STAGE Solti



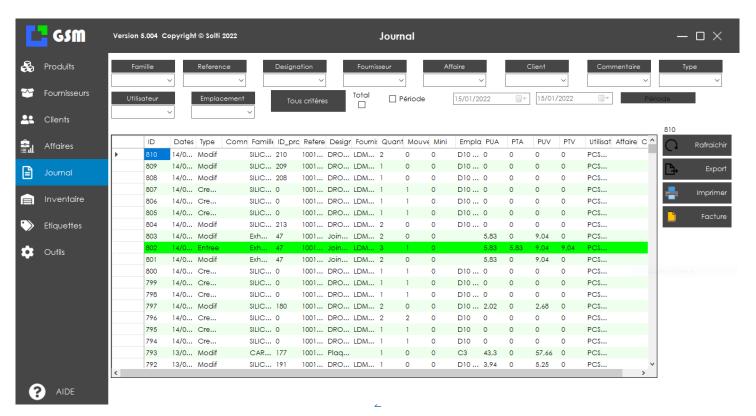
Sommaire

Introduction :	. 2
Mes missions à Solti :	. 3
Facture à partir du journal :	. 3
Conclusion	. 6
Niko Niko	. 6

<u>Introduction</u>:

Solti est une entreprise basée à Seyssinet-Pariset qui commercialise différents logiciels concernant la gestion de stock. Le logiciel nommé GSM (Gestion de stock modulaire) est le plus utilisé, il permet aux entreprises de gérer leur stock. Elle possède actuellement plus de 1550 client à travers 75 pays différents.

Durant mes semaines de stages mes mission seront l'ajout de nouvelles fonctionnalités sur leurs différents logiciels en fonction des besoins des clients et de l'entreprise.



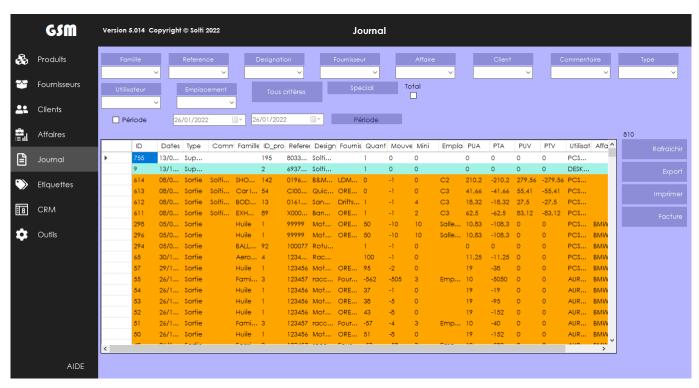
Mes missions à Solti :

Semaine 3:

Début de la troisième semaine, un client à fait la demande d'ajouter une nouvelle fonctionnalité dans le logiciel. Il voulait que dans le journal des sorties il puisse, en sélectionnant les lignes du tableau, créer une facture automatiquement. Cette mission m'a été confié car ayant passé les premières semaines de mon stage à manipuler des fichier Excel je me sentais à l'aise pour ajouter cette fonctionnalité.

Facture à partir du journal :

Le journal :



Il m'a donc fallu créer un bouton facture auquel j'ai ajouté la fonction permettant de créer une facture.

Dans la fonction ci-dessous j'ai d'abord récupéré les différentes informations utiles de chaque ligne du DataGridView que j'ai mis dans une liste. J'ai ensuite récupéré les informations du clients grâce à une requête SQL.

