Hakim Tsouria

Guillaume Vial

ITII P22M

Projet d'Algorithmique et Programmation

Langage C – Techniques de programmation en C

13/12/24

Sommaire

1. Introduction

Dans le cadre du cours d'algorithmique et de programmation de notre formation d'ingénieur à l'ISEN Marseille, nous avons développé une application en langage C visant à gérer un système de gestion de location/réservation. Ce projet, réalisé en binôme, répond à un cahier des charges imposé, combinant aspects techniques et méthodologiques, mais dont le thème était libre.

Ainsi notre base de données sera sur le thème d'un festival de musique, le "BÊTA Festival".

Notre application se concentre sur deux modules principaux qui interagissent entre eux :

1. **La gestion des artistes** : permettant d'ajouter, modifier, supprimer et afficher les informations relatives aux artistes participant au festival.
2. **La gestion des concerts** : offrant la possibilité de planifier, consulter et organiser les concerts en fonction des artistes disponibles.

Ce document détaille notre méthodologie, les choix techniques effectués, ainsi que les résultats obtenus.

2. Spécifications fonctionnelles et techniques

● Expression des besoins

Notre application vise principalement à offrir un outil facile, performant et solide pour gérer les informations essentielles d'un festival. Son objectif est de faciliter l'organisation et la consultation des informations concernant les artistes et les concerts, tout en respectant les exigences du cahier des charges.

L'application doit répondre aux besoins suivants:

Gestion des artistes :

- Ajouter un artiste avec des informations comme : nom, prénom, genre musical, nombre de followers....
- Modifier les informations d'un artiste.
- Supprimer un artiste de la base de données.
- Afficher la liste des artistes.

Gestion des concerts :

- Planifier un concert en spécifiant la date, l'heure, les artistes participants, le nombre de places disponibles…
- Modifier les informations d'un concert.
- Supprimer un concert.
- Afficher la liste des concerts programmés par date

Gestion de la base de données :

- Sauvegarder les données des artistes et des concerts dans des fichiers.
- Faire interagir les bases entres elles

● Liste des exigences

Ce projet dispose d'un cahier des charges complet qui décrit les attendus et les obligations concernant la création de notre application. Celle-ci doit :

- Fonctionner selon une gestion de location/prêt
- Être validé au préalable par le client
- Être codé en C sous Linux
- Posséder entre 500-1000 lignes
- Contenir un menu
- Contenir 2 structures
- Contenir différentes techniques de programmation (fonctions, pointeurs, tableaux, chaînes de caractères, structures, mémoire dynamique, fichiers, modularités, librairies standards)

● Partage des tâches

Pour mener correctement ce projet il a été essentiel de se partager les tâches et pour cela nous avons réalisé un tableau contenant l'attribution des rôles:

Rôle	Nom
Concepteur	Guillaume
Architecte	Hakim
développeur	Hakim & Guillaume
Communication	Guillaume
Qualité	Hakim & Guillaume
Suivi de projet	Hakim

