

Rapport de stage

Dornier Simon

Développeur Divalto Weavy



**Étudiant en Licence
professionnelle :**

Métiers de l'Informatique:
Applications web, Systèmes
intra/internet pour
l'entreprise

Tuteur de stage :

Slimane Mahamdi

Consultant gestion

Tutrice universitaire :

Sandrine Lanquetin

Enseignante chercheuse

Établissement :

UFR - Sciences et techniques

Université de Bourgogne

Entreprise :

OCI 21

Saint-Apollinaire

Date du stage :

Du 05/03 au 03/09/2021

Rapport effectué le :

15/07/2021



OCI INFORMATIQUE & DIGITAL



Remerciements

Je tiens tout d'abord à remercier mon tuteur de stage, Slimane Mahamdi, consultant gestion, pour m'avoir accueilli dans l'entreprise et m'avoir apporté son aide et son soutien tout au long de mon stage.

Je remercie aussi Gilles Millière, chef de projet et responsable développement, pour m'avoir formé et être resté disponible pour répondre à toutes mes questions.

Pierre Zanetti, directeur du pôle gestion, qui m'a fait confiance et m'a permis de rejoindre l'équipe d'OCI où j'ai pu travailler dans un cadre sérieux et convivial.

Sandrine Lanquetin et toute l'équipe pédagogique de la licence professionnelle MIAW, pour m'avoir accompagné durant cette année en distanciel complexe en raison du contexte sanitaire actuel.

Table des matières

Remerciements	1
Table des matières	2
Table des figures	3
1. Présentation générale	4
1.1. Introduction	4
1.2. OCI.....	4
1.3. L'agence de Dijon	5
1.4. Weavy	5
1.5. Studio.....	6
1.6. Problématique au sein du service	7
1.7. Mes missions	8
1.8. Les difficultés rencontrées.....	8
1.9. Annonce du plan	9
2. Développement.....	9
2.1. Standard OCI	9
.....	9
2.2. Standard Web OCI.....	16
.....	16
2.4. Critique du Studio et de l'application	19
3. Conclusion	20
3.1. Mes résultats.....	20
3.2. Les apports personnels.....	20
.....	20
3.3. Mon avenir	21
4. Bibliographie.....	21
.....	21
.....	22
5. Annexe	22

Table des figures

Figure 1 : Répartition des agences OCI.....	4
Figure 2 : Hiérarchie d'OCI 21	5
Figure 3 : Tableau de bord de Divalto Weavy.....	5
Figure 4 : Schéma des surcharges/couches Divalto Weavy.....	8
Figure 5 : Écran de multi-cadence original	9
Figure 6 : Capture d'écran des trois lignes ajoutées.....	11
Figure 7 : Écran de création d'une fonctionnalité.....	14
Figure 8 : Écran de ligne de commande dans le Studio	22
Figure 9 : Écran de ligne de commande dans l'application.....	22
Figure 10 : Onglet de validation de commande	23
Figure 11 : Onglet de validation de commande dans le Studio.....	23
Figure 12 : Écran de Saisie multiple réduit.....	24
Figure 13 : Écran de Saisie multiple dans le Studio réduit	24
Figure 14 : Écran des objectifs globaux.....	25
Figure 15 : Écran des objectifs d'un commercial par mois.....	25
Figure 16 : Écran des objectifs d'un commercial par mois par client.....	26

1. Présentation générale

1.1. Introduction

J'ai connu OCI car mon père a travaillé pour un de leur employé et, après avoir postuler pour me faire engager en alternance l'année dernière, j'ai finalement été pris en stage chez eux. J'ai choisi OCI car c'est une grande entreprise offrant de nombreuses opportunité et je savais que j'allais intégrer une équipe qui distribue des logiciels de gestion dont le but est de faciliter la vie des gens.

1.2. OCI

Le groupe OCI¹ a été fondé en 1979 à Mundolsheim, une ville proche de Strasbourg qui est aujourd'hui l'endroit où est basé le siège social de la société. À l'époque, c'était l'une des seules entreprises à proposer des services numériques comme la création de site internet. Le groupe a beaucoup évolué depuis et en 2019, il rachète Scriba, une entreprise de prestation informatique, ce qui permet au groupe de continuer de s'élargir et de s'exporter dans l'ouest de la France.

Aujourd'hui, le président directeur général d'OCI est Frédéric VACHER et l'entreprise compte environ 750 employés répartis dans les 17 agences OCI à travers la France. OCI est une entreprise informatique ESN² qui accompagne les entreprises, associations et collectivités locales sur l'ensemble de leur système d'information et sur leur transformation digitale. Elle offre un grand panel de services :

- Réseaux et câblage
- Infrastructure³
- Cloud⁴
- Mobilité
- Solutions de gestion
- Web et digital
- Téléphonie



Figure 1 : Répartition des agences OCI

¹ OCI : Organisation de la Communication par l'Informatique

² ESN : Entreprise de services du numérique

³ Infrastructure: Une infrastructure informatique comprend les composants nécessaires au fonctionnement et à la gestion des environnements informatiques d'entreprise.

⁴ Cloud : Il fait référence aux serveurs accessibles via l'internet ainsi que les logiciels et bases de données qui s'exécutent sur ces serveurs.

1.3. L'agence de Dijon

L'agence OCI 21 basée à Saint-Apollinaire est composée d'une trentaine de personnes : des commerciaux, des techniciens, des consultants et des développeurs.

La répartition dans l'agence est organisée de manière logique : les consultants et les développeurs sont regroupés dans un open-space en fonction du logiciel sur lequel ils travaillent. Les techniciens, eux, sont dans un second open-space à côté d'un atelier. Puis les commerciaux sont dans des bureaux à l'entrée de l'agence. L'open-space permet de pouvoir se déplacer à sa guise pour travailler avec les personnes qui sont sur le même projet que nous.

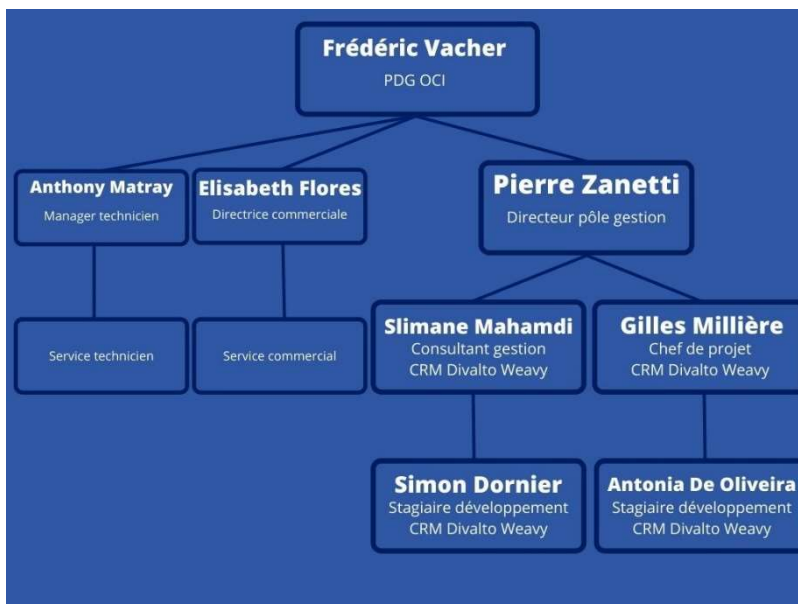


Figure 2 : Hiérarchie d'OCI 21

J'ai effectué mon stage au sein d'une équipe de développement travaillant sur le CRM⁵ Divalto Weavy. Notre groupe est composé de Gilles Millière, chef de projet et responsable développement, Slimane Mahamdi, consultant en gestion et mon tuteur de stage, Antonia De Oliveira et moi-même, stagiaires en développement.

