Rapport de stage

Logiciel de gestion pour la vente et la livraison de bois de chauffage

---

Fabian Tschirhart – Master informatique première année

26 Avril 2021- 28 mai 2021

Faculté des sciences et techniques à Mulhouse

Djamel Meghzili 9 rue d'aspach Cernay

Djamel Meghzili (pas de logo)

Maître de stage: Frederic Cordier

Préambule

Dans le cadre de la formation de Master, il était possible de faire un stage dans le domaine informatique.

C'est pour cela que j'ai contacter l'entreprise de vente de bois de chauffage et de vente de palette qui je savait avait besoin de pouvoir gérer ses ventes de façon plus simple et informatiser.

L'entreprise avait besoin de gérer :

- Enregistrement des clients et accessibilité des clients sous format informatique.
- Ajout/modification/suppressions de livraisons
- Impression de devis/facture concernant une livraison
- Calcul du total des ventes sur une année

Ce stage a été réalisé sans tuteur informatique et entièrement de façon autonome.

Remerciements

Je remercie Djamel Meghzili pour son accueil chaleureux dans son entreprise, sa bonne humeur et son professionnalisme ainsi qu'a sa femme qui m'a accompagné sur la partie conception et qui sera l'utilisatrice principale du programme.

Table des matières

Définition de la partie métier

L'entreprise vend du bois de chauffage mais aussi des palettes mais le logiciel concerne seulement la partie vente de bois de chauffage.

Le bois de chauffage ce vend en stère et en longueur, ces termes étant utilisé voici les définitions Wikipédia ci-dessous :

- *Le stère1 (du grec stereos, solide) est une unité de mesure de volume, valant un mètre cube, utilisée pour mesurer les volumes de stockage de bois de chauffage ou de charpente.*
- *La longueur du bois est la distance entre les 2 coupures, l'épaisseur de bois quand a lui varie beaucoup et n'est pas calculer.*

Ainsi peut importe la longueur du bois acheter par le client, s'il achète un stère il aura la même quantité de bois.

Environnement de travail

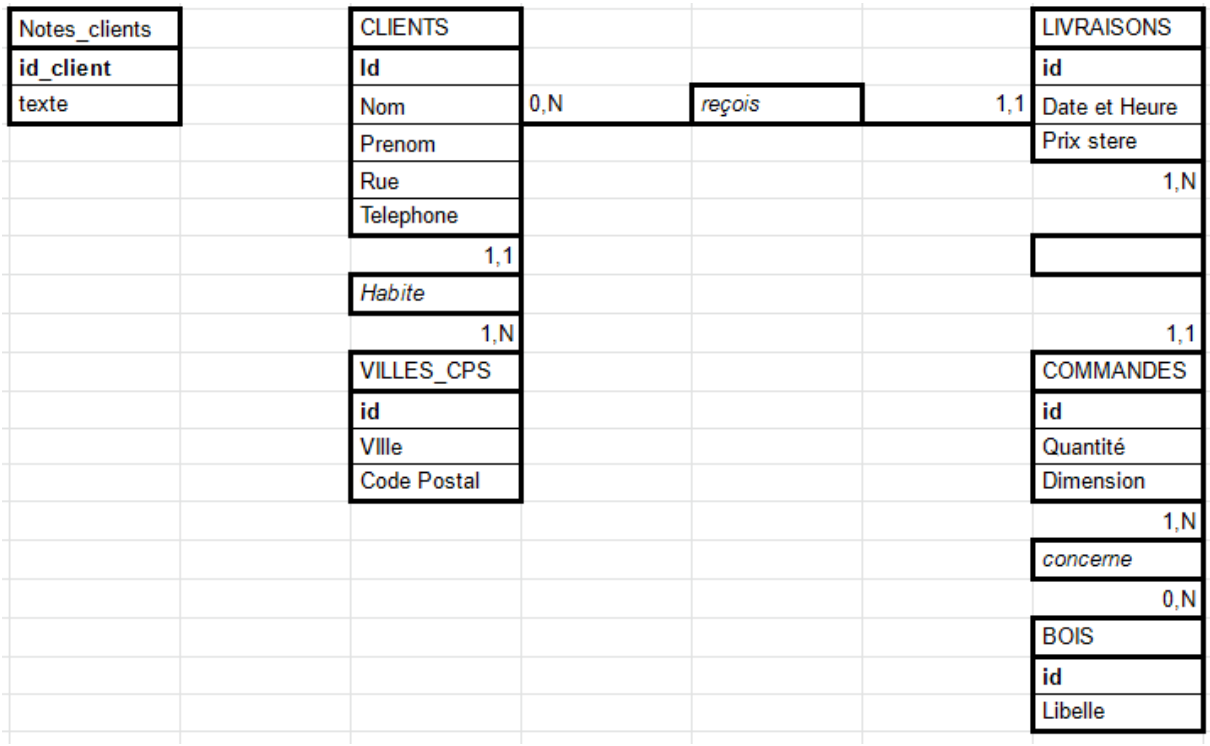
Le travail a été réalise sur un pc fixe et un pc portable et le contact avec l'entreprise était en direct quand cela était nécessaire.

