RAPPORT DE PROJET

Simulation Interactive de Gestion d'Entrepôt

---

1. Introduction

La gestion des stocks est un enjeu central pour les entreprises logistiques, commerciales et industrielles. Une mauvaise gestion peut entraîner des ruptures de stock, des surcoûts ou une insatisfaction client.

Dans ce contexte, ce projet vise à concevoir une **simulation interactive de gestion d'entrepôt**, permettant de visualiser, analyser et corriger l'état des stocks en temps réel à travers un environnement 3D immersif.

Le projet a été développé sous **Unity 3D** en utilisant le langage **C#**, avec une approche modulaire et orientée métier.

---

2. Objectifs du projet

Objectif général

Développer une application interactive simulant la gestion d'un entrepôt avec des produits, leurs états de stock et des actions de correction.

Objectifs spécifiques

- Charger dynamiquement des produits depuis un fichier de données
- Visualiser les articles sous forme d'objets 3D
- Gérer les niveaux de stock (OK, Warning, Error)
- Permettre à l'utilisateur d'interagir avec les produits
- Fournir un état global de stabilité de l'entrepôt
- Mettre en place une architecture claire et extensible

---

3. Technologies utilisées

Technologie	Rôle
Unity 3D	Moteur de jeu et environnement 3D
C#	Langage de programmation
JSON	Stockage et chargement des données produits
TextMeshPro	Interface utilisateur (UI)
Raycasting	Sélection d’objets en 3D
Character Controller	Déplacement FPS

---

4. Architecture globale du système

Le projet repose sur une **séparation claire des responsabilités** :

- **Données** : fichier JSON
- **Logique métier** : scripts C#
- **Interface utilisateur** : UI Unity

Flux de fonctionnement :

1. Chargement des données depuis un fichier JSON
2. Application des données aux objets 3D de la scène
3. Interaction utilisateur (sélection, actions)
4. Mise à jour visuelle et logique en temps réel
5. Évaluation globale de l’état de l’entrepôt

---

5. Gestion des données (JSON)

Les produits sont définis dans un fichier `warehouse_data.json`.

Chaque produit contient :

- un identifiant unique
- un nom
- une quantité
- un seuil d’alerte

Exemple :

```
{
  "items": [
    { "id": "Item_Food_01", "name": "Riz", "quantity": 5,
"warningThreshold": 10 }
  ]
```

}

Ce choix permet :

- une modification simple des stocks
- une extensibilité facile
- une séparation claire entre données et logique

---

6. Description des scripts principaux

6.1 DataManager

- Charge le fichier JSON
- Associe les données aux objets de la scène
- Synchronise les quantités et seuils

6.2 WarehouseItem

- Représente un produit de l'entrepôt
- Gère :
  - quantité
  - seuil
  - statut
- Change la couleur de l'objet selon l'état du stock

6.3 StockStatus

Enum définissant trois états :

- OK
- Warning
- Error

---

7. Interaction utilisateur

Déplacement

- L'utilisateur se déplace librement dans l'entrepôt en vue FPS
- Contrôle clavier + souris

Sélection d'un produit

- Sélection par clic gauche
- Utilisation du raycasting
- Effets visuels :

- surélévation de l’objet
- activation d’un contour (outline)
- affichage des informations dans l’UI

---

8. Actions métier disponibles

Restock

- Ajout de stock sur un produit sélectionné
- Déclenché par la touche **R**

Fix

- Correction d’un produit en rupture de stock
- Déclenché par la touche **F**

Un feedback visuel est affiché à chaque action pour informer l’utilisateur.

---

9. État global de l’entrepôt

Un gestionnaire global analyse en temps réel :

- le nombre de produits en état OK
- Warning
- Error

Deux états globaux possibles :

- **ENTREPÔT STABILISÉ** : tous les produits sont OK
- **ACTION REQUISE** : au moins un produit est en Warning ou Error

Cela permet de simuler une logique de supervision globale.

---

10. Rendu visuel

- Environnement 3D d’entrepôt
- Rayonnages et articles visibles
- Code couleur intuitif
- UI claire et lisible

Le rendu visuel renforce la compréhension de l’état des stocks et améliore l’expérience utilisateur.

---

11. Résultats obtenus

- Application fonctionnelle et stable
- Interaction fluide avec les produits
- Mise à jour en temps réel des données
- Architecture modulaire et évolutive

Le projet répond pleinement aux objectifs fixés.

---

12. Limites du projet

- Absence de persistance des données (pas de sauvegarde)
- Commandes clients non implémentées
- Intelligence artificielle non intégrée
- Assets visuels simples (version gratuite)

---

13. Perspectives d’amélioration

- Ajout d’un système de commandes clients
- Sauvegarde des données (base de données)
- IA de prédiction de rupture de stock
- Amélioration graphique avancée
- Version multi-utilisateur

---

14. Conclusion

Ce projet a permis de concevoir une simulation réaliste et interactive de gestion d’entrepôt, en combinant logique métier, visualisation 3D et interaction utilisateur.

Il constitue une base solide pour des extensions futures dans les domaines de la logistique intelligente, de l’optimisation des stocks et de l’intelligence artificielle appliquée.