1.4. Weavy

Lors de mon stage, j'ai travaillé sur le logiciel Weavy qui appartient à Divalto et dont OCI est un des distributeurs. Weavy, anciennement Swing Mobility, est un CRM, c'est-à-dire que c'est un logiciel de gestion de la relation client, qui offre aux clients des fonctionnalités comme : la liste de leurs clients, le catalogue de produit, les statistiques de vente, les stocks, rédiger des devis ou des commandes, consultation de l'agenda et des tâches du jour, etc.

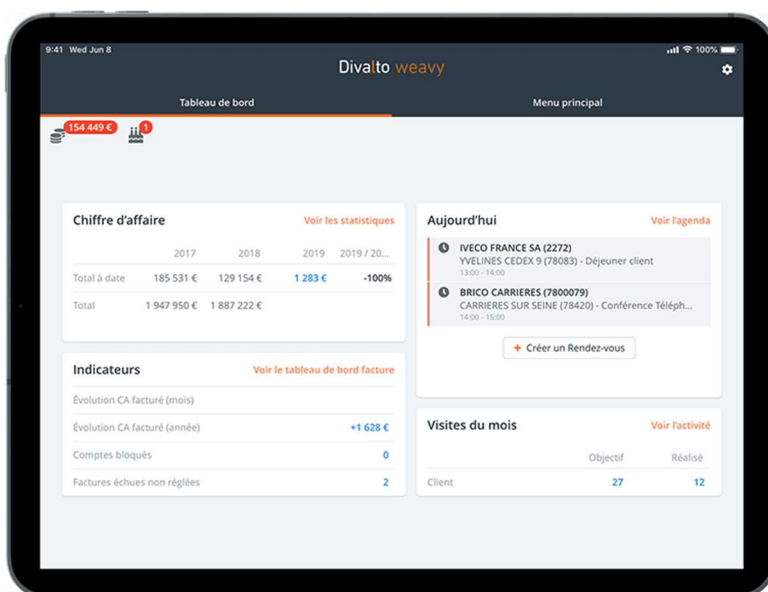


Figure 3 : Tableau de bord de Divalto Weavy

⁵ CRM : Customer Relationship Management, il s'agit d'une application de gestion de la relation client.

Weavy est une solution qui fonctionne en mode SaaS, “Software as a Service”, en français “logiciel en tant que service”. Une solution dite “SaaS” signifie qu’il s’agit d’une solution logicielle applicative hébergée dans le cloud et exploitée en dehors de l’entreprise qui développe ce logiciel. Cette solution n’est donc pas hébergée chez OCI, ce qui leur permet d’éviter l’installation du logiciel sur leurs serveurs et reste facilement accessible grâce à une connexion internet.

Ce logiciel existe sous deux versions, une application mobile et un site web. L’application est principalement utilisée par les commerciaux qui se déplacent chez les clients, car elle n’a pas besoin de connexion internet pour fonctionner, les utilisateurs font une mise à jour manuelle tous les jours pour envoyer et récupérer les informations sur le serveur, ce qui leur permet d’avoir accès à l’application et aux données même s’ils n’ont pas accès à internet durant leurs déplacements.

La version web est accessible en ligne et est mise à jour en direct, elle est principalement utilisée par les équipes de gestion qui ont un accès stable à internet. Cette partie a été longtemps mise de côté par l’équipe d’OCI car personne n’était formé au développement dessus jusqu’à maintenant, mais j’ai pu travailler dessus quelques temps car un nouveau développeur connaissant cette partie a été engagé durant l’été.

1.5. Studio

Comme dit précédemment, le logiciel Weavy appartient à Divalto, nous n’avons pas accès à son code source, nous utilisons un logiciel nommé Studio qui nous permet d’apporter des modifications à l’application Weavy. Dans ce Studio, nous avons plusieurs parties qui nous permettent chacune de gérer une partie du fonctionnement de l’application :

- **Le dictionnaire de données** : Gestion des champs dans les tables de données.
- **Le designer** : Gestion de ce qu’on veut afficher sur les différents écrans.
- **L’unisearch** : Gestion des différentes listes, comme la liste des clients ou des produits.
- **L’arborescence** : Gestion des boutons de l’application.
- **Synchronisation** : Gestion des échanges entre l’application et le Cloud.
- **Événement** : Gestion des envois de mails/notifications, import des traductions, export de fichier en CSV⁶, etc. Ces événements se répètent régulièrement, pour la plupart, une fois par jour.
- **Backend** : Gestion de l’import/export des tables de données entre Weavy et d’autres logiciels de gestion. Ces imports/exports se répètent toutes les nuits.
- **Report** : Gestion du contenu des fichiers à imprimer.
- **Structure commerciale** : Gestion des niveaux d’accès des utilisateurs.

⁶ CSV: Comma-Separated Values, format texte représentant des données tabulaires sous forme de valeurs séparées par des virgules.

En plus de ces outils, le Studio propose un outil d'éditeur de script (en langage propriétaire nommé Diva), un explorateur de base de données en MySQL, un explorateur de fichier serveur, un gestionnaire des clés de traduction, etc.

Quant aux systèmes de gestion de base de données (SGBD), les trois utilisés sont des SGBD relationnels, ce qui veut dire qu'ils permettent de synthétiser n'importe quel lot d'informations en exploitant les données à l'aide de jointure.

- MySQL est utilisé pour de simple recherches dans la base de données via l'explorateur de base de données de l'IDE⁷ ainsi que les éditions automatiques.
- SQLite est principalement présent dans les scripts ou l'affichage d'Unisearch. On peut aussi être amené à l'utiliser afin de compléter certaines requêtes dans les Reports.
- SQL Server sert à l'intégration automatique des données dans les différentes tables de la base de données que ce soit du serveur à l'application ou inversement.

1.6. Problématique au sein du service

Lors de mon stage, j'ai intégré l'équipe qui développe des solutions spécifiques⁸ sur le CRM Divalto Weavy. Initialement, ce CRM offre aux clients des fonctionnalités appartenant au standard de Weavy.

L'équipe avait profité du premier confinement pour commencer à développer un standard OCI qui permettrait d'accélérer le déploiement chez les nouveaux clients. En effet, les clients demandant régulièrement les mêmes fonctionnalités ne faisant pas partie de l'application de base, l'équipe devait les recopier depuis la version d'un autre client, ce qui prenait beaucoup de temps.

Actuellement, ce standard est utilisé comme base pour chaque nouveau client et comme outil de présentation, mais de nombreux bugs restent présent et certaines fonctionnalités sont manquantes, ce qui fait perdre beaucoup de temps à l'équipe qui doit corriger les problèmes sur chaque nouvelle base client. Le développement de ce standard avait été mis en pause par manque de temps, c'est pourquoi j'ai intégré cette équipe.

⁷ IDE : Integrated Development Environment : c'est un ensemble d'outils qui permettent de développer un logiciel.

