ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR
Membre de HONORIS UNITED UNIVERSITIES

HONORIS UNITED UNIVERSITIES

Rapport de projet

3ème année

Ingénierie Informatique et Réseaux

GESTION DE MAGASIN

Etudiant : EL QOUR Soufiane

Classe : 3IIR3

Encadré par : Dr. Mariame Amine

Année Universitaire 2023 -2024

Introduction :

Le projet de gestion de magasin en C++ vise à concevoir un système informatique complet pour la gestion des produits dans un magasin. En utilisant les concepts d'héritage et de polymorphisme, cette solution offre une approche flexible et efficace pour gérer la diversité des produits, tout en offrant des fonctionnalités avancées de recherche et d'affichage.

Objectif :

L'objectif principal de ce projet était de développer un système de gestion de magasin en C++ permettant de gérer efficacement les produits disponibles dans le magasin. En utilisant des classes de base et des classes dérivées, le système permet d'ajouter différents types de produits avec des attributs spécifiques et de les afficher de manière détaillée. De plus, le système permet la recherche de produits par nom ou prix pour faciliter la navigation des clients.

Fonctionnalités attendues :

Le système comprendra une classe de base pour les produits, avec des classes dérivées pour les différents types de produits. Le magasin sera représenté par une classe centrale gérant un tableau dynamique de pointeurs de produits, permettant l'ajout, la recherche et l'affichage des produits. Les fonctionnalités principales incluent l'ajout de nouveaux types de produits, la recherche avancée et l'affichage détaillé des résultats.

Implémentation :

1. Classes de base et dérivées

Nous avons implémenté trois classes principales :

- **Produit (classe de base)** :
  - Contient les attributs nom (chaîne de caractères) et prix (double).
  - Déclare une méthode virtuelle pure afficher() qui sera redéfinie dans les classes dérivées.
- **Alimentaire (classe dérivée)** :
  - Ajoute l'attribut dateExpiration (chaîne de caractères).
  - Redéfinit la méthode afficher() pour inclure la date d'expiration.
- **Electronique (classe dérivée)** :
  - Ajoute l'attribut garantie (entier représentant la durée de garantie en mois).
  - Redéfinit la méthode afficher() pour inclure la durée de garantie.

2. Classe Magasin :

La classe Magasin gère un tableau dynamique de pointeurs vers des objets Produit.

- **Méthodes** :
  - ajouterProduit(Produit* produit) : Permet d'ajouter un produit au magasin.
  - afficherProduits() : Affiche les détails de tous les produits du magasin.
  - rechercherParNom(string nom) : Recherche un produit par nom et l'affiche.
  - rechercherParPrix(double prix) : Recherche un produit par prix et l'affiche.

Travail réalisé :

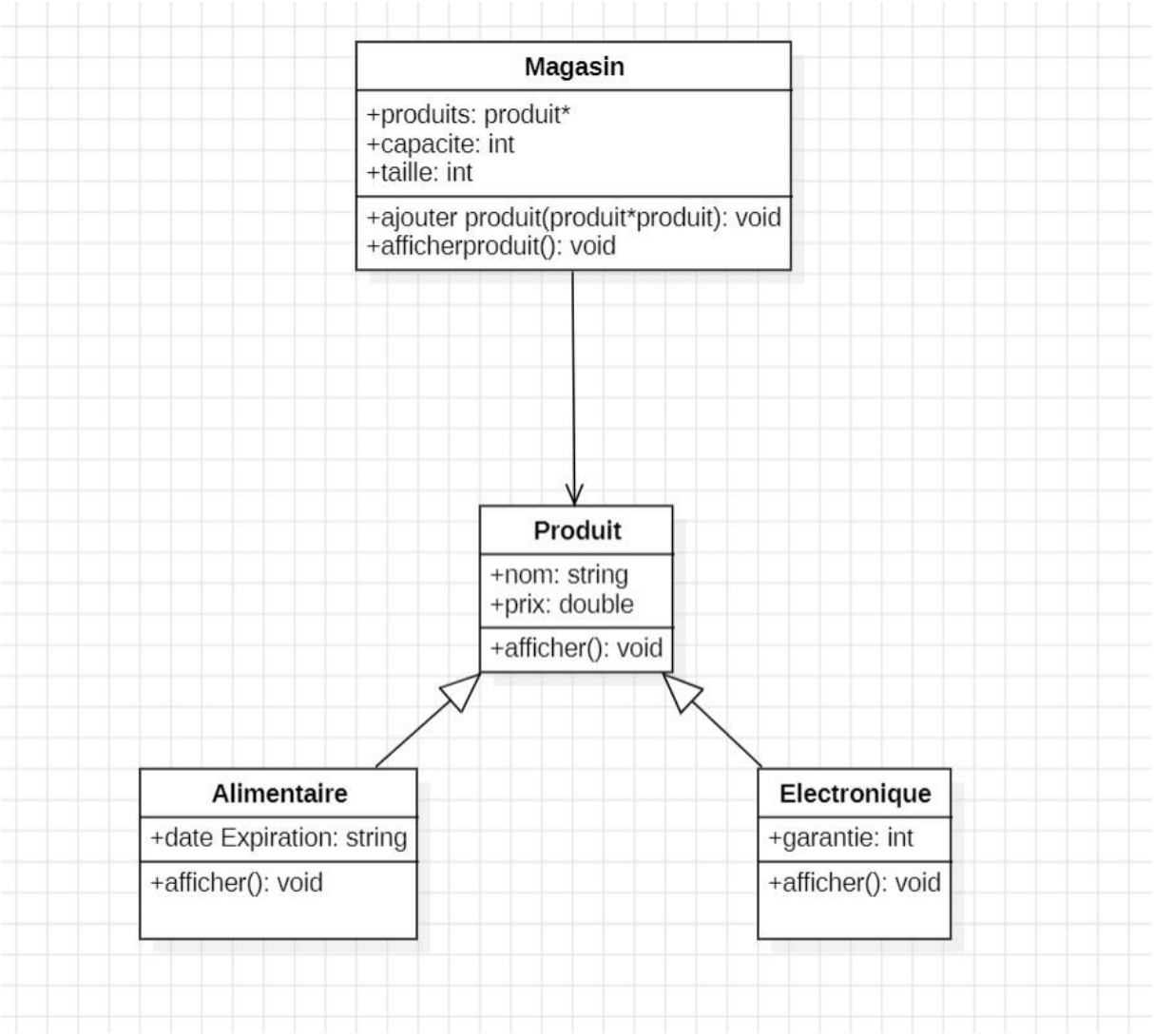
Implémentation des classes Produit, Alimentaire, Electronique et Magasin en respectant les spécifications fournies.

