Sommaire :

* Introduction

- Brève introduction de l'objectif de la présentation 2min sur le sommaire et nos intentions comme si on parlais au clients.

* Planification et collecte des besoins

-RP Client/Archi (bulles dans le ppt, Omar)

* Conception

- Présentation des schémas : Respecter les principles SOLID montrer les diagrammes de classe, cas d'utilisation (Omar & Modibo), décision pour une game (Nawfel), séquence pour une game (Modibo)

 - Explications des choix : ADR (Omar), DDD (Nawfel & Modibo)

* Développer

-Model C4 (uniquement les 3 premiers, pas de code, Omar) Piliers POO respecté

-BDD (Modibo & Nawfel)

* Essais et assurance de la qualité

-Parler du TDD (en fonction de nos choix sans détaillé car no code, Omar, Nawfel, Modibo)

* Déploiement

-Expliquer les méthodes de déploiement et de mise en prod (AWS par exemple, Omar & Modibo)

* Conclusion (Nawfel)

- Récapitulatif des points clés, montrer qu'on BIENNNN appliquer tout le cours au projet (toutes les notions vu en cours)

- Ouverture : parler des prochaines étapes possibles pour le projet, améliorations ou fonctionnalités supplémentaires envisagées.

Introduction :

Ce projet a pour but de créer un jeu vidéo type board game, inspiré du mode Battleground de Hearthstone.

Naturellement, on commence par lire la documentation de Battleground avant de jouer à quelques parties, afin de comprendre son fonctionnement et composants.

Planification et collecte des besoins :

Notre client

Tout d'abord, nous commençons par de simples Use Cases pour appréhender les besoins, c'est-à-dire des chemins types, empruntés par un utilisateur. Par exemple : L'utilisateur s'inscrit, confirme son Email, se connecte et lance une partie. Ou encore, l'utilisateur ouvre la boutique, choisis une Card puis passe au paiement.

Conception :

Le matchmaking dans des jeux de type "battleground" ou "Teamfight Tactics" (TFT) est un processus crucial qui influence grandement l'expérience des joueurs. Ce processus doit être à la fois juste et rapide pour maintenir les joueurs engagés. La gestion du matchmaking et la communication en temps réel (par exemple, avec des WebSockets) sont deux aspects distincts mais interdépendants du développement de jeux multijoueurs. Voici comment ces éléments peuvent fonctionner ensemble :

### Matchmaking

1. \*\*Système de matchmaking\*\* : Le système de matchmaking commence généralement par la collecte de requêtes des joueurs qui cherchent à rejoindre une partie. Ces requêtes incluent souvent des données sur le joueur, comme son niveau de compétence, son rang, ou d'autres statistiques pertinentes.

2. \*\*Algorithme de matchmaking\*\* : L'algorithme de matchmaking analyse les requêtes et regroupe les joueurs en fonction de critères prédéfinis pour créer des parties équilibrées. Ces critères peuvent inclure la compétence, la latence réseau (pour minimiser le lag), la langue, ou d'autres facteurs pertinents pour l'expérience de jeu.

3. \*\*Création de la partie\*\* : Une fois un groupe de joueurs assortis, le système crée une nouvelle instance de partie et assigne les joueurs à cette instance. Cette étape peut également impliquer la sélection d'un serveur de jeu qui offre la meilleure performance pour tous les joueurs du groupe.

### Communication en temps réel avec WebSockets

1. \*\*Établissement de la connexion\*\* : Après la création de la partie, chaque client (jeu du joueur) établit une connexion WebSocket avec le serveur de jeu. Cette connexion reste ouverte pour la durée de la partie, permettant une communication bidirectionnelle en temps réel.

2. \*\*Échange de messages\*\* : Les WebSockets sont utilisés pour échanger des messages entre le client et le serveur, tels que les actions des joueurs, les mouvements, les mises à jour de l'état du jeu, etc. Cela permet au jeu de se dérouler de manière synchronisée pour tous les joueurs.