⁸ Solution spécifique : changements réalisés sur un projet en fonction des besoins et demandes d'un client

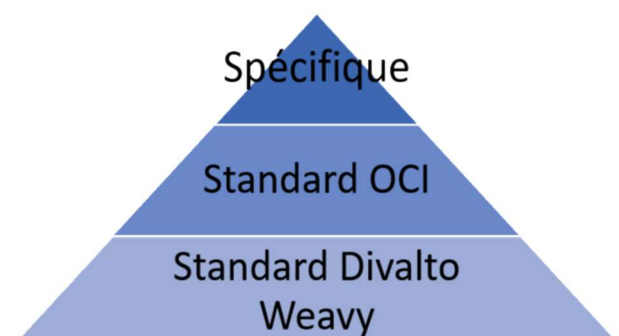


Figure 4 : Schéma des surcharges/couches Divalto Weavy

1.7. Mes missions

Ma mission de base a été de continuer le développement du standard OCI. J'ai commencé par être formé à l'utilisation du Studio et, ensuite, j'ai passé deux/trois semaines pour générer et implémenter les clés de traductions manquantes dans toutes les solutions spécifiques créés par Gilles.

Une fois cela fini, j'ai pu commencer le développement, j'ai créé un écran permettant de faire des commandes multiples, recréé et réadapté un écran de statistique qui existait déjà dans une ancienne version, et j'ai créé un écran pour la partie Web.

Enfin, après avoir fini de travailler sur le standard OCI, j'ai commencé à faire diverses modifications et ajouts pour des clients actuels.

1.8. Les difficultés rencontrées

Lors de mon stage, j'ai rencontré deux principales difficultés. La première étant la nouveauté du logiciel et du langage que je n'avais jamais vu, ni utilisé auparavant.

Grâce à la formation reçue en tout début de stage, j'ai pu acquérir les principales bases, mais cela ne suffisait pas toujours et le manque de documentation à parfois posé problème et ralenti l'avancement de mes tâches. Le responsable développement, Gilles Millière, s'est montré particulièrement disponible pour m'aider pour débloquer certaines étapes de mes tâches ce qui m'a permis de mener mes missions à bien.

La seconde difficulté a été mon manque de connaissance dans le domaine commercial. Il m'est arrivé plusieurs fois de ne pas comprendre des concepts ou des termes techniques, ce qui me bloquait dans mon travail puisque je ne peux pas développer si je ne comprends pas ce que je dois faire. Mon tuteur de stage, Slimane Mahamdi, m'a beaucoup aidé à ce niveau-là pour que je ne reste pas bloqué.

1.9. Annonce du plan

Dans la suite de mon rapport, je vais détailler mes missions au travers des quatre étapes du développement : l'analyse, la conception, la réalisation et les tests. Je vais commencer par parler du développement sur le Standard OCI, ensuite, j'aborderai mon développement sur la version Web et je finirai en faisant une critique du Studio de développement et de l'application.

2. Développement

Ma première mission a été d'améliorer une fonctionnalité nommée la multi-cadence. Cette fonctionnalité, qui se trouve dans l'onglet de passage des commandes, permet de commander le même produit en quantités potentiellement différentes et d'attribuer chaque commande à un client différent ou au client présélectionné. Gilles Millière avait déjà créé la possibilité de le faire pour cinq autres clients en plus du client présélectionné et ma mission a été d'augmenter le nombre de clients pour la multi-cadence à huit et d'ajouter un écran permettant de choisir à quels clients les commandes sont destinées. Cet ajout a été demandé par des commerciaux qui utilisent Weavy et qui voudrait pouvoir faire plus de multi-commande.

Annuler		Pied		Valider	
Panier	Livraison multiple	Livraison	Signature		
Commande Principale		0		0,00	
Vous devez saisir adresse(s) client(s)					
Client L2	-	Aucune date sélectionnée	0	0,00	Idem Client
Client L3	-	Aucune date sélectionnée	0	0,00	Idem Client
Client L4	-	Aucune date sélectionnée	0	0,00	Idem Client
Client L5	-	Aucune date sélectionnée	0	0,00	Idem Client
Client L6	-	Aucune date sélectionnée	0	0,00	Idem Client
TOTAL		0		0,00	

Figure 5 : Écran de multi-cadence original

2.1.1. Analyse

Le besoin de ces commerciaux est de pouvoir ajouter plusieurs quantités du même produit et sélectionner le client qu'ils veulent pour chaque quantité. Pour cela, j'ai observé ce qui était déjà présent sur l'application. Dans l'onglet de sélection d'un produit, il y avait déjà six champs servant à ajouter des quantités et l'onglet de validation de commande comportait des champs pour sélectionner les clients, mais le premier ne récupérait pas les informations entrées et le deuxième manquait d'option.

2.1.2. Conception

Ensuite, j'ai réfléchi à la manière d'ajouter les nouveaux éléments. Pour l'ajout des quantités, il suffit de copier-coller les champs de quantités déjà présents tout en veillant à ce que les scripts liés à ces champs récupèrent bien les données dans la base et sans diminuer l'ergonomie de l'écran.

Cela se complique un peu pour l'onglet de sélection des clients, en effet cet onglet existe déjà dans la validation de la commande, mais il n'est pas complet et il faut en plus que j'en crée un autre qui sera plus facilement accessible et avec des options différentes.

Commençons par parler des ajouts sur l'écran déjà existant : un bouton pour annuler la sélection d'un client, les adresses des clients (car un client peut avoir plusieurs adresses et cela permet de les différencier) et les trois lignes liées aux trois nouvelles quantités. De plus, je vais rendre non cliquables les lignes où il n'y a pas de quantités sélectionnées pour éviter que le commercial ne valide une commande sans client.

Pour finir, je vais copier-coller cet écran pour en créer un nouveau beaucoup plus permissif, qui s'appellera "Saisie multiple", pour que le commercial puisse choisir les clients avant de choisir une quantité. Cet écran sera accessible depuis le catalogue, la liste et le carrousel des produits et depuis l'onglet de commande d'un produit.

2.1.3. Réalisation

La première étape de cette phase a été d'aller dans le Dictionnaire de données du Studio pour créer les nouveaux champs dans la base de données. Après quelques recherches, j'ai trouvé les deux tables qui permettent d'enregistrer une commande, *orderheader*, qui est l'entête de commande où on enregistre les détails de la commande liés au client, comme son nom, son adresse, la date de livraison, etc. Et la deuxième table, *orderdetail*, où on enregistre les détails liés au produit comme le nom, le prix, la quantité, etc, ainsi qu'une clé étrangère qui fait référence à la table *orderheader*.

Dans la table *orderdetail* il a seulement fallu ajouter des champs pour la quantité, puisque le produit ne change pas d'un client à l'autre, seule la quantité de ce dernier change. Par contre, dans la table *orderheader* il a fallu ajouter des lignes pour le client, son adresse et la date de livraison.

