PREPARER PAR :

LABIB MARZOUG-YASSIR SMOUNI-ANASS HADRAOUI

GESTION DE STOCK

PROJET ENCADRE

Rapport de projet

APPLICATION DE GESTION DE STOCK EN LANGAGE C

01/03/2030 - 30/06/2030

NTRODUCTION :

CE RAPPORT PRÉSENTE LE DÉVELOPPEMENT D'UNE APPLICATION DE GESTION DE STOCK EN LANGAGE C, AVEC UNE GESTION DE FICHIERS POUR STOCKER LES DONNÉES DES PRODUITS ET DES UTILISATEURS. L'OBJECTIF PRINCIPAL ÉTAIT DE CRÉER UNE APPLICATION CONVIVIALE ET FONCTIONNELLE, PERMETTANT AUX UTILISATEURS D'INTERAGIR FACILEMENT AVEC LES DONNÉES DE STOCK.

Fonctionnalités

1. AJOUT DE PRODUIT : LES UTILISATEURS PEUVENT AJOUTER DE NOUVEAUX PRODUITS EN SAISISSANT LEURS DÉTAILS, Y COMPRIS LE NOM, LA DESCRIPTION ET LE PRIX.
2. MODIFICATION DE PRODUIT : LES UTILISATEURS PEUVENT MODIFIER LES DÉTAILS D'UN PRODUIT EXISTANT EN SPÉCIFIANT SON IDENTIFIANT.
3. SUPPRESSION DE PRODUIT : LES UTILISATEURS PEUVENT SUPPRIMER UN PRODUIT DE LA LISTE EN SPÉCIFIANT SON IDENTIFIANT.
4. AFFICHAGE DES PRODUITS : LE PROGRAMME AFFICHE LA LISTE DES PRODUITS AVEC LEURS DÉTAILS, Y COMPRIS L'IDENTIFIANT, LE NOM, LA DESCRIPTION ET LE PRIX.
5. SORTIE DU PROGRAMME : LES UTILISATEURS PEUVENT QUITTER LE PROGRAMME À TOUT MOMENT.

Conception

STRUCTURES DE DONNÉES

LE PROGRAMME UTILISE DEUX STRUCTURES DE DONNÉES PRINCIPALES :

- PRODUIT : REPRÉSENTE UN PRODUIT AVEC DES ATTRIBUTS TELS QUE L'IDENTIFIANT, LE NOM, LA DESCRIPTION ET LE PRIX.
- UTILISATEUR : REPRÉSENTE UN UTILISATEUR AVEC UN NOM.

FONCTIONS

LES PRINCIPALES FONCTIONS DU PROGRAMME SONT :

- ADDPRODUCT : PERMET D'AJOUTER UN PRODUIT À LA LISTE.
- MODIFYPRODUCT : PERMET DE MODIFIER LES DÉTAILS D'UN PRODUIT EXISTANT.
- DELETEPRODUCT : PERMET DE SUPPRIMER UN PRODUIT DE LA LISTE.
- DISPLAYPRODUCTS : AFFICHE LA LISTE DES PRODUITS.

Utilisation

1. AU LANCEMENT, LE PROGRAMME DEMANDE À L'UTILISATEUR DE SAISIR SON NOM.
2. ENSUITE, L'UTILISATEUR PEUT CHOISIR PARMI LES OPTIONS DISPONIBLES POUR AJOUTER, MODIFIER, SUPPRIMER DES PRODUITS OU AFFICHER LA LISTE DES PRODUITS.
3. POUR CHAQUE OPTION, LE PROGRAMME GUIDE L'UTILISATEUR À TRAVERS LES ÉTAPES NÉCESSAIRES.