Ajout de plusieurs types de produits avec des attributs spécifiques, tels que demandé.

Implémentation de la recherche de produits par nom ou prix dans le magasin.

Test de l'application en ajoutant plusieurs produits au magasin et en affichant les détails de tous les produits, ainsi que les résultats de la recherche par nom ou prix.

3. Conception :



4. Tests :

Nous avons effectué les tests suivants :

- Ajout de produits alimentaires et électroniques au magasin.
- Affichage de tous les produits dans le magasin.
- Recherche de produits par nom.
- Recherche de produits par prix.

Code Source

```cpp
#include <iostream>
#include <string>

using namespace std;

// Classe de base Produit
class Produit {
protected:
    string nom;
    double prix;

public:
    // Constructeur avec liste d'initialisation
    Produit(string n, double p) : nom(n), prix(p) {}

    // Méthode d'affichage virtuelle
    virtual void afficher() {
        cout << "Nom: " << nom << ", Prix: " << prix << " euros" << endl;
    }

    // Getter pour le nom du produit
    string getNom() const {
        return nom;
    }

    // Getter pour le prix du produit
    double getPrix() const {
        return prix;
    }

    // Destructeur virtuel
    virtual ~Produit() {}
};
```

```cpp
// Classe dérivée Alimentaire
class Alimentaire : public Produit {
protected:
    string dateExpiration;

public:
    // Constructeur avec liste d'initialisation
    Alimentaire(string n, double p, string d) : Produit(n, p), dateExpiration(d) {}

    // Méthode d'affichage redéfinie
    void afficher() override {
        Produit::afficher();
        cout << "Date d'expiration: " << dateExpiration << endl;
    }
};

// Classe dérivée Electronique
class Electronique : public Produit {
protected:
    int garantie;

public:
    // Constructeur avec liste d'initialisation
    Electronique(string n, double p, int g) : Produit(n, p), garantie(g) {}

    // Méthode d'affichage redéfinie
    void afficher() override {
        Produit::afficher();
        cout << "Garantie: " << garantie << " mois" << endl;
    }
};
```

```cpp
// Classe Magasin
class Magasin {
private:
    int capaciteMax; // Capacité maximale du magasin
    Produit** produits; // Tableau de pointeurs vers des objets Produit
    int taille; // Nombre actuel de produits dans le magasin

public:
    // Constructeur avec capacité maximale
    Magasin(int capacite) : capaciteMax(capacite), produits(new Produit*[capacite]), taille(0) {}

    // Méthode pour ajouter un produit
    void ajouterProduit(Produit* produit) {
        if (taille < capaciteMax) {
            produits[taille++] = produit;
        } else {
            cout << "Capacite max atteinte" << endl;
        }
    }

    // Méthode pour afficher tous les produits
    void afficherProduits() {
        for (int i = 0; i < taille; i++) {
            produits[i]->afficher();
        }
    }

    // Méthode pour rechercher un produit par son nom et retourner un pointeur vers le produit
    Produit* rechercherParNom(string nom) {
        for (int i = 0; i < taille; i++) {
            if (produits[i]->getNom() == nom) {
                return produits[i];
            }
        }
    // Méthode pour rechercher un produit par son prix et retourner un pointeur vers le produit
    Produit* rechercherParPrix(double prix) {
        for (int i = 0; i < taille; i++) {
            if (produits[i]->getPrix() == prix) {
                return produits[i];
            }
        }
        return nullptr;
    }

    // Destructeur
    ~Magasin() {
        delete[] produits; // Libération de la mémoire allouée pour le tableau de produits
    }
};
```