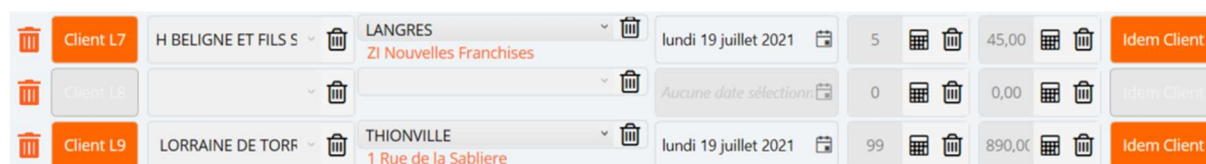
L'étape suivante consistait à ajouter trois champs de saisie à l'écran de la ligne de commande (L7-L8-L9) et à les lier à un script pour qu'ils enregistrent les valeurs inscrites. Pour cela, je suis allé dans le Designer du Studio et j'ai copié-collé les boutons déjà existant, j'ai modifié l'affichage qui était en ligne pour le mettre en colonne pour que ça soit plus pratique (cf : [ValidationDeCommande](#)), puis, j'ai modifié leurs scripts:

```
VARUPDATE_DB_BYID( "sw_data_orderdetail",
"<var>orderdetail_ID|OrderLine</var>", "oci_quantityL9=<ctrl>upQuantity9</ctrl>" )
```

Cette fonction du langage propriétaire Diva sert à mettre à jour une ligne d'une table dans la base de données. Elle comporte trois arguments : la table concernée, l'identifiant qui permet de retrouver la bonne ligne et la valeur à mettre dans la colonne définit (oci_quantityL9). Les balises <ctrl> permettent de récupérer la valeur inscrite dans le champ nommé "upQuantity9" et qui correspond donc au champ que l'on est en train de modifier. Une fois que le champ est créé et relié à la base de données, il faut chercher dans les scripts si les anciens champs étaient appelés et ajouté à ces appels les nouveaux champs, par exemple dans le script d'ouverture de la ligne de commande, il y a des lignes qui permettent d'initialiser les champs des quantités si elles sont déjà renseignées.

J'ai fait un autre petit ajout sur cet écran, car le M de multi cadence n'était qu'un simple logo à la base et j'ai proposé d'en faire un bouton, ce qui permettrait d'ouvrir l'écran de saisie multiple depuis la ligne de commande, ce qui a été accepté. J'ai donc supprimer l'élément dans le Designer qui permettait d'afficher le logo pour le remplacer par un bouton, avec la même image, qui appelle la fonctionnalité qui affiche l'écran de saisie multiple.

Passons maintenant à la modification et à l'ajout des lignes de sélection des clients dans la validation de commande. Au début, il n'y avait que le bouton de sélection du client, son nom, la date, la quantité, le prix total et le bouton Idem Client (ce dernier n'était pas fonctionnel). J'ai ajouté un champ de sélection de la ville du client (qui affiche désormais l'adresse exacte), un bouton supprimer qui vide les champs de la ligne (sauf la quantité et le prix)* !t, et j'ai fait fonctionner le bouton Idem Client. Une fois les lignes complétées, j'ai ajouté les trois nouvelles lignes pour les trois nouvelles quantités.



Client L7	H BELIGNE ET FILS S	LANGRES ZI Nouvelles Franchises	lundi 19 juillet 2021	5	45,00	Idem Client
Client L8			Aucune date sélectionnée	0	0,00	Idem Client
Client L9	LORRAINE DE TORF	THIONVILLE 1 Rue de la Sabliere	lundi 19 juillet 2021	99	890,00	Idem Client

Figure 6 : Capture d'écran des trois lignes ajoutées

J'ai commencé par le bouton Idem Client. Le principe de ce bouton est de récupérer le client sélectionné pour la commande originale et de remplir le champ du nom du client, pour que les commerciaux puissent facilement créer plusieurs commandes chez ce client, mais avec des quantités et des dates différentes. Pour

éviter de créer huit scripts différents, tous les boutons déclencheront le même script, il faut donc commencer par savoir quel bouton a été déclenché, grâce à la fonction : *controle = CONTROL_GETNAME()* qui permet de récupérer le nom du bouton dans une variable et une fois que j'ai récupéré ce nom, je peux m'en servir pour tester sur quelle ligne on est : *IF(EQUALS("", "btCustomerIdemPrincipall9")) THEN*, en l'occurrence, je teste si la variable contrôle créé plus tôt est égale au nom du neuvième bouton et, si oui, on déclenche plusieurs lignes du script :

- *CONTROL_ENABLE("cbCustomeraddress_ID_LivL9", "1")*
CONTROL_ENABLE("dpDateL9", "1")

J'ai commencé par rendre les boutons de sélection de la date et de l'adresse cliquable, car ces champs ne doivent pouvoir être remplis que quand le client est sélectionné.

- *CONTROL_SET_VALUE ("cbCustomer_ID_L9", "<var>customer_ID|Order</var>")*

Ensuite, j'affiche le nom du client en récupérant son identifiant dans le groupe de variable « Order » qui a été créé à l'initialisation de la commande au moment où le client a été sélectionné.

- *CONTROL_FILLCOMBO_SQL("cbCustomeraddress_ID_LivL9", "SELECT customeraddress_ID, city FROM sw_data_customeraddress WHERE customer_ID = " ORDER BY name ASC", "", "")*

Pour les adresses du client, j'ai utilisé une fonction qui permet de remplir une liste déroulante avec une requête SQL. J'ai créé la requête en utilisant l'explorateur de données du Studio. Cette requête récupère l'identifiant de l'adresse et la ville du client, depuis la table des adresses, tout en testant si l'adresse est bien liée au client sélectionné.

- *datlivL9 = CONTROL_GET_VALUE("dpDateL9")*
IF(EQUALS(datlivL9, "") == 1) THEN
 CONTROL_SET_VALUE("dpDateL9", "%today+7%")
ENDIF

Et pour finir, je teste si le champ de date est vide et, s'il l'est, je l'initialise au même jour qu'aujourd'hui, mais dans une semaine.

Le bouton de suppression fonctionne de la même manière :

- Je teste quelle ligne je dois supprimer
 - *controle = CONTROL_GETNAME()*
IF (EQUALS(controle, "ButtonL9") == 1) THEN
- Je mets tous les champs de cette ligne à vide
 - *CONTROL_SET_VALUE("cbCustomer_ID_L9", "")*
CONTROL_SET_VALUE("cbCustomeraddress_ID_LivL9", "")

```

CONTROL_SET_VALUE( "lbCustomerAdress_9", "" )
CONTROL_SET_VALUE( "dpDateL9", "" )
CONTROL_FILLCOMBO_SQL( "cbCustomeraddress_ID_LivL9", "", "",
"" )

```

- Je rends les champs date et adresse non-cliquable.
 - `CONTROL_ENABLE("cbCustomeraddress_ID_LivL9", "0")`
 - `CONTROL_ENABLE("dpDateL9", "0")`

Comme nous l'avons vu juste avant, la liste déroulante des villes se remplit grâce aux scripts de sélection du client. Mais ce n'est pas suffisant pour être sûr d'avoir sélectionné la bonne adresse, car certains clients ont plusieurs adresses dans la même ville. Donc, pour cela, j'ai décidé d'afficher le détail de l'adresse en dessous de la ville. Donc quand la ville est sélectionnée, cela déclenche un script qui va afficher l'adresse correspondante dans le champ situé en dessous. Pour cela, le script va récupérer ce qui est sélectionné dans la liste, puis faire une requête SQL pour récupérer l'adresse correspondante et enfin afficher cette adresse dans le champ correspondant.

```

customer_address_L9 = CONTROL_GET_VALUE( "cbCustomeraddress_ID_LivL9" )
adresseclient9 = VARGET_DBSQL( "SELECT address1 from
sw_data_customeraddress WHERE customeraddress_ID = '<varscript>customer_ad
dress_L9</varscript>'" )
CONTROL_SET_VALUE( "lbCustomerAdress_9",
"<varscript>adresseclient9</varscript>" )

```

Et pour finir avec cet écran, j'ai ajouté les trois nouvelles lignes de sélection de client. Pour cela j'ai copié-collé les lignes déjà existantes dans le Designer (cf : [ValidationDeCommande](#)) puis je suis allé chercher tous les scripts qui pourraient contenir des appels aux anciens champs pour y ajouter les nouveaux, ce qui m'a demandé de chercher tous les noms des champs dans les scripts pour être sûr de n'en rater aucun.