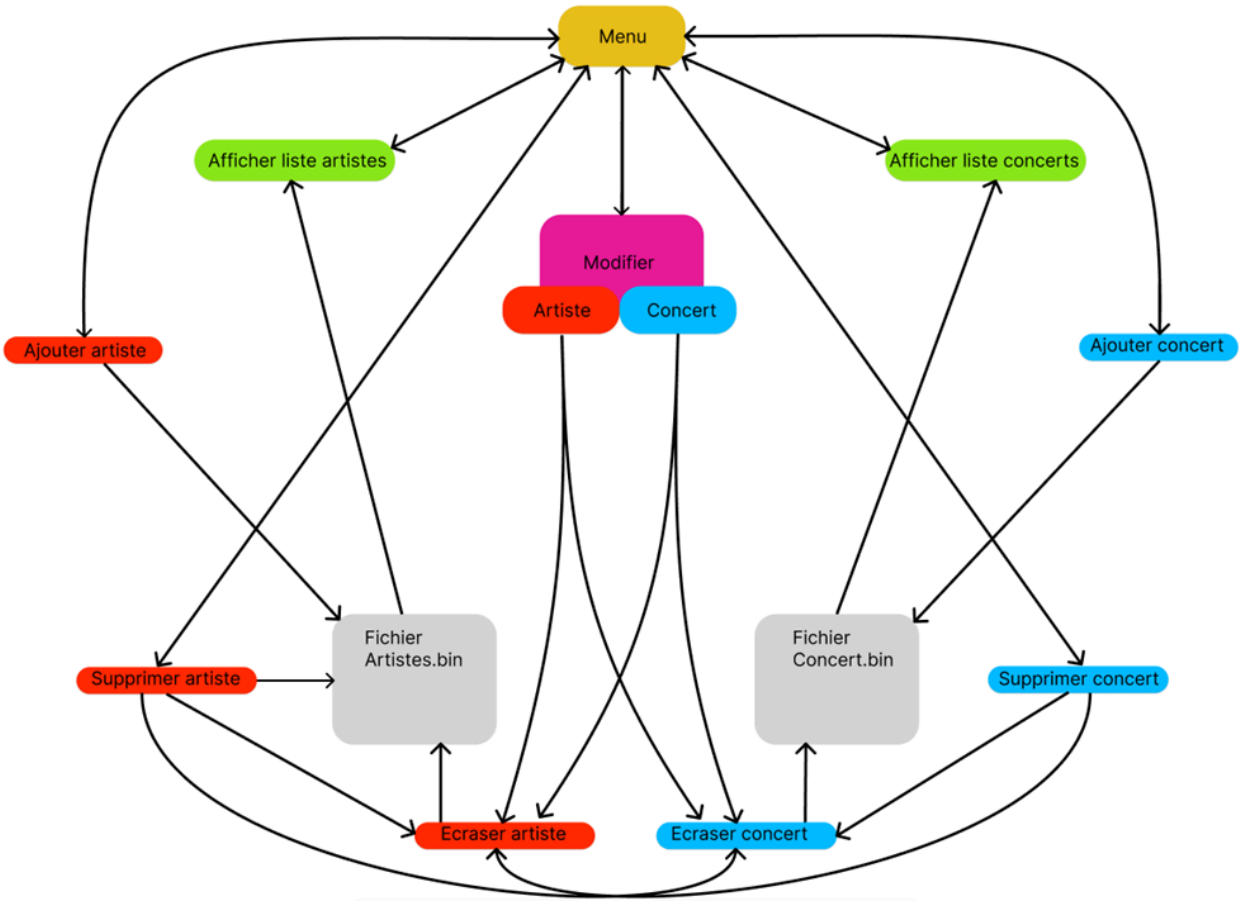
3. Conception logicielle

- Description des structures (artistes, concerts)

Pour débuter le projet nous avons dû concevoir 2 structures qui définissent les paramètres de nos bases de données. La première intitulée “Artistes” concerne tous les paramètres des artistes qui seront renseignés par l’utilisateur. On y retrouve le nom, le nombre de followers, le style de musique ainsi qu’un paramètre qui définit si l’artiste est programmé ou non “is booked”.
La seconde intitulée “Concerts”, qui concerne tous les paramètres des concerts, contient le nom du concert, le nombre de places, la date et l’heure, le nombre de parties qui définit le nombre d’artistes présents à ce concert (qui est de 5 maximum) et les artistes.

- Architecture générale du programme

L’architecture de notre programme peut se visualiser grâce à ce schéma qui nous montre les liens entre le “main”, les fonctions et les fichiers :



4. Développement

- ## Approche modulaire et utilisation des techniques de programmation (pointeurs, boucles, etc.)

Pour programmer notre application nous avons fait appel à la programmation modulaire. Cette méthode permet de simplifier le code, ainsi que le travail en binôme, en décomposant le programme en plusieurs modules :

- ➢ les interfaces: les fichiers.h (fonctions.h & projet_festival.h)
- ➢ les implémentations des fonctionnalités des interfaces: les fichiers.c (fonctions.c & projet_festival.c)

Nous nous sommes également servis des techniques de programmation que nous avons acquises en cours d'algorithmique et programmation afin de développer le programme. Tels que les variables, instructions, les boucles, les structures, les fonctions, les pointeurs, les tableaux, les fichiers ou encore l'allocation dynamique de mémoire.