Le logiciel a été développer sur un ordinateur Windows 10 avec l'environnement de travail (IDE) Eclipse avec le langage de programmation java (jre 15) en utilisant Maven qui permet d'ajouter plus simplement des libraires externes et de build plus simplement le projet.

Pour ce qui concerne la partie base de données j'ai utilise WampServer qui héberge la base de données.

Une connexion internet fibré était équiper à l'ordinateur permettant la recherche d'information sur internet et la gestion de la partie google du projet.

Travail sur la base de données



Modèle conceptuel de la base de données 1

Ce modèle est le model final de la base de données, j'ai pensé dès le départ à faciliter la vie concernant le remplissage des **villes** pour l'utilisateur étant donné que c'est souvent les mêmes **villes** ou les **livraisons** sont effectuer.

Comme une livraison peut avoir plusieurs commandes différentes j'ai mis une table **commandes**, sinon j'aurai directement mis les informations de **commandes** dans la table **livraisons.**

Une **commande** n'appartient qu'à une seule **livraison** et une livraison n'appartient qu'a un seul **client**, ainsi on peut retrouver les **livraisons** d'un **client** et les **commandes** d'un **client**.

En cours de route j'ai remarqué que parfois nous avions besoin d'un peut plus d'informations sur les **clients**, c'est pourquoi j'ai rajouté une table **notes_clients** qui doit avoir le même identifiant que le client concerner par cette **note**, la plupart du temps les **clients** n'ont pas de **note**.

Les informations stockées sont nécessaires au bon fonctionnement du programme, les informations du **client** sont nécessaires pour le contacter, son adresse de livraison est nécessaire pour la livraison.

La date et l'heure de la **livraison** est nécessaire pour la programmation des **livraisons**, le prix du stère est nécessaire pour le devis et/ou la facture. Le programme ne gère pas les réductions, c'est l'utilisateur du logiciel qui s'en charge à la main.

La quantité (en stères) et la dimension (en cm) d'une **commande** est aussi nécessaire pour la programmation, les types de bois voulus sont aussi nécessaires même si la plupart du temps les clients n'ont pas de préférence de bois. Les types de **bois** sont au nombre de 4, Hêtre, chêne, charme et Acacias. Dans les rares cas où il pourrait vendre un autre type de **bois** l'utilisateur du logiciel le noterai a part. Mais a priori il ne pense pas vendre d'autres type de **bois**.

Partie programmation, partie non visible.

J'ai utilisé le modèle d'accès au données (DAO) pour accéder aux données, de ce fait j'ai créé une classe par table du modèle et une classe DAO de cette même table qui exploite les données.

Le but étant d'avoir une représentation java de l'objet pour éviter d'avoir à travailler avec des objets SQL dans notre travail.

Chaque classe est construit de cette façon :

- Une propriété privée par propriété de la table du modèle
- Un constructeur sans identifiant utilisée lors de la création d'une instance dans la base de données qui choisit l'identifiant
- Un constructeur avec identifiant pour la lecture et la modification ainsi que la suppression
- Des getters et setter pour chaque propriété
- Pas de setter pour l'identifiant

Pour la classe **clients** on stock aussi la classe ville.