Améliorations Possibles

BIEN QUE LE PROGRAMME ACTUEL OFFRE DES FONCTIONNALITÉS DE BASE POUR GÉRER UN INVENTAIRE DE PRODUITS, PLUSIEURS AMÉLIORATIONS PEUVENT ÊTRE APPORTÉES, NOTAMMENT :

- GESTION DES ERREURS ET VALIDATION DES ENTRÉES UTILISATEUR.
- STOCKAGE DES DONNÉES DANS DES FICHIERS CSV POUR UNE PERSISTENCE DES DONNÉES ENTRE LES EXÉCUTIONS DU PROGRAMME.
- INTERFACE UTILISATEUR PLUS CONVIVIALE ET INTERACTIVE.
- AJOUT DE FONCTIONNALITÉS AVANCÉES TELLES QUE LA RECHERCHE DE PRODUITS, LE TRI, LES RAPPORTS SUR LES VENTES, ETC.

Conclusion

LE PROGRAMME DE GESTION DE STOCK EN LANGAGE C OFFRE UNE SOLUTION SIMPLE MAIS FONCTIONNELLE POUR GÉRER UN INVENTAIRE DE PRODUITS. AVEC DES AMÉLIORATIONS SUPPLÉMENTAIRES, IL PEUT DEVENIR UN OUTIL PLUS PUISSANT POUR LES PETITES ENTREPRISES ET LES ENTREPRENEURS.

```
#INCLUDE <STDIO.H>
#INCLUDE <STDLIB.H>
#INCLUDE <STRING.H>

#DEFINE MAX_NAME_LENGTH 50
#DEFINE MAX_DESCRIPTION_LENGTH 100
#DEFINE MAX_PRODUCTS 100
#DEFINE FILENAME_PRODUCTS "PRODUCTS.CSV"
#DEFINE FILENAME_USERS "USERS.CSV"

// STRUCTURE DE DONNÉES POUR UN PRODUIT
TYPEDEF STRUCT {
   INT ID;
   CHAR NAME[MAX_NAME_LENGTH];
   CHAR DESCRIPTION[MAX_DESCRIPTION_LENGTH];
   FLOAT PRICE;
} PRODUCT;

// STRUCTURE DE DONNÉES POUR UN UTILISATEUR
TYPEDEF STRUCT {
   CHAR NAME[MAX_NAME_LENGTH];
} USER;

// FONCTION POUR AJOUTER UN PRODUIT
VOID ADDPRODUCT(PRODUCT PRODUCTS[], INT *COUNT) {
   PRINTF("ENTER PRODUCT DETAILS:\N");
   PRINTF("NAME: ");
   SCANF("%S", PRODUCTS[*COUNT].NAME);
   PRINTF("DESCRIPTION: ");
   SCANF("%S", PRODUCTS[*COUNT].DESCRIPTION);
   PRINTF("PRICE: ");
   SCANF("%F", &PRODUCTS[*COUNT].PRICE);
   PRODUCTS[*COUNT].ID = *COUNT + 1;
   (*COUNT)++;
}
```

```
// FONCTION POUR AFFICHER LA LISTE DES PRODUITS
VOID DISPLAYPRODUCTS(PRODUCT PRODUCTS[], INT COUNT) {
   PRINTF("PRODUCT LIST:\N");
   PRINTF("ID\TNAME\TDESCRIPTION\TPRICE\N");
   FOR (INT I = 0; I < COUNT; I++) {
      PRINTF("%D\T%S\T%S\T%.2F\N", PRODUCTS[I].ID, PRODUCTS[I].NAME,
PRODUCTS[I].DESCRIPTION, PRODUCTS[I].PRICE);
   }
}

// FONCTION POUR MODIFIER UN PRODUIT
VOID MODIFYPRODUCT(PRODUCT PRODUCTS[], INT COUNT) {
   INT ID;
   PRINTF("ENTER THE ID OF THE PRODUCT YOU WANT TO MODIFY: ");
   SCANF("%D", &ID);

   FOR (INT I = 0; I < COUNT; I++) {
      IF (PRODUCTS[I].ID == ID) {
         PRINTF("ENTER NEW DETAILS FOR THE PRODUCT:\N");
         PRINTF("NAME: ");
         SCANF("%S", PRODUCTS[I].NAME);
         PRINTF("DESCRIPTION: ");
         SCANF("%S", PRODUCTS[I].DESCRIPTION);
         PRINTF("PRICE: ");
         SCANF("%F", &PRODUCTS[I].PRICE);
         PRINTF("PRODUCT MODIFIED SUCCESSFULLY.\N");
         RETURN;
      }
   }
   PRINTF("PRODUCT WITH ID %D NOT FOUND.\N", ID);
}

// FONCTION POUR SUPPRIMER UN PRODUIT
VOID DELETEPRODUCT(PRODUCT PRODUCTS[], INT *COUNT) {
   INT ID;
   PRINTF("ENTER THE ID OF THE PRODUCT YOU WANT TO DELETE: ");
   SCANF("%D", &ID);
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_NAME_LENGTH 50
#define MAX_DESCRIPTION_LENGTH 100
#define MAX_PRODUCTS 100
#define FILENAME_PRODUCTS "products.csv"
#define FILENAME_USERS "users.csv"

// Structure de données pour un produit
typedef struct {
    int id;
    char name[MAX_NAME_LENGTH];
    char description[MAX_DESCRIPTION_LENGTH];
    float price;
} Product;

// Structure de données pour un utilisateur
typedef struct {
    char name[MAX_NAME_LENGTH];
} User;

// Fonction pour ajouter un produit
void addProduct(Product products[], int *count) {
    printf("Enter product details:\n");
