**DESCRIPTION DU JEU**

**1. Interface du Jeu**

* **Écran Principal :**
  + **Menu de Navigation :** Accueil, Classement, Équipement, Aide, Quitter…
  + **Vue de la Surface :** Représentation graphique de la mer avec différents environnements (plage, port, eaux profondes, etc.).
  + **Barre d’Actions :** Indicateurs de temps de pêche, nombre de poissons attrapés, et valeur totale.
* **Écran de Pêche :**
  + **Vue Divisée :**
    - Partie supérieure pour la surface de l'eau, avec animations d'effets de vagues.
    - Partie inférieure pour le fond marin, montrant différents types de poissons et obstacles (coraux, épaves, etc.). Les poissons effectueront des allers et retour et seront éparpillés dans la couche d’eau.
  + **Animation de Lancer :** Montre le lancer de l'hameçon et sa descente à travers l'eau, avec les poissons visibles.

**2. Animation**

* **Lancer et Ramener (vertical) :**
  + **Animation de Lancer :** Un mouvement fluide lorsque le joueur lance l'hameçon, avec un effet de traînée.
  + **Animation de Descente :** L’hameçon s’enfonce lentement, et la caméra suit son mouvement, permettant aux joueurs de voir les poissons et les obstacles.
  + **Animation de Remontée :** Le joueur pourra contrôler grâce à sa souris le déplacement horizontal de son hamecon.
* **Animation de Capture :** Lorsqu’un poisson mord, une animation dynamique montre la lutte pour le ramener à la surface, avec des mouvements de secousse pour simuler la résistance.

**3. Système de Points et Classement**

* **Collecte de Points :**
  + Chaque type de poisson a une valeur différente (par exemple, un thon pourrait valoir plus qu’une sardine).
  + Bonus pour attraper des espèces rares ou en série (ex: attraper trois poissons identiques à la suite).
* **Classement :**
  + Un système de classement en temps réel pour afficher les meilleurs pêcheurs sur une période donnée (quotidien, hebdomadaire).
  + Possibilité d'ajouter des défis ou des tournois pour inciter à la compétition.

**4. Équipement et Améliorations**

* **Types d’Équipement :**
  + **Canne à Pêche :** Différentes cannes ayant des caractéristiques variées (longueur, flexibilité, rapidité de remontée).
  + **Appâts :** Divers types d'appâts qui attirent différents poissons. Les joueurs peuvent débloquer des appâts spéciaux.
* **Système d'Amélioration :**
  + Les joueurs peuvent utiliser les points gagnés pour améliorer leur équipement (ex: augmenter la force de la canne ou la capacité de l'appât).
  + Éventuellement, un système de crafting pour créer des équipements à partir de matériaux collectés pendant le jeu.

**5. Éléments Additionnels**

* **Événements En Direct :**
  + Organiser des événements temporaires avec des récompenses spéciales (ex: un week-end de pêche à un endroit spécifique où certains poissons sont plus fréquents).
* **Modes de Jeu :**
  + **Multijoueur :** Compétitions en temps réel avec plusieurs joueurs sur la même carte.
  + **Défis Quotidiens :** Missions spécifiques à réaliser pour gagner des récompenses.

**Priorisation des tâches**

------------------------------------------------------------------------------------------------------------

En détails :

**Étape 1 : Mécanique de Lancer et de Remontée (Haute Priorité)**

* **Objectif** : Implémenter la fonctionnalité de lancer l'hameçon et de le ramener.
* **Actions à entreprendre** :
  + Créer une classe ou un module pour gérer le lancer de l'hameçon.
  + Développer l'animation du lancer avec des effets de traînée.
  + Programmer la logique pour la descente de l'hameçon.
  + Ajouter la capacité de contrôler la remontée de l'hameçon via la souris.

**Étape 2 : Types de Poissons et Valeurs Associées (Haute Priorité)**

* **Objectif** : Développer différents types de poissons avec des valeurs variées.
* **Actions à entreprendre** :
  + Créer une base de données ou des structures de données pour les poissons (nom, valeur, rareté).
  + Intégrer des sprites ou modèles 3D pour les différents types de poissons.
  + Programmer la logique de mouvement des poissons dans l'eau.

**Étape 3 : Système de Classement (Haute Priorité)**

* **Objectif** : Mettre en place un système de classement des pêcheurs.
* **Actions à entreprendre** :
  + Concevoir une structure pour stocker les données des joueurs (nom, score).
  + Créer une interface utilisateur pour afficher le classement.
  + Développer la logique pour mettre à jour le classement en temps réel.

**Étape 4 : Modes de Jeu (Multijoueur) (Haute Priorité)**

* **Objectif** : Intégrer un mode multijoueur simple.
* **Actions à entreprendre** :
  + Choisir une solution pour le réseau (par exemple, Unity Networking, Photon).
  + Créer des systèmes de matchmaking et de gestion de sessions.
  + Implémenter la synchronisation des actions des joueurs dans le jeu.

