



RAPPORT DE STAGE

FORMATION : *Concepteur Développeur
d'Applications*

STAGIAIRE : *DAIRAINÉ Alexis*

SOMMAIRE

<i>I – Remerciements</i>	<i>page 2</i>
<i>II - Introduction</i>	<i>page 3</i>
<i>III – Présentation de l’entreprise</i>	<i>page 4</i>
<i>IV – Résumé en anglais</i>	<i>page 5</i>
<i>IV – Projet</i>	<i>page 6</i>
1 – Objectifs	page 6
2 – Outils	page 6
<i>V – Les Technologies</i>	<i>page 7</i>
<i>VI – Apprentissage de WinDev</i>	<i>page 9</i>
<i>VII – Réalisations</i>	<i>page 11</i>
1 – Projet Fictif	page 11
2 – Projet réel	page 19
3 – Formation	page 36
4 – Maintenance	page 37
<i>VIII – Conclusion</i>	<i>page 38</i>
<i>IX – Annexes</i>	<i>page 39</i>
1- Analyseur de performances	page 39
2- Envoi d'un e-mail	page 40
3- Connexion manuelle BDD	page 40
<i>X – Références</i>	<i>page 41</i>

REMERCIEMENTS

Je tiens à remercier l'ensemble des intervenants durant mes périodes de formation et de stage qui ont contribué à faire en sorte que tout se passe pour le mieux et qui ont su répondre à mes interrogations et mes besoins.

Plus particulièrement, je voudrais remercier M. François-Régis CAUMARTIN, M. Emmanuel BERTHOME, M. Germain SIPIERE ainsi que Mme Marion BRUSADELLI, tous formateurs à l'AFPA d'Amiens, qui m'ont accompagné dès le début de ma formation en me dispensant de précieux conseils qui m'ont permis de toujours progresser tout au long de ce parcours.

Aussi, je tenais à remercier M. Florian BRUNET, mon maître de stage au sein de Servi-Plus, qui a su me guider et m'aider tout au long de mon stage de par ses conseils avisés et ses explications limpides.

Un grand merci également à M. Laurent PARSIS, gérant de Servi-Plus et M. Arnaud BRASSART, employé chez Servi-Plus, sans qui ce stage n'aurait pas été possible et qui m'ont permis de le passer dans la convivialité, le sérieux et la bonne humeur.

Enfin, je n'oublie pas non plus mes collègues de formation et plus particulièrement ma partenaire, de projet et surtout dans la vie, Juliette qui m'aura aidé et soutenu tout le long de ce sinueux parcours ainsi que mes camarades et amis : Florian, Nathan, Mohamed, Thomas, Scotty et Léa sans qui cette formation n'aurait pas été vécue aussi joyeusement.

INTRODUCTION

Etant depuis petit attiré par les nouvelles technologies, suivre cette formation était pour moi un réel plaisir. En effet, j'ai toujours été un passionné de l'informatique et le choix de me spécialiser dans le développement est né d'une envie de découvrir de nouveaux outils et d'apprendre un nouveau métier qui me tient à cœur.

Après avoir suivi plusieurs formations en ce sens (TB/MS/CDA), il m'a donc été demandé d'effectuer un stage en entreprise d'une durée de 3 mois. J'ai donc eu la chance de pouvoir l'effectuer dans l'entreprise Servi-Plus basée à Drucat. Là-bas j'y ai appris énormément sur un logiciel et un langage tout à fait nouveau pour moi j'ai nommé WinDev.

Ne sachant pas manipuler ce nouvel outil, j'ai donc dû apprendre toutes les bases afin de pouvoir aider et travailler au sein de l'entreprise tout en étant efficace.

Je débiterais donc ce rapport en parlant de ce que j'ai appris sur WinDev, et je poursuivrais en présentant les différents travaux qui m'ont été demandé de réaliser puis de la petite période durant laquelle j'ai eu un jeune stagiaire avec moi.

PRESENTATION DE L'ENTREPRISE

Servi-Plus est une SARL (Société à Responsabilité Limitée) basée à Drucat, à quelques kilomètres d'Abbeville, et est en activité depuis maintenant 26 ans (créée en 1996). Elle est spécialisée dans le secteur d'activités du commerce de gros (commerce interentreprises) d'ordinateurs, d'équipements informatiques périphériques et de logiciels.

Au fil du temps, elle a su développer son activité et propose aujourd'hui une large gamme de services :

- Maintenance : logiciel et matériel.
- Mise en place de réseaux : - Switch
 - Réseaux WI-FI
 - Serveur NAS
- Formation : - Internet
 - Pack Office
 - Utilisation système d'exploitation (Windows)
- Assistance du lundi au vendredi de 8h-12h à 13h30-18h

Plus précisément, l'entreprise vend du matériel informatique, des logiciels confectionnés par les employés, forme sur les logiciels créés ainsi que sur ceux de Microsoft et dépanne du matériel informatique (sur site ou télémaintenance).

RESUME EN ANGLAIS

I did my training at Servi-plus, a company located in Drucat, near Abbeville (80). This company sells computer hardware, software made by employees, creates websites and has three people, the manager, Mr Laurent Parsis, and two employees, Mr Arnaud Brassart and Mr Florian Brunet.

During my internship, my supervisor asked me to develop several programs to improve the work of my colleagues with customers and in general. For learn how to use the software WinDev, and understand the language I have started to develop a fictional project and on the inside I have learned all the basics. For example, I learned to use the windows system, the variables, the procedures and how to integrate all the differents components necessary. This fictional project served me uniquely for learning and training. The first step was to create a window with a table inside and in this table we insert all the quotations in the database. On this datas, I have created some buttons with differents functions, one for create a new quotation, another to update a quotation and a last for delete a quotation. After that, I have develop a new program for retrieve the differents datas we had in another external database and perform the same operations we made for the first table. At the end of the fictional project, my supervisor made me work on a real project to manage all the quotations we already had. My first exercice in this project was to create a simple window which contain all the usefull informations for a quotation (costs, VAT, number of pupils,...). Like I said before, this informations are usefull for make a quotation for the customers. Afterwards, I have created another window for validate a quotation and make operations on the datas. This functions insert a follow up message and a date on the tracking window. I have created more functions which I wouldn't speak here but later in this internship report. For my last week in the enterprise, I've had with me another younger trainee coming from college. So, I've learned to teach to another person how to develop using Windev and WLangage. It's a new experience for me and I think it's not an easier thing because for explain to another person how a software work and how to use It, It's essential to use an adapted vocabulary and It's not a simple thing. Also, I made external intervention in customer base for troubleshoot with my other colleague in charge of maintenance and assistance. All the experiences were interesting like all my period of internship.

PROJET

1 – Objectifs

Le nombre de collectivités faisant appel à Servi-Plus pour leur programme de gestion de cantine étant en constante augmentation, l'entreprise a développé un logiciel pour son utilisation exclusive en interne afin de gérer au mieux et de manière plus efficace les centaines de collectivités qui requièrent leurs services.

De fait, j'ai donc été missionné par mon maître de stage de développer plusieurs fonctionnalités et corriger certains dysfonctionnements dans ce logiciel afin de faciliter leur travail et de proposer une assistance toujours plus efficace aux clients.

2 – Outils

Pour réaliser les différentes tâches qui m'ont été confiées j'ai donc utilisé principalement : **WinDev** et **VS Code**.

Pour m'assister j'ai aussi utilisé **WAMP Server** afin de disposer d'un serveur local pour tester mes scripts en PHP, **Postman** pour pouvoir essayer mes différentes requêtes http ainsi que **Filezilla** afin de manipuler des fichiers sur un serveur FTP.

LES TECHNOLOGIES

WinDev est un atelier de génie logiciel (AGL) développé par la société française PC Soft, orienté pour développer principalement des applications Windows, mais également Linux, .NET et Java. On peut lui associer aussi WebDev (pour la conception d'applications web) et WinDev Mobile (pour la conception d'applications mobiles).

A l'installation, WinDev comprend différents éditeurs comme un éditeur d'analyses, de fenêtres, de requêtes SQL, d'états, de code, d'UML, etc...

Sous WinDev, les fenêtres et tous les éléments qui la composent sont créés dans un premier temps de manière visuel, ce qui permet de positionner et de styliser l'élément en question de la manière que l'on souhaite puis ensuite nous pouvons manipuler à souhait cet élément par programmation puisque les composants de WinDev embarquent avec eux tout une panoplie de propriétés. Ensuite, WinDev embarque également avec lui un langage qui lui est propre : le WLangage.

Ce langage est conçu en français ce qui le différencie des langages plus traditionnels (PHP, JS,..) et n'est évidemment utilisable que sous WinDev.

Le logiciel est livré avec la base de données HFSQL qui est gratuite avec WinDev et set disponible en mode Classic, Client/Serveur, Réseau, Cloud, Mobile, ...

Autre particularité de WinDev, c'est que le code à exécuter est généralement placé dans le composant avec lequel il interagit, ce qui permet donc une excellente lisibilité et une meilleure maintenabilité.

Enfin, une documentation en ligne est disponible ainsi que des exemples et des aides dans WinDev directement.

En revanche, en raison de son coût, et d'autres raisons WinDev est bien moins répandue à travers le monde que ses concurrents ce qui entraîne donc une documentation moins fournie.



Lors de la réalisation des tâches qui m'ont été confiées, j'ai également utilisé l'éditeur de texte développé par Microsoft : Visual Studio Code.

Bien plus répandu que WinDev (et surtout gratuit), VS Code embarque avec lui un système de débogage, la mise en évidence de la syntaxe, permet aussi d'intégrer des extensions développées par d'autres utilisateurs et beaucoup d'autres fonctionnalités que je ne développerais pas ici.

Cet éditeur m'a donc permis d'utiliser le langage PHP qui m'a été très utile durant ce stage.



Pour pouvoir tester mes scripts PHP, je me suis servi de WampServer qui est une plateforme de développement web qui permet de disposer d'un serveur local pour faire fonctionner les scripts. WampServer intègre 3 serveurs (Apache, MySQL et Maria DB) et phpMyAdmin pour l'administration Web des bases MySQL.



Pour terminer ce tour d'horizon, je vais vous présenter rapidement l'application Postman qui m'aura été utile dans les tests de mes API. Cette application permet donc d'envoyer des requêtes http vers une adresse et renvoi la réponse de cette dernière. Et enfin, le très connu Filezilla qui permet des fichiers à souhait sur un FTP.



APPRENTISSAGE DE WINDEV

Etant complètement novice dans l'utilisation de cet outil, il m'a fallu apprendre les bases de WinDev avant de commencer à réellement travailler pour mes collègues.

Afin d'apprendre les principes fondamentaux du logiciel, j'ai commencé par suivre le manuel d'autoformation fourni par mon maître de stage. Ce dernier m'a fait comprendre le fonctionnement global de WinDev. Ceci passe donc par l'apprentissage de la syntaxe, des procédures ou bien encore du système de fonctionnement de base de données intégré à WinDev.

Pour ce faire, en plus du manuel d'autoformation, j'ai également réalisé les exemples fournis pour approfondir et mieux comprendre. Cette période a été assez courte (1 jour et demi plus ou moins) et s'est clôturée par un exercice donné par mon maître de stage qui consistait à créer une fenêtre réunissant toutes les informations nécessaires pour la création d'un devis.

The screenshot shows a window titled "Infos Devis" with a close button (X) in the top right corner. The window contains two main sections, each with a yellow header and a dark background.

Logiciel de gestion de cantine

At the top of this section is a large white input field. To the right, "Taux Tva" is set to "0,00".