- ## Description des fonctions

La suite du programme se concentre sur les fonctions. Elle possède les fonctions:

- ➢ ajouter_artist

Permet d'entrer les données d'un nouvel artiste, de l'ajouter dans le fichier artistes.bin, et d'ajouter cet artiste dans la structure d'un concert de notre choix (dans le fichier concert.bin), si l'on souhaite programmer l'artiste.

- ➢ menu;

Menu principal avec 9 options (exemple : ajouter un concert, supprimer un artiste, etc.).

- ➢ save_artist:

Ajoute un artiste de plus dans le fichier artistes.bin.

- ➢ afficher_list_artistes

Affiche la liste des artistes stockés dans le fichier artistes.bin.

- ➢ affiche_artiste

Affiche les informations d'un artiste.

- ➢ ajouter_concert

Permet d'entrer les données d'un nouveau concert, de l'ajouter dans le fichier concert.bin, et d'y sélectionner un ou plusieurs artistes.

- ➢ save_concert

Ajoute un concert de plus dans le fichier concert.bin.

- ➢ getAll_unprogramed_artists:

Retourne le nombre d'artistes non programmés et renseigne les artistes non programmés dans un tableau.

- programmer_artist:

Modifie l'état de la structure Artiste pour indiquer si l'artiste est programmé ou non (is_booked)

- trouver_artiste:

Cherche dans le fichier artistes.bin et renvoie l'artiste correspondant au nom donné en paramètre.

- supprimer_artiste:

Supprime un artiste donné du fichier artistes.bin.

- afficher_list_nom_artistes:

Affiche le numéro et le nom des artistes du fichier artistes.bin.

- remove_artist_from_concert:

Supprime un artiste des listes d'artistes de tous les concerts dans le fichier concert.bin.Par exemple, si l'artiste "Hakim" est présent dans le fichier concert.bin, il sera supprimé,et le nombre d'artistes pour ce concert sera décrémenté.

- supprimer_concert:

Supprime un concert donné du fichier concert.bin.

- tableau_artistes:

Lit le fichier artistes.bin, renseigne tous les artistes dans un tableau et renvoie leur nombre.

- tableau_concerts:

Lit le fichier concert.bin, renseigne tous les concerts dans un tableau et renvoie leur nombre.

- ecraser_artistes:

Remplace les données du fichier artistes.bin par celles de la liste donnée en paramètre.

- ecraser_concert:

Remplace les données du fichier concert.bin par celles de la liste donnée en paramètre.

- afficher_concert:

Affiche un concert à partir de son adresse.

- afficher_list_concerts:

Affiche tous les concerts du fichier concert.bin.

- modifier:

Permet de modifier les données d'un concert ou d'un artiste.

- leave:

Efface la console, affiche un texte, attend une réponse (1 ou 2), et retourne cette réponse.

- Ajt_artiste_dans_concert_dispo:

Permet de sélectionner un concert et d'y programmer un artiste (si le nombre de places le permet).

- ➢ Ajt_artiste_dans_concert_dispo:

Transforme une chaîne de caractères en deux entiers (heures et minutes).

- ➢ synchroniser_artiste:

Lit les artistes depuis le fichier artistes.bin et met à jour leurs données dans le fichier concert.bin pour synchroniser les informations. Par exemple, si un style de musique d'un artiste est modifié dans artistes.bin, cette fonction s'assure que concert.bin reflète ce changement.

- ➢ saisir_date:

Demande une date au format 10/12, 11/12 ou 12/12 et retourne un entier correspondant (0, 1 ou 2).

- ➢ saisir_heure:

Redemande la saisie tant qu'une heure au format xxhxx (où x est un entier de 0 à 9) n'est pas entrée.

- ➢ is_0to9:

Vérifie si un caractère est un chiffre entre 0 et 9.

- ➢ message_supprimer:

Affiche "supprimer ?", attend une réponse (1 ou 2), et retourne cette réponse.

- ➢ nb_un_to_cinq:

Vérifie que le caractère tapé est un entier entre 1 et 5, et retourne la réponse.

5. Tests et validation

- ## Tests unitaires et d'intégration

  Pour ce projet, nous avons réalisé des tests unitaires individuellement pour chaque fonction afin de nous assurer que chacune fonctionne correctement de manière isolée. À chaque problème rencontré, nous avons pris soin de le noter pour pouvoir le résoudre plus facilement si jamais il réapparaissait plus tard dans le projet.

