14 juin 2016

LINGE1322
Rapport de Projet

Informatique Analyse et conception de systèmes d'information

Denauw Antoine

Table des matières

1 Introduction

Dans le cadre du cours LINGE1322 (Informatique Analyse et conception de système d'information) de l'UCL, nous avons été amené à réaliser un petit système permettant de gérer une base de données de location de véhicules. Ce projet pouvait être fais en groupe mais j'ai décidé de le faire seul et de poser juste quelques questions à mes collègues. Ma petite application va donc vous permettre de gérer la base de données fournie avec la clé USB. J'ai décidé d'apprendre le PHP, comme vous l'avez conseillé, pour faire le lien entre le site et la DB.

2 Comment lancer l'application ?

Afin de pouvoir utiliser le PHP et de pouvoir lancer mon application en local, j'ai décidé d'installer XAMPP (l'équivalent de WAMP sous Windows ou MAMP sous Mac OS X) étant donné que je suis sous Ubuntu. Une fois le logiciel installé, vous pouvez le lancer via la commande : "sudo /opt/lampp/lampp start" dans l'invite de commande sous Linux. Il ne faudra pas oublier de l'arrêter une fois que vous aurez fini d'utiliser l'application en faisant : "sudo /opt/lampp/lampp stop".

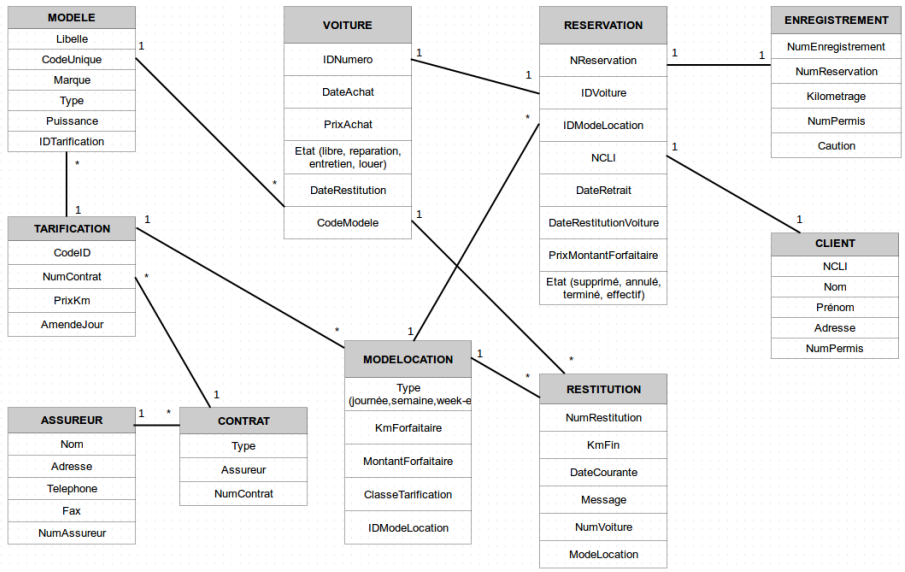
Une fois le logiciel lancé, rendez vous sur notre navigateur habituel que ce soit Firefox, Google Chrome ou IE ne change rien. Dans la barre d'adresse taper : "localhost/Projet_LINGE1322/Site/index.php" ce qui vous enverra directement à la page d'accueil de mon site.

Il vous suffit ensuite de vous rendre à l'adresse : "localhost/dashboard/" et de cliquer sur le bouton phpmyadmin. Si vous voulez jeter un oeil à la base de données de l'application, il vous faudra importer le fichier *Denauw-Antoine-DB-Projet1322.sql* en cliquant sur l'onglet "Importer" de phpmyadmin. Une fois sur la page de phpmyadmin, vous pourrez voir la base de données Projet_LINGE1322 avec les tables que j'ai ajoutées et les données.

