Mathys BRONNEC Groupe G

Rapport de projet

J’ai commencé le projet sans penser faire une interface graphique au début, j’avais codé l’algorithme de Dijkstra durant le TD sur les graphes. J’ai commencé par créer la classe abstraite Personne, puis leurs sous classes Salarie et Client. J’ai aussi créé une classe Salaries et Clients qui sont la liste des Salariés et la liste des Clients. Globalement quand je créé une classe et la même classe au pluriel c’est pour l’objet et la liste (ex : Vehicule et Vehicules). J’ai ensuite décidé de faire une interface graphique.

Toutes les classes au « pluriel » sont utilisées pour faire de l’écriture et de la lecture de fichier (Salaries, Clients, Vehicules et Commandes). Globalement j’ai codé dans l’ordre de l’énoncé du problème. J’ai eu du mal avec le module Commande qui a été ma plus grosse difficulté, j’ai du ajouter une classe EmploiDuTemps qui crée pour chaque Chauffeur un fichier avec la liste des dates où il a des commandes. J’utilise le numéro de sécurité sociale comme clé primaire pour les clients et les salariés et les plaques d’immatriculations comme clé primaire pour l’identification des Véhicules. J’ai aussi ajouté un attribut ListeDatesOccupees pour avoir les disponibilités des Véhicules. De cette manière, quand je veux créer une nouvelle commande, je sélectionne une date et le programme me laisse choisir parmi les salariés et les véhicules disponible le jour choisi.

**Les ajouts à noter :**

Dans le **module salarié :**

Possibilité de remplacer un salarié, le renvoi ne peut se faire que si l’employé n’a personne à sa charge et, dans le cas des chauffeurs, s’il n’a pas de livraison à venir. Le bouton « Sauvegarder les modifications » sert à écrire les modifications d’attributs dans le fichier. Cependant, si l’on fait des modifications et que l’on oublie de les sauvegarder, si l’on tente de quitter la page , on a un popup demandant si on veut sauvegarder les modifications.

Dans le **module client :**

Même système de sauvegarde que dans le module salarié, le tri s’effectue sur les liste avec des List.Sort et des delegates. On peut supprimer un ou plusieurs clients à la fois. On ne peut modifier que les attributs qu’ils ont dit pouvoir être modifiés dans l’énoncé.

Dans le **module commande :**

On a la liste des commandes en cours qui s’affiche, on peut supprimer n’importe quelle commande en cours ce qui libère le véhicule et le chauffeur pour la date où était la commande. On peut afficher le chemin à emprunter. Pour ajouter une nouvelle commande, on choit une date et un type de véhicule. **Les dates où tous les salariés sont occupées sur les 30 prochains jours sont en gras sur le calendrier**. On peut ensuite voir la liste des véhicules du type sélectionné qui sont disponible le jour sélectionné, on choisit un des véhicules dans la liste, puis la liste des chauffeurs disponibles le jour sélectionnés s’affiche, on choisit ensuite un chauffeur pour effectuer la course, puis un client, on entre les informations nécessaires à la commande (point de départ, point d’arrivée et spécificités du véhicule). Le prix de la commande est calculé grâce à la méthode   
**Vehicule virtual double Prix(string PointA, string PointB)**. Chaque type de véhicule a son override de cette méthode. Par exemple, le prix d’une camionnette est fixe à la journée, le prix d’un car s’effectue par rapport au nombre de place dans le car et à la distance parcourue, le prix d’un camion citerne par rapport au type de cuve et son volume etc…

Dans le **module statistique :**

Module sans ajout particulier, simple d’utilisation.

**2 nouveaux modules :**

**-Module Gestion des fichiers :**

Permet de faire une sauvegarde de la base de donnée actuelle ou de charger une ancienne sauvegarde de base de donnée (liste des clients, liste des véhicules, liste des commandes, liste des salariés, emploi du temps de chaque véhicule et de chaque salarié)

**-Module Garage :**

Permet de voir la liste de tous les véhicules de la société, on peut en ajouter à notre guise et l’on peut les supprimer seulement s’il ne sont impliqué dans aucune commande à venir.

Pour conclure, j’ai fait une interface graphique très simple d’utilisation et sans mode d’emploi nécessaire, on comprend facilement ce que l’on a le droit de faire ou non grâce aux différents messages d’erreurs que l’on peut rencontrer pendant la navigation dans l’interface graphique et quand l’on veut faire quelque chose, on sait où se rendre et comment le faire je pense. Une utilisation du programme sera beaucoup plus explicite que ce Rapport.