Rapport de Projet

Système de Gestion de Salle de Sport

Informations du Projet

Cours :	Programmation C
Date :	Décembre 2025
Langage :	C (ISO Standard)
Type :	Application console

Présenté par :

KACEM Nour et JABOU Mohamed Amine

Professeur : MANSOURI Sameh

18 décembre 2025

Table des matières

1 Introduction

Ce rapport présente le développement d'un **système de gestion de salle de sport** réalisé en langage C. Il s'agit d'une application console permettant aux membres de gérer leurs abonnements et aux administrateurs de gérer les ressources de la salle de sport (plans d'abonnement, équipements et membres).

1.1 Contexte du Projet

Dans le cadre du cours de Programmation C, ce projet vise à mettre en pratique les concepts fondamentaux de la programmation structurée en C, notamment la manipulation de structures de données, la gestion de fichiers, et la conception d'interfaces utilisateur en mode console.

1.2 Objectifs Principaux

— Créer un système fonctionnel de gestion de salle de sport en langage C
— Implémenter un système d'authentification (Membre et Administrateur)
— Fournir des opérations CRUD (Create, Read, Update, Delete) pour les ressources
— Assurer la persistance des données via des fichiers texte
— Appliquer les bonnes pratiques de programmation modulaire

2 Fonctionnalités du Système

Le système offre deux interfaces distinctes selon le type d'utilisateur : les membres et les administrateurs.

2.1 Fonctionnalités pour les Membres

Les membres disposent d'un espace personnel leur permettant de :

1. **Créer un compte** : Enregistrement avec nom d'utilisateur et mot de passe
2. **Se connecter** : Authentification pour accéder à l'espace personnel
3. **Consulter les plans** : Visualisation des plans d'abonnement disponibles
4. **Souscrire à un plan** : Choix et activation d'un abonnement mensuel
5. **Voir le profil** : Consultation de l'abonnement actuel et des informations personnelles

2.2 Fonctionnalités pour les Administrateurs

Les administrateurs ont accès à un panneau de contrôle complet :

1. **Connexion sécurisée** : Authentification avec identifiants admin
2. **Gestion des plans d'abonnement** :
   — Ajouter de nouveaux plans
   — Consulter tous les plans
   — Modifier les plans existants

— Supprimer des plans

3. **Gestion des équipements** :
   — Ajouter du matériel
   — Consulter l'inventaire
   — Modifier les équipements
   — Retirer des équipements
4. **Gestion des membres** :
   — Visualiser tous les membres inscrits
   — Consulter leurs abonnements

3 Architecture Technique

3.1 Concepts de Programmation Utilisés

Le projet met en œuvre plusieurs concepts fondamentaux de la programmation C :

Concept	Utilisation dans le Projet
Structures (`struct`)	Définition des modèles de données pour Plan, Équipement et Membre
Tableaux	Stockage des collections de plans, équipements et membres
Pointeurs	Passage efficace de données entre les fonctions
Fonctions	Modularisation du code en composants réutilisables
Entrées/Sorties Fichiers	Sauvegarde et chargement des données depuis des fichiers texte
Switch/Case	Navigation dans les menus de l'application
Manipulation de Chaînes	Gestion des noms d'utilisateur, mots de passe et saisies textuelles

TABLE 1 – Concepts de programmation C utilisés

3.2 Structures de Données

Le système utilise trois structures principales pour modéliser les entités :

```c
typedef struct {
    int id_plan;
    char name[50];
    float price;
    char description[100];
} Plan;
```

Listing 1 – Structure Plan

```c
typedef struct {
    int id_equipment;
    char name[50];
    int quantity;
```

```
    char description[100];
} Equipment;
```

Listing 2 – Structure Equipment

```
typedef struct {
    int id_member;
    char username[50];
    char password[50];
    char name[100];
    int id_current_plan;  // -1 si pas d'abonnement
} Member;
```

Listing 3 – Structure Member

3.3 Organisation des Fichiers

Le projet est organisé de manière modulaire pour faciliter la maintenance :