3 Les schémas

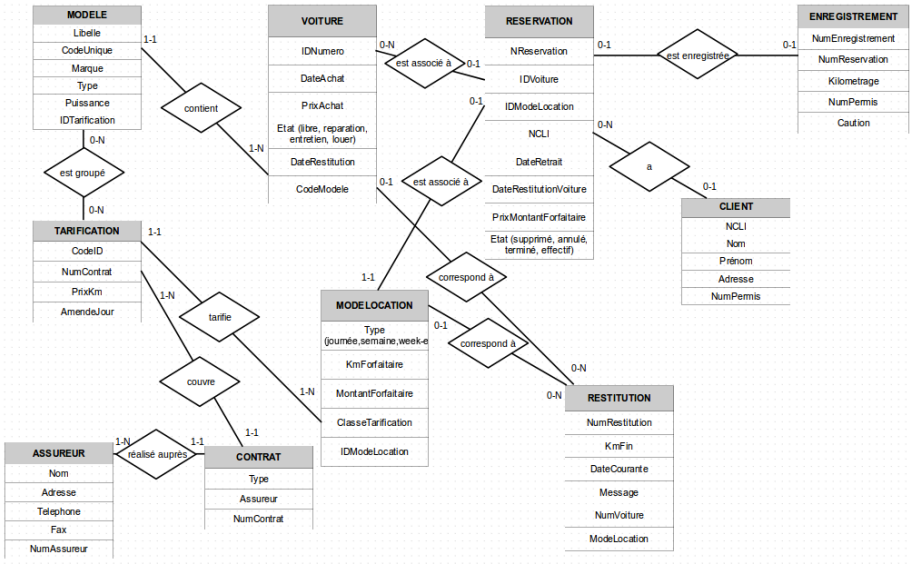
Pour réaliser l'application, il était demandé de réaliser les schémas relationnel et entité-association de notre base de données. Si vous ne les voyez pas bien, ils sont sous format pdf dans ce même dossier.

3.1 Le schéma relationnel



Nous avons donc ici les différentes tables de ma base de données reliée entre elles par des relations One-to-one (1 1) et One-to-Many (1 *). Les détails de ces relations seront détaillés à l'oral.

3.2 Le schéma entité-association



Ici encore nous avons les différentes tables reliées par des relations tel que vus au cours, encore une fois les détails de ces relations seront détaillés à l'oral si besoin.