3. \*\*Gestion des événements en temps réel\*\* : Le serveur utilise la connexion WebSocket pour envoyer des mises à jour en temps réel aux clients, comme les changements dans l'environnement de jeu, les résultats des combats, et les notifications de jeu. Les clients peuvent également envoyer des informations au serveur, comme les décisions tactiques ou les mouvements des joueurs.

### Infrastructure serveur

- \*\*Serveurs de jeu dédiés\*\* : Pour supporter le matchmaking et la communication en temps réel, il est courant d'utiliser des serveurs de jeu dédiés. Ces serveurs hébergent les instances de jeu, gèrent les connexions WebSocket, et exécutent la logique de jeu.

- \*\*Serveurs de matchmaking\*\* : Distincts des serveurs de jeu, les serveurs de matchmaking sont responsables de faire fonctionner l'algorithme de matchmaking et de gérer les files d'attente des joueurs.

### Conclusion

Le matchmaking et la communication en temps réel via WebSockets sont essentiels pour créer une expérience de jeu multijoueur engageante et fluide. Le matchmaking assure que les parties sont équitables et compétitives, tandis que les WebSockets permettent une interaction en temps réel entre les joueurs et le serveur de jeu, essentielle pour les jeux dynamiques comme les battlegrounds et TFT. La mise en place d'une infrastructure solide et d'un algorithme de matchmaking efficace est primordiale pour le succès d'un jeu multijoueur en ligne.

JOURNAL DE BORD :

Donc on discutais d’abord dans quel ordre aborder les choses, qui s’occupe de quoi en essayant d’estimer la longueur et la complexité entre chaque chose. Il est vrai que l’envie de coder est très présente mais on comprend bien.

Notre objectif c'est de répartir le travail et être efficient dans les 3 jours qu'on à pour ce projet ambitieux. On a pris conscience que le temps de développement n'est pas notre priorité mais plutôt préparer le terrain pour une équipe de développment en mettant en place une documentation explicative avec des notions forte de Carftmanship. Donc avec un BDD et un DDD.

07/02 - 16h30

Nous avons fait les diagramme de cas d'utilisation, diagramme de décision, diagramme de classe.

Nous nous sommes mis d'accord sur le Gameplay du jeu et composants métier grâce à la démo de Hamza.

Cet après midi Nous avons commencé le Power Point et réfléchi sur les solutions à apporter pour répondre aux enjeux de notre projet tel que la mise en ligne internationnal via des serveurs par région, le module i18n, et les Web Socket.

Le faire en microservice est envisagé

Le matchmaking sera basé sur la cote des joueurs et leur région. Nous réflechissons à comment incoporer le Web Socket à la recherche de joueurs.

Diagramme de séquence en cours...

07/02 - 20h00

Tâches réparties entre l'ADR, le modèle C4 et BDD et DDD.

Réflexion sur le modèle de BDD et language ubiquitaire.

08/02 - 14h

Modibo avance sur le diagramme séquence.

Omar termine le diaporama

Nawfel avance sur le DDD et BDD

08/02 - 17h35

Diagramme séquence et de classe faites

BDD et DDD

Diapositive partagé pour préparation oral et vérification.

VBA mise pour l'avancé visuelle de la présentation

Présentation à faire demain à 11h.

09/02 - 10h30

Finalisation du power point et préparation finale

DDD :

**Language Ubiquitaire :**

**Vocabulaire du jeu**

**Classes de Base**:

* **Card**: Classe de base pour tous les éléments jouables. Elle peut inclure des attributs communs comme le nom de la carte et le coût en mana pour l'utiliser.
* **Minion** et **Spell**: Sous-classes de Card. Minions a des attributs comme les points d'attaque, les points de vie, un attribut Race, des effets etc. Sort, d'autre part, possède surtout un effet.
* **Effect**: Les effets peuvent être traités via le polymorphisme, en définissant une interface Effect avec une méthode Trigger() et Power() qui décrivent respectivement la condition de déclenchement et le pouvoir de l'effet lui-même.