La dernière étape consistait à recréer entièrement l'écran précédent dans un autre écran pour que les commerciaux puissent y accéder différemment et aussi être plus libres dans leurs modifications. Là où l'écran de validation de commande n'est accessible que depuis le bouton qui valide la commande et est très peu permissif pour éviter les commandes incomplètes, le nouvel écran de saisie multiple est, lui, accessible depuis le catalogue, la liste, le carrousel et la fiche des articles, et tous les champs sont modifiable même s'il n'y a pas de quantité sélectionnée.

La première chose à faire quand on veut créer un nouvel écran dans le Studio, c'est de créer une fonctionnalité avec un nom, une table de données et un ou plusieurs

écrans liés. Une fois créé, je pourrai l'appeler dans les scripts pour ouvrir l'écran correspondant. De plus, si les écrans renseignés en bas n'existent pas, le Studio en crée de nouveaux.

Ecrans	Format
OrderMultipleSaisieCustomer	Common
OrderMultipleSaisieCustomer	Smartphone

Figure 7 : Ecran de création d'une fonctionnalité

Maintenant, que j'ai de quoi appeler l'écran, je peux aller dans les écrans de la fiche article ou du catalogue pour y ajouter un bouton et lui lier le script suivant :

```
CALL_SCRIPT( "Func_OpenFunctionality", "OrderMultipleSaisieCustomer", "edit",
"orderheader_ID|Order", "", "", "Oci_multipleEntry", "1120", "580" )
```

Ce script permet de d'appeler une fonctionnalité avec comme arguments : le nom de la fonctionnalité, si on peut modifier ses informations ou non, à quelle commande on fait référence, le titre de l'écran (avec une clé de langue qui permet de l'avoir en français et en anglais) et ses dimensions.

Une fois que cela est fait, il nous faut des champs dans cet écran. Pour cela, j'ai regardé ce qui avait été fait pour l'écran de validation de commande et j'ai recréé le même écran, champ par champ, car, dans le Studio, il n'y a pas la possibilité de copier-coller des champs d'un écran à l'autre. Ensuite, j'ai relié les nouveaux boutons aux scripts déjà créés pour les anciens boutons et j'ai laissé tous les champs modifiables.

Avant les tests, il a fallu que je rajoute une dernière petite chose. Puisque toutes les entreprises à qui nous allons proposer cette fonctionnalité n'auront pas forcément besoin des huit multi-commandes supplémentaires et que ça serait plutôt long de venir supprimer ces lignes à chaque nouvelle entreprise, j'ai décidé de rajouter une variable dans la table de données des options, cette variable allant de un à neuf. Quand la variable est à un, aucune des lignes de multi-commandes n'apparaît et les lignes

réapparaissent une par une quand on l'augmente. Pour cela, j'ai dû ajouter à tous les scripts d'ouverture un test pour chaque ligne et cacher chaque champ de cette ligne quand la variable est inférieure au numéro de la ligne.

Script pour cacher la ligne 9 :

```
IF ( "NbLivraisonsChoisi" < "9" ) THEN  
    CONTROL_VISIBLE( "btCustomerL9", "0" )  
    CONTROL_VISIBLE( "cbCustomer_ID_L9", "0" )  
    CONTROL_VISIBLE( "cbCustomeraddress_ID_LivL9", "0" )  
    CONTROL_VISIBLE( "lbCustomerAdress_9", "0" )  
    CONTROL_VISIBLE( "dpDateL9", "0" )  
    CONTROL_VISIBLE( "nuQTL9", "0" )  
    CONTROL_VISIBLE( "nuMtl9", "0" )  
    CONTROL_VISIBLE( "btCustomerIdemPrincipall9", "0" )  
ENDIF
```

2.1.4. Test

La dernière phase du développement est toujours la phase de test. Dans une application comme celle-ci, où il y a énormément de fonctionnalités diverses qui agissent sur les mêmes données, il est important de bien vérifier que les données entrées par les utilisateurs dans les champs sont bien sauvegardées et il faut aussi surveiller que tous les boutons fonctionnent dans tous les cas. C'est-à-dire que pour être sûr qu'il n'y a pas de bugs, j'ai testé tout ce que j'ai rajouté ou modifié un par un, et ce, dans tous les sens possibles.

J'ai, par exemple, testé en ajoutant une quantité, puis en liant un client à cette quantité depuis l'écran de saisie multiple. Ensuite, je suis allé voir dans l'écran de validation et j'ai remarqué que le client ne s'affichait pas car un script à l'ouverture de cet écran vidait mes informations.

J'ai testé si les adresses des clients étaient bien les bonnes en comparant mes résultats à ce que je trouvais dans la base de données. J'ai eu quelques difficultés avec cette partie car la base de données qu'il y a sur le serveur de Divalto et les données contenues dans l'application ne sont pas exactement les mêmes car quand on se connecte à l'application, on le fait en tant que commercial et les commerciaux n'ont pas accès à toutes les informations. Il faut donc pour installer un explorateur de base de données (comme DB Browser) pour comparer les informations qu'il y a d'afficher dans l'application : celle qu'il y a sur la base de l'application et celle qu'il y a sur la base de Divalto afin de s'assurer que tous les clients ont bien les bonnes adresses et aussi que les commandes sont bien complètes quand on les valide.

Globalement, il est assez difficile d'être sûr et certain que tout fonctionne parfaitement, car il faudrait beaucoup trop de temps pour tester toutes les données une par une. Mais on peut supposer que tout fonctionne si on teste la plupart des possibilités différentes.

2.1.5. Conclusion

En conclusion, les phases d'analyse, de conception et de réalisation de cette application passent par beaucoup de recherches dans ce qui est déjà présent pour modifier ou recopier et adapter les différents écrans, scripts, liste, etc. Et la phase de test demande beaucoup de rigueur pour s'assurer que les données enregistrées soient bien les bonnes.

2.2. Standard Web OCI

Après avoir fini ma mission sur le standard OCI, on m'a confié la mission de développer un écran de statistique sur la version Web.

Le développement sur la version Web de Weavy se fait grâce à un Studio où on trouve des pages de code avec des scripts, des requêtes, des composants, des templates, des schémas, etc. Pour faire simple, on récupère les données grâce aux requêtes, puis on donne un nom à ces données dans les schémas pour pouvoir ensuite les appeler dans les composants et le template détermine ensuite l'affichage.

2.2.1. Analyse

Ma mission était de créer un écran pour les objectifs de chiffre d'affaires des commerciaux par mois et par client. Il me fallait donc un affichage des objectifs des commerciaux, puis les objectifs des commerciaux par mois, et enfin les objectifs des commerciaux par mois et par client.