4 L'interface de l'application

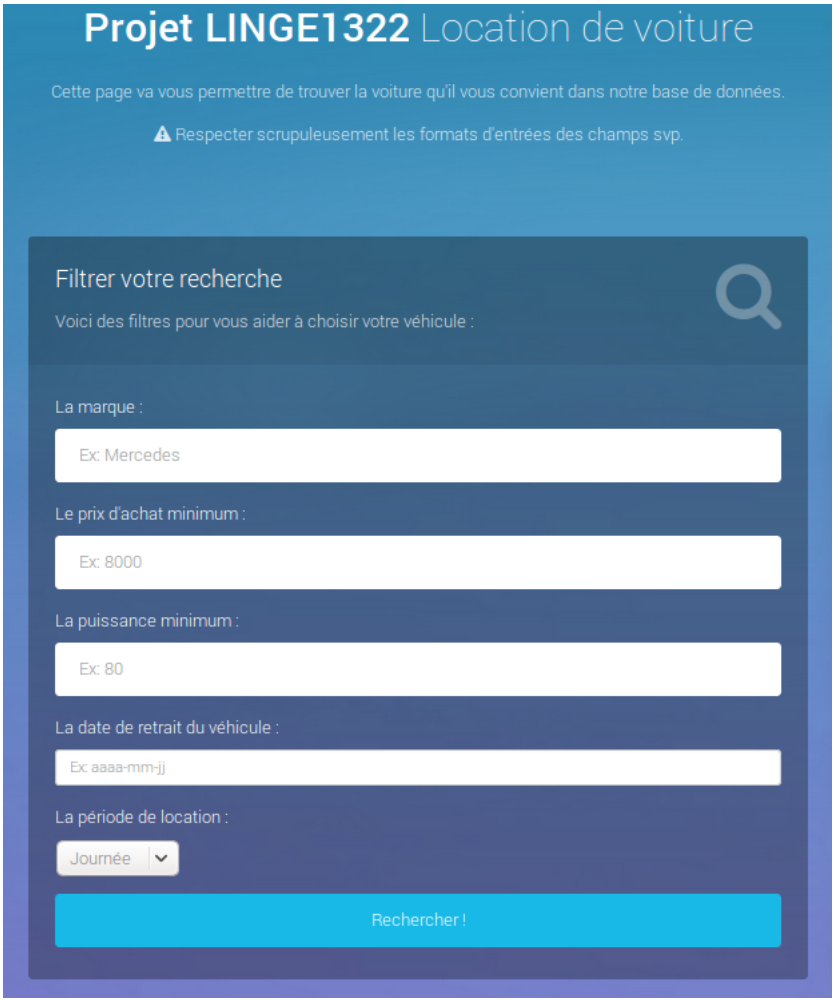
Pour l'interface, j'ai décidé de faire quelque chose de simple, épuré mais néanmoins très efficace. En effet, j'ai décidé de créer une page d'accueil qui va contenir les 5 options principales de l'application et d'ajouter un bouton "Accueil" à chaque fin de page ce qui va permettre à l'utilisateur de retourner quand il le souhaite à la page d'accueil si par exemple le client s'en va, change d'avis ou autre. L'interface a été réalisé à partir d'un template Bootstrap qui comportait une seule page de formulaire mais, par la suite, il a été modifié par mes soins de manière rapide et assez radicale car je me suis rendu compte que cette application était bien plus qu'une simple suite de formulaires.

5 Les options

Il y a 5 options principales dans mon application qui ont chacune un rôle bien précis pour chaque circonstance.

5.1 Ajouter une réservation

Cette option va permettre à un client qui se présente de pouvoir réserver un véhicule pour plus tard ou le jour même. Lorsqu'on clique sur le bouton "Ajouter", nous arrivons sur une autre page php qui va permettre au client de "filtrer" notre base de données à l'aide de ses préférences qu'il pourra noter dans les différents champs disponible :



Vous constaterez que les champs à compléter sont, d'après moi, assez nombreux et sélectif pour le moment étant donné que notre base de données est plutôt petite. Toutefois, ces filtres sont assez facile et rapide à ajouter en SQL.

Voici le code SQL que j'ai utilisé pour filtrer la base de données (disponible dans la page reserver.php) : [1]

```php
<?php
$req = $bdd->prepare('SELECT DISTINCT V.DateRestitution, V.IDNumero, L.IDModeLocation, V.PrixAchat,
                                      M.Marque, M.Type, M.Libelle, M.Puissance, L.MontantForfaitaire
                      FROM VOITURE V, MODELE M, TARIFICATION T, MODELOCATION L, RESERVATION R
                      WHERE V.CodeModele = M.CodeUnique
                      AND T.CodeID = L.ClasseTarification
                      AND V.DateRestitution <= ?
                      AND L.Type = ?
                      AND V.PrixAchat >= ?
                      AND M.Marque = ?
                      AND M.Puissance >= ?
                      ORDER BY V.IDNumero');
$req->execute(array($_POST['form-date-retrait'], $_POST['form-periode'], $_POST['form-prix-achat'],
                    $_POST['form-marque'], $_POST['form-puissance']));
```

Pour vous prouver que je suis capable d'ajouter d'autres filtres, je vous propose de penser à un filtre qui permettra à l'utilisateur de mettre une des options présente dans le libellé du modèle. Pour se faire, l'utilisateur rentrera une option par exemple "3 portes" et le filtre recherchera dans le libellé tout les modèles qui comporte les caractères : "3 portes".
Voici le code SQL :

