Cahier des Charges – Développement d'un Jeu Stratégique en Java ISIL 25/26

1. Contexte du projet

Le projet consiste à développer un **jeu vidéo stratégique** basé sur une gestion de ressources, de territoires et de combats. Le jeu sera réalisé en **langage Java**, en utilisant la Programmation Orientée Objet (POO), les **interfaces**, les **classes abstraites**, les **collections**, ainsi qu'une architecture logicielle propre et extensible.

---

2. Objectifs du projet

- Concevoir un jeu stratégique jouable sur PC.
- Appliquer les concepts POO (héritage, polymorphisme, encapsulation).
- Mettre en place une architecture modulaire permettant d'ajouter des unités, cartes ou niveaux facilement.

---

3. Description générale du jeu

3.1 Concept

Le joueur contrôle une faction sur une carte divisée en cases.
Il doit :

- collecter des ressources,
- construire des bâtiments,
- entraîner des unités,
- étendre son territoire,

3.2 Type de jeu

- Jeu de stratégie au tour par tour **(ou en temps réel selon choix du groupe)**.
- Mode solo contre machine.

---

4. Fonctionnalités attendues

4.1 Carte de jeu

- Générée statiquement ou procéduralement.
- Composée de cases (grass, water, mountain…).

- Chaque case possède :
    - type,
    - accessibilité,
    - bonus/malus.

4.2 Ressources

- Types : Or, Bois, Pierre, Nourriture (modifiable).
- Gestion en temps réel ou par tour.
- Utilisation des Collections Java (Map<String, Integer>).

4.3 Bâtiments

- Trois types minimum :
    - Centre de commandement
    - Camp d'entraînement
    - Mine / Ferme / Scierie
- Chaque bâtiment aura :
    - un coût,
    - un temps de construction,
    - une fonction (production de ressources, création d'unités).

4.4 Unités

- Unité abstraite : **Classe abstraite `Unit`**.
- Dérivées : Soldat, Archer, Cavalier, etc.
- Caractéristiques :
    - Points de vie,
    - Attaque,
    - Défense,
    - Portée,
    - Coût.

4.5 Système de combat

- Déplacements sur la carte.
- Attaques selon règles définies.
- Résolution du combat :
    - formule de dégâts,
    - prise en compte d'aléatoire (Random),
    - mort et retrait des unités.

4.6 Interface utilisateur

- Menu principal : Nouvelle partie / Charger / Quitter.
- Carte affichée.
- État du joueur : ressources, unités, bâtiments.
- Notifications d'événements.

---

5. Exigences techniques

5.1 Langages et technologies

- Java 11+
- Outils : IntelliJ / Eclipse / VS Code
- Collections : List, Map, HashMap, ArrayList

6. Contraintes

- Respect des principes SOLID
- Code organisé en packages
- Documentation

---

7. Livrables

1. Cahier des charges
2. Code source complet
3. Manuel utilisateur
4. Rapport technique détaillé
5. Présentation finale (slides)

---

8. Critères d’évaluation / Validation

- Fonctionnalités respectées
- Architecture propre et extensible
- Performance du jeu
- Bonne organisation POO
- Qualité du code