Below the header, there are several rows of input fields and calculated values:

- Nom logiciel**: A white input field.
- Coût HT**: A white input field with "0,00 €" next to it.
- Coût TTC**: A white input field with "9 999 999 999,99 €" next to it.
- Coût Ouverture Compte HT**: A white input field with "0,00 €" next to it.
- Coût Ouverture Compte TTC**: A white input field with "9 999 999 999,99 €" next to it.
- Total Mise En Place HT**: A white input field with "9 999 999 999,99 €" next to it.
- Total Mise En Place TTC**: A white input field with "9 999 999 999,99 €" next to it.

Abonnement Annuel

At the top of this section, "Nombre d'Enfants" is set to "0" and "Taux Abonnement" is set to "0,00".

Below the header, there are several rows of input fields and calculated values:

- Prix Annuel Élèves HT**: A white input field with "9 999 999 999,99 €" next to it.
- Prix Annuel Élèves TTC**: A white input field with "9 999 999 999,99 €" next to it.
- Frais Fixe HT**: A white input field with "0,00 €" next to it.
- Frais Fixe TTC**: A white input field with "9 999 999 999,99 €" next to it.
- Total Abonnement HT**: A white input field with "0,00 €" next to it.
- Total Abonnement TTC**: A white input field with "0,00 €" next to it.

At the bottom right of the window is a red button with a white "X" icon and the text "Fermer".

Bien que le visuel de cette fenêtre soit basique, elle permet de regrouper toutes les informations d'un devis sur une seule fenêtre bien que ces dernières soient dans différentes tables de la base de données.

Aussi, j'ai rempli certains champs par programmation pour pouvoir effectuer des calculs (TVA dans le cas présent) puisque ces informations ne sont pas disponibles directement en base.

Pour afficher certains champs de cette fenêtre, j'ai fait face à certains problèmes que tout débutant de WinDev doit connaître. En effet, les champs concernés (données se rapportant

aux paramètres et ceux du logiciel de comptabilité) ne s'affichaient pas du fait qu'elles n'étaient tout simplement pas en mémoire. Pour rectifier ce souci, il a donc fallu rechercher ces informations en base afin de pouvoir les afficher à l'écran.

Ceci s'effectue avec les fonctions de recherche du logiciel (HLitRecherche, ...) qui reviennent en quelque sorte à exécuter une requête SQL classique.

Pour revenir aux calculs que j'ai évoqué précédemment, je me suis retrouvé face à un autre problème qui est que les calculs ne s'effectuaient pas, ce qui s'explique simplement par un mauvais placement du code. Comme je vous le disais, dans WinDev, le code est placé dans le composant avec lequel il interagit. Dans le cas présent, les instructions de calculs étaient placées avant l'initialisation de la fenêtre, ce qui a eu pour effet d'effectuer les calculs avant d'initialiser cette page et donc WinDev affichait le rendu sans ces calculs. Il m'a juste été nécessaire de déplacer le code dans la partie de fin d'initialisation de la fenêtre pour que tout fonctionne correctement.

```
Fin d'initialisation de FEN_Info_Devis
FichierVersEcran()
HLitRecherche(LogicielCompta, IDLogicielCompta, SAI_IDLogicielCompta)
SI HTrouve(LogicielCompta) ALORS
  FichierVersEcran()
FIN

HLitPremier(Paramètres)
SI HTrouve(Paramètres) ALORS
  FichierVersEcran()
FIN

// SI PAIEMENT AVANCE ET PAYFIP REGIE
SI SAI_Mode = 3 ET SAI_PAYFIP = 3 ALORS
  SAI_CoutHT = 59
FIN

//SI PAIEMENT AVANCE ET PAYFIP AUCUN
SI SAI_Mode = 3 ET SAI_PAYFIP = 0 ALORS
  SAI_CoutHT = 0
FIN

// SI PAIEMENT FIN DE MOIS ET NON ASAP ET PAYFIP REGIE
SI SAI_Mode = 2 ET INT_EstEnAsap = Faux ET SAI_PAYFIP = 3 ALORS
  SAI_CoutHT = 59
FIN

SAI_CoutOuvertureCompteTTC = SAI_CoutOuvertureCompteHT * 1.2
SAI_FraisFixeTTC = SAI_FraisFixeHT * 1.2
SAI_PrixAnnuelElevesHT = SAI_NombreEnfants * 1.5
SAI_PrixAnnuelElevesTTC = SAI_PrixAnnuelElevesHT * 1.2
SAI_CoutTTC = SAI_CoutHT * 1.2
SAI_TotalMiseEnPlaceHT = SAI_CoutOuvertureCompteHT + SAI_CoutHT
SAI_TotalMiseEnPlaceTTC = SAI_CoutOuvertureCompteTTC + SAI_CoutTTC
SAI_TotalAbonnementHT = SAI_FraisFixeHT + SAI_PrixAnnuelElevesHT
SAI_TotalAbonnementTTC = SAI_FraisFixeTTC + SAI_PrixAnnuelElevesTTC
```

Cette image démontre l'importance du placement du code dans WinDev. Chaque élément d'une page (fenêtre, bouton, champ de saisie, ...) possède plusieurs « sections » où du code peut y être écrit et cela définit donc l'ordre d'exécution de ce code.

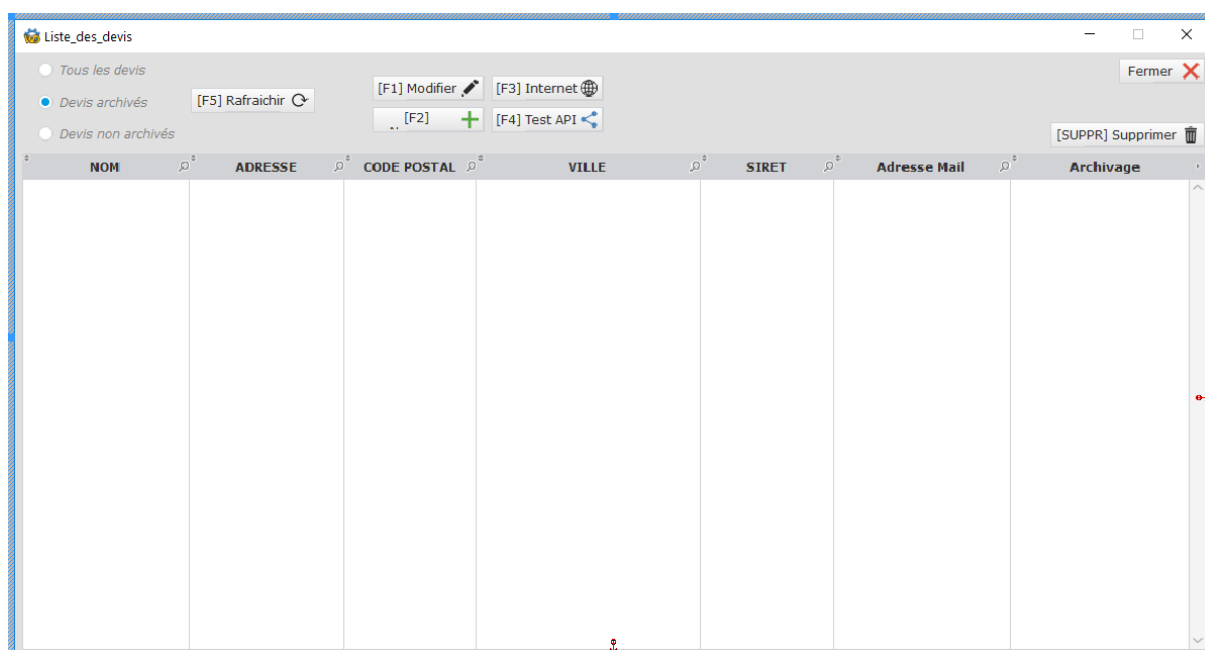
Ces différents cas de figure, clôturèrent donc mon apprentissage des principes de base de WinDev et du WLangage.

REALISATIONS

1 – Projet Fictif

Avant de travailler concrètement sur l'application de l'entreprise mon maître de stage m'a fait travailler sur un projet fictif afin de découvrir de nouvelles fonctionnalités de WinDev et consolider aussi certains acquis.

Dans un premier temps, j'ai été missionné de créer une fenêtre tout simple, affichant un tableau rempli par une requête SQL.



Ici vous avez donc la fenêtre finale du projet fictif, avec toutes les fonctionnalités que j'ai pu ajouter et que je vais vous présenter par la suite.

En haut à gauche, nous voyons donc un sélecteur pour pouvoir faire le tri entre des devis archivés, non archivés ou bien afficher l'intégralité des informations sans distinctions.

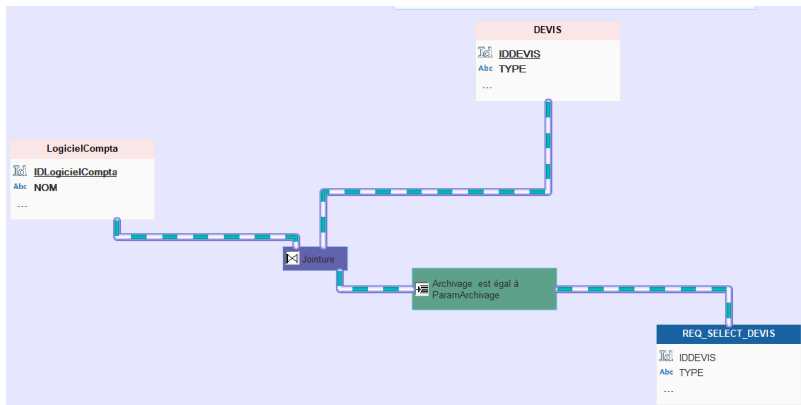
Du côté de la programmation, nous avons donc le champ sélecteur qui renvoie une valeur entre 0, 1 et 3. On capte cette valeur à l'initialisation du tableau comme ci-dessous :

```
Var est un Variant
Var = SELECTEUR_PARAM

SI Var = 3 ALORS
    Var = Null
FIN

REQ_SELECT_DEVIS.ParamArchivage = Var
```

Puis on la transmet ensuite dans la requête en tant que paramètre afin de pouvoir exécuter la requête avec la valeur du sélecteur comme paramètre.



```
WHERE
    LogicielCompta.IDLogicielCompta = DEVIS.IDLogicielCompta
AND
(
    DEVIS.Archivage = {ParamArchivage}
)
```

Selon la valeur reçue, la requête renvoie des résultats différents.

Dans les illustrations précédentes, nous pouvons constater que si la valeur du sélecteur vaut 3 alors la valeur renvoyée vaut NULL. Et si un paramètre vaut NULL en SQL sous WinDev alors la condition présente dans la requête est ignorée, ici cela implique que tous les devis archivés ou non seront affichés.

Ensuite, j'ai créé une fenêtre pour afficher les informations d'un devis sélectionné dans le tableau précédent.

En appuyant sur le bouton modifier présent au-dessus du tableau, on exécute ce code :

```
HLitRecherche(DEVIS, IDDEVIS, IDDEVIS)
SI HTrouve ALORS
    HLitRecherche(LogicielCompta, IDLogicielCompta, IDLogicielCompta)
    SI HTrouve ALORS
        HBloqueNumEnr(DEVIS, hNumEnrEnCours, hBlocageEcriture)
        Ouvre(FEN_Fiche_Devis)
        HDébloqueNumEnr(DEVIS, hNumEnrEnCours)
    FIN
FIN
```

Ce dernier effectue une recherche dans le fichier de données « DEVIS » grâce à l'id du devis que nous avons sélectionné dans le tableau. La deuxième recherche permet de récupérer une autre information dans un autre fichier de données (LogicielCompta ici) en fonction d'un devis.

Pour continuer, si on trouve un résultat sur la deuxième recherche, on bloque l'enregistrement de la table en question pour éviter à deux utilisateurs de modifier les informations en même temps et ainsi éviter d'éventuels conflits puis on ouvre la fenêtre d'informations du devis.

A la fin de l'initialisation de cette fenêtre, on exécute cette ligne de code :

```
Fin d'initialisation de FEN_Fiche_Devis
FichierVersEcran()
```

Elle nous permet de récupérer les informations en base de données et de les afficher directement dans les champs reliés au fichier de données.

