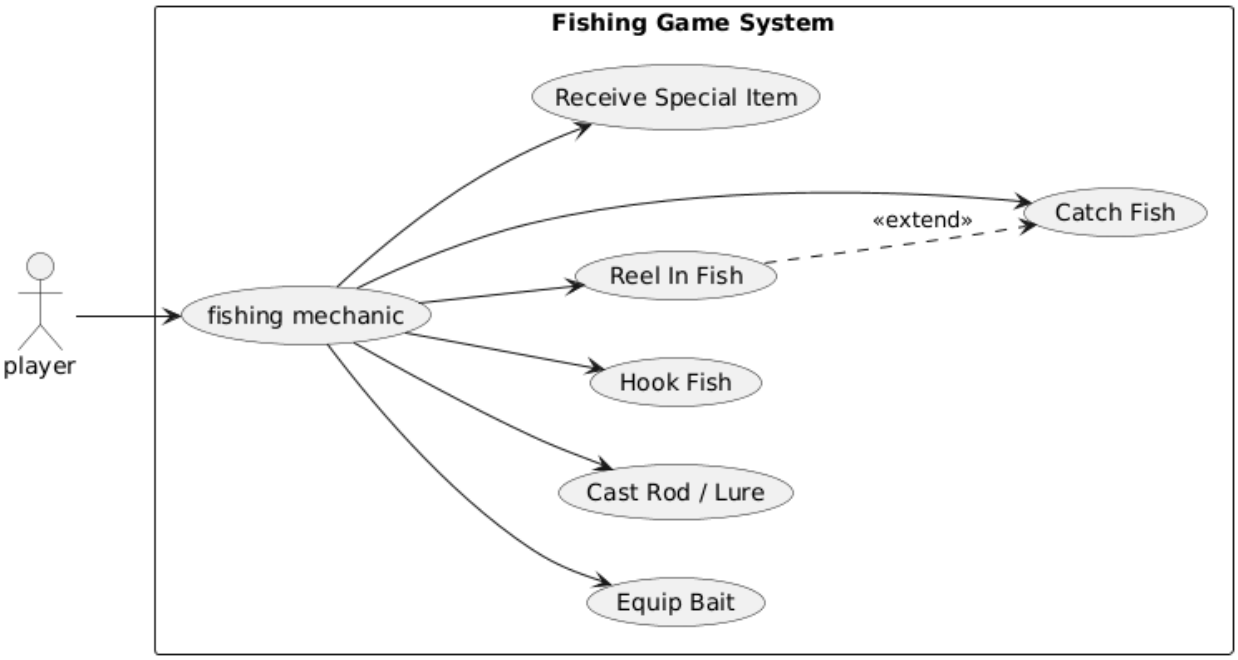
When the boat launches from shore and goes into sea, I need to make sure the player has plenty of fish to hopefully catch. The fish that spawn when the player goes to sea will be randomly generated from a set of fish that will be different every time the player goes out.