**Étape 5 : Écran de Pêche (Vue Divisée) (Haute Priorité)**

* **Objectif** : Concevoir l'interface graphique pour l'écran de pêche.
* **Actions à entreprendre** :
  + Développer l'interface utilisateur pour la vue divisée (surface et fond marin).
  + Créer les animations d'effets de vagues et de mouvements sous-marins.
  + Intégrer les éléments graphiques pour les poissons et les obstacles.

**Étape 6 : Collecte de Points et Système de Récompenses (Moyenne Priorité)**

* **Objectif** : Implémenter le système de points pour chaque poisson attrapé.
* **Actions à entreprendre** :
  + Créer un système de gestion des points dans le code.
  + Développer des mécanismes pour attribuer des points lors de la capture de poissons.
  + Concevoir une interface pour afficher les points gagnés.

**Étape 7 : Types d’Équipement (Canne à Pêche et Appâts) (Moyenne Priorité)**

* **Objectif** : Intégrer différents types d'équipement pour les joueurs.
* **Actions à entreprendre** :
  + Créer une base de données pour les équipements (caractéristiques, niveaux).
  + Développer une interface utilisateur pour gérer l'équipement.
  + Intégrer les mécaniques d'amélioration d'équipement.

**Étape 8 : Animation de Capture (Moyenne Priorité)**

* **Objectif** : Ajouter l'animation lors de la capture d'un poisson.
* **Actions à entreprendre** :
  + Développer des animations pour la lutte entre le joueur et le poisson.
  + Créer des effets visuels pour rendre l'animation plus immersive.

**Étape 9 : Événements en Direct (Basse Priorité)**

* **Objectif** : Planifier et organiser des événements temporaires.
* **Actions à entreprendre** :
  + Développer un système de gestion des événements dans le jeu.
  + Créer des interfaces pour afficher les événements à venir.

**Étape 10 : Défis Quotidiens (Basse Priorité)**

* **Objectif** : Ajouter des missions spécifiques à réaliser.
* **Actions à entreprendre** :
  + Concevoir un système pour suivre les défis et récompenser les joueurs.
  + Intégrer des notifications dans le jeu pour informer les joueurs des défis.

### Classe FishingRod

Cette classe gère les interactions liées à la canne à pêche et l'hameçon.

#### Attributs

* **public float castDistance** : Distance où l'hameçon est lancé.
* **public float reelSpeed** : Vitesse à laquelle l'hameçon remonte.
* **public bool isCasting** : Indique si l'hameçon est en train d'être lancé.
* **public bool isReeling** : Indique si l'hameçon est en train d'être remonté.
* **public Vector3 castPosition** : Position actuelle de l'hameçon.

#### Méthodes

* **public void Cast()**
  + Description : Lance l'hameçon à une distance. Calcul la position de l’hamecon
  + isCasting vrai + calcul de position de l’hamecon avec CastDistance
* **public void HandleCasting()**

gere la descente de l’hamecon

Si isCasting vrai on diminue la position verticale de l’hamecon de castPosition jusqu’à atteindre le fond de l’eau

* **public void ReelIn(float horizontalInput)**
  + Description : Remonte l'hameçon, avec un contrôle horizontal.
  + Met isReeling vrai
  + Modifie la position de castPosition verticalement et surtout horizontalement avec horizontalInput
  + Vitesse de remontée avec reelSpeed
* **public void StopReeling()**
  + Description : Arrête la remontée de l'hameçon
  + isReeling faux

### Classe Fish

Cette classe représente les poissons que le joueur peut attraper.

#### Attributs

* **public string fishName** : Nom du poisson.
* **public float value** : Valeur en points du poisson.
* **public bool isCaught** : Indique si le poisson a été attrapé.

#### Méthodes

* **public void OnCaught()**
  + Description : Action à réaliser lorsque le poisson est attrapé.
  + Logique :
    - isCaught vrai

### Classe FishingGameManager

Cette classe gère l'état global du jeu de pêche.

#### Attributs

* **private FishingRod fishingRod** : Instance de FishingRod.
* **private List fishList** : Liste de poissons disponibles dans le jeu.
* **private int score** : Score actuel du joueur.

#### Méthodes

* **public void StartFishing()**
  + Description : Initialise le jeu de pêche.
  + Logique :
    - Crée une instance de FishingRod
    - Remplit fishList avec des instances de Fish
* **public void AttemptCatch()**
  + Description : Vérifie si un poisson a été attrapé lorsque l'hameçon est remonté.
  + Logique :
    - Parcourt fishList et vérifie si un poisson est à la position de l'hameçon.
    - Si un poisson est attrapé, appelle onCaught() sur le poisson et met à jour le score.