**Application des Principes SOLID**

1. **Single Responsibility Principle :** Chaque classe a une responsabilité unique. Card gère les attributs de base, Minions ajoute des spécificités des créatures comme les points de vie et d'attaque, etc.
2. **Open/Closed Principle :** Le système permet l'implementation facile de nouvelles extention de cartes, races, ou effets sans avoir à modifier celles existantes.
3. **Liskov Substitution Principle :** Les sous-classes de Card peuvent être utilisées à la place de Card sans que cela ne casse le comportement du programme. Par exemple, là où le jeu attend une carte, un Minion ou un Spell peut être utilisé.
4. **Interface Segregation Principle :** Les interfaces sont spécifiques et non surchargées.
5. **Dependency Inversion Principle :** Les détails d'implémentation, comme les effets des sorts, dépendent des abstractions (comme l'interface Effet) plutôt que l'inverse. Cela facilite l'ajout de nouveaux effets sans perturber les classes existantes.

**Illustration avec l'Exemple Donné**

Prenons l'exemple d'une carte serviteur spécifique, "Tueur de Dragons", qui a un effet déclenché à son arrivée sur le terrain :

* **Classe Minion**: Hérite de Card et a des attributs pour les points d'attaque et de vie. Elle possède un attribut Effect pour gérer son effet spécial.
* **Attribut Race**: Peut être utilisé pour identifier les serviteurs comme dragons.
* **Implémentation de l'Effet**: L'effet de "Tueur de Dragons" est une classe qui implémente Effet, avec une méthode Trigger() qui parcourt toutes les cartes sur le terrain et applique l'effet spécifique (par exemple, ajouter un point d'attaque).

**Alliés :** ceux de votre côté du terrain.

**Ennemis :** ceux du côté adverse du terrain.

**Adjacent / à côté de :** les serviteurs à droite et à gauche d'un serviteur.

**Trigger:**

Débarquement : L'effet qui suit se déclenche uniquement lorsque vous posez le serviteur sur le plateau.

Héritage éternelle : L'effet qui suit se déclenche uniquement lorsque le serviteur décède.

Renaissance : L'effet qui suit se déclenche uniquement lorsque le serviteur revient sur le terrain la deuxième fois.

Vengeance : L'effet qui suit se déclenche uniquement lorsque le serviteur est blessé, c'est-à-dire moins que son max health.

Riposte : L'effet qui suit se déclenche à chaque fois que le serviteur est frappé ou ciblé par un sort.

Synchronie : L'effet qui suit se déclenche à chaque utilisation de votre capacité de Héros.

Enchaînement:  L'effet qui suit se déclenche uniquement lorsque une autre carte fut invoqué avant dans le même tour.

**Power :**

Assaut doublé : Attaque deux fois de suite.

Impétueux : Attaque dès qu'il a été invoqué.

Belliqueux : Devient l'unique cible pour les adversaires.

Mirage éphémère : Annule une instance de dégât puis disparait.

Invisible : Devient entièrement inciblable.

Muet : Rend inefficace tout les mots-clés ou tout effets du serviteur.

Stase Glaciale : L'attaque du personnage est réduit à 0 jusqu'à la fin du tour. Peut toujours recevoir un buff.

Invocation : fait apparaître 3 cartes aléatoires devant le joueur qui doit en choisir une.

+X Arcane: Ajoute un bonus de X dégâts à vos sorts.

+X Force: Ajoute un bonus de X dégâts à vos serviteurs.

Siphon de mana : Récupère X cristaux de mana à l'ennemi.

Vampirisme : Récupère autant de points de vie que de dégâts infligé. Quand il frappe le Héros adverse, soigne notre Héros à la place.

Rage sanguinaire : Si il tue un serviteur, il frappe également le héros ennemi directement aussi.

Aura de bravoure : Donne +1 / +1 aux serviteurs adjacents

Eau de jouvence : Récupère tout son max health si il survit au tour.

Cachot : Exclu un serviteur ennemi du jeu tant que le serviteur est vivant.

Force vitale : Attaque toujours avec son nombre de health uniquement.

Renforcé : Résiste de 1 à toutes instance de dégâts reçu.