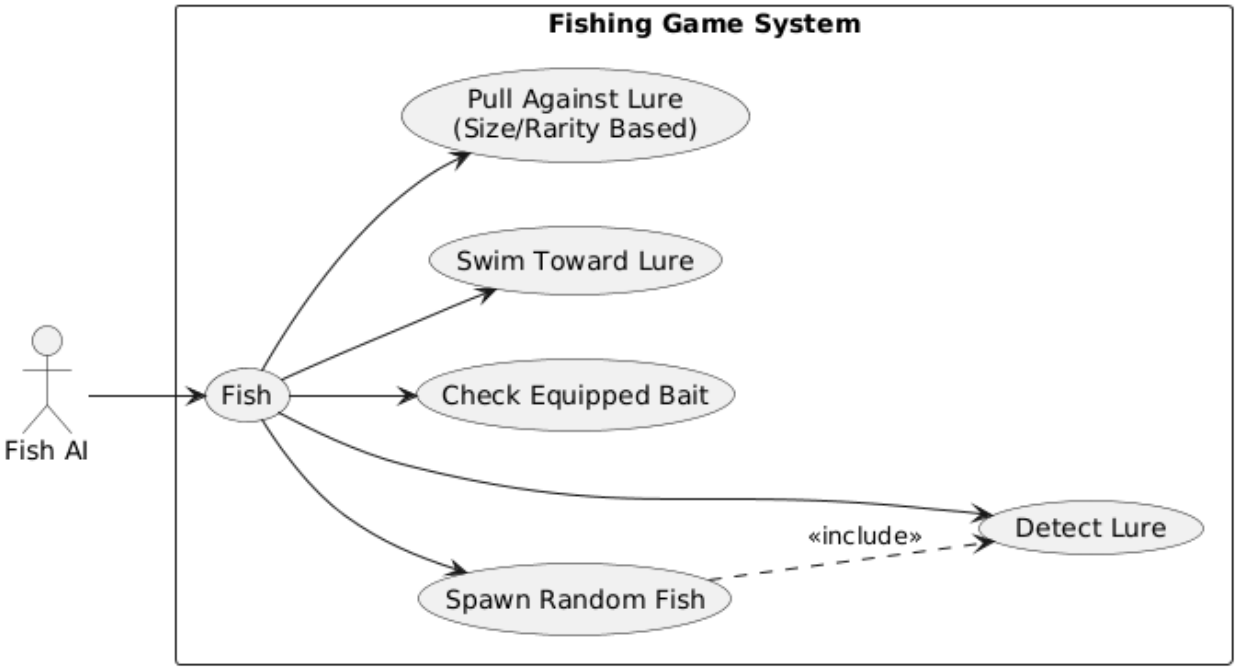
The fish will also need to fight against the player so that the game is not too easy. Each fish will fight differently, with some fighting harder than others. The fish that fight harder may weigh more or they may be a rarer fish that is worth more than a basic fish.

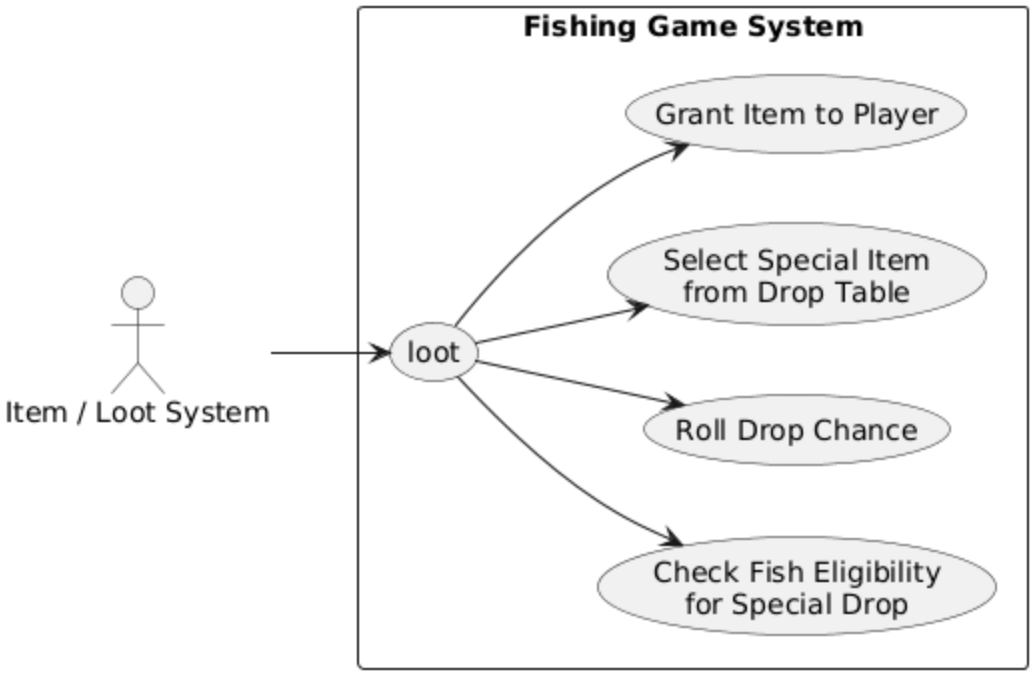
I will also need to make sure that when the player uses bait, that the fish are more likely to be attracted to the user's rod. How attracted the fish are to the lure will most likely depend on the rarity of the bait used.

2. Use case diagram with scenario _14

Use Case Diagrams







Scenarios

Scenario 1: (Player Diagram):

Name: fishing mechanic

Summary: Based on what bait the player has equipped and if they hook a fish, the player can receive different items. They can receive the fish itself if they catch it, and they may or may not get special items from that certain fish.

