



Haute École  
Libre de Bruxelles

# 2021-2022 Rapport Projet HELBPark 2.0

Al Haddaoui Marwane

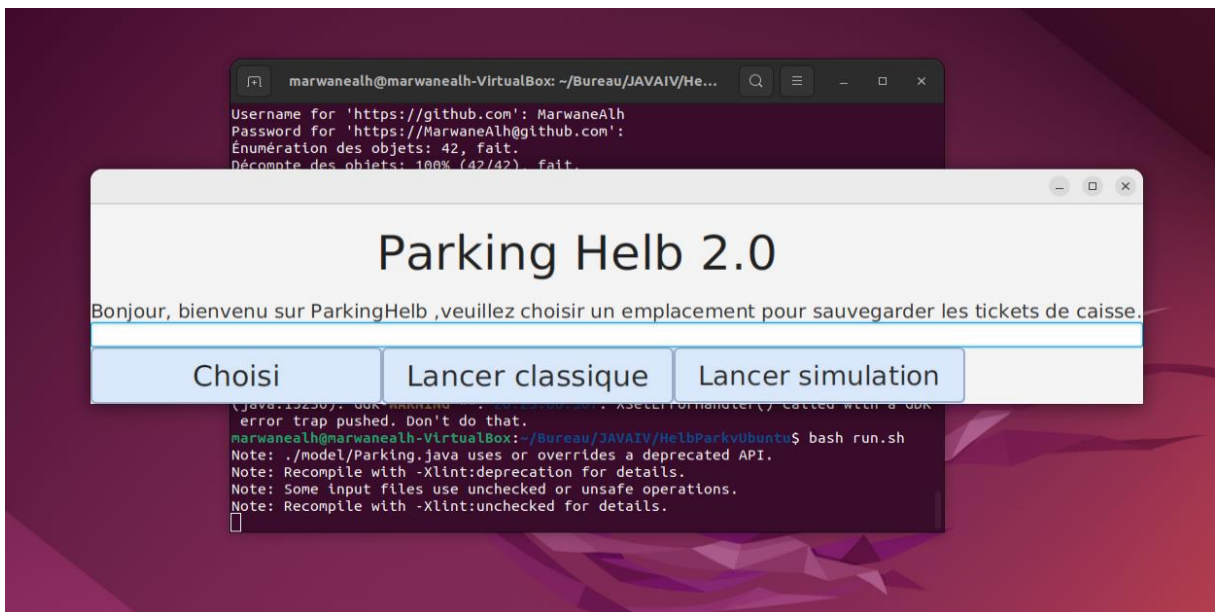
## Introduction :

Dans le cadre du cours de java IV, un projet nous a été demandé d'être implémenté, le projet a pour but de gérer le parking de la société HELBPARK, accompagné d'un certain nombre de fonctionnalité, son implémentation sera vue de manière détaillée au sein de ce rapport.

## Présentation de l'interface graphique :

Ayant opté pour une application dynamique qui change la contenu de la fenêtre affichée en fonction du besoin utilisé, l'application est composé de 3 interfaces graphiques qui n'existent pas de façon simultanée, chacune est appelée par rapport à un besoin spécifique.

