



23 NOVEMBRE 2022

Document présenté à  
Jean-Christophe Demers


Dans le cadre du cours  
420-C61-IN Projet Synthèse

# ESCAPE ROOM MANAGER

RAPPORT DE MI-MANDAT 420-C61-IN

Belony Carlens, Guindon Maxence

23 novembre 2022



## Fonctionnalités

Maxence Guindon

### Liste double chaînée

Pour le projet, nous avons décidé de réaliser une liste double chaînée. Au moment de la présentation de mi-mandat, celle-ci était presque achevée. Il restait à implémenter la fonction *reverse* et une fonction pour aller sélectionner un item dans la liste directement. Autrement, il était possible d'insérer ou de retirer des éléments à l'intérieur de la liste ainsi que d'itérer à travers celle-ci.

Le plus gros défi pour la création de la liste a été la recherche et l'agencement des différentes sources d'information pour créer une liste plus complète en utilisant notamment les fonctions *dunder* de python. En tout, il y a au moins une demi-douzaine de sources d'information différentes qui m'ont aidé à construire la liste double chaînée et la plupart d'entre elles expliquent comment redéfinir une fonction *dunder*.

Pour l'instant, la classe de la liste n'est pas encore appelée dans le code, mais éventuellement, elle servira lorsque nous implémenterons une logique pour organiser les réservations et les envoyer de la base de données à la vue notamment. Autrement, la classe est dans le fichier `DoubleLinkedList.py`. Son chemin absolu est le suivant : `EscapeRoomManager\Dev\App\Modele\DoubleLinkedList.py`.

### Algorithme *BackTracking* et *ForwardsChecking*

L'une des grosses fonctionnalités à implémenter et qui sera très importante pour la suite du projet est le design pattern *Strategy* qui nous permettra d'encapsuler plusieurs algorithmes différents de création d'horaires. Les deux plus importants à implémenter dans l'ordre pour le projet sont les algorithmes de *BackTracking* et de *ForwardsChecking*. Si nous avons encore un peu de temps, nous pourrions tenter d'implémenter l'algorithme *MinConflict*, mais comme il est efficace pour de grosses simulations, et un peu moins avec des petites, il n'est pas essentiel au projet.

Le défi le plus significatif au niveau de ce développement se situe au niveau des algorithmes eux-mêmes. Après en avoir parlé avec Frédéric Thériault, il nous a assurés que même s'il y avait peu de ligne de code, c'est la compréhension en elle-même des algorithmes qui est plus complexe.

### Carlens Belony

#### Middleware en architecture orienté-objet

Le middleware est ce qui communique entre la Vue, le modèle et la base de données. Au moment de la présentation, elle communiquait que très peu dans la base de données, mais maintenant il permet de prendre les informations de celle-ci et mettre à jour les informations reçus dans la base de données dans les objets du modèle qui seront utilisés pour modifier la vue. Pour insérer une donnée, le `middleWare` l'insère tout d'abord dans la BD et lorsque la confirmation est reçue, on l'insère aussi dans le modèle.

Le plus gros défi a été de trouver, puis comprendre et adapter une librairie (de middleware) toute nouvelle à une *autre* librairie toute nouvelle (React). On avait déjà essayé d'implémenter un middleware avec Redux mais le nombre de bugs et le nombre de lignes de codes nécessaires pour faire une action le rendait trop lourd; on a dû changer de librairie (Mobx) et changer d'architecture plusieurs fois dans celle-

ci pour finalement avoir quelque chose qui fonctionne, puis remodeler l'architecture en format MVC une fois que notre compréhension de la librairie s'est affinée avec le temps.

Le middleware le plus complet et structuré à ce jour est le plus gros et le plus important des 2 middlewares développés, l'*AccueilStore* que l'on peut trouver dans le chemin suivant :

`EscapeRoomManager/dev/FrontEnd/src/Middlewares/AccueilStore.ts`

### Affichage d'un graphique en React

Le prochain gros défis pour moi sera d'implémenter un graphique montrant les salles les plus performantes en React lié à mon middleware. Puisque je n'ai jamais fait de graphiques en Web, et encore moins en React, et encore moins avec des données d'un middleware, ce sera encore une fois un terrain nouveau pour moi.

### Avancement total du projet

Nous estimons que nous avons accompli environ 60 % de tout ce que nous voulions accomplir à ce point-ci. Même si nous avons avancé beaucoup sur plusieurs fronts, il reste encore beaucoup de choses à accomplir avant que nous ayons un projet complet et utilisable.