BEN ALI Adnane
CORNUEJOULS Damien
L2 CUPGE 2021-2022

Projet BAC - Book a classroom

Rapport

Dans le cadre de l'unité d'enseignement "Structure de données", nous avons dû réaliser un programme en langage C en binôme. L'objectif de ce projet consiste en un gestionnaire d'emploi du temps de type "scolaire" : réservation d'un créneau horaire, gestion d'une liste de profs, de formation, de salles,... Nous évoquerons dans un premier temps la structuration du projet : comment nous avons prévu d'organiser le programme, les interrogations qui nous sont venues et les connaissances que nous jugions manquantes mais nécessaires à ce projet. Dans un second temps, nous verrons les difficultés que nous avons rencontrées : la confrontation de nos idées, la planification du travail ainsi que l'implémentation de ressources externes. Enfin, nous verrons les difficultés restantes ainsi que des possibles voies d'amélioration.

I - Structuration du projet

a - Organisation du programme

La toute première étape fût d'imaginer le résultat final de notre travail. Nous l'avons imaginé comme suit :

- Les utilisateurs se découpent en 2 groupes : les clients et les administrateurs.
- Les clients ont uniquement un rôle de consultation : ils peuvent visualiser les emplois du temps de n'importe quel prof, salle ou formations sans pouvoir les modifier.
- Les administrateurs ont accès à un environnement, protégé par un mot de passe chiffré, qui leur permet de gérer les emplois du temps de n'importe quelle salle, prof, ou formations. Ils peuvent également gérer la liste des salles, profs et formations disponibles.
- Pour ne pas perdre toute modification effectuée, l'ensemble des données (mot de passe chiffré, liste des emplois du temps, listes des profs, des formations, des salles,...) sera stocké dans des fichiers JSON qui seront lus au début du programme et remplacés à la fin.

Au sein du programme, les données sont représentées par des structures : s_cours, s_prof,, s_formation, s_salle et s_UE. Pour stocker l'ensemble des emplois du temps, profs,

formations ,UE par formations et salles, Nous avons décidé d'utiliser des listes chainées, représentées par d'autres structures.
Nous avions également penser à des fonctionnalités additionnelles telles que :

- Un système de remplissage automatique des emplois du temps
- Un système fonctionnant avec plusieurs types d'utilisateurs : profs, étudiants, admin,... où chacun n'aurait accès qu'aux données le concernant.
- Une version graphique

Cependant, nous nous sommes questionnés sur la faisabilité et l'utilité de chacune de ces fonctionnalités, ce qui nous amène aux interrogations qui nous sont venues lors de notre travail.

b - Les interrogations qui nous sont venues

Nous nous sommes d'abord demandés s'il était important de considérer chaque semaine indépendamment ou si nous décidions que chaque semaine serait identiques aux autres. Pour y résoudre, nous nous sommes dit que nous cherchions à faire un gestionnaire d'emploi du temps dans un cadre scolaire, où les semaines se suivent et se ressemblent. Nous nous sommes donc dit que différencier chaque semaine n'était pas forcément pertinent.
Une autre interrogation qui nous est venue concerne l'implémentation de différents types d'utilisateurs "clients" (prof, étudiants,...). Nous nous sommes dit que cette fonctionnalité compliquerait notre travail sans pour autant être d'une grande utilité. Nous avons donc pris la décision de mettre de côté cette idée.

c - Les connaissances que nous devions acquérir

Les contraintes qui nous étaient imposées contenaient l'utilisation du format de fichier JSON (JavaScript Object Notation) pour stocker les données après l'exécution du programme. Cependant, ce format nous était jusqu'alors inconnu et il nous a fallu nous documenter et expérimenter pour mieux l'appréhender. Une autre contrainte nous imposait l'utilisation d'une mot de passe chiffré pour les administrateurs. Le chiffrement est un domaine dans lequel nous n'avions aucune compétence, ce qui a donc représenté une quantité d'informations supplémentaires à comprendre avant de l'implémenter. Cependant, ce ne fût pas si simple , ce qui nous amène à notre deuxième partie : les difficultés rencontrées.