- Cahier de recette

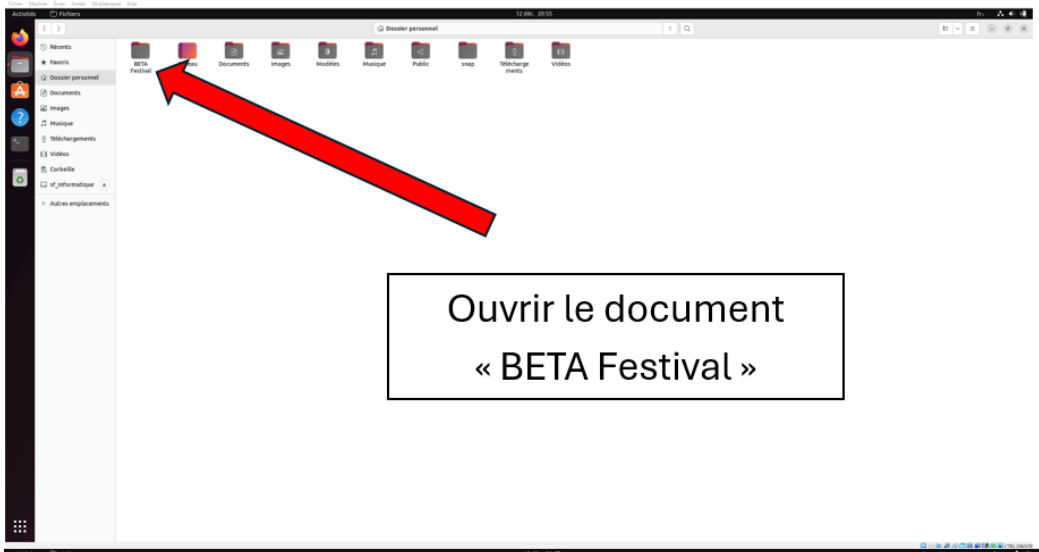
Voici le cahier de recette de notre projet qui contient chacune des fonctions et leur validation au tests unitaires qui ont été effectués.

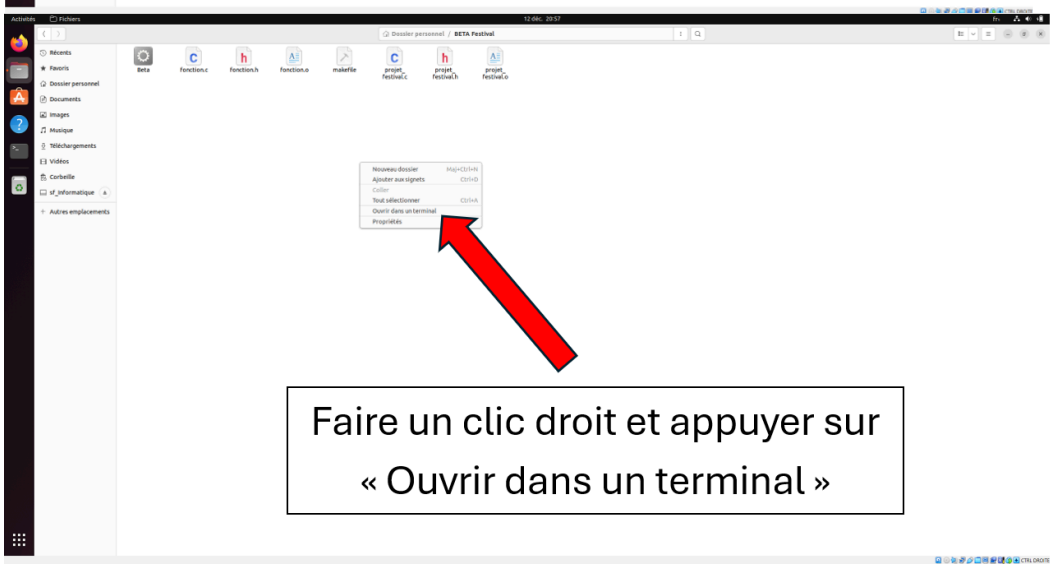
Numéro	Fonctions	Checking
1	menu	OK
2	ajouter_artiste	OK
3	save_artist	OK
4	afficher_list_artistes	OK
5	affiche_artiste	OK
6	ajouter_concert	OK
7	save_concert	OK
8	getAll_unprogramed_artists	OK
9	programmer_artist	OK
10	trouver_artiste	OK
11	supprimer_artiste	OK
12	afficher_list_nom_artistes	OK
13	remove_artist_from_concert	OK
14	supprimer_concert	OK
15	tableau_artistes	OK
16	tableau_concerts	OK
17	ecraser_artistes	OK
18	ecraser_concert	OK
19	afficher_concert	OK
20	afficher_list_concerts	OK
21	modifier	OK
22	leave	OK
23	Ajt_artiste_dans_concert_dispo	OK
24	get_int_time	OK
25	synchroniser_artiste	OK
26	saisir_date	OK
27	saisir_heure	OK
28	is_0to9	OK
29	message_supprimer	OK
30	nb_un_to_cinq	OK