Structure du Projet

```
projet/
|-- src/
|   |-- main.c          - Point d'entree, menu principal
|   |-- member.c/h      - Gestion des comptes membres
|   |-- admin.c/h       - Authentification et gestion admin
|   |-- plans.c/h       - Operations CRUD sur les plans
|   |-- equipment.c/h   - Operations CRUD sur les equipements
|   +-- utils.c/h       - Fonctions utilitaires
|
+-- data/
    |-- plans.txt       - Stockage des plans d'abonnement
    |-- equipment.txt   - Stockage des equipements
    +-- members.txt     - Stockage des comptes membres
```

4 Flux de l'Application

4.1 Diagramme de Flux Principal

4.2 Persistance des Données

Toutes les données sont stockées dans des **fichiers texte** utilisant un délimiteur pipe (|) :

Exemple - plans.txt

```
3
1|Musculation Only|50.00|Acces aux equipements de musculation
2|Cardio Only|40.00|Acces aux machines cardio
3|Combined|70.00|Acces complet a tous les equipements
```

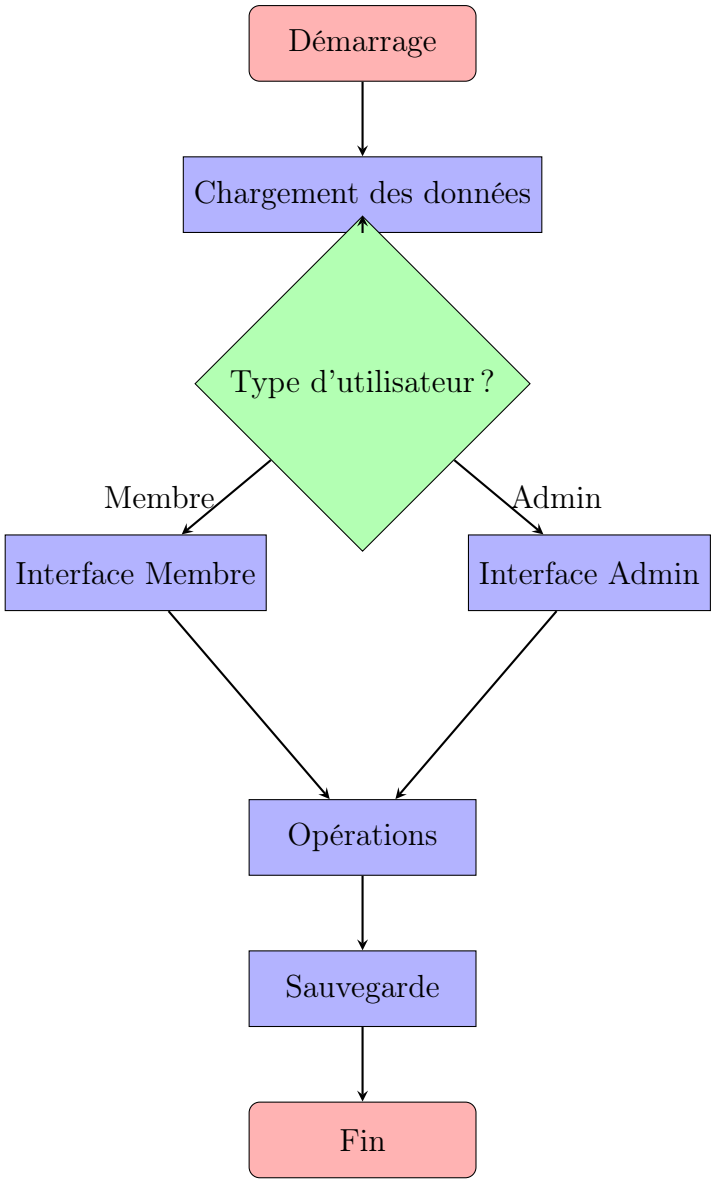

FIGURE 1 – Flux principal de l'application

Format : La première ligne contient le nombre d'éléments, suivie d'une ligne par élément au format `id|nom|valeur|description`.