Suite à cette fenêtre, j'ai continué de travailler sur ce projet fictif et j'ai développé une fenêtre pour créer un nouveau devis et un bouton pour supprimer un devis. La fenêtre de création d'un devis est la même que celle de modification, mais pour pouvoir l'utiliser en « mode création » j'exécute ce code sur le clic du bouton :

```
Clic sur BTN_Nouveau
HRAZ(DEVIS)

Ouvre(FEN_Fiche_Devis)
TableAffiche(Table_REQ_SELECT_DEVIS, taCourantEnreg)
```

Avant d'ouvrir la fenêtre, on fait appel à la fonction HRAZ() pour indiquer au programme que l'on compte créer un nouvel enregistrement dans le fichier de données passé en argument de cette fonction et donc à l'ouverture tous les champs seront vides car nous créons un nouvel enregistrement. Ensuite à la fermeture de la fenêtre on rafraichit le tableau regroupant tous les devis afin de voir notre nouveau devis.

Pour pouvoir enregistrer nos données en base on utilise ce code :

```

Clic sur BTN Modifier
EcranVersFichier()
SI DEVIS..NouvelEnregistrement ALORS
    HAjoute(DEVIS)
SINON
    HModifie(DEVIS)
FIN
Ferme()

```

Comme vous pouvez le voir, il est utilisable aussi bien en modification qu'en création puisque nous captions si l'enregistrement que nous sommes en train d'éditer est un nouvel enregistrement grâce à la combinaison de la fonction HRAZ() utilisée précédemment et de la propriété ..NouvelEnregistrement. Dans ce cas précis, on ajoute nos données dans le fichier de données sinon on modifie simplement l'enregistrement en cours.

Comme je vous le disais plus tôt, j'ai également développé un bouton pour pouvoir supprimer un devis dans le tableau et en base, bouton dont voici le code :

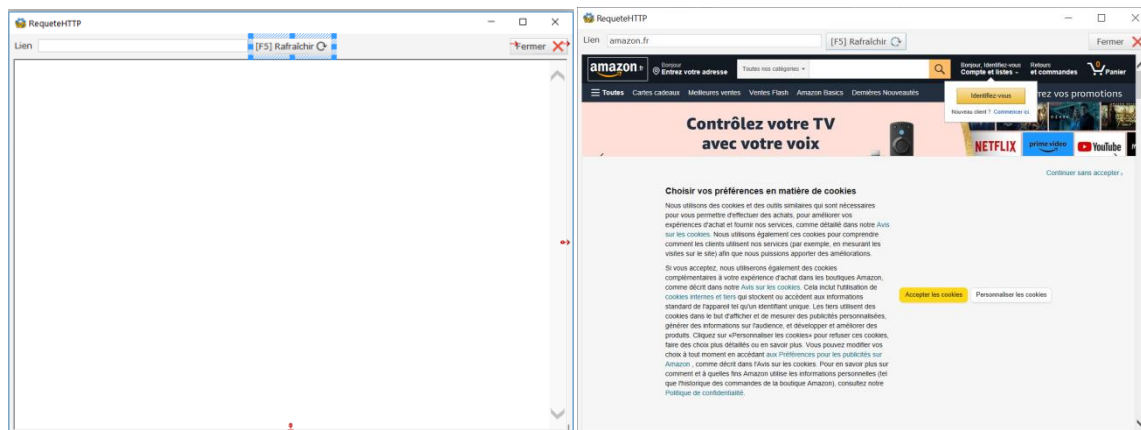
```

Clic sur BTN_Suppr
Si Erreur :
SI OuiNon("Voulez-vous supprimer ce devis ?") ALORS
    HlitRecherche(DEVIS, IDDEVIS, IDDEVIS)
    SI Htrouve(DEVIS) ALORS
        Hlit()
        HSupprime()
        TableAffiche(Table_REQ_SELECT_DEVIS)
    FIN
FIN

```

Simplement, le programme demande à l'utilisateur s'il souhaite supprimer le devis. Si la réponse est bien « Oui », alors on effectue une recherche sur le devis en question et si on le trouve on le supprime, il ne reste plus qu'à rafraichir le tableau pour voir qu'il a bien été supprimé.

Ensuite, j'ai dû développer une autre fenêtre qui permettait à l'utilisateur d'afficher une page internet directement au sein de l'application. Pour ce faire, dans la nouvelle page, j'ai inséré un champ HTML. Ce champ intégré à WinDev permet d'afficher des pages HTML ou bien du code. Et pour pouvoir afficher une page internet il suffit simplement d'assigner un lien au champ HTML.



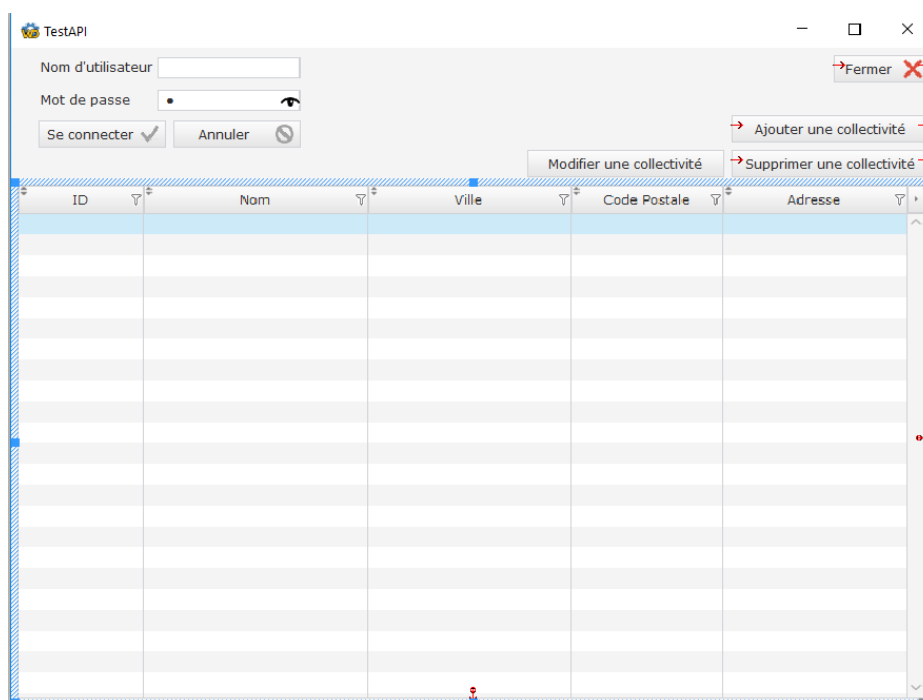
Ici vous pouvez donc voir la fenêtre vierge et cette même fenêtre affichant une page internet sur laquelle nous pouvons naviguer. Le code permettant ceci se trouve dans le bouton « Rafraîchir » et voici son contenu :

```
//LanceAppliAssociée("https://amazon.fr")
HTML1 = "https://" + SAI_Lien
```

Comme vous pouvez le voir, le code est extrêmement simpliste et parfaitement fonctionnel.

Ma dernière réalisation sur ce projet fictif était de créer des scripts en PHP avec VS Code, permettant de capter des données d'une base de données externe et de les afficher dans l'application. Autrement dit, il m'a été demandé d'utiliser les API et d'apprendre à utiliser les requêtes http avec WinDev.

Pour pouvoir réaliser cette tâche, une nouvelle fenêtre était nécessaire, avec un tableau rempli par programmation. Le résultat final ressemblait donc à ceci :



Vous voyez donc maintenant tout un tas de fonctionnalités que je n'ai pas encore pris le temps de vous présenter.

Pour résumer, les champs en haut à gauche permettent de s'authentifier sur l'API et permettent donc d'effectuer des requêtes dessus, et les boutons à droite permettent d'envoyer chacun une requête http différentes sur l'API et donc de modifier le contenu du tableau.

A l'appui sur le bouton de connexion, on déclenche une première requête http permettant de récupérer les données dont il est questions puis de les afficher dans le tableau. Cette requête ressemble donc à ceci :

```
h est un httpRequête
h.Méthode = httpGet
h.URL = "http://localhost/script_garde_cant.php"
h.ContentType = "application/json"
h.Utilisateur = SAI_User
h.MotDePasse = SAI_MotDePasse
.....
r est un restRéponse = RESTEnvoie(h)

RéponseVariant est un Variant

RéponseVariant = JSONVersVariant(r.Contenu)
POUR TOUT Collectivite DE RéponseVariant
    TableAjouteLigne(Table_Collectivite_API,Collectivite.id, Collectivite.Nom, Collectivite.Ville, Collectivite.CodePostal, Collectivite.Adresse)
FIN
```

On déclare donc une variable de type « httpRequête », cette variable hérite donc de plusieurs propriétés qui permettent de définir les différents paramètres pour une requête http comme l'url, la méthode ou bien encore l'utilisateur.

Une fois nos informations complétées, on utilise la fonction restEnvoie (ou httpEnvoie) pour envoyer notre requête et récupérer la réponse du serveur dans notre variable (ici « r »).

Comme pour tout API, la réponse est au format JSON donc nous formatons ce JSON afin d'obtenir des chaînes de caractères que nous pouvons exploiter et que nous pouvons ajouter dans les différentes colonnes de notre tableau avec la fonction TableAjouteLigne().

Voilà donc pour une requête GET, passons maintenant à la méthode POST dont voici l'exemple mis en œuvre dans le projet fictif :

```

NouvelleCollectivite est une chaîne
NouvelleCollectivite = [
{
    "id": "1",
    "Nom": "Communauté d'agglomération Baie de Somme ",
    "Ville": "ABBEVILLE",
    "CodePostal": "80100",
    "Adresse": "Garopôle \r\nPlace de la Gare"
}
]
.....
MaReq est un restRequête
LaRéponse est un restRéponse

MaReq.URL = "http://localhost/script_garde_cant.php"
MaReq.Méthode = httpPost
MaReq.Contenu = NouvelleCollectivite
MaReq.ContentType = "application/json"
MaReq.Utilisateur = "test"
MaReq.MotDePasse = "test"

LaRéponse = RESTEnvoie(MaReq)

```

Dans le même principe qu'une requête GET, on déclare une variable de type requête (http ou REST) pour hériter des différentes propriétés nécessaires à l'élaboration d'une requête que l'on envoie pour ensuite récupérer une réponse. Le traitement de cette réponse est similaire à la requête GET. A titre d'exemple, ici j'ai déclaré une variable de type chaîne qui correspond à un modèle de collectivité que l'on peut retrouver en base de données afin de pouvoir tester le bon fonctionnement de ma requête.

Je vais m'arrêter ici pour la présentation des requêtes http, le fonctionnement étant le même pour les autres, sauf pour la méthode DELETE qui n'est pas totalement intégré à WinDev et qui par conséquent ne fonctionne pas.

Pour terminer cette présentation du projet fictif, je vais vous expliquer le fonctionnement de la connexion.

Tout simplement, dans le script PHP, on a cette ligne de code :

```

if ( ($_SERVER ['PHP_AUTH_USER'] == 'test' && ($_SERVER ['PHP_AUTH_PW'] == 'test' )))

```

Bien que très basique, cette ligne permet d'indiquer que pour effectuer une requête sur l'API on attend un nom d'utilisateur et un mot de passe (tous deux définis ici sur « test »).