### 1) Écran de lancement



## Explication :

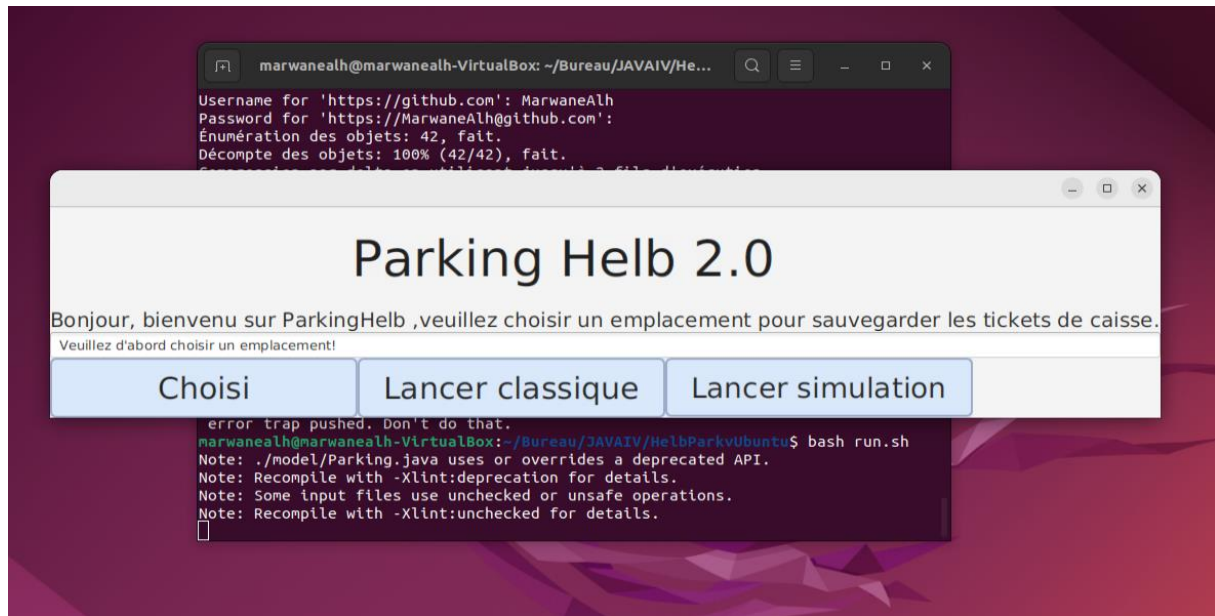
Au lancement de l'application une fenêtre affichant trois boutons ainsi que le titre et un message de bienvenue se lance, l'un de cet écran réside dans le choix de l'emplacement de stockage pour le dossier contenant les tickets à générer cette solution permet d'avoir un chemin toujours valide qui s'adapte en fonction de la machine ainsi que de l'utilisateur et évite d'avoir des problèmes liés à un changement de machine.

Au lancement de l'application une fenêtre affichant trois boutons ainsi que le titre et un message de bienvenue se lance, l'un de cet écran réside dans le choix de l'emplacement de stockage pour le dossier contenant les tickets à générer cette solution permet d'avoir un chemin toujours valide qui s'adapte en fonction de la machine ainsi que de l'utilisateur et évite d'avoir des problèmes liés à un changement de machine.

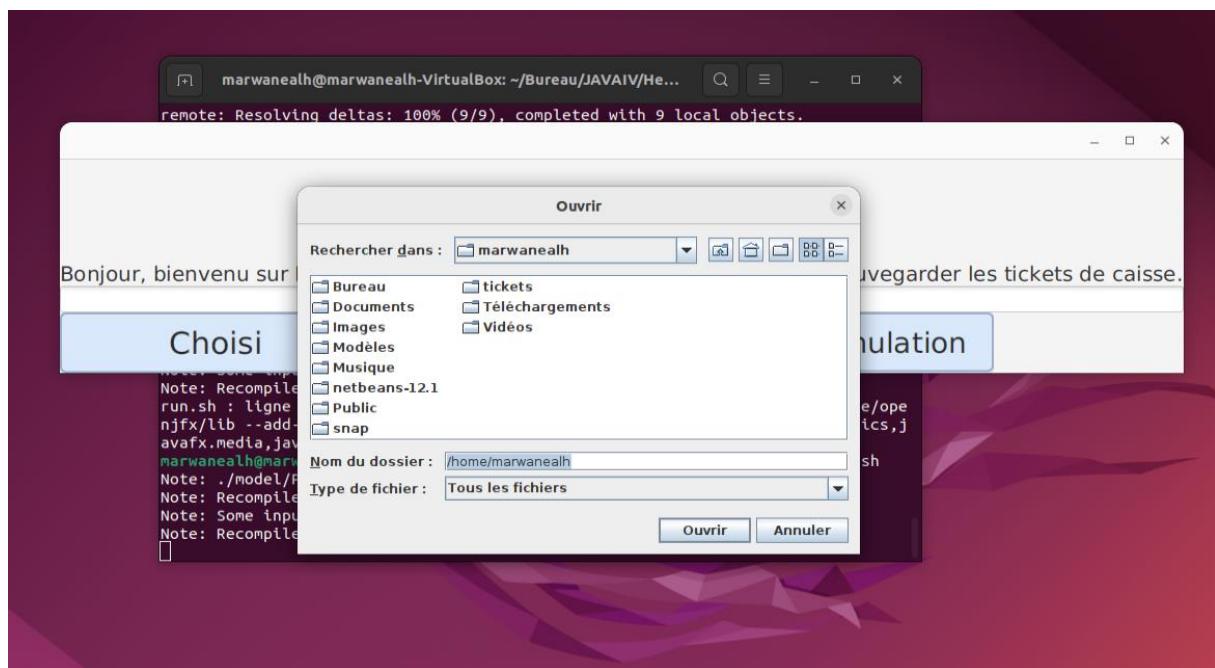
Note : si aucun chemin pour l'emplacement du dossier tickets n'est choisi un message est affiché dans le textfield et les boutons sont obsolètes.