- SELECT * FROM 'MODELE' WHERE 'Libelle' LIKE '%3 portes%' ;

Ce qui vous retournera les modèles avec "3 portes" dans leur libellé :

				Libelle	CodeUnique	Marque	Type	Puissance	IDTarification
☐	Modifier	Copier	Effacer	3 portes, toit ouvrant, climatisation, bluetooth	2	Ford	Fiesta	50	10
☐	Modifier	Copier	Effacer	3 portes, toit ouvrant, autoradio, sieges ajustabl...	4	Fiat	Punto	20	4
☐	Modifier	Copier	Effacer	bluetooth, 3 portes, climatisation,	5	Renault	Captur	80	5
☐	Modifier	Copier	Effacer	3 portes, bluetooth, climatisation, toit ouvrant, ...	9	VW	Golf	90	5

Il suffit maintenant de savoir si le client est dans notre base de données ou pas. Pour se faire, un bouton va vous permettre de les afficher et si il n'est pas dans la listes, un autre bouton va vous permettre de le rajouter.

Les modèles disponibles sont donc maintenant affichés au client et il peut compléter le formulaire en dessous de la page. Toutes les informations étant sur cette même page, le formulaire est très facile à compléter. Une fois le formulaire complété, une réservation est créée dans notre base de données (à l'aide du code SQL ci-dessous) et est ensuite affichée à la page suivante :

1. A noter que les "?" sont remplacé par les valeurs en dessous (le premier "?" fait donc référence à la date de retrait que l'utilisateur aura choisi dans les champs du formulaire à la page précédente, le deuxième "?" à la deuxième valeur, etc)

```php
$req = $bdd->prepare('INSERT INTO RESERVATION (IDVoiture, IDModeLocation, NCLI,
                                DateRetrait, DateRestitutionVoiture, PrixMontantForfaitaire, Etat)
                                VALUES(?, ?, ?, ?, ?, ?, "effectif")');
$req->execute(array($_POST['form-IDVoiture'],
                    $_POST['form-mode-location'],
                    $_POST['form-NCLI'],
                    $_POST['form-date-retrait'],
                    $_POST['form-date-restitution'],
                    $_POST['form-prix-indicatif']));
```

Il ne reste plus qu'à l'imprimer en deux exemplaire et le client pourra revenir à la date qu'il a mentionné pour retirer son véhicule avec sa réservation à la main.

5.2 Retirer un véhicule

Vous devez appuyer sur ce bouton dans le cas où le véhicule est directement disponible pour le client. Si le véhicule n'est pas disponible quand le client arrive, il vous faudra cliquer sur le bouton "Remboursement", décris à la section suivante. Si le client désire quand même louer un autre véhicule après avoir été remboursé, il faudra cliquer sur ce bouton.

Une fois que vous avez cliqué sur ce bouton, il vous est affiché tout les véhicules disponible actuellement dans la base de données. Le réceptionniste de garage va donc pouvoir vérifier de visu si le véhicule est disponible (dans l'étude de cas il est stipulé "de visu" j'en ai conclu que ça se passait comme ça).
Le code SQL pour afficher les véhicules disponible est plutôt simple :

- SELECT * FROM VOITURE WHERE Etat = "libre" ;

Il y a de nouveau la liste des clients pour connaître le numéro de permis du client à partir de son numéro de client. Une fois le formulaire complété, un ENREGISTREMENT est ajouté de la même façon que nous avions ajouté une RESERVATION précédemment. Ensuite, vous êtes redirigé vers un formulaire qui va vous permettre d'imprimer le contrat en 2 exemplaires. Une fois sur la page où vous pourrez imprimer le contrat, vous allez devoir vous rendre dans la base de données pour dire que le véhicule (rentrer sont ID) est maintenant dans un état "louer" jusqu'à la date que vous allez devoir rentrer également, tout ça en remplissant le petit formulaire et en appuyant sur le bouton. Sur la dernière page, il vous faudra juste rentrer le numéro de réservation pour la marquer comme terminée. Vous serez ensuite redirigé vers l'accueil.
Le code SQL pour modifier ces propriétés :