C'est une façon extrêmement simpliste de « fermer » l'API à tous ceux qui n'ont pas les informations de connexion, toutefois l'API n'est pas sécurisée pour autant.

```

case 'GET': //Pour récupérer toutes les collectivités
if ( ($_SERVER ['PHP_AUTH_USER'] == 'test' && ($_SERVER ['PHP_AUTH_PW'] == 'test' ))) {

    try{
        $bdd = new PDO('mysql:host=localhost;dbname=gardecant;charset=utf8', 'root', '');
    }
    catch (Exception $e){
        die('Erreur : '.$e->getMessage());
    }

    header('content-type:application/json');

    $reponse = $bdd -> query('SELECT IDDOSSIER, Nom, Ville, Code_Postal, Adresse FROM collectivite');

    foreach($reponse as $collectivite) {
        $json[] = array(
            'id'          => $collectivite['IDDOSSIER'],
            'Nom'         => $collectivite['Nom'],
            'Ville'       => $collectivite['Ville'],
            'CodePostal'  => $collectivite['Code_Postal'],
            'Adresse'     => $collectivite['Adresse']
        );
    }
    $jsonstring = json_encode($json, JSON_UNESCAPED_UNICODE);
    echo $jsonstring;
}

```

Et voici un exemple de comment est géré l'envoi de données coté PHP. Dans le cas où la requête est de méthode « GET », on vérifie les données de connexion, si tout est bon on se connecte à la base de données et on effectue notre requête pour récupérer nos données.

Ensuite pour chaque réponse obtenue par la requête, on insère les données dans un tableau que l'on encode au format JSON.

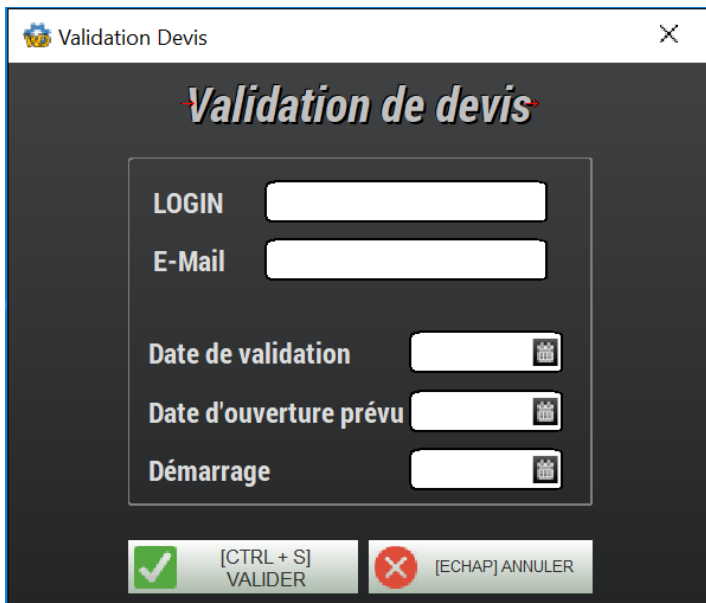
Voici qui clôture donc les travaux sur le projet fictif.

2 – Projet Réel

Après cette phase d'apprentissage et les travaux sur le projet fictif, j'ai donc pu commencer à réellement travailler pour l'entreprise et mes collègues.

Mon maître de stage m'a demandé de corriger une erreur fatale présente dans le projet qui se déclenchait lorsque l'on fermait la page de création d'un nouveau contact. Suite à l'analyse du code déjà présent (et explications de mon maître de stage), il s'est avéré que le problème venait du fait qu'à la fermeture de cette page, nous faisons appel à une fonction appelé HDébloquerNumEnr (qui permet d'éviter aux utilisateurs de rentrer des informations différentes sur un même enregistrement) mais que cette dernière fonctionne de pair avec la fonction HBloque qui n'était pas présente dans le code. Pour corriger ce dysfonctionnement, j'ai donc ajouté la fonction de blocage, lorsque nous ouvrons la fenêtre en mode modification uniquement car c'est à ce moment que nous modifions des données déjà existantes. En mode création cette fonction est donc inutile car nous créons un nouvel enregistrement.

Il m'a ensuite été demandé de créer une nouvelle fenêtre, qui permettrait de valider un devis sans faire tout un tas de manipulation au préalable.



Cette page permet donc de valider un devis donné.

Elle récupère les informations nécessaires (login, e-mail, dates) depuis la base de données en fonction de l'id qui lui est passé et les affiche.

Si les informations conviennent à l'utilisateur, on appuie sur le bouton de validation qui déclenche une série d'instructions qui permettent de considérer un devis comme validé.

Voici donc un extrait du code du bouton de validation.

Logiquement, lorsqu'un devis a déjà été validé, il n'est pas possible de la valider une seconde fois.

```
// Lecture des informations saisies
EcranVersFichier()

// Ajoute / Modifie l'enregistrement
sEmail_Devis = SAI_MailDevis
Login_Devis = SAI_LOGIN
Date_Démarrage = SAI_DateDebut
dDate_Accord = SAI_BonPourAccord_Date
dDate_Ouverture = SAI_DateOuverturePrévu
DEVIS.BonpourAccord = 1

HEnregistre(DEVIS)

Suivi.IDDEVIS = DEVIS.IDDEVIS
Suivi.date = DateSys
Suivi.Observations = "Réception bon pour accord."

HAjoute(Suivi)

// Ferme la fenêtre
Ferme()
```

Pour décharger un peu le tableau regroupant tous les devis existant en base de données, j'ai aussi réalisé un système d'archivage dans l'application.

Pour y arriver, j'ai rajouté un champ de type booléen dans la table des devis ce qui permet donc de définir facilement et rapidement si un devis est archivé ou non.

Si c'est le cas, le devis apparaît donc un tableau prévu à cet effet dans la fenêtre d'archivage et j'ai intégré les mêmes fonctions que pour les devis non archivés afin de pouvoir effectuer des modifications même s'il est archivé.

Dans le but de faciliter et accélérer le travail de mes collègues, on m'a ensuite demandé de travailler sur une fonctionnalité qui permet à l'utilisateur de créer des devis automatiquement depuis un « squelette » existant.

Cette fonctionnalité était déjà existant dans le projet mais n'était pas finalisée et donc non fonctionnelle.

J'ai donc repris le code existant et l'ai modifié afin qu'il puisse créer ces fameux devis automatiquement sur Word. En effet, dans le modèle existant, les valeurs qui changent en fonction des devis sont remplacé par des variables et ensuite le programme se charge de chercher ces variables et de les remplacer par la valeur correspondante.

Enfin, il ne reste plus qu'à enregistrer le devis et l'imprimer ce qui fluidifie et rend bien plus agréable le travail de mes collègues.

Côté programmation, au chargement de cette page, on récupère toutes les informations nécessaires à la création du devis puis à l'appui du bouton « Création du document Word » on utilise une fonction qui permet de rechercher une chaîne de caractère dans le modèle et de la remplacer par la valeur spécifiée. Fonction dont voici un exemple d'utilisation :

```
word>>Selection>>Find>>Execute("@desc",Faux,Vrai,Faux,Faux,Faux,Vrai,1,Faux,Remplace(SAI_Description, RC,Caract(13)),2)
```

Pour continuer dans ce tour d'horizon de mes réalisations, je vais maintenant vous parler du projet d'intégration d'une carte (comme Google Map par exemple) au sein même de l'application.

Cette carte a pour but d'identifier rapidement les éventuels clients à l'autre bout du téléphone lors d'un appel téléphonique. Elle fournit à l'utilisateur des informations concernant une recherche effectuée et permet donc de cibler précisément le client, et de voir les éventuelles communes déjà clientes à proximité. Elle permet aussi à l'utilisateur de voir l'ensemble des communes en base de données et de savoir grâce à la couleur des marqueurs quelles sont les communes ayant simplement fait une demande de devis et celles qui possèdent déjà un compte actif.

Pour réaliser cette fonctionnalité, j'ai utilisé l'API OpenStreetMap (celle de Google Map n'étant pas totalement gratuite) et la bibliothèque JavaScript Leaflet JS.

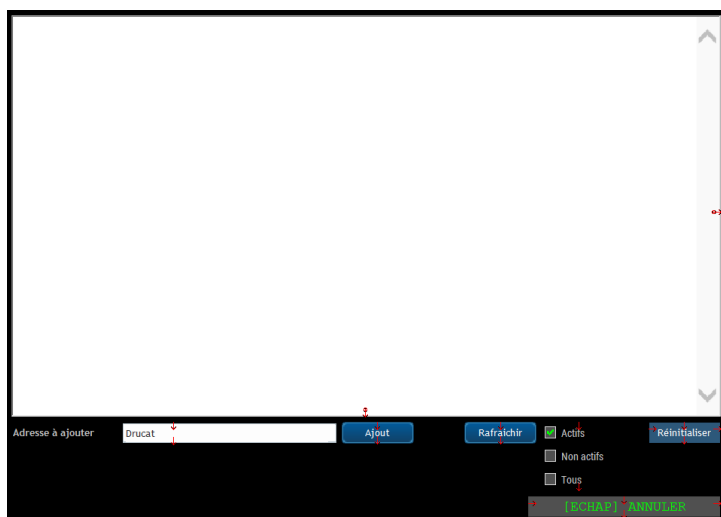
Afin de disposer des marqueurs sur les communes présentes en base de données, il m'a fallu faire une recherche géo localisée sur ces communes. Cette recherche renvoie donc des coordonnées GPS (latitude et longitude) et un marqueur est placé à l'endroit où renvoie ces coordonnées.

Dans un premier temps, la carte effectuait une recherche sur l'intégralité des communes en base à chaque rafraichissement de la carte ce qui avait donc pour effet de créer de gros ralentissements et long temps d'attente avant de voir la carte s'afficher avec les marqueurs.

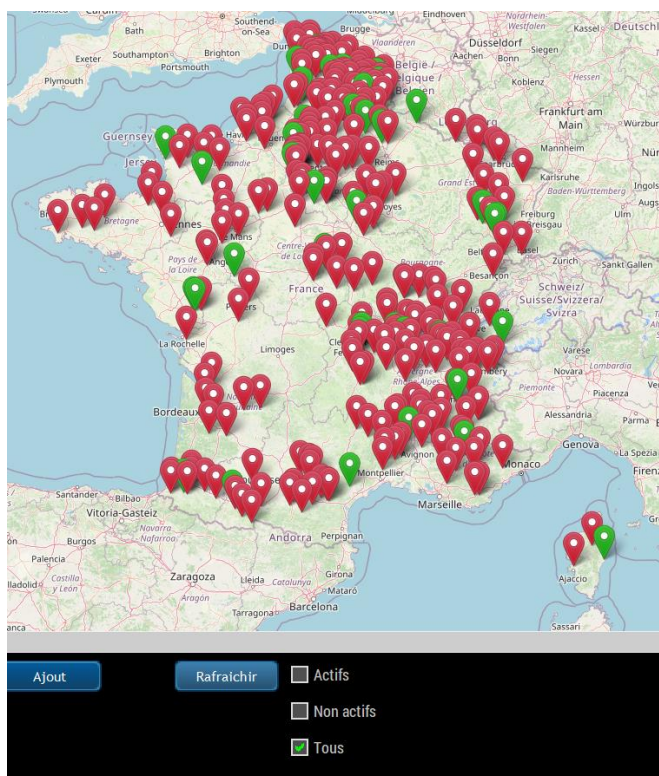
Pour corriger ce problème, j'ai réécrit le programme de manière à ce que si un devis ne possédait pas de coordonnées GPS en base, une recherche était effectuée pour le devis en question puis les coordonnées sont enregistrées en base de données pour pouvoir enfin positionner un marqueur à l'endroit désigné. Et si un devis possède déjà des coordonnées, alors un marqueur est placé directement sur la position.

Cette condition a donc eu pour effet de réduire considérablement le chargement de la carte et donc d'optimiser l'application.