4.3 Fonctions Clés

4.3.1 Système Membre

— `create_member_account()` : Enregistrement de nouveaux membres
— `member_login()` : Authentification des membres
— `subscribe_to_plan()` : Attribution d'un plan à un membre
— `save_members_to_file()` : Persistance des données membres

4.3.2 Système Administrateur

— `admin_login()` : Authentification admin

— `add_plan_interactive()` : Création de nouveaux plans
— `modify_plan()` : Mise à jour des plans existants
— `delete_plan()` : Suppression de plans
— `save_plans_to_file()` : Persistance des données plans

4.3.3 Fonctions Utilitaires

— `get_int_input()` : Saisie sécurisée d'entiers avec validation
— `get_string_input()` : Gestion sécurisée des chaînes de caractères
— `print_header()` : Formatage des en-têtes de menu
— `clear_screen()` : Effacement de la console

5 Compilation et Exécution

5.1 Compilation du Projet

Pour compiler le projet, utilisez la commande suivante :

Commande de Compilation

```
gcc -o gym_app.exe src\main.c src\member.c src\admin.c
    src\plans.c src\equipment.c src\utils.c -Wall
```

5.2 Exécution de l'Application

Lancement de l'Application

```
.\gym_app.exe
```

5.3 Prérequis Système

— Compilateur C (GCC recommandé)
— Terminal Windows/Linux/macOS
— Aucune bibliothèque externe requise (uniquement la bibliothèque standard C)

6 Techniques de Programmation Avancées

6.1 Validation des Entrées

Toutes les saisies utilisateur sont validées pour prévenir les erreurs :

```
1 int get_int_input() {
2     int value;
3     while (scanf("%d", &value) != 1) {
4         clear_input_buffer();
5         printf("Entree invalide ! Veuillez entrer un nombre : ");
6     }
```

```
    clear_input_buffer();
    return value;
}
```

Listing 4 – Validation des entrees entieres

6.2 Sauvegarde Automatique

Les modifications sont sauvegardées immédiatement après chaque opération pour éviter la perte de données :

```
// Apres l'ajout d'un plan
add_plan_interactive(plans, count);
save_plans_to_file(plans, *count);  // Sauvegarde auto
```

Listing 5 – Sauvegarde automatique apres ajout

6.3 Conception Modulaire

Chaque fonctionnalité est séparée dans son propre module (fichier), rendant le code :

— Facile à comprendre

— Facile à maintenir

— Facile à tester

— Réutilisable

7 Défis et Solutions

Défi	**Solution Implémentée**
Persistance des données	Implémentation d'E/S fichiers avec délimiteur pipe
Validation des entrées	Création de fonctions utilitaires de validation
Navigation dans les menus	Utilisation de switch/case avec choix numériques clairs
Encodage des caractères	Remplacement des caractères spéciaux par du texte compatible
Cohérence des données	Ajout de sauvegarde automatique après chaque opération

TABLE 2 – Défis rencontrés et solutions apportées

8 Conclusion

Ce projet démontre la maîtrise des concepts fondamentaux de la programmation C, notamment :

— La programmation structurée avec fonctions et modules

— L'utilisation de structures de données avec `struct`

— La gestion de fichiers pour la persistance des données

— La validation des entrées utilisateur

— Le développement d'applications console avec menus

— La gestion de la mémoire avec des tableaux

L'application est entièrement fonctionnelle, gère les cas limites, et fournit une solution complète de gestion de salle de sport en utilisant uniquement les fonctions de la bibliothèque standard C.

8.1 Perspectives d'Amélioration

Pour étendre ce projet, les améliorations suivantes pourraient être envisagées :

1. Ajout d'un système de cryptage pour les mots de passe
2. Implémentation d'un historique des abonnements
3. Ajout de statistiques pour les administrateurs
4. Développement d'une interface graphique (GUI)
5. Migration vers une base de données relationnelle

9 Annexes

9.1 Compte de Démonstration

Identifiants Administrateur

— **Nom d'utilisateur :** `admin`

— **Mot de passe :** `admin123`

9.2 Données par Défaut

— 3 plans d'abonnement (Musculation, Cardio, Combine)

— 5 equipements (Halteres, Tapis de course, Banc de presse, etc.)

— 1 compte membre de test (nom d'utilisateur : `nour`, mot de passe : `nour`)

Fin du Rapport