1: Player equips bait on their rod

2: Player casts lure into water

3: A fish bites the lure and player hooks the fish

4: Player reels the fish into the boat

5: Rewards are given to player, the fish is guaranteed, special items maybe being earned.

Exceptions:

1: The fish doesn't go for the players' lure, therefore no chance to earn rewards

2: The player heads back to shore with nothing to show for their expedition.

Scenario 2 (Fish AI):

Name: Fish

Summary: The spawning of the fish will change daily to avoid repetition, some fish may stay though as pool of fish will not be very big. The behavior of the fish will be determined by a couple of parameters, such as how rare the fish is, how big it is, and lure used by the player.

1: System spawns' fish when player goes to sea.

2: Fish detects lure and calculates interest in it

3: Fish goes to lure and attempts to bite

4: If hooked, the fish fights the player based on size/rarity

Exceptions:

1: System spawns' fish at sea

2: Fish detects lure and calculates interest in it

1: Fish does not go for the lure, as it is not interested in it.

2: Fish swims off and despawns off screen.

Scenario 3(Item/Loot system)
Name: Loot

Summary: When the player catches a fish, the loot system will decide what to award to the player based off several factors of the fish caught. This could potentially be the rarity of the fish, the weight of the fish, and if the fish has a special drop table.

1: Player catches a fish

2: Loot System checks what fish was caught

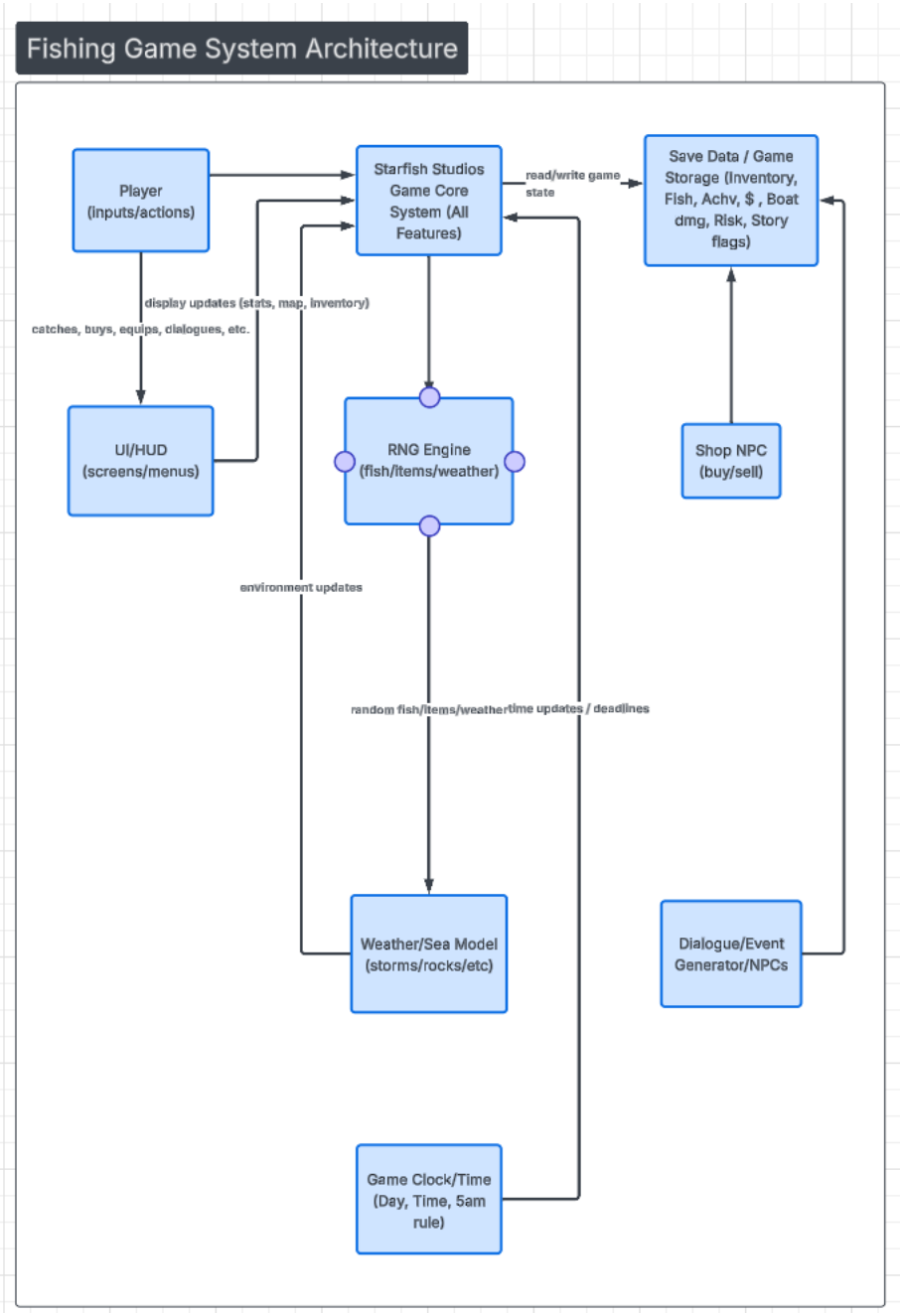
3: If able to, systems system checks drops chances