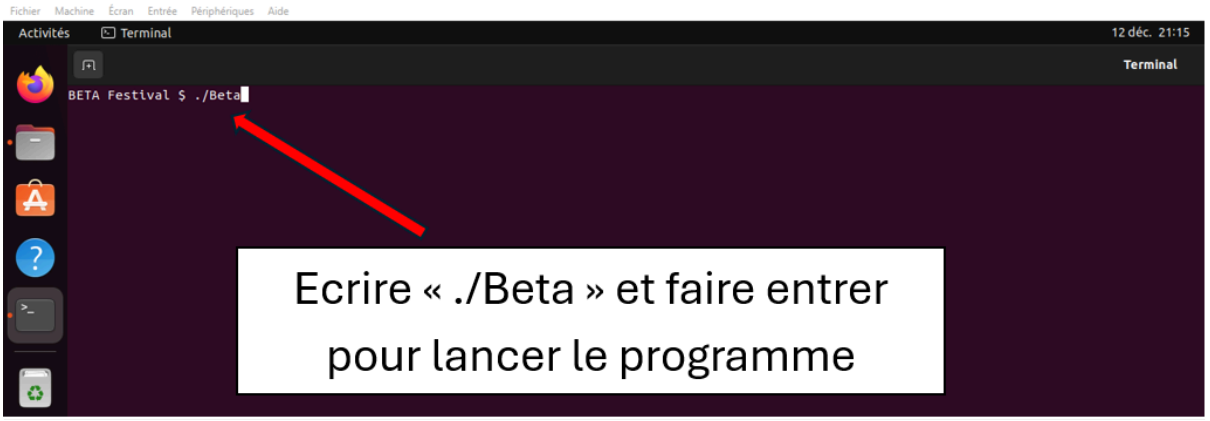
6. Documentation utilisateur

- Guide d’utilisation de l’application

Pour utiliser cette application il faut posséder un ordinateur sous Linux et suivre le tutoriel suivant:







7. Bilan et perspectives

- Analyse des objectifs atteints et des difficultés rencontrées

Ce projet de gestion de festival a été une expérience très enrichissante, qui nous a permis de mieux comprendre les bases de la programmation modulaire et la gestion de données avec des fichiers binaires. Trois points principaux se dégagent de ce travail :

D'abord, l'approche modulaire s'est révélée être un véritable avantage, notamment pour le travail en binôme. En organisant le code en plusieurs fichiers distincts, nous avons pu répartir les tâches efficacement et travailler en parallèle sans trop de confusion. Cela a aussi simplifié le débogage et la compréhension globale du projet, un aspect que nous avons particulièrement apprécié.

Ensuite, nous avons pris conscience de l'importance d'un bon plan d'opérations avant de commencer à coder. À plusieurs reprises, nous avons dû réécrire ou ajuster des fonctions parce qu'on s'est rendu compte qu'elles étaient redondantes ou mal conçues. Cela a entraîné une perte de temps qui aurait pu être évitée avec une meilleure anticipation. C'est une leçon précieuse pour nos futurs projets.