J'ai maintenant récupéré 2 listes qui me seront utile pour ma prochaine fonction qui me servira à remplir la facture avec en paramètre les 2 listes d'information dont j'ai besoin pour remplir une facture et le chemin du modèle de facture.

```
private void CreateInvoice()
   List<List<string>> factureList = new List<List<string>>();
   string client = "
    foreach (DataGridViewRow row in dataGridView1.SelectedRows)
       last_client = client;
       List<string> infoRow = new List<string>();
       string cellValue = CellValue("Type", row.Index);
string mouvement = CellValue("Mouvement", row.Index);
       int move = Tools.ConvToInt(mouvement) * -1;
        mouvement = Tools.ConvToString(move);
        if (cellValue == "Sortie" ^ cellValue == "Output")
            infoRow.Add(row.Cells["Reference"].Value.ToString());
            infoRow.Add(row.Cells["Designation"].Value.ToString());
            infoRow.Add(mouvement);
            infoRow.Add(row.Cells["PUV"].Value.ToString());
            factureList.Add(infoRow);
            client = row.Cells["Client"].Value.ToString();
            if (client == "")
                MessageBox.Show("Aucun client n'est renseigné.");
            else if (last_client != "" & client != last_client)
                    MessageBox.Show("Plusieurs clients ont été renseignés");
```

```
else
{
    MessageBox.Show("Une des lignes selectionnés n'est pas une sortie");
    return;
}

if (factureList.Count == 0)
{
    MessageBox.Show("Aucune ligne séléctionnée");
    return;
}
int err = m_GSM.Open();
string cmd = "SELECT ID,Societe,Adresse,Code,Ville FROM t_clients WHERE Societe = '" + client + "';";
List<object> objClient = m_GSM.ExecuteReader(cmd);
    m_GSM.Close();
    m_GSM.Close();
    m_GSM.Error(err);
List<string> infosClient = Tools.ConvObjListToStringList(objClient);

string src = Environment.CurrentDirectory + @"\MODELE\Facture_Motorsports.xls";

FillInvoiceCustomer(factureList,infosClient,src);
}
```

Dans FillInvoiceCustomer je me sert de la class CExcelDoc dans laquelle il y à quelques méthodes que l'entreprise se sert pour manipuler des fichier Excel et où j'ai moi-même rajouter quelques méthodes dont j'aurais besoin. Le procéder est assez similaire à ce que j'ai pu faire la semaine précédente où je devais transformer un devis en facture.

```
private void FillInvoiceCustomer(List<List<string>> Tablist, List<string> infosClient,string src)
   CExcelDoc DT = new CExcelDoc();
   int err = DT.OpenExcelDocument(src);
   if (err != 0) MessageBox.Show("Erreur sur " + src);
   string date = DateTime.Now.ToString("d");
   string societe = infosClient[1];
   string adresse = infosClient[2];
   string CP = infosClient[3];
   string ville = infosClient[4];
   DT.Replace2("§DATE§", date, false);
   DT.Replace2("§SOCIETE2§, §ADRESSE2§, §CP2§, §VILLE2§", societe + " " + adresse + " " + CP + " " + ville, false);
DT.Replace2("§SOCIETE3§", societe, false);
   DT.FillTable(15, 3, TabList);
   DT.Write2(11, 7, date);
   DT.Write2(12, 4, societe + " " + adresse + " " + CP + " " + ville);
   DT.Replace2("§SOCIETE3§", societe, false);
   string factureName = DT.SetFactureName("GSM_Facture.xlsx", societe);
   string dst = Environment.CurrentDirectory + @"\DOC\" + factureName;
   string[] tokenFacture = factureName.Split('_');
   string invoiceNumber = tokenFacture[0];
   DT.Write2(12, 7, invoiceNumber);
   int row = 29;
   int dest_col = 3;
    while (DT.ReadCell(row - 1, dest_col) == "")
       DT.DeleteRow(dest_col, row);
   DT.SaveAs(dst);
    SoltiProcess.Start(dst);
```

Conclusion

Durant cette semaine j'ai été très occupé à chercher des solutions aux différents problèmes lié à ma nouvelle fonctionnalité et aussi à corrigé les nombreux problèmes qu'il y avait avec la fonctionnalité que j'avais mise en place la semaine dernière. Je découvre donc que le débogage est une partie importante du métier de développeur.

Niko Niko

Lundi	Mardi	Mercredi	Jeudi	Vendredi