2.2.2. Conception

J'ai commencé par regarder un écran qui existait déjà et qui se nomme Objectif CA (Chiffre d'Affaires) et je me suis demandé comment je pouvais recopier et adapter cet écran pour qu'il affiche les informations voulues. Cet écran m'a offert une bonne base car il avait déjà un premier écran avec les objectifs totaux de l'entreprise et une liste de commerciaux avec un résumé de leurs objectifs. De plus, il y a la possibilité de cliquer sur un commercial pour accéder à un autre écran ressemblant au premier, mais

avec le détail des objectifs par mois (cf : [ÉcranDesObjectifsDUnCommercialParMois](#)).

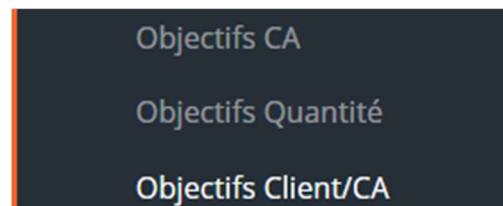
Une fois parti de cet écran, il me suffirait d'ajouter un nouvel écran, qui s'afficherait quand on clique sur un mois et qui donnerait donc les détails des objectifs pour un commercial par mois par client (cf : [ÉcranDesObjectifsDunCommercialParClient](#)). Pour savoir quelles données mettre dans le nouvel écran, j'ai dû faire de nombreux schémas de base de données et des tests dans l'explorateur de données.

2.2.3. Réalisation

La première étape a été de créer un bouton permettant d'accéder à ce futur nouvel écran. Pour cela, j'ai cherché où et comment était créé les autres boutons et j'ai trouvé une page de code se nommant 'menu' qui détermine les boutons que l'on trouve sur la gauche de l'écran.

Après avoir trouvé comment faire un bouton, j'ai pu créer le nouveau bouton auquel j'ai donné le titre 'Objectif Client/CA' grâce à une clé de langue dans le label. Ensuite, je lui ai donné comme cible le nom de la page du futur écran.

```
{
  "Label": "Oci_customersObjectivesCA",
  "Link": {
    "Target": "usergoal.customerCA"
  }
}
```



Une fois cela fait, j'ai recréé l'écran Objectifs Client et celui avec les détails par mois. Cela m'a demandé de recréer et de copier-coller presque trente pages de code, ce qui a été assez fastidieux étant donné que j'ai dû changer tous les noms de pages. J'ai également dû retrouver tous les endroits du code où ces pages étaient appelées et les remplacer. J'ai aussi dû rendre les mois de la liste cliquable pour qu'on puisse accéder aux détails par client. Pour cela, je leur ai donné comme cible la page des détails par client tout en passant l'identifiant du client (baseuser_ID) et le mois sélectionné (monthIndex=#monthIndex#) dans l'URL.

```
"monthName": {
  "title": "Gen_Month",
  "type": "string",
  "Link": {
```

```
"Target": "usergoal.customerCAMonth/#=baseuser_ID#?monthIndex=#=monthIndex#"
```

```
},
```

```
},
```

Passons maintenant à la création de l'écran des détails par client, ce dernier est inspiré de l'écran avec la liste des commerciaux, sauf qu'on la remplace par la liste des clients liés au commercial sélectionné et on affiche les informations pour chaque client en fonction du commercial et du mois sélectionné. On ajoute donc ceci à la requête SQL:

```
WHERE bu.baseuser_ID = {{id}}
```

```
AND months.month = {{monthIndex}}
```

2.2.4. Test

J'ai testé si les données présentes correspondaient avec celles dans la base de données et j'ai surtout testé si les modifications fonctionnaient. Pour cela, j'ai modifié un objectif dans le premier écran, donc pour un commercial, cet objectif s'est divisé en douze pour remplir chaque mois de l'écran suivant mais sans ajouter d'objectif dans le troisième écran avec les clients. Ensuite, si on modifie un objectif pour un client, cela modifie l'objectif dans le mois de l'écran précédant ainsi que celle de l'année. Une fois qu'il y a un objectif dans un client, si on modifie l'objectif de l'année ou du mois, l'objectif du client n'est pas modifié et il est ajouté au nouvel objectif de l'année ou du mois. Mais si on modifie un mois et ensuite l'année, la valeur du mois est recalculée et la modification est donc perdue. Malgré le fait que cette fonctionnalité me semble étrange cela fonctionnait déjà comme ça sur la version de base, je l'ai donc laissé ainsi.

2.2.5. Conclusion

En conclusion, comme pour la partie application, le développement passe par beaucoup de copier-coller et des tests poussés pour s'assurer que les données sont bien enregistrées comme on le souhaite.

2.4. Critique du Studio et de l'application

Dans cette partie, je vais parler des différents inconvénients et problèmes que peuvent comporter l'application Weavy et le Studio, qui sont, pour la plupart, dû au fait que l'application date de 2003 et que, malgré les mises à jour, il manque des choses qui pourraient être utiles.

Premier problème du Studio qu'on utilise pour développer, il n'y a pas de technique exacte pour retrouver un écran de l'application dans le studio, le moyen le plus rapide étant de regarder le titre de l'écran dans l'application puis d'essayer de trouver son nom dans la liste des écrans du Designer. Mais un écran n'a pas toujours son nom en titre.

Par exemple, l'écran de validation d'une commande se nomme seulement "Pied" et aucun écran du Designer ne se nomme "Foot" ou "Footer", il faut donc aller chercher dans l'arborescence le bouton permettant d'afficher la liste des pièces commerciales, la fonctionnalité liée à ce bouton va permettre de retrouver l'écran qui affiche la liste des commandes, depuis là on peut trouver le bouton qui crée une nouvelle commande et depuis l'écran de la nouvelle commande on trouve le bouton pour valider cette commande et on trouve enfin l'écran de validation de commande.

Tout ceci est très fastidieux et fait perdre beaucoup de temps, surtout au début. Ce problème est, pour moi, lié au fait que nous avons les mêmes accès à l'application qu'un commercial. Pour régler ce problème, on pourrait avoir une version de l'application presque identique, mais sur laquelle on se connecterait en tant que développeur, ce qui nous donnerait plus d'accès. Par exemple, on pourrait ajouter la possibilité de faire un clic droit sur des écrans ou des champs de l'application pour avoir leurs informations, comme leurs noms, leur arborescence dans le Studio (pour les retrouver facilement), les requêtes SQL ou les scripts liés.

Et surtout, cela permettrait de pouvoir activer une console qui listerait tout ce qui est appelé (requêtes, script, écran, ...) au fur et à mesure de l'utilisation de l'application car, par exemple, le seul moyen pour l'instant de vérifier si c'est le script qui ne récupère aucun client ou si c'est l'affichage de la liste qui ne fonctionne pas est d'ajouter une boîte de dialogue avec le résultat de la requête qui est censé donner la liste et cette solution pose des problèmes car en plus d'être longue à exécuter, elle augmente le risque d'oublier des boîtes de dialogues et cela risque de gêner les clients.

Second problème, l'affichage de l'application n'est pas responsive, c'est-à-dire que certaines tailles d'écrans, de boutons ou de texte ne s'adaptent pas à la taille de l'appareil utilisé. Pour régler ce problème, on pourrait, comme dit plus tôt, avoir une version de l'application pour développeur, ce qui nous donnerait plus d'accès et par exemple, on pourrait ajouter la possibilité de faire un clic droit sur des écrans ou des champs de l'application pour avoir leurs informations, comme leurs noms, leur arborescence dans le Studio (pour les retrouver facilement), les requêtes SQL ou les scripts liés. Et cela s'étend à toute l'application, les écrans de saisie multiple ou de validation de commande voient leurs boutons de droites disparaître les uns après les

autres si on réduit la fenêtre (cf : [ÉcranRéduit](#)). Ce problème existe aussi sur le Studio, je ne pouvais pas ajouter des champs à droite de l'écran si le Studio était sur l'écran de l'ordinateur portable (cf : [ÉcranRéduitStudio](#)).