II - Difficultés rencontrées

a - Confrontations de nos idées

Un point sur lequel nous n'étions pas d'accord fût la manière de stocker les données dans la mémoire. Les 2 idées opposées étaient les suivantes :

- Une liste doublement chaînée non circulaire
- Une liste doublement chaînée circulaire avec une sentinelle

Une liste doublement chaînée circulaire permet de mieux se déplacer dans la liste, notamment de revenir au début une fois arrivé à la fin et vice-versa. Cependant, une liste doublement chaînée non circulaire permet aussi de se déplacer dans la liste, en utilisant

toutefois un pointeur supplémentaire menant vers la fin de la liste. La liste circulaire permet de se déplacer plus fluidement mais nécessite une complexité accrue du code sans pour autant avoir des atouts indispensables. Notre choix s'est donc tourné vers la liste chaînée circulaire.
Globalement, le travail en groupe nous a inévitablement amenés à des différents sur diverses questions, ce qui nous a permis de confronter nos points de vues et d'en tirer le meilleur pour mener à bien notre projet.

b - Planification et organisation du travail

Nous avions déjà participé à un projet à deux, par conséquent nous connaissions déjà les horaires de chacun. Pour se partager le travail, ce que l'on faisait de prime abord, c'était qu'ensemble on réfléchissent à une sorte de "plan d'action" à court terme. Par exemple, pour faire le calendrier.c, on préparait ensemble le calendrier.h pour savoir quelles fonctions coder. Après ça, on se répartissait les fonctions de manière égale et on se donnait une date pour finir d'écrire ces fonctions. Généralement on finissait le travail dans la même journée ou bien le jour suivant.
Et c'est là que l'on a remarqué que la partie de codage, n'est pas ce qu'il y avait de plus dur mais c'était surtout au niveau de la planification. Dès que l'on planifie, on sait où aller et cela rend les choses beaucoup plus simples. Néanmoins, une de nos plus grosses erreurs lors de ce projet à était de sous-estimer cette partie de notre travail.
Planifier à l'avance nous aurait donné un plan d'actions efficace. Nous aurions pu savoir si nous étions en retard et donc essayer de négocier un temps supplémentaire avec notre professeur, ou bien de modifier notre plan d'action, nous aurions pu nous concentrer sur des éléments plus compliqués de notre projet comme l'implémentation de librairies externes si nous étions en avance.
On avait parfois l'impression de se diriger à l'aveugle. Malgré tout, bien que l'on avait aucune difficulté à construire et mener à bien les plans court terme, il nous manquait la finalité, l'objectif recherché.
En finalité, un outil comme le diagramme de gantt nous aurait permis d'éviter de finir en retard, de prévoir certaines difficultés et avoir une meilleur vue d'ensemble de ce projet.

III - Difficultés restantes et voies d'amélioration

a - L'implémentation de librairies externes

L'une des difficultés majeures fût l'implémentation du parsing JSON. En effet, JSON n'est pas pris en charge nativement en C, nous avons donc dû chercher et choisir une librairie qui soit à la fois simple à implémenter et suffisante pour notre projet. Nous avons retenu les librairies suivantes : JSON-c ; cJSON et Parson. cJSON et Parson sont des librairies légères (un .c et un header chacune) tandis que JSON-c est très fournie et nous laisse une grande liberté dans notre manière d'aborder le parsing JSON. Nous nous sommes donc tournés vers cette dernière. Cependant, la documentation de chacune de ces librairies n'est pas évidente à comprendre pour des novices et nous avons conscience que cette partie de notre programme n'est pas la plus aboutie et reste une voie d'amélioration.

Une autre difficulté principale fût l'ajout de la gestion d'un mot de passe chiffré. Pour notre programme, nous avons cherché des librairies qui soient facilement implémentables et éprouvées, c'est-à-dire existants depuis suffisamment longtemps pour avoir été testées et approuvées de très nombreuses fois. Dans un premier temps, nous nous sommes tournés vers les librairies OpenSSL, qui correspondent à nos critères quant à la robustesse du programme. Cependant, OpenSSL étant très complet, cela le rend difficile à installer et à exploiter. De plus, un seul algorithme nous suffisait pour ce que nous voulions faire. Nous nous sommes donc tournés vers la librairie "crypto-algorithms" : GitHub - B-Con/crypto-algorithms: Basic implementations of standard cryptography algorithms, like AES and SHA-1. Qui a pour avantages d'être très simple à implémenter (un .c et un header) ainsi que d'être pensée pour l'éducation ce qui la rend facilement compréhensible pour des novices.

b - Possibilités d'amélioration

Pour notre programme, nous avons pensé tout au long de notre processus de travail à des fonctionnalités qui ne semblent pas essentielles mais qui peuvent représenter des améliorations à notre programme. Parmi elles :

- Un algorithme de remplissage automatique. Cette idée nous a été déconseillée par notre professeur, en raison de la complexité d'un tel algorithme.
- Une version graphique.
- Une gestion semaine par semaine
- Des contraintes supplémentaires : la taille et l'équipement d'une salle, les disponibilités de chaque professeur,...
- Une version prenant en charge des utilisateurs clients de différents types.

Conclusion

Nous avons donc parlé de la structuration de notre projet, des difficultés rencontrées, ainsi que des difficultés restantes et des améliorations que l'on pourrait apporter.

Pour conclure, ce projet nous a permis de mesurer les véritables difficultés auxquelles nous pouvons être confrontés dans le futur lors de travail de groupes.