- UPDATE RESERVATION SET Etat = "termine" WHERE RESERVATION.NReservation = $_POST['form-numero-reservation'] ;

Vous remarquerez que le bouton ”Accueil” n’est plus présent sur ces deux dernières pages. C’est volontaire : pour empêcher l’utilisateur de l’application de louer un véhicule à une personne jusqu’à une telle date sans que celui-ci ne soit dans un état ”louer” (donc indisponible pour les autres clients) jusqu’à cette date. Ou qu’une réservation ne soit pas notée comme terminée alors qu’elle l’est.

5.3 Remboursement

Si le véhicule n’est pas disponible (si un client le garde plus longtemps que prévu ou qu’il est en réparation, ...), le client doit être immédiatement dédommagé d’un montant équivalent à l’amende journalière prévue pour la catégorie de modèles à laquelle appartient le véhicule qu’il avait réservé, il vous faudra alors cliquer sur ce bouton.

Le code SQL derrière cette manipulation fonctionne en sélectionnant l’amende journalière de la classe de tarification auquel le modèle fait référence auquel l’ID de la voiture fait référence :

```php
<?php
$req = $bdd->prepare('SELECT DISTINCT T.AmendeJour FROM RESERVATION R, TARIFICATION T, MODELE M, VOITURE V
                      WHERE V.IDNumero = ?
                      AND V.CodeModele = M.CodeUnique
                      AND M.IDTarification = T.CodeID');
$req->execute(array($_POST['form-IDVoiture']));
```

Maintenant, retour à l’accueil et le client est libre soit d’en choisir un autre, soit de partir.

5.4 Annulation

Cette option va tout simplement permettre au client d’annuler sa commande. Dans ce cas, le numéro de la réservation sera demandé afin de la mettre dans un état ”annule”. Attention, le client a le droit de l’annuler deux jours avant la date de retrait du véhicule après cela, il ne sera plus possible d’annuler sa réservation sans perdre sa caution.
Le code SQL :

```php
$req = $bdd->prepare('UPDATE RESERVATION SET Etat = "annuler" WHERE RESERVATION.NReservation = ? ');
$req->execute(array($_POST['form-numero-reservation']));
```

5.5 Restitution du véhicule

La première page va simplement permettre au garagiste de bien vérifier le kilométrage final du véhicule ainsi que les dommages éventuel causé à ce dernier. Une fois le formulaire rempli, nous affichons la RESTITUTION que vous venez de compléter afin que vous puissiez l’imprimer en deux exemplaires (un pour la personne responsable du service de gestion des réparations et un pour le client qui devra se présenter avec au services des paiements). Une nouvelle RESTITUTION est alors rajoutée de la même façon que nous avions rajouté une RESERVATION.

La restitution est donc affichée à l'aide d'une requête assez intéressante, en effet nous allons sélectionner la RESTITUTION avec l'ID le plus grand (soit celle que nous venons d'ajouter) et l'afficher. Nous avons besoin d'une sous-requête pour rechercher l'ID le plus grand de cette même table :

```php
<?php
$req = $bdd->prepare('SELECT * FROM RESTITUTION

                      WHERE NumRestitution = (SELECT MAX(NumRestitution)

                                              FROM RESTITUTION)'

                     );
$req->execute(array());
```