Pour la classe **commande** j'ai stocker son identifiant de livraison et sa liste de bois.

Pour la classe **livraisons** je stock la liste de commandes et je stock l'identifiant de son client.

La ou je stock certains identifiant pour faire le lien, le fait de stocker une classe dans une autre permet de faire moins de requêtes.

Dans la partie DAO le même schéma est utilisé tout le temps

1) Chargement des pilotes
2) Connexion a la base de données
3) Fait une requête
4) Retour au deuxième point tant que l'on utilise l'application

Dans chacune des partie DAO il y a :

- Ajout
- Modification
- Suppression
- Accès a une liste en temps direct

Dans la classe **boisDAO** on gère aussi le lient entre les **bois** et **commandes** étant donné que le lien est géré dans une autre table.

ClientDAO utilise **villeDAO** pour ajouter la **ville** dans la classe du **client**. Ainsi on connait la **ville** du **client** sans avoir à la gérer plus tard dans la vue.

De la même façon **commandeDAO** utilise **boisDAO** et **livraisonDAO** utilise **commandeDAO**.

Pour travailler avec une base de données MySQL j'ai choisi l'extension jdbc.

On utilise avec jdbc :

- Une Connection qui sert à se connecter à la base de données
- Un PreparedStatement qui est une requete préparée qui sert à préparer les requêtes à l'avance
- Et un ResultSet quand l'on a besoin lors que l'on cherche dans la base de données des entités.

Tout ce qui est utiliser est fermer après utilisation car ce sont des instances

Génération d'un PDF

La génération d'un fichier PDF est nécessaire pour les factures et devis.

J'ai fait le choix d'utiliser itextpdf qui m'a permis de rajouter des informations.

Je voulais au départ juste modifier une Template mais cela na pas été possible donc j'ai créé le document de a à z avec des informations qui sont choisit selon le client, la livraison et les commandes.

Le principe est que l'on peut crée un document dans lequel on peut ajouter des paragraphes, des tableau et d'autres choses.

Dans les tableaux on peut ajouter des cellules et dans les paragraphes des textes.

Finalement j'ai codé les fichier PDF comme si je les avais faites à la main sous Word et divisées en plusieurs fonctions pour chaque partie.

Partie planning

Dans la partie planning le choix a été fait de travailler avec google calendar et donc de travailler avec l'interface de programmation (API) de google

Il m'a fallu pour sa crée un projet google et crée un fichier credentials inclus dans le projet.

J'ai donc fait une classe planning qui permet la création d'un évènement sur le calendrier google de la personne connectée.

Lors de la première connexion a l'application l'utilisateur devra se connecter à son compte google et autoriser le logiciel à modifier son calendrier google.

On fait appel à cette classe lors de l'ajout/modification et suppression d'une livraison et pareillement pour les commandes qui lui sont assigner.

Lors d'un changement on supprimer et recrée tout bêtement l'évènement.

Les lignes de codes de l'authentification sont entièrement fournies par google.