Murmure : Infuse ses mot-clés à tous ses alliées sur le terrain.

**Races :**

**Identification du Domaine Métier :**

**Modélisation du domaine :**

**Entités** :​

Représentent les objets du domaine qui ont une identité distincte et une existence indépendante.

X (avec des attributs comme ID, nom, prix, description)​

**Valeurs d'Objet** :​ Concept de domaine qui n'a pas d'identité distincte en dehors de ses attributs.

**Language Ubiquitaire** :​

**Agrégats** :​ Un un groupe d'objets qui sont traités comme une seule unité. Utilisé pour encapsuler des données et garantir leur cohérence et leur intégrité.

**Repositories** : Les repositories sont utilisés pour abstraire l'accès aux données et fournir une interface standardisée pour interagir avec la source de données sous-jacente, qu'il s'agisse d'une base de données, d'un service web, ou autre.

**Services de Domaine** : Encapsule la logique métier d'une application et fournit une interface pour interagir avec cette logique.

**Contextes Bornés :​**

**Gestion des Produits** : Gère l'information et la disponibilité des produits.​

**Traitement des Commandes** : Gère la création et le suivi des commandes des clients.​

**Services de Domaine :​**

Un service pour gérer les opérations du panier, comme ajouter ou retirer des produits.​

Un service pour traiter les commandes, incluant la validation, le paiement et la logistique.​

BDD :

**1: Inscription d'un nouvel utilisateur**

Fonctionnalité: Inscription des utilisateurs

 En tant que nouvel utilisateur

 Je veux créer un compte

 Afin de pouvoir jouer au HearthStone Battleground

 Scénario: Inscription réussie

   Étant donné que je suis sur la page d'inscription

   Quand je saisis un nom d'utilisateur unique et un mot de passe valide

   Et que je soumets le formulaire d'inscription

   Alors je reçois une confirmation d'inscription

   Et je suis redirigé vers la page de connexion

**2: Matchmaking Basé sur la Cote**

Fonctionnalité: Matchmaking basé sur la cote

 Afin d'assurer des parties équilibrées

 En tant que joueur

 Je veux être matché avec un adversaire de cote similaire

 Scénario: Trouver un adversaire de cote similaire

   Étant donné que je suis un joueur avec une cote de 1500

   Quand je lance une partie

   Alors le système me matche avec un adversaire ayant une cote comprise entre 1450 et 1550

  Et la partie commence

**3: Sélection de Langue et de Région**

Fonctionnalité: Sélection de langue et de région pour l'internationalisation

 Afin de fournir une expérience personnalisée aux joueurs du monde entier

 En tant que joueur

 Je veux sélectionner ma langue et ma région préférées dans les paramètres du jeu

 Scénario: Changer la langue et la région du jeu

   Étant donné que je suis sur l'écran des paramètres du jeu

   Quand je sélectionne "Français" comme langue et "Europe" comme région

   Alors le jeu affiche le contenu en français et connecte aux serveurs européens

**4: Connexion et Reconnexion au Match**

Fonctionnalité: Connexion et reconnexion au match

 Afin de minimiser l'impact d'une déconnexion accidentelle

 En tant que joueur

 Je veux pouvoir me reconnecter à mon match en cours

 Scénario: Reconnexion à un match en cours après une déconnexion

   Étant donné que j'ai été déconnecté d'un match en cours

   Quand je relance le jeu et sélectionne "Rejoindre le match en cours"

   Alors je suis reconnecté au match que j'avais quitté

**5: Acheter/Revendre un article**

Fonctionnalité: Acheter/Vendre des articles dans le shop

 Afin d'obtenir de nouveaux objets/récupérer des pièces

 En tant que joueur

 Je veux acheter/revendre des articles quand le shop est ouvert

 Scénario: Acheter/Vendre un article

   Étant donné que le shop est ouvert

   Quand je choisis d'acheter un article

   Alors l'article est ajouté/retiré à mon inventaire

   Et le montant correspondant est déduit/ajouter de mes crédits

**6: Reroll le shop**

Fonctionnalité: Reroll le shop

 Afin de voir de nouveaux articles disponibles à l'achat

 En tant que joueur

 Je veux utiliser l'option "Reroll" pour rafraîchir les articles disponibles

 Scénario: Reroll le shop

   Étant donné que le shop est ouvert

   Quand je choisis l'option "Reroll"