Le client se présente maintenant au services des paiements où il va devoir compléter le formulaire pour permettre à l'application de calculer ce qu'il devra payer. Cette partie de l'application est implémentée de façon à ce que le code SQL ci-dessous tienne compte des éventuels jours de retards, des Km qui ont été dépassés (au delà du Km forfaitaire), si il a payé la caution ou pas, etc.
Voici donc la plus grande et sûrement la requête SQL la plus compliquée de ce projet : [2]

```php
<?php
$req = $bdd->prepare('SELECT  DISTINCT I.DateCourante, V.DateRestitution, L.KmForfaitaire, E.Kilometrage,

                                       I.KmFin, E.Caution, L.MontantForfaitaire, T.PrixKm, T.AmendeJour

                      FROM VOITURE V, MODELOCATION L, RESTITUTION I, RESERVATION R, ENREGISTREMENT E, TARIFICATION T

                      WHERE I.NumVoiture = ?

                      AND V.IDNumero = I.NumVoiture

                      AND I.ModeLocation = ?

                      AND L.IDModeLocation = I.ModeLocation

                      AND V.IDNumero = R.IDVoiture

                      AND E.NumReservation = R.NReservation

                      AND L.ClasseTarification = T.CodeID

                      ');
$req->execute(array($_POST['form-numero-vehicule'], $_POST['form-mode-location']));
```

Vous remarquerez que seulement les résultats sont affichés, en effet mon application permet de renseigner tout ce qu'il faut pour permettre à l'utilisateur de l'application de calculer, à l'aide d'une calculatrice, ce que le client doit payer. Par exemple : je mentionne le nombre de Km à la fin, le nombre de Km avant qu'il prenne la voiture, le nombre de Km qu'il pouvait parcourir sous le couvert du Km forfaitaire et enfin ce qu'il devra payer par Km dépassé. Il suffira à l'utilisateur de soustraire les Km à la fin par les Km au début pour obtenir le nombre de Km parcouru et voir si il a dépassé le Km forfaitaire ou pas. Si il a dépassé le Km forfaitaire, alors il calcule le nombre de Km qu'il a dépassé et le multiplie par ce qu'il doit payé par Km dépassé. Je n'ai pas implémenté cette partie car d'après moi, cela n'avait rien à voir avec du code SQL et donc une telle démarche (et le nombres d'heures passé dessus) aurait été inutile car ce n'est pas sur quoi nous devons être évalué de mon humble avis.

2. Si vous ne la voyez pas bien, toutes les images de ce rapport se trouve dans le dossier images. Celle-ci s'appelle "calculerPaiement.php"

Il suffit maintenant de donner l'ID du véhicule afin de le mettre en état "libre" et de mettre la date de restitution au "CURRENT_TIMESTAMP" par sécurité. Vous êtes maintenant redirigé vers l'accueil ce qui clôture notre tour des options de l'application.

6 Difficultés rencontrées

La première grosse difficulté que j'ai rencontré durant ce projet, a été de bien extraire les données de l'étude de cas. En effet, bien comprendre l'étude de cas (de 5 pages) et en extraire correctement ses données est primordial car sans cela, on se lance dans une application instable avec une base de données tout aussi instable.

La deuxième difficulté a été l'agencement de l'application car je ne savais pas trop comment cela devait se dérouler avec le client et pour être honnête je n'ai pas compris la situation correctement à la première lecture de l'étude de cas. Donc j'ai supposé une vraie application dans la vie réelle avec un client qui se présente, on lui imprime un vrai formulaire de réservation en papier, il faut qu'il se présente avec sa réservation sinon nous ne pouvons pas lui donner le véhicule, etc.

Apprendre le PHP n'a pas été extrêmement dur et je trouve que c'est un langage extrêmement intéressant à connaître à l'heure actuelle car la plupart des sites web l'utilise.

7 Conclusion

En conclusion, j'ai utilisé les langages SQL et PHP (mais aussi HTML, CSS et JS) pour vous présenter une application convaincante qui respecte tout les critères que vous avez mentionnés dans les consignes. J'espère vous avoir prouvé que j'ai acquis une bonne connaissance du SQL et des bases de données à travers ce projet (requêtes imbriquées, jointure, alias, fonction agrégative (Max), etc). Je vous rappelle que tout mon code est disponible dans les dossiers de la clé USB donc vous pourrez vérifier que mes requêtes SQL fonctionnent ou même que mon site est fonctionnel.