4: On success, system awards loot to player.

5: Notify player of loot gained and value of loot.

3. Data Flow diagram(s) from Level 0 to process description for your feature _____14

Data Flow Diagrams



Process Descriptions

Gameplay loop(Fish AI and Fishing mechanics):

1: Player casts the lure

2: Fish are selected from random pool with RNG.

3: Fish AI checks

3a: Bait

3b: Lure

3c: attraction

4: If fish is attracted, bite the players lure

5: When bite occurs, player hooks the fish

6: After bite, the fish fights the player by pulling on the players line, the fight depends on-

6a: Rarity of fish

6b: Fish strength

6c: Size of fish

7: After fish is caught or escapes the hook, loop back to 1 to keep the gameplay going.

Fish Loot and special item generation:

1: When fish caught, fish loot will receive 3 attributes from fish:

1a: Weight

2b: Size

1c: Rarity

2: Fish loot checks which loot to give

3: If eligible, fish loot checks for special drop on fish
4: On success, loot is given to the player.

4. Acceptance Tests _____9

Test 1: Fish fighting back test:

Assume a fight model like:

- pullStrength = base * sizeMultiplier * rarityMultiplier
- Player reeling applies opposite force
- When fish pulls back, a tension meter goes uo

AT-B1: Pull Strength Scales Correctly

Purpose: Bigger/rarer fish pull harder.

Inputs

- Fish A: size = min (e.g., 0.5), rarity = Common
- Fish B: size = max (e.g., 5.0), rarity = Legendary
- Same lure/line stats, same reel input pattern

Steps

1. Hook Fish A and record average pull force (or distance gained per second).
2. Hook Fish B and record same metric.

Expected Outputs

- Fish B pull metric > Fish A pull metric (clearly).
- UI values (tension/distance/stamina) update consistently.

Boundary Cases

- **Minimum size + minimum rarity** → pullStrength should be **>= 0** and not negative.
- **Maximum size + maximum rarity** → pullStrength should not exceed safe caps (no physics explosions).
- If you allow “size = 0” → fish should pull **0** (or clamp to min).

Test 2: Special Items from Certain Caught Fish

Assume:

- Only some fish types have a drop table.
- Drop chance may depend on rarity.

Eligible Fish Can Drop Items

Purpose: Catching an eligible fish performs a drop roll and possibly awards an item.

Inputs

- Fish type: “TBD” eligible with drop chance 25%
- RNG seed fixed to produce a known “success” roll (or override roll)
- Inventory has space

Steps

1. Catch eligible fish.
2. Trigger loot generation.

Expected Outputs

- System checks eligibility: True
- Performs roll
- If roll succeeds: item added to inventory, notification shown, saved to player data.

Boundary Cases

- Drop chance = 0% → Output: never drops.
- Drop chance = 100% → Output: always drops.
- Empty drop table → Output: no item awarded, log/config error handled safely.