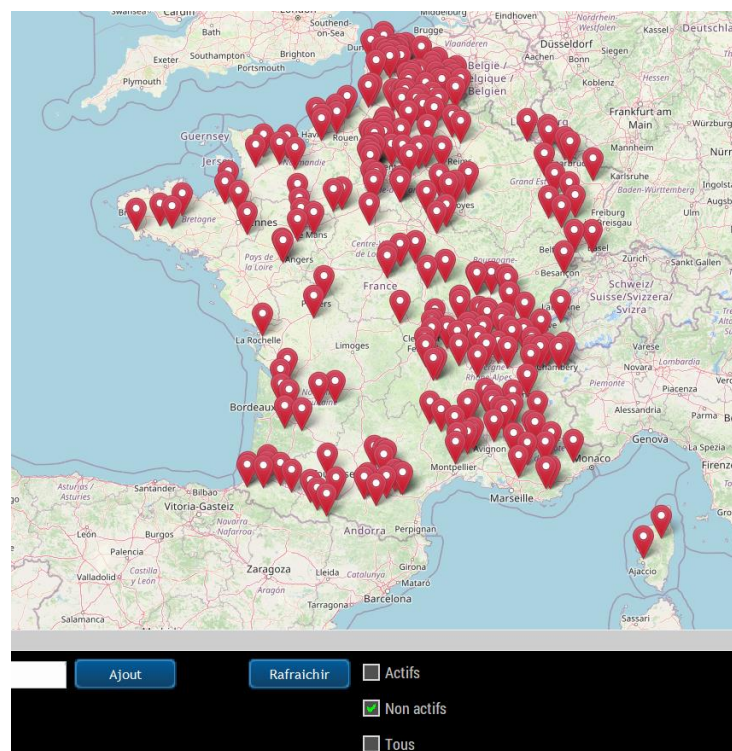
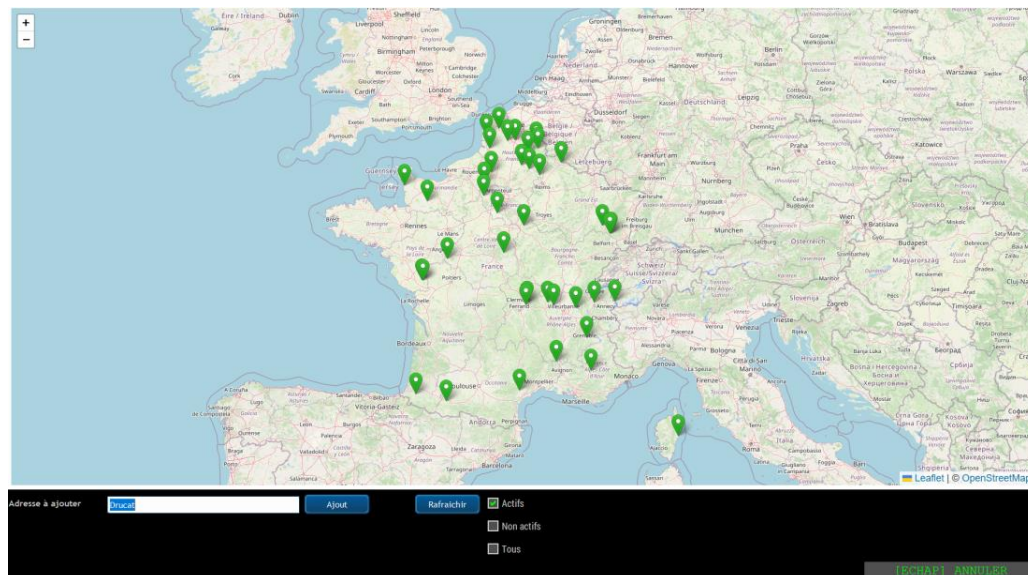
Voici donc la fenêtre de la carte.



Une fois chargée voici à quoi cela peut ressembler :



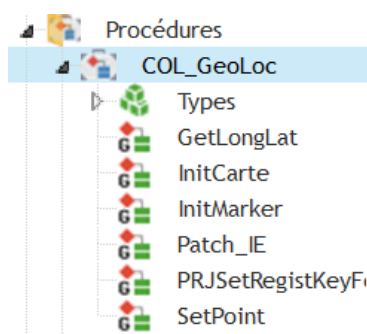
Et enfin les différents états possibles de la carte :



Vous aurez sans doute remarqué que la couleur des marqueurs change en fonction de la case cochée dans le sélecteur en dessous, c'est ce qu'il m'a été demandé d'implémenter par la suite.

En effet, mes collègues voulaient que l'on puisse faire la différence entre les devis dits « acceptés » et ceux qui sont juste resté à l'état de devis. Les devis acceptés correspondent donc aux marqueurs verts et les rouges aux devis non acceptés. Aussi, lorsqu'on ajoute un marqueur manuellement avec la recherche un marqueur bleu apparaît sur la carte ce qui indique que le marqueur est temporaire.

Côté programmation, on a donc plusieurs scripts JavaScript et tout un tas de procédures déclarées dans une collection de procédures dans le projet.



Chaque procédure correspond à une fonctionnalité de la carte. Pour exemple la procédure intitulée « GetLongLat » permet d'obtenir les coordonnées GPS d'un lieu.

```

SI $AdressToFind<>"" ALORS
  $MyRequest = ChaîneVersUTF8("https://nominatim.openstreetmap.org/search?q="+$AdressToFind+"&limit=1&format=json")
  HTTPRequête($MyRequest)
  MyResult=HTTPDonneRésultat()
  vMyData=JSONVersVariant(MyResult)
  stGeoResult.$Geo_Lib=vMyData[1].display_name
  stGeoResult.rGeo_Lat=vMyData[1].lat
  stGeoResult.rGeo_Lon=vMyData[1].lon
  stGeoResult.rGeo_Trust=vMyData[1].importance
FIN
RENOYER stGeoResult

```

Ci-dessus une partie du code de GetLongLat. On récupère le lieu à chercher dans « AdressToFind » qu'on transmet à « nominatim » qui est un outil de recherche lié à OpenStreetMap qui nous renvoie tout un tas d'informations dont les coordonnées GPS dont nous nous servons plus tard pour initialiser des marqueurs sur la carte.

Viens ensuite le temps de déployer le travail produit sur les postes de mes collègues afin de tester les différentes fonctionnalités que j'ai pu créer sur serveur.

C'est à cet instant que nous avons pu constater que la création de devis automatique, qui pourtant était parfaitement fonctionnelle en local, ne marchait pas sur serveur. Si le document Word destiné à devenir le devis final était créé correctement, les variables à l'intérieur n'étaient pas remplacées du tout.

Après quelques recherches sur ce sujet, il s'est avérée que la fonction utilisée jusqu'à présent (l'objet « OLE ») ne marchait pas sur serveur, j'ai donc dû repenser et revoir complètement la manière dont les choses étaient gérées.

Pour pouvoir remplacer l'objet OLE et réussir à utiliser Word depuis WinDev, j'ai utilisé les fonctions Doc... de WinDev directement (DocOuvre, DocRemplace,...).

```
fCopieFichier(sFichierACopier,sFichierCopié)
Doc est un Document = DocOuvre(sFichierCopié)
DocRemplace(Doc,"@NomOption",SAI_Nom_de_l_option)
```

Ces fonctions, en plus de réduire la taille de la ligne, réduisait également les temps de chargement de la fenêtre.

En effet, l'objet OLE permettait de piloter un logiciel externe (Word en l'occurrence) mais son exécution prenait donc plus de temps et les temps d'attentes pour l'utilisateur s'en trouvaient rallongés. Avec les fonctions intégrées de WinDev, ces temps d'attentes sont quasiment supprimés et l'expérience utilisateur devient bien plus agréable.

Après toutes ces modifications, nous avons donc déployés la mise à jour de l'application sur les autres postes et nous avons pu constater son bon fonctionnement.

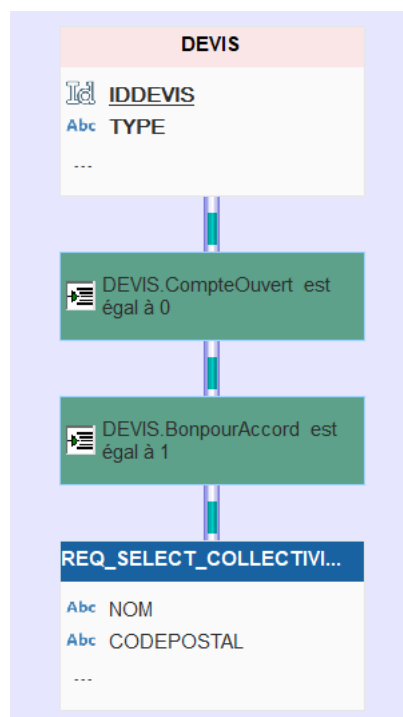
Pour poursuivre, j'ai développé une fonctionnalité qui permet à l'utilisateur de créer des comptes clients directement sur le site depuis l'application. Le logiciel se connecte donc à la base de données via PDO en PHP, une fois connecté on peut effectuer des requêtes pour réaliser des actions sur la base. Une fois les informations insérées, elles sont récupérées sur le site et le compte client est créé. Cela produit donc un gain de temps considérable car l'utilisateur n'a plus à alterner entre le site et l'application pour récupérer ce dont il a besoin afin de créer le compte client puisque tout est géré de manière automatique. Certaines informations sont donc déjà présentes en base de données depuis l'application et d'autres sont définies dans le programme ou dans la base de données du site comme valeur par défaut.

La fenêtre de cette page ressemble donc à ceci :



Cette fenêtre était déjà existante avant mon arrivée, j'ai donc repris le travail et l'ai complété.

Ce tableau affiche donc uniquement les devis dont le compte est ouvert et dont le bon pour accord a bien été reçu. Une fois ces deux conditions remplies, on peut considérer que le devis est accepté et que le compte client est prêt à être créer sur le site. Ce tableau est rempli par cette requête SQL :



On a également la présence d'un bouton « Modifier » pour pouvoir modifier les informations du devis si nécessaire avant de créer le compte et donc aussi le fameux bouton pour la création.

Pour pouvoir réaliser cette action, dans le programme nous avons cette ligne qui permet de récupérer les informations du devis en question :

```
HLitRecherche(DEVIS, IDDEVIS, COL_IDDevis)
```

On effectue simplement une recherche dans le fichier de données « DEVIS » sur la rubrique « IDDEVIS » avec la valeur « COL_IDDEVIS » qui correspond à la valeur de l'id du devis dans le tableau.

Cette recherche nous permet alors de récupérer les informations du devis dont on a besoin et les exploiter. Pour ce faire, on crée donc une variable de type « Variant » qui est en fait un objet dans lequel nous pouvons stocker les informations dans une seule et même variable un peu à la manière d'un JSON.

```
ContenuAAjouter est un Variant
ContenuAAjouter.Nom = DEVIS.TITRE_DOC
ContenuAAjouter.Ville = DEVIS.VILLE
ContenuAAjouter.Code_Postal = DEVIS.CODEPOSTAL
ContenuAAjouter.Adresse = DEVIS.ADRESSE
ContenuAAjouter.LOGIN = DEVIS.LOGIN
ContenuAAjouter.Mail = DEVIS.MailDevis

HLitRecherche(Contact, IDDEVIS, COL_IDDevis)
ContenuAAjouter.Telephone = Contact.Telephone
SI Contact.Telephone = "" ALORS
    ContenuAAjouter.Telephone = Contact.Portable
FIN
SI DEVIS.MailDevis = "" ALORS
    ContenuAAjouter.Mail = Contact.Mail
FIN

ContenuAAjouter.dateCreation = DateDuJour()
ContenuAAjouter.Contrat_Assistance = DateDuJour()
ContenuAAjouter.first_open = 1
ContenuAAjouter.type_reservation_cantine = "S"
ContenuAAjouter.dt_butoir_jour = 1
ContenuAAjouter.dt_butoir_heure = "090000"
ContenuAAjouter.dt_butoir_numero_jour = 4
ContenuAAjouter.LiaisonCodeArticle = "7067"
ContenuAAjouter.LiaisonCodeArticle_Garderie = "7067"
ContenuAAjouter.LiaisonCodeArticle_TAP = "7067"
ContenuAAjouter.LiaisonCodeArticle_Transport = "7067"

SI DEVIS.Mode = 2 ALORS //Si Facturation Fin de Mois
    ContenuAAjouter.ModeEnvoi_Facture = 2
    ContenuAAjouter.FACTURATION_Pas_ExtraiteRoleExecutoire = 1
    ContenuAAjouter.min_facturation = 15
    ContenuAAjouter.option_report = 1
FIN
```

Vous pouvez voir que nous effectuons une deuxième recherche sur le fichier de données « Contact », ceci dans le but de pouvoir récupérer les informations des personnes associées au devis et de les associer à notre variant.