    printf("Name: ");
    scanf("%s", products[*count].name);
    printf("Description: ");
    scanf("%s", products[*count].description);
    printf("Price: ");
    scanf("%f", &products[*count].price);
```

```c
    products[*count].id = *count + 1;
    (*count)++;
}

// Fonction pour afficher la liste des produits
void displayProducts(Product products[], int count) {
    printf("Product List:\n");
    printf("ID\tName\tDescription\tPrice\n");
    for (int i = 0; i < count; i++) {
        printf("%d\t%s\t%s\t%.2f\n", products[i].id, products[i].name, products[i].description, products[i].price);
    }
}

// Fonction pour modifier un produit
void modifyProduct(Product products[], int count) {
    int id;
    printf("Enter the ID of the product you want to modify: ");
    scanf("%d", &id);

    for (int i = 0; i < count; i++) {
        if (products[i].id == id) {
            printf("Enter new details for the product:\n");
            printf("Name: ");
            scanf("%s", products[i].name);
            printf("Description: ");
            scanf("%s", products[i].description);
            printf("Price: ");
            scanf("%f", &products[i].price);
            printf("Product modified successfully.\n");
            return;
        }
    }
```

```c
    printf("Product with ID %d not found.\n", id);
}

// Fonction pour supprimer un produit
void deleteProduct(Product products[], int *count) {
    int id;
    printf("Enter the ID of the product you want to delete: ");
    scanf("%d", &id);

    for (int i = 0; i < *count; i++) {
        if (products[i].id == id) {
            for (int j = i; j < *count - 1; j++) {
                products[j] = products[j + 1];
            }
            (*count)--;
            printf("Product deleted successfully.\n");
            return;
        }
    }
    printf("Product with ID %d not found.\n", id);
}

// Fonction principale
int main() {
    Product products[MAX_PRODUCTS];
    User user;

    int productCount = 0;

    // Interface utilisateur
    printf("Welcome to Stock Management System\n");
    printf("Enter your name: ");
    scanf("%s", user.name);

    int choice;
    do {
        printf("\n1. Add Product\n");
        printf("2. Modify Product\n");
        printf("3. Delete Product\n");
        printf("4. Display Products\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addProduct(products, &productCount);
                break;
            case 2:
                modifyProduct(products, productCount);
                break;
            case 3:
                deleteProduct(products, &productCount);
                break;
            case 4:
                displayProducts(products, productCount);
                break;
            case 5:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 5);
```

```c
        printf("\n1. Add Product\n");
        printf("2. Modify Product\n");
        printf("3. Delete Product\n");
        printf("4. Display Products\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addProduct(products, &productCount);
                break;
            case 2:
                modifyProduct(products, productCount);
                break;
            case 3:
                deleteProduct(products, &productCount);
                break;
            case 4:
                displayProducts(products, productCount);
                break;
            case 5:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 5);

    return 0;
}

```

MERCI:)