Enfin, malgré ces ajustements, nous sommes satisfaits d'avoir atteint l'objectif principal. Le programme est pleinement fonctionnel et répond aux besoins de gestion d'un festival : ajout, modification, suppression d'artistes et de concerts, tout cela en assurant un stockage fiable grâce aux fichiers binaires.

- Évolutions potentielles du projet

Tout d'abord, un point important à améliorer serait la gestion des erreurs de saisie. Actuellement, si l'utilisateur entre un mauvais type de donnée, comme un caractère alors qu'un entier est attendu, la console plante.

Bien que nous ayons déjà ajouté quelques protections contre ce type d'erreur, nous aimerions rendre le programme complètement imperméable à ce genre de situation, en gérant mieux les erreurs de saisie pour éviter que le programme ne se bloque.

Ensuite, une autre amélioration serait de trier les concerts non seulement par date, mais aussi par heure. Bien que nous ayons déjà mis en place un tri par date, l'ajout d'un tri par heure permettrait de gérer de manière plus précise l'ordre des concerts. Nous avons déjà commencé à travailler sur cette fonctionnalité à travers la fonction get_int_time, qui convertit l'heure en une somme d'heures et de minutes, rendant ainsi le tri plus simple et plus performant.

Nous avons même pu réfléchir à la manière dont ce tri pourrait avoir lieu. L'idéal pour nous serait de convertir l'heure de début de concert en une somme de minutes. Ainsi, nous pourrions comparer ces sommes entre elles pour chaque concert.

Enfin, nous aimerions rendre impossible le chevauchement des concerts. Actuellement, un concert peut se superposer à un autre, ce qui peut entraîner des conflits dans l'organisation. Par exemple, un concert de 5 parties qui commence à midi devrait se terminer à 17 heures, et il serait illogique qu'un autre concert commence avant. Nous pourrions ajouter une fonctionnalité qui vérifie les horaires de début et de fin des concerts, afin d'empêcher tout chevauchement. Cela garantit une meilleure gestion du planning et une organisation plus réaliste des concerts.

8. Conclusion

- Résumé des résultats obtenus

En termes de résultats, nous avons atteint tous les objectifs fixés pour ce projet. Le programme fonctionne parfaitement, sans aucun problème. Toutes les fonctionnalités sont opérationnelles : ajout, modification, suppression d'artistes et de concerts, gestion des bases de données, et tout est bien enregistré dans les fichiers. Le système est fluide et fiable, comme prévu.

Autrement nous pouvons souligner que nous avons réussi à terminer le projet avec moins de 1000 lignes de code, annotations comprises. Cela montre que nous avons respecté les contraintes techniques, mais aussi que notre code est bien structuré et lisible. Nous sommes satisfait du résultat, malgré les défis rencontrés en cours de route (et il y en a eu beaucoup), nous avons su bien gérer notre développement et notre temps pour obtenir un programme propre et fonctionnel.

● Conclusion personnel

Guillaume Vial:

Pour ma part, ce projet a été une réelle opportunité pour moi d'apprendre à coder en C puisque je n'en avais jamais fait auparavant. Malgré les difficultés lors des premiers cours, j'ai réussi à surmonter les obstacles et à améliorer mes compétences au fil du temps. Ce projet m'a permis non seulement de développer des connaissances techniques, mais aussi de renforcer mes capacités de travail en équipe, car coder à plusieurs est une tâche difficile qui nécessite beaucoup d'organisations et une communication claire et concise pour comprendre l'autre et se faire comprendre.
Pour conclure, je suis très fier d'avoir pu réaliser de A à Z ce projet d'algorithmique et de programmation en seulement un semestre avec Hakim.

Hako:

Personnellement, ce projet a été une étape marquante dans mon apprentissage. Lorsque nous avons commencé, je n'avais pas de compétences particulières en C et j'avais l'impression de partir de zéro. En tant qu'étudiant en reprise d'études, cela faisait très longtemps que je n'avais pas codé, et ce projet m'a vraiment donné l'opportunité de rattraper mes lacunes. J'ai l'impression d'avoir réalisé mon premier vrai projet en programmation, ce qui a été une expérience très enrichissante.

Aujourd'hui, on est capable de créer un programme qui gère plusieurs bases de données interconnectées.

Je vois bien le potentiel que ça représente. C'est motivant et concret .