Ce variant est ensuite changé en JSON puis est envoyé par requête http à notre script PHP afin d'y être traité puis d'enfin pouvoir effectuer notre création.

```
IDJSON est un JSON
IDJSON = VariantVersJSON(ContenuAAjouter)

MaReq est une restRequête
MaReq.URL = Paramètres.SourceDonnees+"api_rest/collectivites/Creer.php"
MaReq.Méthode = httpPost
MaReq.Contenu = IDJSON
MaReq.ContentType = "application/json"
MaReq.AuthToken = MonToken

reponse est une restRéponse = RESTEnvoie(MaReq)

SI reponse.CodeEtat = 201 ALORS

    DEVIS.CompteOuvert = Vrai
    HEnregistre(DEVIS)

    Suivi.IDDEVIS = DEVIS.IDDEVIS
    Suivi.date = DateSys
    Suivi.Observations = "Ouverture de compte."

    HAjoute(Suivi)
    Info("Collectivité ajoutée")

    HExécuteRequête(REQ_SELECT_COLLECTIVITE_A_CREER)
    TableAffiche(TABLE_REQ_SELECT_COLLECTIVITE_A_CREER)
```

Comme je vous le disais juste avant, nous transformons notre variant en JSON grâce à la fonction VariantVersJSON() de WinDev ce qui nous permet par la suite de pouvoir envoyer notre contenu dans la requête http et comme présenté plus tôt dans ce rapport nous envoyons le tout à notre script PHP et si le code de la réponse vaut 201 (ce qui équivaut à une création réussie) nous pouvons alors effectuer toutes sortes d'actions et informer que la création à bien été exécutée.

```

public function creer()
{
    // Ecriture de la requête SQL en y insérant le nom de la table
    $sql = "INSERT INTO " . $this->table . " (Nom, Ville, Code_Postal, Adresse, Telephone, Mail, dateCreation,
VALUES (:Nom, :Ville, :Code_Postal, :Adresse, :Telephone, :Mail, :dateCreation, :LOGIN, :option_report, :m
INSERT INTO parametreccantine (IDDossier) VALUES (LAST_INSERT_ID());
INSERT INTO parametrescompta (IDDossier) VALUES (LAST_INSERT_ID());
INSERT INTO transport_parametres (IDDossier) VALUES (LAST_INSERT_ID());

    // Préparation de la requête
    $query = $this->connexion->prepare($sql);

    // Ajout des données protégées
    $query->bindParam(":Nom", $this->Nom);
    $query->bindParam(":Ville", $this->Ville);
    $query->bindParam(":Code_Postal", $this->Code_Postal);
    $query->bindParam(":Adresse", $this->Adresse);
    $query->bindParam(":Telephone", $this->Telephone);
    $query->bindParam(":Mail", $this->Mail);
    $query->bindParam(":dateCreation", $this->dateCreation);
    $query->bindParam(":LOGIN", $this->LOGIN);
    $query->bindParam(":option_report", $this->option_report);
    $query->bindParam(":min_facturation", $this->min_facturation);
    $query->bindParam(":dt_butoir_jour", $this->dt_butoir_jour);
    $query->bindParam(":dt_butoir_heure", $this->dt_butoir_heure);
    $query->bindParam(":type_reservation_cantine", $this->type_reservation_cantine);
    $query->bindParam(":Contrat_Assistance", $this->Contrat_Assistance);
    $query->bindParam(":first_open", $this->first_open);
}

```

Voici, ci-dessus, un petit extrait du script PHP afin de mieux saisir son fonctionnement. Nous avons donc sous les yeux, une fonction « créer » qui effectue une simple requête « INSERT INTO » SQL avec des paramètres afin d'éviter les injections SQL et donc de sécuriser notre requête.

Il est bon de préciser que les données que nous passons en paramètres sont récupérées d'un autre script PHP que voici :

```

// On récupère les informations envoyées
$donnees = json_decode(file_get_contents("php://input"));

if (!empty($donnees->Nom) && !empty($donnees->Ville) && !empty($donnees->Code_Postal))
{
    // Ici on a reçu les données
    // On hydrate notre objet
    $collectivite->Nom = $donnees->Nom;
    $collectivite->Ville = $donnees->Ville;
    $collectivite->Code_Postal = $donnees->Code_Postal;
    $collectivite->Adresse = $donnees->Adresse;
    $collectivite->Mail = $donnees->Mail;
    $collectivite->LOGIN = $donnees->LOGIN;
    $collectivite->Telephone = $donnees->Telephone;
    $collectivite->dateCreation = $donnees->dateCreation;
    $collectivite->Contrat_Assistance = $donnees->Contrat_Assistance;
    $collectivite->first_open = $donnees->first_open;
    $collectivite->type_reservation_cantine = $donnees->type_reservation_cantine;
    $collectivite->dt_butoir_jour = $donnees->dt_butoir_jour;
    $collectivite->dt_butoir_heure = $donnees->dt_butoir_heure;
    $collectivite->dt_butoir_numero_jour = $donnees->dt_butoir_numero_jour;
    $collectivite->LiaisonCodeArticle = $donnees->LiaisonCodeArticle;
}

```

Nos données envoyées depuis l'application WinDev sont captées avec `file_get_contents()` puis décodées avec `json_decode`.

Après cela nous avons à disposition des chaînes de caractères manipulables à souhait.

Comme indiqué, on assigne ensuite ces différentes données à notre objet « Collectivite » afin de réaliser l'insertion désirée. Lors du développement de cette fonctionnalité, j'ai fait face à plusieurs problèmes. Le premier était que la création ne s'effectuait pas correctement pour la simple et bonne raison que nous n'envoyions pas assez de données pour remplir l'intégralité des champs de la table alors ceux qui ne possédaient pas de valeur par défaut et que nous ne renseignions pas ne recevaient rien et cela bloquait la requête donc le compte n'était pas créé. Ensuite, le second souci était que les fonctions PHP `htmlspecialchars()` et `strip_tags()`, généralement utilisées en PHP pour sécuriser des données rentrées par des utilisateurs et les empêcher d'insérer des éléments html comme (`<`, `/`, `\`,...), bloquaient également la création du compte.

Une fois les différents problèmes réglés, mon maître de stage m'a précisé que lors de l'insertion des données, il fallait également récupérer l'identifiant associé au compte fraîchement créé et l'insérer dans 3 autres tables de la base de données. Pour réaliser ceci, j'ai simplement utilisé l'instruction SQL : `LAST_INSERT_ID()`. Cette instruction permet de récupérer simplement et facilement le dernier id (en incrémentation automatique) qui a été inséré dans n'importe que table de la base de données. De fait, avec ceci, il était aisé d'insérer un id dans d'autres tables, j'ai simplement écrit 3 autres requêtes « INSERT INTO » sur les tables concernées et inscrit cette instruction dans « VALUES ». Après avoir vérifié que tout fonctionnait comme convenu, nous avons donc mis à jour le logiciel et déployé cette fonctionnalité sur les autres postes de l'entreprise.

Ensuite, le prochain travail qu'il m'ait été demandé de réaliser consistait en la création d'une nouvelle fenêtre qui regrouperait tous les devis existants et qui les classerait en fonction de leur département.

Le tableau se découperait donc en plusieurs parties, séparées par les noms du département auquel le devis appartient, le code postal et le nom de la région. La première chose à faire était donc de récupérer l'intégralité des départements et régions de France et les insérer dans notre base de données.

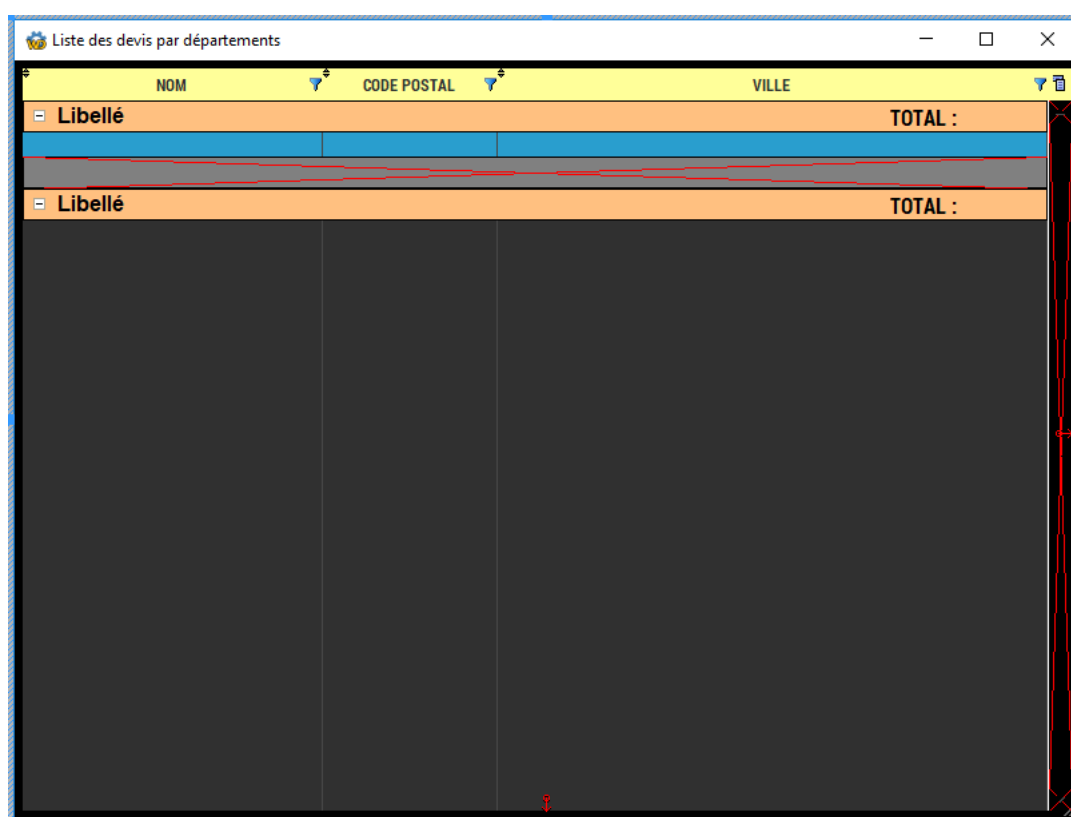
Pour réaliser ceci, je suis parti d'un fichier XLS trouvé sur le web, puis dans l'analyse du projet, j'ai simplement fait un glisser-déposer à l'intérieur. Cela a eu pour effet d'ouvrir un assistant qui a converti les données du fichier XLS en données HFSQL tout en créant un nouveau fichier de données dont nous pouvons nous servir ensuite.

Après avoir récupérer tous nos départements, j'ai créé le tableau qui regroupera l'intégralité des devis de la base de données, puis à l'aide d'une requête SQL, d'une boucle et la fonction `TableAjouteLigne` je l'ai rempli. A cette étape, nous disposons donc de tous les devis mais ceux-ci n'étaient pas encore classés. Pour pouvoir réaliser ce fameux classement, il fallait donc définir dans notre tableau des séparateurs que l'on appelle : des ruptures.