Partie Main :

```cpp
int main() {
    // Création du magasin avec une capacité maximale de 2 produits
    Magasin monMagasin(2);

    // Demander à l'utilisateur d'ajouter des produits
    cout << "=== Ajout de produits au magasin ===" << endl;

    for (int i = 0; i < 2; ++i) {
        string typeProduit;
        string nomProduit;
        double prixProduit;

        cout << "Quel type de produit ajouter (Alimentaire / Electronique) ? ";
        cin >> typeProduit;
        cin.ignore(); // Pour consommer le retour chariot laissé par cin

        cout << "Nom du produit ? ";
        getline(cin, nomProduit);

        cout << "Son prix ? ";
        cin >> prixProduit;

        if (typeProduit == "Alimentaire") {
            string dateExpiration;
            cout << "Date d'expiration ? ";
            cin >> dateExpiration;
            monMagasin.ajouterProduit(new Alimentaire(nomProduit, prixProduit, dateExpiration));
        } else if (typeProduit == "Electronique") {
            int garantie;
            cout << "Garantie (en mois) ? ";
            cin >> garantie;
            monMagasin.ajouterProduit(new Electronique(nomProduit, prixProduit, garantie));
        } else {
            cout << "Type de produit non valide !" << endl;
        }
    }

    // Affichage de tous les produits dans le magasin
    cout << "=== Détails de tous les produits dans le magasin ===" << endl;
    monMagasin.afficherProduits();

    // Recherche d'un produit par nom
    string nomRecherche;
    cout << "Entrez le nom du produit à rechercher : ";
    cin >> nomRecherche;
    cout << "Résultat de la recherche par nom : " << endl;
    Produit* produitNom = monMagasin.rechercherParNom(nomRecherche);
    if (produitNom) {
        produitNom->afficher();
    } else {
        cout << "Produit non trouvé." << endl;
    }

    // Recherche d'un produit par prix
    double prixRecherche;
    cout << "Entrez le prix du produit à rechercher : ";
    cin >> prixRecherche;
    cout << "Résultat de la recherche par prix : " << endl;
    Produit* produitPrix = monMagasin.rechercherParPrix(prixRecherche);
    if (produitPrix) {
        produitPrix->afficher();
    } else {
```

```
182     } else {
183         cout << "Produit non trouvé." << endl;
184     }
185
186     return 0;
187 }
188
```

Affichage des tests :

- Ajout d'un produit Alimentaire au magasin

```
=== Ajout de produits au magasin ===
Quel type de produit ajouter (Alimentaire / Electronique) ? Alimentaire
Nom du produit ? Beurre
Son prix ? 15
Date d'expiration ? 2025
=== Détails de tous les produits dans le magasin ===
Nom: Beurre, Prix: 15 euros
Date d'expiration: 2025
```

- Rechercher le produit Alimentaire par son nom :

```
Entrez le nom du produit à rechercher : Beurre
Résultat de la recherche par nom :
Nom: Beurre, Prix: 15 euros
Date d'expiration: 2025
```

- Rechercher le produit Alimentaire par son prix :

```
Entrez le prix du produit à rechercher : 15
Résultat de la recherche par prix :
Nom: Beurre, Prix: 15 euros
Date d'expiration: 2025
```

- Ajout du produit Electronique au magasin

```
=== Ajout de produits au magasin ===
Quel type de produit ajouter (Alimentaire / Electronique) ? Electronique
Nom du produit ? Téléphone
Son prix ? 1455
Garantie (en mois) ? 24
=== Détails de tous les produits dans le magasin ===
Nom: Téléphone, Prix: 1455 euros
Garantie: 24 mois
```

- Rechercher le produit Electronique par son nom :

```
Entrez le nom du produit à rechercher : Téléphone
Résultat de la recherche par nom :
Nom: Téléphone, Prix: 1455 euros
Garantie: 24 mois
```

- Rechercher le produit Electronique par son prix :

```
Entrez le prix du produit à rechercher : 1455
Résultat de la recherche par prix :
Nom: Téléphone, Prix: 1455 euros
Garantie: 24 mois
```

Conclusion

Ce projet nous a permis de mettre en pratique les concepts d'héritage, de polymorphisme et de gestion de la mémoire en C++. Nous avons créé un système de gestion de magasin flexible et fonctionnel. En respectant les directives du cahier des charges, nous avons réussi à implémenter toutes les fonctionnalités requises.