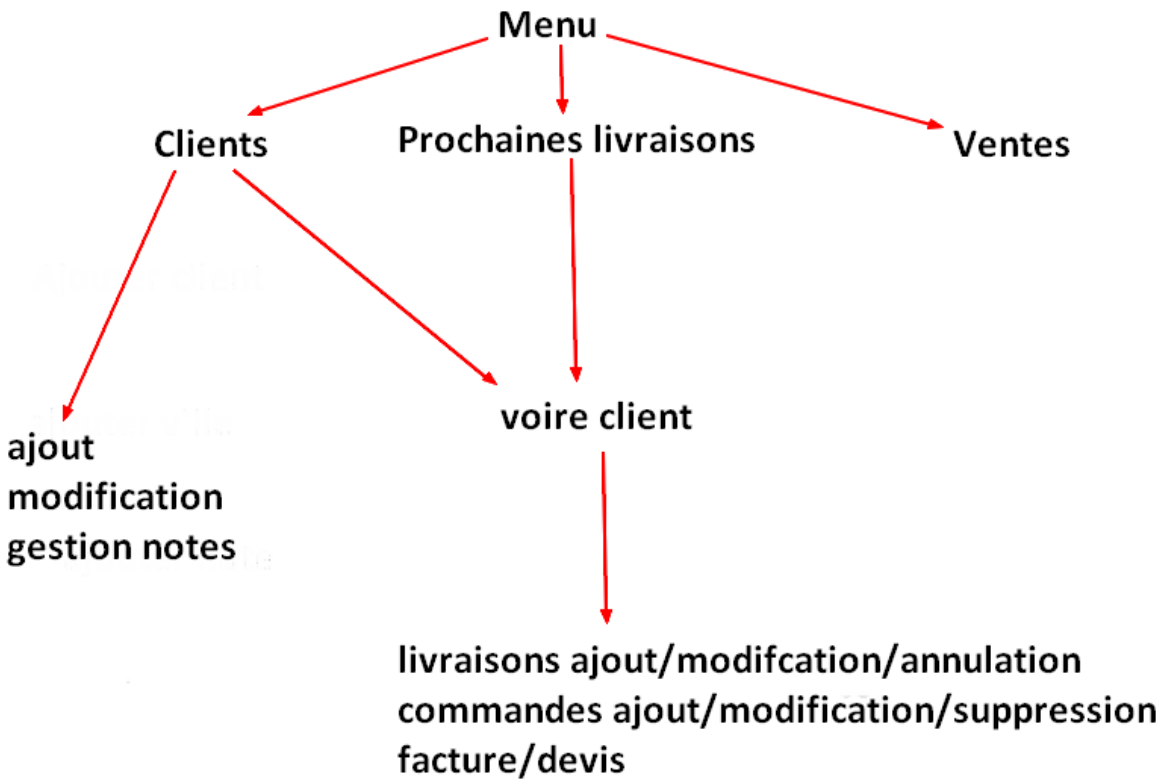
Partie interface utilisateur

Pour voire les pages du logiciel référer vous au manuel utilisateur.

Les pages du logiciel sont très peut mélanger avec du code qui ne concernent pas le visuel, le code sépare la plupart du temps les éléments de la page dans plusieurs fonctions car je me suis rendu compte que c'était plus facile à lire et à gérer après avoir travailler sur les premières pages.

J'ai utilisé swing pour faire le visuel, la partie la plus délicate selon moi était la disposition des éléments sur l'écran, pour éviter d'avoir de choisir la disposition des éléments a la main on utilise des gestionnaires de positionnements.

Hiérarchie des pages :



Aucune maquette n'on été faites, le choix des interfaces a été fait par moi.

Page clients

Fonctionnalités :

- Affiche la liste des clients
- Recherche d'un client avec une barre de recherche
- Ajout d'un client et ajout d'une nouvelle ville si nécessaire
- Modifier un client
- Ajouter une note sur un client
- Accéder à la page d'un client (voire un client)

Page ventes

Fonctionnalités :

- Visualisé les vente pour une année

Pages prochaines livraisons

Fonctionnalités

- Affiche la liste des prochaines livraisons
- Accéder à un client qui est concerner par une prochaine livraison

Page voire un client

Fonctionnalités :

- Ajout/modification/annulation livraison
    - Utilisation d'une extension JDatePicker pour afficher un calendrier pour le choix de la date.
- Ajout/modification/suppression commande
    - La modification ne pré remplie pas les cases à cocher des bois
- Modifier client ou sa note
- Génération fichier PDF du devis ou de la facture d'une livraison

Déploiement

L'exécutable est créé avec le jar générer par Maven directement sur l'ordinateur de l'entreprise a l'aide du logiciel launch4j et du jre 15 nécessaire pour le bon fonctionnement du logiciel.

De plus nous installons WampServer et nous importons la base de données avec les informations que j'ai déjà remplies, comme les premiers clients et les types de bois qui sont déjà connues.

L'exécutable de WampServer et du logiciel sont dans un dossier qui contient aussi le manuel utilisateur.

Conclusion

Annexes

Sont considérer comme annexes :

- Le code commenté
- Le manuel utilisateur qui rajoute des informations et les images des interfaces utilisateurs.