## Démonstration :

### Cas : Pas choisi d'emplacement



### Cas clique bouton choisi :



## 2) Écran de parking



### Explication :

Suite au clique sur un des deux boutons de lancement de l'écran précédent, un écran affichant le parking est généré, le style ainsi que le code couleur de l'application est fortement inspiré par le document d'explication de projet et donc un affichage similaire est généré.

La grille qui est affichée est en fait un ensemble de boutons cliquables qui permet d'ouvrir une fenêtre d'édition que nous verrons par la suite.

## 1) Écran d'Édition

### Explication :

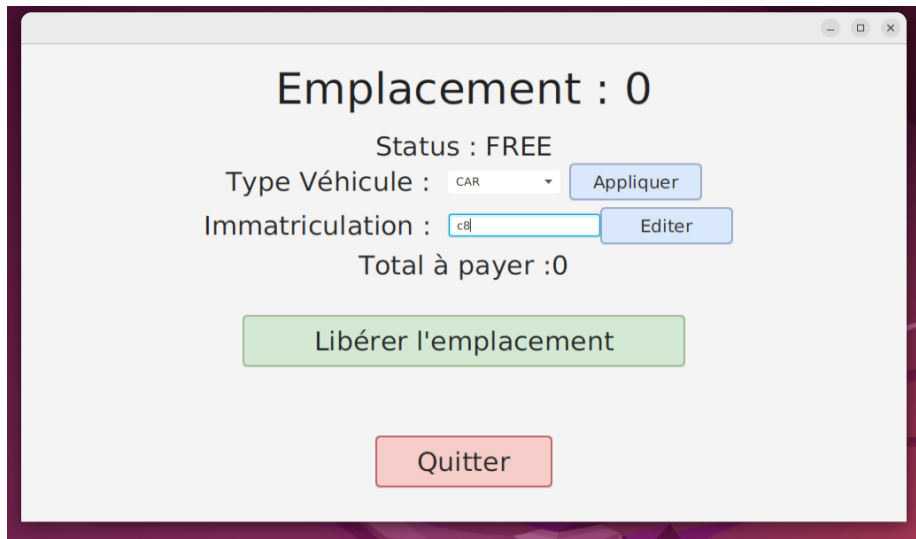
Après avoir cliqué sur un des emplacements du parking, la fenêtre change et une fenêtre d'édition s'affiche.

Sur cette fenêtre d'édition, les valeurs de la place de parking sont affichées, son statut correspond à si oui ou non elle est occupée, le type du véhicule est affiché ainsi que la plaque d'immatriculation sans oublier le prix de la place.

Comme vous pouvez le remarquer les champs de texte sont désactivés afin de pouvoir les modifier l'utilisateur doit cliquer sur les boutons "édités" ou "changer" qui active la modification du champ de texte, les boutons libérer l'emplacement ainsi que quitter permette chacun une action, le bouton libérer l'emplacement permet de libérer la place, il redirige la page vers l'affichage du parking sans devoir appuyer sur le bouton quitter quant au bouton quitter il permet de quitter la fenêtre ainsi que de sauvegarder les changements effectués.

### Démonstration :

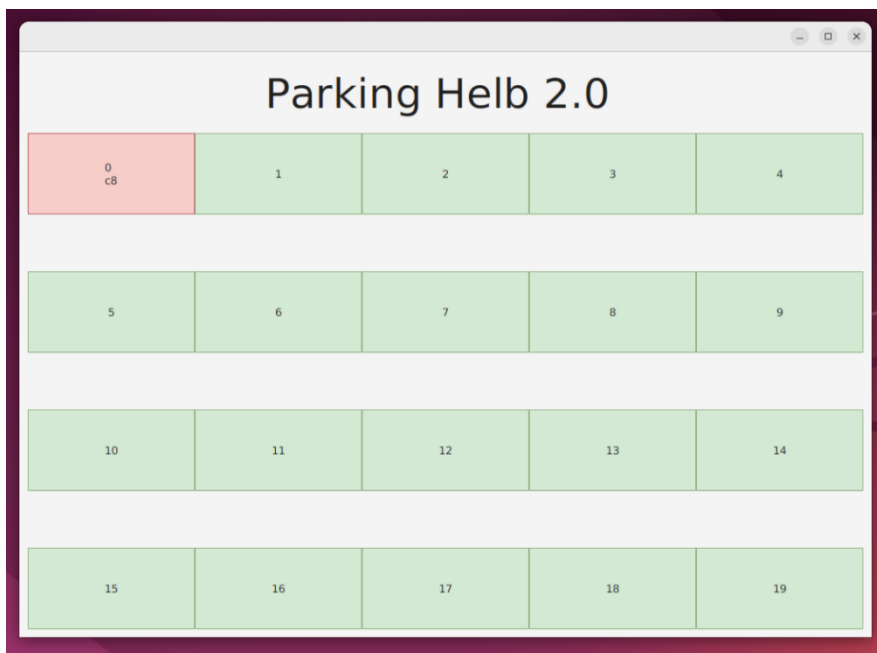
#### Cas appuie sur les boutons éditer et changer :