3. Conclusion

3.1. Mes résultats

Le travail que j'ai effectué lors de ce stage est totalement fonctionnel et est par ailleurs déjà utilisé, de par les modifications apportées sur les applications des clients ou de par le standard OCl qui est utilisé pour les nouveaux clients.

Le standard n'est pas encore totalement achevé mais il a beaucoup avancé grâce à mon travail et celui de mon binôme. Les erreurs qu'il y avait au départ sont corrigées ce qui permet de le déployer pour nos futurs clients et il est beaucoup plus complet qu'à l'origine.

Mes missions n'ont pas pu complètement aboutir car les cahiers des charges de celles-ci sont en continuels évolutions, nous trouvons continuellement des éléments à ajouter ou modifier très souvent ce qui fait que rien n'est jamais réellement finalisé. Mais malgré cela j'ai fait progresser de nombreux projets durant mes cinq mois de stage.

3.2. Les apports personnels

Mon stage fut extrêmement intéressant à réaliser. J'ai tout d'abord pu intégrer une réelle équipe de développement où nous venions tous de parcours différents ce qui est très enrichissant et didactique. Il m'a permis de mettre en application les méthodes de travail acquises lors de ces trois années à la faculté de Dijon ainsi que les connaissances en base de données et en algorithmique qui m'ont grandement aidé durant mon stage. J'ai également bien compris l'importance de bien commenter son code pour qu'il soit facilement lisible quand on travaille à plusieurs sur le même projet.

De plus, j'ai énormément gagné en maturité grâce à la découverte du milieu professionnel. J'ai pu voir à quoi ressemblait l'informatique en dehors du cadre scolaire et cela me conforte dans mon choix de l'informatique et dans mon envie de rentrer dans le monde du travail dès maintenant.

3.3. Mon avenir

Ce stage m'a confirmé mon envie de continuer dans l'informatique, il m'a aussi montré les avantages à travailler dans une entreprise au sein d'une équipe qui peut m'aider dans mon travail et apporter une bonne ambiance de travail. Mais cela m'a aussi donné envie de me lancer en freelance pour pouvoir gérer moi-même toutes les phases du développement et être libre dans mes choix. C'est donc ce que je vais essayer de faire en septembre.

4. Bibliographie

<https://www.oci.fr/>

<https://www.divalto.com/logiciel-crm/>

5. Annexe



Figure 9 : Écran de ligne de commande dans l'application

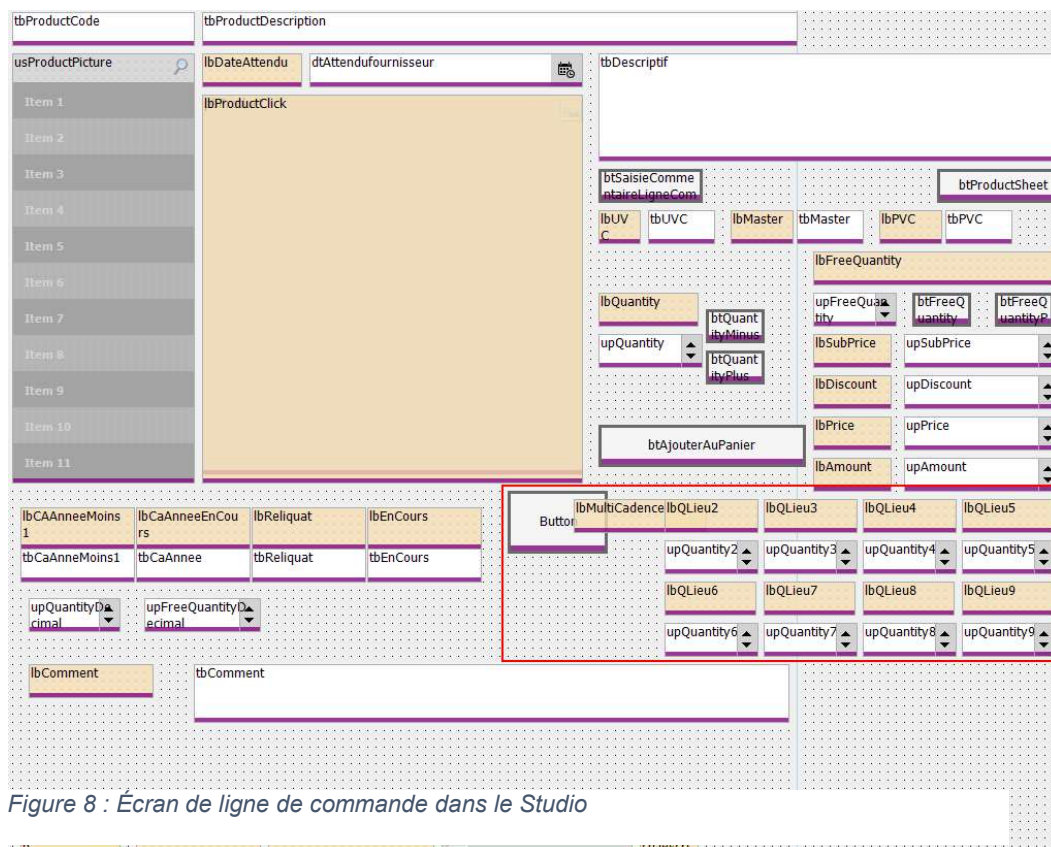


Figure 8 : Écran de ligne de commande dans le Studio

Annuler Pied Valider

Panier Livraison Livraison multiple Règlement Signature

LORRAINE DE TORREFACTION Commande Principale 1 9,00

Vous devez saisir 3 adresse(s) client(s)

	Client L2					Aucune date sélectionn.	0			0,00			Idem Client
	Client L3					Aucune date sélectionn.	0			0,00			Idem Client
	Client L4					Aucune date sélectionn.	0			0,00			Idem Client
	Client L5					Aucune date sélectionn.	0			0,00			Idem Client
	Client L6	BONDA		BISCHOFFSHEIM		mardi 13 juillet 2021	25			225,00			Idem Client
	Client L7	H BELIGNE ET FILS S		LANGRES		mardi 20 juillet 2021	5			45,00			Idem Client
	Client L8					Aucune date sélectionn.	0			0,00			Idem Client
	Client L9	LORRAINE DE TORF				mardi 20 juillet 2021	99			890,00			Idem Client

