Mardi 17 décembre 2019

A picture containing object

Description automatically generated

Développement informatique avancé : Orienté Applications

-

Projet JAVA : Réalisation d’une application utilitaire

*« Application de gestion des parkings étudiants »*

Groupe 6

DE LEENER Nathan – GORTZ Gaëtan – LUK Brian

I) *Cahier des charges*

1. Contexte du projet :

Dans le cadre du cours « Développement informatique avancé :application (théorique) », les élèves de TI doivent créer une application web qui permet aux étudiants de l’Ephec de pouvoir réserver une place de parking durant un certain temps.

1. Description du projet :

L’application doit être capable de gérer le parking à disposition des étudiants. Chaque utilisateur doit créer un compte afin de pouvoir utiliser l’application et réserver une place de parking (nom utilisateur, mot de passe, plaque du véhicule, email, …etc.). Les étudiants auront aussi la possibilité de modifier les informations de leur voiture, de réserver une place pour une période déterminée et de signaler les utilisateurs qui enfreignent les règles (système de pénalité).

1. Objectifs du projet :

L’objectif est, pour les étudiants de l’Ephec, de pouvoir stationner leur véhicule en toute sécurité et de pouvoir trouver une place sans que les autres usagers entravent la circulation.

1. Contraintes techniques :
   * Développement en Java
   * MVC
   * L’interface de programmation JDBC
2. Besoins fonctionnels :

En tant qu’utilisateur, je souhaite avoir une interface graphique simple et intuitive permettant de réserver facilement les places disponibles. Une fois que la place a été choisi l’application envoie les données dans la base de données qui se met à jour et réserve la place. Il est aussi possible que je signale un utilisateur mal garé.

A screenshot of a cell phone

Description automatically generated*II) Diagramme UML*

*III) Nos choix d’implémentation*

La réalisation de notre application utilitaire permettant de gérer les parkings étudiants, nous avons décidé d’utiliser une base de données. Cela nous semble être le choix le plus judicieux pour réaliser notre application, car pour mettre à jour nos données nous pouvons utilisé l’API « Java.SQL ».

Pour la mise en place de notre base de données, à défaut d’avoir accès à un serveur distant, nous avons utilisé Xampp. Ce qui nous a permis de créer un serveur local sur nos machines et ainsi de développer sans encombres notre application sans pour autant devoir avoir un serveur physique.

Ensuite, nous avons mis en place et développé notre base de données à l’aide de phpMyAdmin, ce qui nous a permis de coder cette dernière en y intégrant des donnée de bases, indispensable pour effectuer nos tests.

Concernant l’interface graphique, nous avons utilisé pour le layout « GridLayout » ce qui nous semble idéal pour réaliser un programme avec une fenêtre de connexion.

*IV) Les difficultés que nous avons rencontrées*

Lors de la réalisation de ce projet, nous avons rencontré plusieurs difficultés…

Tout d’abord, nous avons éprouvé des difficultés à mettre en place le projet en se servant correctement de l’outil GitHub. Il a fallu quelques *commits* de test et une concertation d’équipe pour se lancer correctement dans le projet.

Ensuite, nous avons eu besoin de pas mal de concertation afin de se répartir au mieux les tâches à réalisées. Notamment, lorsqu’il s’agissait de savoir qui allait réaliser quelle partie du code. En effet, certaines parties étant liées à d’autres, il fallait que chacun puisse avancé sans dépendre des autres. Pour cela nous devions définir les lignes directrices de notre code, comme la manière de nommer nos variables, les conventions de codages que nous utilisions et bien entendu la structure de nos classes.

Pour cette étape, le diagramme UML nous a bien servi et nous a permis d’avoir une bonne vision globale.

Par la suite, nous avons rencontré certaines difficultés par rapport à l’élaboration de l’interface utilisateur, tant pour l’interface graphique que pour l’interface console. Notre problème étant que les deux interfaces étaient dépendantes l’une de l’autre. C’est-à-dire que nous tentions de faire fonctionner les deux en parallèle, ce qui nous retournait des erreurs lors de l’exécution. Nous avons compris finalement notre erreur et avons séparé les deux.

*V) Les pistes d’amélioration*

Nous avons de nombreuses pistes concernant les améliorations possibles à apporter à notre application.

Premièrement, nous pourrions améliorer l’interface graphique en la rendant plus attractive visuellement. Notamment en y ajoutant des image/icones pour les boutons, des listes déroulantes pour les parkings à réserver, des boutons supplémentaires pour effectuer des retours ou des déconnections.

Deuxièmement, nous pourrions créer une accès administrateur afin de rajouter, supprimer ou modifier les parkings, places et utilisateurs sans devoir nécessairement modifier la base de donnée manuellement. Nous avons déjà créé certaines méthodes dans le code permettant d’agir en tant qu’administrateur.

Troisièmement, nous pourrions mettre la base de donnée dans un serveur distant afin que celle-ci soit accessible par quiconque souhaite utiliser l’application.

Quatrièmement, il faudrait réécrire le code de manière plus propre, claire et lisible en essayant de garder une réelle uniformisation dans la manière de codée entre les différentes classes. Il faudrait aussi revoir sa structure afin que les différents composants du code soient plus structurés.

Cinquièmement, les messages d’erreur -par exemple lorsque l’on a trop d’infractions- n’apparait pour le moment que dans l’interface console et il serait judicieux de faire apparaître ceux-ci également dans l’interface graphique. De plus lorsqu’une entrée n’est pas valide, nous sommes pour le moment obligé de relancer le programme. Il faudrait plutôt afficher un message d’erreur annonçant l’erreur et permettant à l’utilisateur de ré-entrer les données erronées.

*VI) Conclusion personnelle des membres du groupe*

* Nathan DE LEENER
* Gaëtan GORTZ
* Brian Luk