The screenshot shows a window titled "Emplacement : 0". It displays the following information and controls:

- Status : FREE
- Type Véhicule : CAR (with a dropdown arrow)
- Immatriculation : c8 (with a text input field)
- Total à payer : 0
- Buttons: "Appliquer" (next to Type Véhicule), "Editer" (next to Immatriculation), "Libérer l'emplacement" (large green button), and "Quitter" (red button).

#### Cas clique ensuite sur bouton quitter :



The screenshot shows a window titled "Parking Helb 2.0". It displays a grid of 20 parking spots, numbered 0 to 19. Spot 0 is highlighted in red and contains the text "0 c8". The other spots (1-19) are green and empty.

0 c8	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19

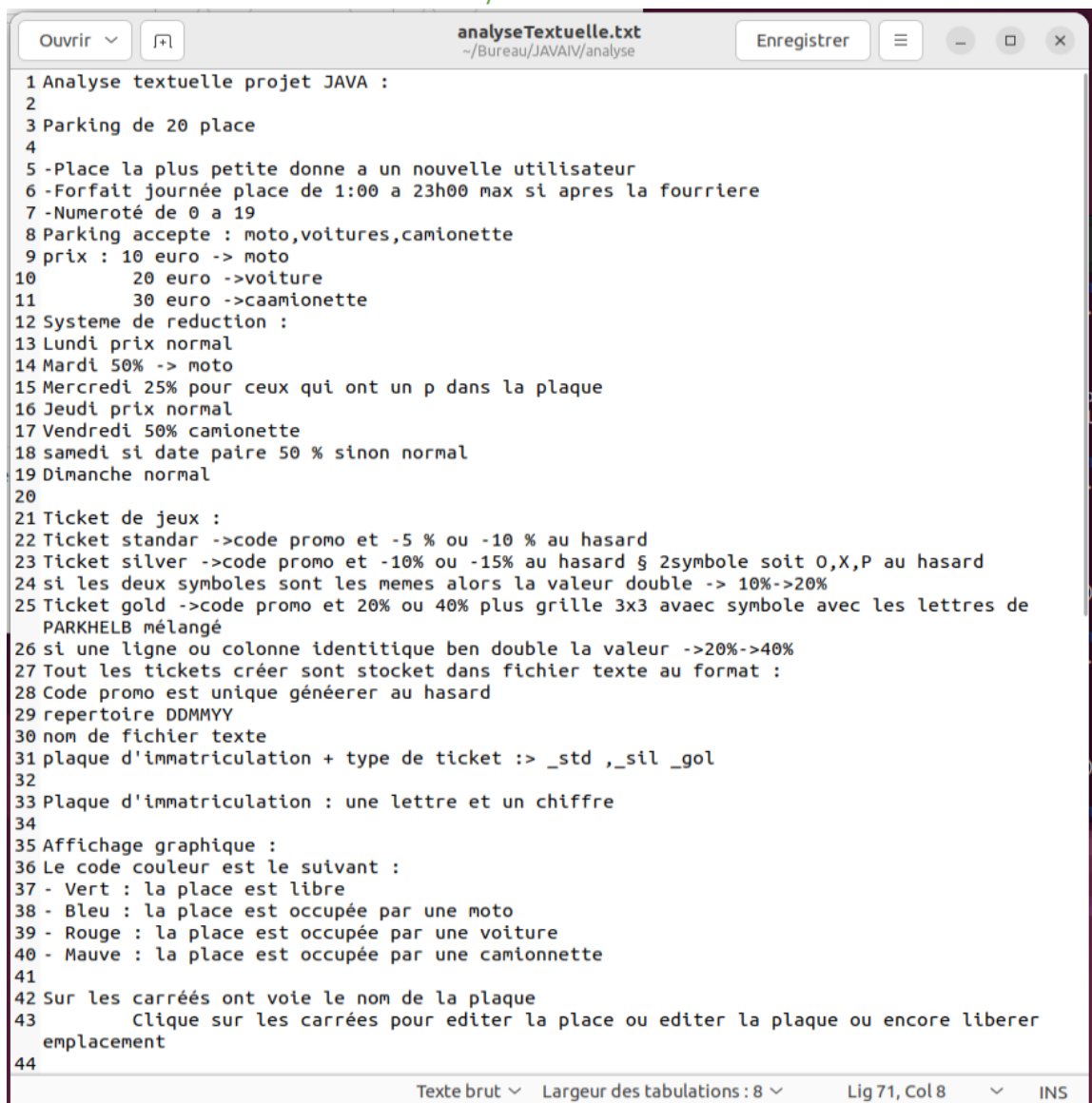
## Analyse et Application des Design Patterns :