TOTAL : 130 1 169,

Figure 10 : Onglet de validation de commande

lbName		lbCommandePrincipale		nuQTL1		nuMtl1	
lbnombreAdresse							
Butto nL2	btCustomer L2	cbCustomer_ID_L2	cbCustomeraddress_ID_LivL2	dpDateL2	nuQTL2	nuMtl2	btCustomerIde mPrincipalL2
Butto nL3	btCustomer L3	cbCustomer_ID_L3	cbCustomeraddress_ID_LivL3	dpDateL3	nuQTL3	nuMtl3	btCustomerIde mPrincipalL3
Butto nL4	btCustomer L4	cbCustomer_ID_L4	cbCustomeraddress_ID_LivL4	dpDateL4	nuQTL4	nuMtl4	btCustomerIde mPrincipalL4
Butto nL5	btCustomer L5	cbCustomer_ID_L5	cbCustomeraddress_ID_LivL5	dpDateL5	nuQTL5	nuMtl5	btCustomerIde mPrincipalL5
Butto nL6	btCustomer L6	cbCustomer_ID_L6	cbCustomeraddress_ID_LivL6	dpDateL6	nuQTL6	nuMtl6	btCustomerIde mPrincipalL6
Butto nL7	btCustomer L7	cbCustomer_ID_L7	cbCustomeraddress_ID_LivL7	dpDateL7	nuQTL7	nuMtl7	btCustomerIde mPrincipalL7
Butto nL8	btCustomer L8	cbCustomer_ID_L8	cbCustomeraddress_ID_LivL8	dpDateL8	nuQTL8	nuMtl8	btCustomerIde mPrincipalL8
Butto nL9	btCustomer L9	cbCustomer_ID_L9	cbCustomeraddress_ID_LivL9	dpDateL9	nuQTL9	nuMtl9	btCustomerIde mPrincipalL9
lbTotal					nuQTLTot	nuMTot	

Figure 11 : Onglet de validation de commande dans le Studio

Divalto weavy [5.4.1] - □ ×

Annuler
Saisie multiple
Valider

LORRAINE DE TORREFACTION
Commande principale

1 🗑️ 9 🗑️

	Client	Adresse	Date	Quantité	Prix
🗑️	Client L2		Aucune date s...	0	0,00
🗑️	Client L3		Aucune date s...	0	0,00
🗑️	Client L4		Aucune date s...	0	0,00
🗑️	Client L5		Aucune date s...	0	0,00
🗑️	Client L6	BONDA BISCHOFFSHEIM 8 RUE EMILE MATHIS	mardi 13 juillet	25	225,00
🗑️	Client L7		Aucune date s...	5	45,00
🗑️	Client L8		Aucune date s...	0	0,00
🗑️	Client L9		Aucune date s...	99	890,00
TOTAL				130	1 169,00

Figure 12 : Écran de Saisie multiple réduit

lbName
lbCommandePrincipale
nuQTL1

lbnombreAdresse

Butto nL2	btCustomer L2	cbCustomer_ID_L2	cbCustomeraddress_ID_LivL2	dpDateL2	nuQTL2
			lbCustomerAddress_2		
Butto nL3	btCustomer L3	cbCustomer_ID_L3	cbCustomeraddress_ID_LivL3	dpDateL3	nuQTL3
			lbCustomerAddress_3		
Butto nL4	btCustomer L4	cbCustomer_ID_L4	cbCustomeraddress_ID_LivL4	dpDateL4	nuQTL4
			lbCustomerAddress_4		
Butto nL5	btCustomer L5	cbCustomer_ID_L5	cbCustomeraddress_ID_LivL5	dpDateL5	nuQTL5
			lbCustomerAddress_5		
Butto nL6	btCustomer L6	cbCustomer_ID_L6	cbCustomeraddress_ID_LivL6	dpDateL6	nuQTL6
			lbCustomerAddress_6		
Butto nL7	btCustomer L7	cbCustomer_ID_L7	cbCustomeraddress_ID_LivL7	dpDateL7	nuQTL7
			lbCustomerAddress_7		
Butto nL8	btCustomer L8	cbCustomer_ID_L8	cbCustomeraddress_ID_LivL8	dpDateL8	nuQTL8
			lbCustomerAddress_8		
Butto nL9	btCustomer L9	cbCustomer_ID_L9	cbCustomeraddress_ID_LivL9	dpDateL9	nuQTL9
			lbCustomerAddress_9		
lbTotal					nuQTLTot

Figure 13 : Écran de Saisie multiple dans le Studio réduit

Objectifs Client Actions ▼

Collaborateur ▼
37 éléments(s) Parcourir...

Rechercher Par défaut Effacer

Objectif CA (2020) 0	Objectif CA (2021) 13 688 100,00 % ↑	Objectif CA (2022) 99 -99,28 % ↓
-------------------------	--	--

	Obj. (2020)	Réal. (2020)	% (2020)	Obj. (2021)	Evol. Obj. (2020/2021)	Obj. (2022)	Evol. Obj. (2021/2022)
▼ Agence de Dijon (1)	0	0		4 180		0	
☑ Zanetti Philippe	0	0	100,00 %	4 180	100,00 % ↑	0	-100,00 % ↓
▼ Commerciaux (8)	0	0		9 165		0	
☑ LEPEZ Abdul	0	0	100,00 %	0	0,00 % —	0	0,00 %
☑ ABA Bart	0	0	100,00 %	0	0,00 % —	0	0,00 %
☑ PAMURE Clémentine	0	0	100,00 %	0	0,00 % —	0	0,00 %
☑ CICODE David	0	0	100,00 %	120	100,00 % ↑	0	-100,00 % ↓
☑ TASSION Félicie	0	0	100,00 %	9 024	100,00 % ↑	0	-100,00 % ↓

Figure 14 : Écran des objectifs globaux

Objectifs Client > Zanetti Philippe (PZ) Actions ▼

Objectif CA (2020) 0	Objectif CA (2021) 4 180 100,00 % ↑	Objectif CA (2022) 0 -100,00 % ↓
-------------------------	---	--

Mois	Obj. (2020)	Réal. (2020)	% (2020)	Obj. (2021)	Evol. Obj. (2020/2021)	Obj. (2022)	Evol. Obj. (2021/2022)
Janvier	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Février	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Mars	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Avril	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Mai	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Juin	0	0	100,00 %	4 158	100,00 % ↑	0	-100,00 % ↓
Juillet	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Août	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Septembre	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Octobre	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Novembre	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Décembre	0	0	100,00 %	2	100,00 % ↑	0	-100,00 % ↓
Total				4 180			

Figure 15 : Écran des objectifs d'un commercial par mois

Objectifs Client > Zanetti Philippe (PZ) > Juin Actions

Objectif Client (2020) Année fiscale

0

Objectif Client (2021) Année fiscale

4 158

100,00 % ↑

Objectif Client (2022) Année fiscale

0

-100,00 % ↓

Client	Obj. (2020)	Réal. (2020)	% (2020)	Obj. (2021)	Evol. Obj.	(2020/2021)	Obj. (2022)	Evol. Obj.	(2021/2022)
BOIMARE JOUET SA- VANNES	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
BRESSE DIS 2 SAS - LECLERC	0	0	100,00 %	4 145	100,00 %	↑	0	-100,00 %	↓
CLICHY DISTRIBUTION - CO...	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
CODIS ESPACE CULTUREL LE...	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
CORA COLMAR-HOUSSEN - ...	0	0	100,00 %	5	100,00 %	↑	0	-100,00 %	↓
CORA LENS - VENDIN LEM	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
ESPACE CULTUREL LECLERC ...	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
JC LESPARRE - JOUER EN MÉ...	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
JOUÉ CLUB (HOUSSEN)	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
JOUÉ CLUB - LESTRADE	0	0	100,00 %	0	0,00 %	—	0	0,00 %	—
Total				4 150					

Page 1 de 2 1 - 10 de 20 éléments

Figure 16 : Écran des objectifs d'un commercial par mois par client