Pour pouvoir définir ces ruptures, il faut les baser sur une colonne de notre tableau. Donc notre classement peut-être réaliser aussi bien sur le nom ou bien toute autre donnée. En l'occurrence notre tri était effectué sur les codes postaux (2 premiers chiffres).

Une fois nos ruptures définies correctement, on leur a ajouté un libellé qui correspond en fait au code postal suivi du nom du département et de la région. Toutes ces caractéristiques sont définies dans le code du libellé, pour récupérer les données correspondantes au codes postaux de nos ruptures on effectue une recherche dans notre fichier de données crée précédemment avec la fonction `HLitRecherche` et basant la recherche sur le code postal.

Ceci a pour effet de récupérer nos tout ce qu'il nous faut, il ne nous reste alors plus qu'à assigner ces données au libellé de nos ruptures et ainsi donc notre classement était enfin effectif. Voici donc le rendu final :



NOM	CODE POSTAL	VILLE
Libellé		
TOTAL :		
Libellé		
TOTAL :		

Pour poursuivre, j'ai ensuite développé ce que nous appellerons un « Mode maintenance ». Cette nouvelle fonctionnalité permet d'activer ou au contraire de désactiver la maintenance sur le site depuis l'application, plus précisément quand ce mode est activé, le client est redirigé sur une page lui indiquant que le site est en maintenance et l'accès lui est donc refusé. Dans le cas contraire, tout se passe normalement.

Etant donné qu'à la moindre fausse note l'accès au site pouvait être refusé à tous les clients, j'ai donc logiquement travaillé en premier sur un autre domaine afin de tester le bon fonctionnement de ma fonctionnalité.

Pour pouvoir réaliser ce travail, je devais disposer des accès au FTP (File Transfer Protocol) du site, il m'a donc fallu installer le logiciel Filezilla. Ce logiciel permet d'accéder (avec des identifiants de connexion) aux différents répertoires et fichiers d'un site et d'en modifier son contenu. Ainsi on peut supprimer, copier, coller, créer, etc... des fichiers. Avec ces accès j'ai donc pu modifier le fichier nommé « .htaccess » et en modifier son contenu pour pouvoir déclencher ou non le mode maintenance.

Les scripts d'activation et de désactivation m'ont été fournis par mon maître de stage qui, à chaque qu'il souhaitait activer ou désactiver la maintenance, modifiait le .htaccess manuellement depuis le FTP. Pour pouvoir les exploiter il m'a fallu les sauvegarder quelque part, ma première idée étant de les inscrire directement dans le programme en tant que chaîne de caractères. Dans l'absolu, cela ne posait aucun problème dans le fonctionnement mais, ces scripts pouvant toujours être modifiés il m'a été demandé de les ajouter dans la base de données pour pouvoir les modifier si nécessaire sans passer par le programme.

En revanche, ces scripts contenaient l'adresse ip de la connexion en cours, mais ces adresses étant inscrites directement à l'intérieur des fichiers j'ai donc défini à leur place des balises de type « @ip@ » pour que l'utilisateur du logiciel puisse utiliser le mode maintenance sans être affecté depuis n'importe quelle connexion internet et pas uniquement celle de l'entreprise.

WinDev offre nativement des fonctions qui permettent d'interagir avec un serveur FTP ce qui m'a été très utile dans le cas présent. Le programme commence donc par se connecter au serveur avec FTPConnecte puis une fois connecté il commence par récupérer l'adresse ip publique depuis laquelle l'application est utilisée ceci grâce à une requête sur une API qui renvoi l'adresse en question. Ensuite, une boîte de dialogue s'affiche à l'écran en demandant à l'utilisateur s'il souhaite activer ou désactiver le mode maintenance. En fonction de la réponse reçue, on interroge le serveur pour savoir si le fichier « .htaccess » existe déjà dans le FTP. S'il existe on le récupère sur l'ordinateur, on le supprime du serveur, on modifie son contenu avec le script approprié (en fonction de la réponse obtenue précédemment) puis on l'envoi sur le FTP avant de le supprimer de l'ordinateur. Pour indiquer à l'utilisateur si le mode maintenance est activé ou non, on affiche un cercle vert ou rouge (respectivement activé ou désactivé) à côté de l'option qui permet de déclencher le programme. Pour savoir de quelle couleur est ce cercle, on récupère simplement le fichier .htaccess depuis le FTP puis on vérifie s'il contient cette chaîne de caractères : « #ModeMaintenance ».

Comme vous vous en doutez, s'il la chaîne est trouvée c'est que le script actuel est bien celui de la maintenance et dans le cas contraire c'est l'inverse.

Après toutes ces explications barbares, voici donc le programme du mode maintenance :

```
nResConnecte est un entier = FTPConnecte("ftp.cluster011.hosting.ovh.net", "gestioncjq", "Servipus80", 21 )

HlitPremier(Paramètres)

SI nResConnecte <> -1 ALORS
//La connexion a bien été effectuée
//Récupération de l'adresse IP Publique
sAdresseIPPublique est une chaîne = ""

SI HTTPRequête("http://api.ipify.org") ALORS
sAdresseIPPublique = HTTPDonneRésultat(httpRésultat)
FIN
// info(sAdresseIP)

sHtAccess_Activation est une chaîne

Paramètres.ScriptMaintenance = Remplace(Paramètres.ScriptMaintenance, "@ip1@", ExtraireChaine(sAdresseIPPublique,1,"."))
Paramètres.ScriptMaintenance = Remplace(Paramètres.ScriptMaintenance, "@ip2@", ExtraireChaine(sAdresseIPPublique,2,"."))
Paramètres.ScriptMaintenance = Remplace(Paramètres.ScriptMaintenance, "@ip3@", ExtraireChaine(sAdresseIPPublique,3,"."))
Paramètres.ScriptMaintenance = Remplace(Paramètres.ScriptMaintenance, "@ip4@", ExtraireChaine(sAdresseIPPublique,4,"."))

sHtAccess_Activation = Paramètres.ScriptMaintenance
// Info(sHtAccess_Activation)

sHtAccess_Desactivation est une chaîne
Paramètres.ScriptNormal = Remplace(Paramètres.ScriptNormal, "@ip@", sAdresseIPPublique)
sHtAccess_Desactivation = Paramètres.ScriptNormal
// info(sHtAccess_Desactivation)

SELON Dialogue("Voulez-vous activer le mode maintenance ?",["Activer le mode maintenance", "Désactiver le mode maintenance"])

CAS 1 :
Info("Activation du mode maintenance")
//
SI FTPFichierExiste(nResConnecte,"/www/.htaccess") ALORS
FTPRecupere(nResConnecte,"/www/.htaccess",ComplèteRep(Paramètres.CheminEnregistrement))
FTPSupprimeFichier(nResConnecte,"/www/.htaccess")
fSauveTexte(ComplèteRep(Paramètres.CheminEnregistrement)+".htaccess",sHtAccess_Activation)
FTPEnvoie(nResConnecte,ComplèteRep(Paramètres.CheminEnregistrement)+".htaccess","/www/.htaccess")
fSupprime(Paramètres.CheminEnregistrement+".htaccess")
FTPDéconnecte(nResConnecte)
OPT_Mode_Maintenance..Image = "C:\Mes Projets\GESTION-CANTINE\FLS18820a.png"
SINON
Info("Le fichier n'existe pas.")
FIN

CAS 2 :
Info("Désactivation du mode maintenance")
//
SI FTPFichierExiste(nResConnecte,"/www/.htaccess") ALORS
FTPRecupere(nResConnecte,"/www/.htaccess",ComplèteRep(Paramètres.CheminEnregistrement))
FTPSupprimeFichier(nResConnecte,"/www/.htaccess")
fSauveTexte(ComplèteRep(Paramètres.CheminEnregistrement)+".htaccess",sHtAccess_Desactivation)
FTPEnvoie(nResConnecte,ComplèteRep(Paramètres.CheminEnregistrement)+".htaccess","/www/.htaccess")
fSupprime(ComplèteRep(Paramètres.CheminEnregistrement)+".htaccess")
FTPDéconnecte(nResConnecte)
OPT_Mode_Maintenance..Image = "C:\Mes Projets\GESTION-CANTINE\FLS01340.png"
SINON
Info("Le fichier n'existe pas.")
FIN

FIN
```



Arrivant sur la fin de mon stage, j'ai donc ensuite effectué des tâches moins chronophages, parmi lesquelles une « refonte » de la fenêtre des paramètres qui ressemble désormais à ceci :

Fiche Paramètres

Taux Abonnement

1,50

Frais Fixe

139,00 €

Cout Ouverture Compte

377,00 €

Taux Tva

20,00 %

Squelette Exportation Word

C:\Users\user\Documents\testWord\CreaDevis.d

Chemin Enregistrement

C:\Users\user\Documents\DOC GESTION CANTINE\

Source données

http://localhost/

HTACCESS Maintenance

HTACCESS Normal

[CTRL + S] ENREGISTRER

[ECHAP] ANNULER

Ou bien encore la suppression de certaines fenêtres qui étaient devenues inutiles suite aux différentes modifications que j'ai pu ajouter tout le long de ce stage.

Je peux aussi vous parler de la « ToDoList » que j'ai réalisé. Elle se présente sous la forme d'un tableau qui regroupe les données d'un nouveau fichier de données et sur lesquels nous pouvons effectuer des actions basiques comme l'ajout, la modification et la suppression.

Ce tableau permet donc de regrouper des demandes de clients, qui ont pour but d'améliorer le site et décharge donc l'utilisateur de cette tâche. Dans ce fichier de données fraîchement créé, on retrouve donc l'intitulé de la tâche, la date, le délai, le statut et le niveau de priorité de celle-ci.

[illegible]

Comme vous pouvez le deviner, c'est par le biais de la fiche ci-dessus que nous pouvons effectuer les ajouts ou bien les modifications.

Parmi les différents boutons présents sous le tableau, on trouve également un bouton « Active/Désactive filtre ». Ce dernier permet comme son nom l'indique d'utiliser ou non filtre pour mieux distinguer les choses qui sont déjà faites et celles qui ne le sont pas.

```
// Permet d'activer/Désactiver le filtre en utilisant le booléen
SI bFiltre= Vrai ALORS
    TableDésactiveFiltre(TABLE_EvolutionProgramme.COL_Statut)
    bFiltre=Faux
SINON
    TableActiveFiltre(COL_Statut,filtreEgal,0)
    bFiltre=Vrai
FIN
```

Voici le code de ce bouton :

Pour vous le décrire un peu plus en détail, on a une variable globale « bFiltre » de type booléen. Si cette dernière vaut Vrai on désactive le filtre de la table sur la colonne statut. Autrement, on l'active pour n'afficher que les données dont Statut vaut 0 (Statut étant également un booléen) puis on indique que bFiltre vaut Vrai.

Dans le fonctionnement cette fonctionnalité est plutôt simple, mais reste très utile pour l'utilisateur.

3 - FORMATION

Pendant la dernière semaine de mon stage, nous avons reçu au sein de l'entreprise un second stagiaire qui était en stage d'observation de 3^{ème} et j'ai donc pu m'occuper quelques temps de ce jeune homme intéressé par l'informatique et plus particulièrement par le développement web.

Pour débiter son stage, je lui ai parlé de ce que nous faisons dans cette entreprise puis lui ai demandé s'il avait déjà développé et quels langages il connaissait. Chose à laquelle il m'a répondu qu'il avait très peu pratiqué et uniquement en HTML. Comme cela a pu être mon cas moi quand je suis arrivé dans cette entreprise, j'ai décidé de lui faire suivre le manuel d'auto-formation fourni avec WinDev. Bien évidemment je l'ai accompagné pour lui expliquer le fonctionnement du logiciel et les éventuelles erreurs qu'il pouvait faire. De ce fait, il a pu avoir une première approche de WinDev et du WLangage.