Explicité par le titre de la section, dans cette section nous verrons la partie analyse ainsi que l'utilisation des designs patterns au sein du projet.

### a) Analyse :

La partie analyse du projet s'est faite en deux étapes, la première consiste à extraire du document d'indication de projet, les besoins du projet, sous forme d'analyse textuelle ce qui va permettre d'identifier les besoins du projet ainsi que de permettre par la suite une analyse conceptuelle qui va poser de manière plus précis les bases du projet par rapport à son implémentation.

#### Analyse textuelle :

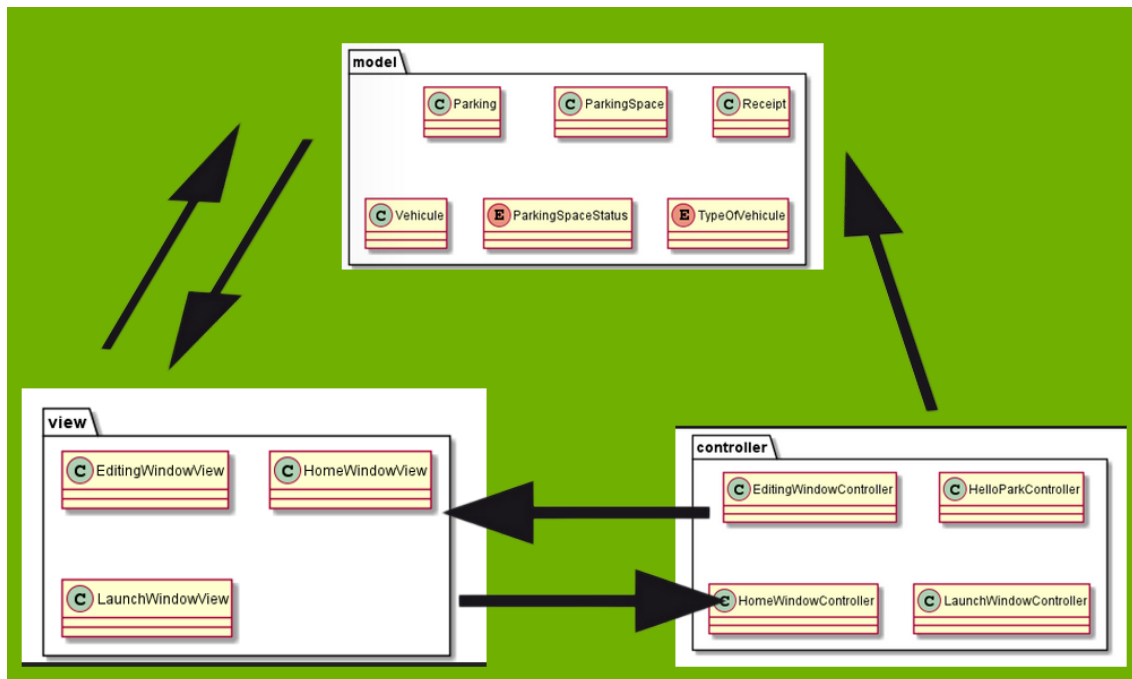


```
1 Analyse textuelle projet JAVA :
2
3 Parking de 20 place
4
5 -Place la plus petite donne a un nouvelle utilisateur
6 -Forfait journée place de 1:00 a 23h00 max si apres la fourriere
7 -Numeroté de 0 a 19
8 Parking accepte : moto,voitures,camionette
9 prix : 10 euro -> moto
10         20 euro ->voiture
11         30 euro ->caamionette
12 Systeme de reduction :
13 Lundi prix normal
14 Mardi 50% -> moto
15 Mercredi 25% pour ceux qui ont un p dans la plaque
16 Jeudi prix normal
17 Vendredi 50% camionette
18 samedi si date paire 50 % sinon normal
19 Dimanche normal
20
21 Ticket de jeux :
22 Ticket standar ->code promo et -5 % ou -10 % au hasard
23 Ticket silver ->code promo et -10% ou -15% au hasard § 2symbole soit O,X,P au hasard
24 si les deux symboles sont les memes alors la valeur double -> 10%->20%
25 Ticket gold ->code promo et 20% ou 40% plus grille 3x3 avaec symbole avec les lettres de
  PARKHELB mélangé
26 si une ligne ou colonne identitique ben double la valeur ->20%->40%
27 Tout les tickets créer sont stocket dans fichier texte au format :
28 Code promo est unique générer au hasard
29 repertoire DDMMYY
30 nom de fichier texte
31 plaque d'immatriculation + type de ticket :> _std ,_sil _gol
32
33 Plaque d'immatriculation : une lettre et un chiffre
34
35 Affichage graphique :
36 Le code couleur est le suivant :
37 - Vert : la place est libre
38 - Bleu : la place est occupée par une moto
39 - Rouge : la place est occupée par une voiture
40 - Mauve : la place est occupée par une camionnette
41
42 Sur les carrées ont voie le nom de la plaque
43 Cliquez sur les carrées pour editer la place ou editer la plaque ou encore liberer
  emplacement
44
```