5. Timeline _____/10

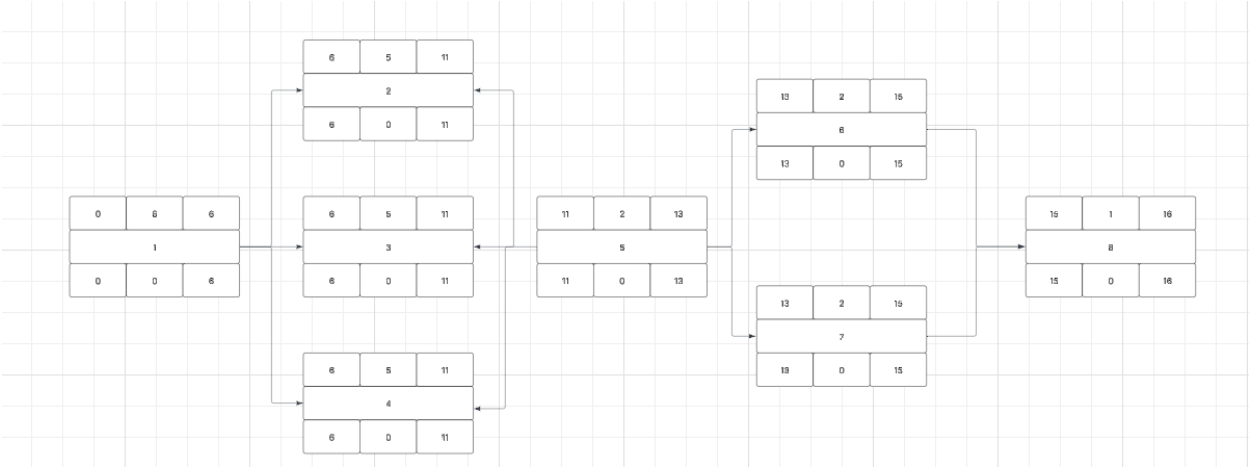
[Figure out the tasks required to complete your feature]

Example:

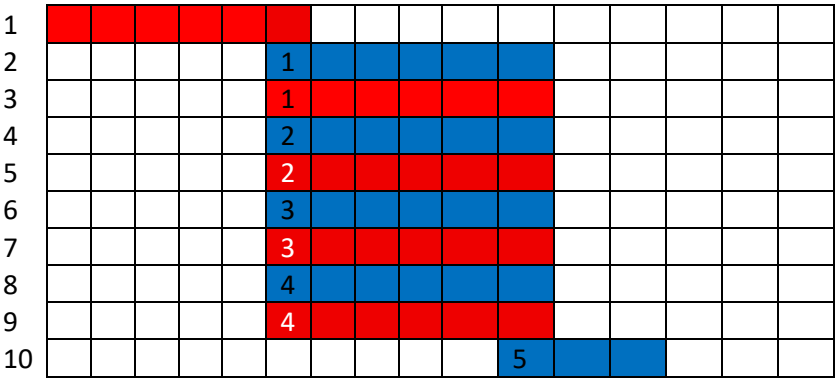
Work items

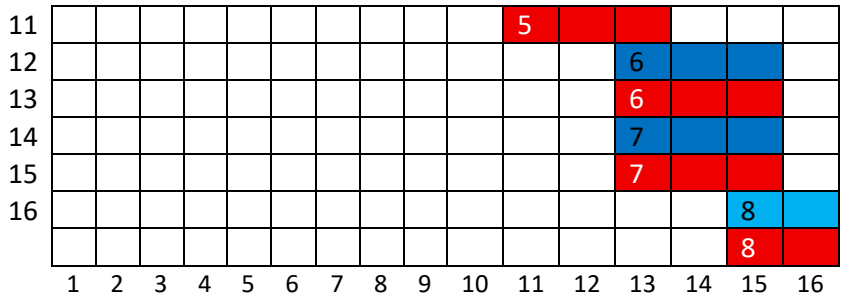
Task	Duration (PWks)	Predecessor Task(s)
1. Requirements Collection	6	-
2. Fish AI programming	5	1
3. Bait/fight programming	5	1
4. Reward programming	5	1
5. test Fish/Bait/Fight/Reward interactions	2	2,3,4
6. Model/Sound Polishing	2	5
7. Final testing/fixing	3	6
8. deployment/Ins	1	6,7

Pert diagram



Gantt timeline




11
12
13
14
15
16
5
6
6
7
7
8
8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16