Nous avons donc débuté par les bases, le fonctionnement des variables, les différentes fonctions et leurs utilités, les bases de données etc...

En suivant le manuel nous avons créé étape par étape un projet dans lequel nous avons un tableau affichant toute une liste de produits que nous pouvons modifier, supprimer et aussi en créer de nouveaux. Cet exemple fait partie des exercices fournis avec WinDev, il a pour but de permettre à l'utilisateur d'avoir une première approche avec le logiciel de manière très simple et complète, tout en passant sur les différentes fonctionnalités et options du logiciel. L'ensemble étant expliqué par des exemples et des exercices.

Par la suite, j'ai décidé de le faire travailler sur du HTML et un peu de CSS pour qu'il puisse voir d'autres choses. Son stage étant un stage d'observation, j'ai pensé que c'était important qu'il puisse voir un maximum de choses mais aussi qu'il puisse pratiquer également vu que le temps de son stage était court (1 semaine) nous n'avions que très peu de temps. Bien sûr je ne me suis pas occupé seul de ce garçon, mes collègues m'ont aussi donné un coup de main et j'ai été assisté par mon maître de stage.

De mon côté, cette expérience, aussi courte soit-elle, m'a permis d'avoir une première approche du monde de la formation du côté des formateurs. Evidemment le travail était tout autre et bien moins important, mais il m'a quand même permis d'avoir une autre vision de la formation et de me faire mon propre avis sur le travail de formateur.

4 – MAINTENANCE

Aussi je te tenais à préciser que cette partie ne concerne pas le développement à proprement parler, mais plutôt le peu de maintenance que j'ai pu effectuer avec mon collègue Arnaud en clientèle et au bureau directement. Même si cela n'a pas de rapport direct avec le travail de programmation, cela est tout de même pour moi une expérience différente et tout aussi enrichissante, c'est pourquoi j'ai décidé de vous en parler dans ce rapport.

Comme je vous l'expliquais précédemment, j'ai eu plusieurs fois l'occasion de partir en déplacement avec mon collègue afin d'observer son travail et, quand l'opportunité se présentait, de l'assister dans les dépannages. Etant tout autant intéressé par le développement que par le dépannage, j'ai tout de suite accepté quand on m'a proposé de l'accompagner. Cela me permettait également de m'aérer l'esprit quand je bloquais sur certains travaux.

J'ai donc, entre autres, appris comment monter un ordinateur, à partir des différentes pièces détachées qui le compose. Aussi, j'ai pu voir et pratiquer quelques travaux en réseaux, tout simplement nous nous sommes rendus dans un cabinet d'architectes pour y installer une nouvelle imprimante.

Ce cabinet étant composé de plus d'une dizaine d'ordinateurs, une fois l'imprimante neuve installée, il a fallu connecter tous les postes sur cette dernière avec les bons pilotes et, lors de la connexion à l'imprimante, faire attention à ce que les adresses IP des postes ne se chevauchent pas pour éviter d'éventuels conflits entre eux.

Ensuite, et je terminerai cette courte présentation là-dessus, nous avons reçu une commande de deux ordinateurs au format tour et d'un serveur NAS, pour un cabinet fabriquant des prothèses dentaires. Mon collègue a donc passé commande auprès de son fournisseur puis nous avons monté les postes et installé Windows dessus afin de gagner du temps lors de l'installation en clientèle.

Le jour venu, nous sommes donc allés installer les deux postes chez le client, évidemment il possédait des données qu'il souhaitait conserver sur les nouvelles machines, il a donc été nécessaire d'effectuer une sauvegarde des données des anciens pc. Nous avons donc commencé par installer le serveur NAS, puis avons effectué la sauvegarde sur ce même serveur. De fait nous avons centralisé les données des deux ordinateurs puis quand nous en avions besoin nous avons récupéré ce qu'il nous fallait sur les nouvelles machines. Nous avons donc procédé ensuite à l'installation des nouveaux ordinateurs avec les écrans et à la récupération des données depuis le serveur.

Voici donc deux interventions, parmi d'autres, que j'ai pu réaliser durant cette période en entreprise. Comme je vous le disais, ce fut une expérience bien différente du développement mais tout aussi intéressante et enrichissante.

CONCLUSION

N'ayant jamais effectué de stage en entreprise c'était donc pour une première expérience dans ce domaine. Evidemment, l'environnement de travail peut varier de bien des façons selon les entreprises dans lesquels nous sommes, et dans mon cas, Servi-plus étant une entreprise de moins de 10 personnes, l'ambiance était des plus chaleureuses et sympathique tout en restant sérieux. Ces éléments réunis ont grandement facilité mon intégration dans l'entreprise et la réalisation de ce stage.

Je n'ai pas le vécu ni le ressenti d'un stage dans une grosse entreprise mais je pense pouvoir affirmer qu'il est plus simple de trouver sa place et de s'adapter dans une entreprise tel que Servi-Plus, l'environnement de travail étant plus « familial ». Dans une grande entreprise, il faut faire sa place et cela demande nécessairement plus d'efforts, néanmoins je ne dis pas que le travail y est plus désagréable ou que c'est moins chaleureux, juste que c'est différent.

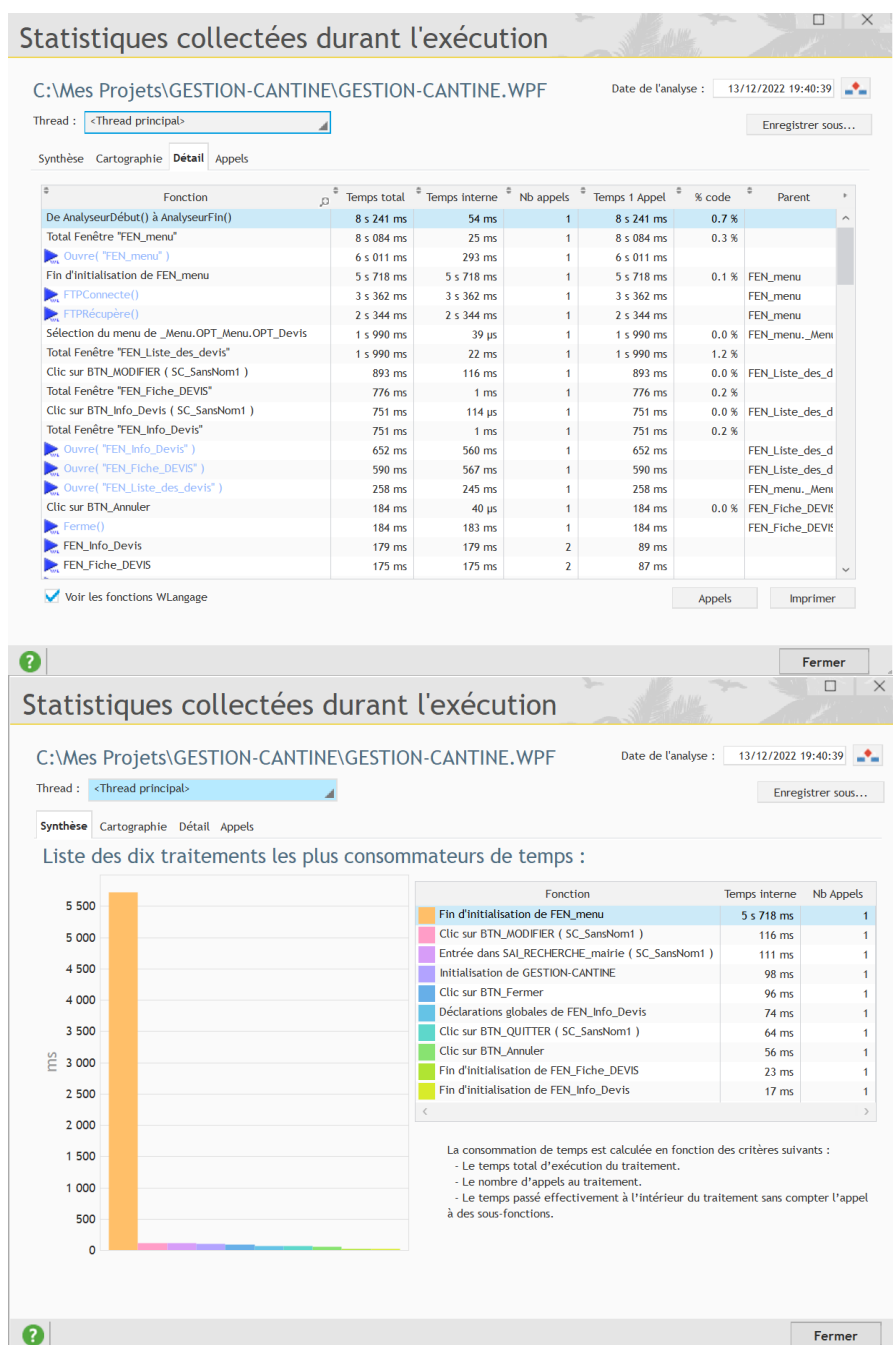
Bien que le logiciel de travail utilisé durant ce stage fût bien différent de celui utilisé lors de la formation, j'ai su apprendre ce qu'il m'était nécessaire de connaître afin d'aider au mieux mes collègues et de réaliser les multiples tâches qui m'ont été confiées. Il faut bien admettre qu'en arrivant face à WinDev et sans aucune pratique antérieure, au départ j'avais quelques réticences mais ces dernières se sont vite dissipées avec l'aide de mes collègues et les explications de mon maître de stage. Bien sûr ce que j'avais pu voir durant la formation n'a pas été du tout inutile pendant ce stage bien au contraire étant donné que j'ai dû utiliser essentiellement du PHP pour les api ainsi que du JavaScript pour le développement de la carte et des requêtes SQL pour communiquer avec les bases de données.

En définitive, je garde de cette période chez Servi-plus un excellent souvenir, tout cela m'a permis d'acquérir de nouvelles compétences et savoir-faire, cela restera donc pour moi une expérience extrêmement enrichissante.

ANNEXES

1- Exemple d'utilisation de l'analyseur de performances.

WinDev embarque avec lui un analyseur de performances, il permet de lancer le projet en cours et calcule les différents temps de réponses des fonctionnalités et fenêtres que le développeur a créées. A la fermeture du test, l'analyseur nous renvoie des graphiques et des tableaux avec les temps de réponse de chacun. En voici un exemple :



2- Exemple d'envoi d'un e-mail

```
Email.Destinataire = "adresse@email.com"  
Email.NbDestinataire = 1  
Email.Sujet = "Titre du mail"  
Email.Message = "Contenu du mail"  
Email.EnvoiMessage()
```

Voici les champs minimaux qu'il est nécessaire de renseigner pour que l'e-mail soit envoyé.

3- Exemple de connexion à une base de données manuellement

Un peu à la manière d'une connexion avec PDO en php, Windev propose également de se connecter à une base de données en utilisant la programmation et pas le centre de contrôle fourni avec le logiciel. Pour ce faire, il y a disposition toute une panoplie de fonctions pour utiliser la programmation, dont voici un extrait :

```
MaConnexion est une Connexion  
// Description de la connexion  
MaConnexion.Utilisateur = "USER"  
MaConnexion.MotDePasse = "PASSWORD"  
MaConnexion.Serveur = "MONSERVEUR"  
MaConnexion.BaseDeDonnées = "Base de données"  
MaConnexion.Provider = hAccèsHFClientServeur  
MaConnexion.Accès = hOLectureEcriture  
MaConnexion.InfosEtendues = "Infos étendues"  
MaConnexion.OptionsCurseur = hCurseurClient  
  
HOuvreConnexion(MaConnexion)
```

REFERENCES

doc.pcsoft.fr

www.php.net

stackoverflow.com

www.commentcamarche.net

www.youtube.com

Manuel d'autoformation / Aide en ligne WinDev

learn.microsoft.com