Note :

L'analyse textuelle dans sa globalité est disponible au lien suivant : lien [git](#).

### Analyse Conceptuelle :



### Explication :

Le diagramme ci-dessus est un diagramme de package, afin de démontrer les liens étroits entre les composants du projet, le diagramme de package est la solution la plus adéquate on peut y voir ici toutes les classes des packages respectifs ainsi que leurs liens à l'aide des flèches, le package-modèle communique avec la view et la réciproque est valable, le package contrôler gère le package view ainsi que les interactions de la view et pour finir le package controller disposent des changements du package modèle.

NOTE : Symbole « E » correspond à une énumération et « C » à une class.

### b) Design Patterns :

#### - MVC :

Comme remarqué dans la section précédente, l'un des patrons utilisés est le patron mvc, celui-ci est en adéquation avec les besoins du projet et ma visualisation de l'implémentation du projet, il assure une certaine division qui correspond avec la division du projet, (ex : la partie interface graphique est dans la partie vue), il assure une maintenabilité plus simple, car chaque composant du projet dans package adéquat...

#### - Créateur :

Ici le Patern créateur a été utilisé pour la partie parking, étant donné un contenant parking qui contient des objets de types parking Place (place de parking), en suivant ce Patern le contenant crée sa contenue et donc l'objet parking créer ses composants.

#### - Le pattern Expert en Information :

Ce pattern à trouver son utilisation afin de répondre au besoin de savoir identifier une place de parking, en effet dans la logique d'utilisation, l' d'une place de parking et une tâche triviale cependant son implémentation requiert de savoir ou implémenter cette fonctionnalité et en suivant le Pattern Expert en information, le contenant parking possède des méthodes permettant d'identifier une place de parking.

#### Réflexion :

Au début du projet, un questionnement quant à l'utilisation de l'observer Observable s'est posé, seulement dans la logique de programmation du projet actuel ce Pattern n'avait pas sa place, car chaque fenêtre est régénérée à son lancement et elle ne s'affiche jamais en simultané

Exemple : le gérant du parking clique sur une place de parking, la fenêtre du parking ferme et la fenêtre édition s'ouvre quand il quitte cette fenêtre la fenêtre du parking s'ouvre à nouveau avec les changements, dû à la modification de l'objet parking.

#### Limitation :

L'absence de test Unitaire, malgré un fichier de simulation qui a pour but de simuler un test, représente une zone d'ombre dans le projet, certaines situations pourraient générer un cas inattendu, malgré une simulation d'un plan de test par scénario certaines situations peuvent avoir échappé à la robustesse du code.

#### Conclusion :

Pour conclure, la majorité des fonctionnalités demandées ont été implémenter tout en respectant les contraintes de Pattern et certains principes de programmation malgré un manque de temps évident dû à la charge de travail général, la réalisation de ce projet m'a permis de mettre des mots voire concept sur certaines pratiques de programmation.