   Alors les articles disponibles sont rafraîchis

   Et je peux voir de nouveaux articles à acheter

**7: Gagner des points de cote après un match**

Fonctionnalité: Système de cote

 En tant que joueur

 Je veux gagner des points de cote après avoir terminé un match dans les 4 premiers

 Afin d'améliorer ma cote globale

 Scénario: Terminer un match dans les 4 premiers

   Étant donné que j'ai terminé un match PvP

   Et que ma position est dans les 4 premiers

   Quand le match se termine

   Alors je gagne un nombre de points de cote proportionnel à ma position finale

   Et ma cote globale est mise à jour en conséquence

**8: Acheter un item dans la boutique in-game**

Fonctionnalité: Acheter des items dans la boutique in-game

 En tant que joueur

 Je veux acheter des items dans la boutique in-game

 Afin de personnaliser et montrer mes cosmétique

 Scénario: Acheter un skin de héros

   Étant donné que je suis dans la boutique in-game

   Et que j'ai suffisamment de monnaie virtuelle

   Quand je sélectionne un skin de héros et confirme l'achat

   Alors le montant correspondant est déduit de ma monnaie virtuelle

   Et le skin de héros est ajouté à mon inventaire

**9: Utilisation d'une emote pendant un match**

Fonctionnalité: Utiliser des emotes pendant un match

 En tant que joueur

 Je veux utiliser des emotes pendant un match

 Afin de féliciter ou de "flame" mon adversaire

 Scénario: Envoyer une emote "Bien joué"

   Étant donné que je suis en plein match PvP

   Quand je sélectionne l'emote "Bien joué" et l'envoie à mon adversaire

   Alors mon adversaire reçoit l'emote "Bien joué"

   Et peut voir l'emote s'afficher sur son écran

**10: Afficher les statistiques du joueur**

Fonctionnalité: Statistiques du joueur

 En tant que joueur

 Je veux voir mes statistiques de jeu

 Afin d'analyser mes performances et m'améliorer

 Scénario: Accéder à l'inventaire statistique

   Étant donné que je suis connecté à mon compte

   Quand je navigue vers la section "Mes Statistiques"

   Alors je vois mes statistiques de jeu affichées

   Incluant le taux de victoire, le héros le plus utilisé, et le nombre total de matchs joués

Contexte

Mise en Situation

Attendus

PLANIFICATION ET COLLECTE DES DONNEES

### ****Rôle Play Client/Archi:****

Client (C): Bonjour, je souhaite développer un jeu inspiré de Hearthstone Battlegrounds. Je vise à intégrer le PvP en ligne, des textures et sons captivants, une boutique de skins, un Battle Pass de saison, et plus encore. Pouvez-vous m'aider à structurer ce projet ?

Architecte Logiciel (A): Bonjour ! C'est un projet ambitieux et passionnant. Commençons par décomposer votre vision pour s'assurer que nous couvrons tous les aspects importants.

C: Le PvP en ligne est essentiel. Je veux que les joueurs du monde entier puissent s'affronter dans des matchs en auto-battle rythmé par des achats de minions. Un système de rareté, et d'évolution des qu'on obtient 3 cartes pareil.

A: Compris. Pour le PvP en ligne, nous allons concevoir une architecture réseau robuste pour supporter des matchs fluides et stables avec peu de latence. Je suggère l'utilisation de WebSocket pour une communication en temps réel entre client et serveur, ce qui est crucial pour les jeux PvP, en incluant toutes ces fonctionnalités.

C: Compris. Et concernant les textures et les sons, comment pouvons-nous les gérer efficacement ?

A: Nous pouvons intégrer un moteur de jeu comme Unity, qui offre un bon support pour la gestion des ressources multimédias et facilite l'ajout de visuels et d'effets sonores de haute qualité.

