**PTS Info: simulateur de parking**

# **Spécifications Fonctionnelles - Simulateur de Parking**



## 

DESJOUIS Louis - SINTES Alexandre - Ethvignot Alexis

[**Spécifications Fonctionnelles - Simulateur de Parking**](#_a91n2h9668z) **1**

[1. Description de l'interface](#_45c2xc17vmug) 2

[2. Fonctionnement de l’interface](#_up2wy5ihf6q9) 3

[Fonctionnalités obligatoires :](#_r4drtqacjqml) 3

[Fonctionnalités Utiles](#_kdvfs6efzg4) 3

[Fonctionnalités Optionnelles](#_qx13rpgye7x6) 3

[3. Modélisation](#_qrkiyuy6oaog) 4

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **1. Description de l'interface**

L’objectif de ce projet est de développer un simulateur de parking qui permet de gérer l’entrée et la sortie des véhicules dans un parking simulé. Ce simulateur vise à faciliter la gestion du stationnement en temps réel, en affichant les places disponibles et en gérant les interactions des utilisateurs avec une entrée, une sortie et des véhicules.

Le simulateur s’adresse principalement à des utilisateurs souhaitant simuler ou tester des scénarios de gestion de parkings, comme les propriétaires de parkings ou les gestionnaires d’espaces de stationnement publics ou privés. L'interface est conçue pour fonctionner en local, avec une interface simple et intuitive, permettant une prise en main rapide par tout type d’utilisateur. Il y aura une version console et une interface graphique.

L’objectif est d’optimiser la gestion des places disponibles dans un parking, en simulant divers comportements des véhicules avec des compteurs et en affichant en temps réel les informations nécessaires à l’utilisateur comme le nombre de places occupées et disponibles, il y aura aussi des led sur chaque place permettant d’indiquer si elle est libre ou non.

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **2. Fonctionnement de l’interface**

### **Fonctionnalités obligatoires :**

1. **Gestion des places de parking** : L'interface doit afficher en temps réel le nombre de places disponibles et occupées.
2. **Entrée et sortie des véhicules** : L’utilisateur peut simuler l’entrée ou la sortie d’un véhicule, ce qui met à jour automatiquement le nombre de places disponibles.
3. **Affichage du statut des véhicules** : Les véhicules présents dans le parking doivent être visibles sous forme de liste ou de représentation graphique, avec des informations telles que le temps de stationnement.

### 

### **Fonctionnalités Utiles**

1. **Réservation de place** : L'utilisateur peut réserver une place de parking à l’avance, et cette place sera marquée comme "réservée" jusqu’à ce que le véhicule arrive.
2. **Leds sur chaque place:**

* verte = place libre
* rouge = place occupée
* 

1. **Parking complet** : Si le parking est complet, il y aura marqué complet, personne ne pourra entrer dans le parking.
2. **Gestion des types de véhicules** : Différencier les places de parking en fonction des types de véhicules (voitures électriques, motos, camions; voitures uxe), ajoutant ainsi une profondeur au simulateur.14
3. **Système de priorités** : Permettre aux véhicules d’urgence ou aux clients privilégiés (ex: clients réguliers, handicapés et ambul) d'accéder à certaines places réservées, simulant ainsi des parkings à priorités multiples.

### **Fonctionnalités Optionnelles**

1. **Statistiques détaillées** : L'interface pourrait fournir des statistiques détaillées sur l’utilisation du parking, comme le temps moyen de stationnement ou les périodes de forte affluence.
2. **Système de paiement simulé** : ajouter une simulation de gestion des paiements, où les utilisateurs "paient" pour leur stationnement avec des tarifs horaires ou forfaitaires simulés.
3. **Gestion des abonnés** : Intégrer un système de gestion des abonnements pour les utilisateurs réguliers du parking, simulant ainsi un parking avec une clientèle fixe.

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **3. Modélisation**

Un diagramme de classes (UML) pour illustrer les interactions entre les différents objets de l’interface (Parking, Véhicule, Place, etc.).

schema indispensable