C: J'insiste aussi sur la nécessité d'une boutique de skins, pour permettre aux joueurs de personnaliser leurs héros et l'équivalent du Bartender.

A: La boutique proposera une variété de skins, d'emotes et de visuels pour les attaques spéciales. Nous veillerons à ce que ces éléments soient purement cosmétiques, pour ne pas affecter l'équilibre du jeu.

C: Il ne faut pas oublier le Battle Pass de saison qui offrira des récompenses uniques.

A: Le Battle Pass intégrera un système de progression avec des récompenses à différents niveaux, encourageant les joueurs à rester engagés tout au long de la saison. Les récompenses incluront des éléments exclusifs comme des skins pour les héros, des emotes, et bien plus.

C: Et pour suivre les progrès, un inventaire, et des statistiques serait apprécié.

A: Nous développerons un système d'inventaire d'un coté et statistiques de l'autres permettant aux joueurs de suivre leurs performances, avec des détails sur les victoires, les stratégies les plus efficaces, et plus. Cela encouragera l'amélioration personnelle et la stratégie.

C: Comment allons-nous assurer que tout cela s'intègre harmonieusement dans le jeu ?

A: Nous adopterons une approche modulaire dans le développement, permettant une intégration fluide de nouvelles fonctionnalités et facilitant les mises à jour. Un cycle de feedbacks.

C: Très bien. J'insiste aussi sur l'importance d'une boutique de skins et d'un Battle Pass de saison. Comment les intégrerons-nous ?

A: Nous pouvons développer un système de microtransactions intégré pour la boutique de skins, en utilisant une base de données sécurisée pour gérer les transactions et les inventaires des utilisateurs. Pour le Battle Pass, un système de progression basé sur l'expérience permettra aux joueurs de débloquer des récompenses au fur et à mesure de leur avancement.

C: Pour approfondir, comment mettrons-nous en œuvre le système de cote, où les 4 premiers joueurs d'une partie montent en cote, et le fonctionnement de la boutique in-game, spécifiquement avec le Bartender ?

A: Commençons par le système de cote. Nous utiliserons un algorithme Elo ou un système similaire pour ajuster les cotes des joueurs après chaque partie. À la fin d'une partie, le système calculera les changements de cote basés sur les positions finales des joueurs, en assurant que les 4 premiers gagnent des points de cote tandis que les derniers en perdent. Après chaque partie, notre serveur de jeu enregistrera les résultats et mettra à jour les cotes dans la base de données des joueurs. Nous programmerons une fonction de calcul de cote qui prend en compte la position du joueur à la fin de la partie et sa cote actuelle, en comparaison avec celles des autres joueurs dans la partie.

C: Comment contez vous traiter les différentes transactions dans le jeu ?

A: Pour la boutique in-game et le rôle du bartender, nous envisageons de créer une interface utilisateur dans le jeu qui permet aux joueurs d'interagir avec le bartender pour acheter des minions avec la monnaie du jeu en commençant avec 3 pièces comme du mana dans d'autres jeu et qui augmente a chaque round.

C Et les achats avec de la monnaie réelle et dans la partie ?

A: Et une boutique en dehors avec des skins, des emotes ou des animations d'attaques. Les transactions seront gérées par un système de gestion des achats sécurisé, qui utiliseront la monnaie réelle, ont envisagera une interface dédié à la transaction avec plusieurs moyens de paiement.

C: Et pour la gestion des transactions et des inventaires ?

A: Pour l'inventaire, chaque joueur aura un inventaire virtuel stocké côté serveur, où tous les articles achetés et les récompenses sont enregistrés. Lorsqu'un joueur achète un article chez le bartender, l'article est immédiatement ajouté à son inventaire.

C: Comment assurerons-nous la sécurité des transactions ?

A: La sécurité des transactions sera assurée par l'utilisation de protocoles sécurisés pour la communication entre le client et le serveur, et par le stockage sécurisé des données de transaction. Nous implémenterons également des vérifications côté serveur pour chaque transaction pour prévenir la fraude et les abus.