Déclaration des variables globales du formulaire de départ :

```
string chcon = @"Provider = Microsoft.ACE.OLEDB.12.0; Data Source = ";
OleDbConnection connec = new OleDbConnection();
DataSet ds = new DataSet();
List<int> choix_ingredients = new List<int>();
List<int> choix_categories = new List<int>();
int valeur;
```

Procédure évènementielle affichage des ingrédients d'une famille :

(Le sender est une RadioButton d'une famille dont le Tag est le numéro de la famille)

```
private void rb_CheckedChanged(object sender, EventArgs e)
        {
            int i = 0;
            int j = 0;
            int max = 0;
            RadioButton rb = (RadioButton)sender;
            if (rb.Checked)
            {
                lbl_famille.Text = rb.Text;
                gb_ingredients.Controls.Clear();
                foreach (DataRow row in ds.Tables["Ingrédients"].Rows)
                {
                    if (row["codeFamille"].ToString() == rb.Tag.ToString())
                    {
                        CheckBox ch = new CheckBox();
                        ch.Text = row["libIngredient"].ToString();
                        ch.Font = new System.Drawing.Font("Verdana", 8.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
                        ch.Tag = row["codeIngredient"];
                        ch.BringToFront();
                        ch.ForeColor = System.Drawing.Color.White;
                        gb_ingredients.Controls.Add(ch);

                        ch.Left = 20+ j;
                        ch.Top = 20 + i;
                        ch.AutoSize = true;
                        i += 26;
                        if (i == 12*26)
                        {
                            j += 210;
                            i = 0;
                        }
                        if (ch.Width > max && ch.Left == 20)
                        {
                            max = ch.Width;
                        }
                        foreach(int val in choix_ingredients)
                        {
                            if (int.Parse(ch.Tag.ToString()) == val)
                            {
                                ch.Checked = true;
                            }
                        }
                        ch.CheckedChanged += new
System.EventHandler(this.cb_CheckedChanged);
```

```
                    }
                }
                foreach(CheckBox ch  in
gb_ingredients.Controls.OfType<CheckBox>())
                {
                    if(ch.Left != 20)
                    {
                        ch.Left = max + 30;
                    }
                }
            }
        }
```

Procédure évènementielle de recherche des recettes :

```
private void frmRecettes_Load(object sender, EventArgs e)
        {
            string path = Directory.GetCurrentDirectory();
            chcon = chcon + path + @"\\baseFrigov2023.mdb";
            connec.ConnectionString = chcon;
            connec.Open();
            if(ingredients.Count > 1)
            {
                for (int i = 0; i < ingredients.Count; i++)
                {
                    OleDbCommand d2 = new
OleDbCommand(contruction_requette(ingredients[i]), connec);
                    recette.Load(d2.ExecuteReader());
                }
            }
            else
            {
                OleDbCommand d2 = new OleDbCommand(contruction_requette(),
connec);
                recette.Load(d2.ExecuteReader());
            }
            connec.Close();
            trier();
            chargement_recette();

        }

private void trier()
        {
            Dictionary<int, int> temp = new Dictionary<int, int>();
            foreach (DataRow row in recette.Rows)
            {
                int num = row.Field<int>(0);
                if (temp.ContainsKey(num))
                {
                    temp[num]++;
                }
                else
                {
                    temp[num] = 1;
                }
            }
            var pourliste = from entry in temp orderby entry.Value descending
select entry;
            foreach (var entry in pourliste)
            {
```

```
                recette_trier.Add(entry.Key);
            }
        }


private String contruction_requette(int ingredient)
        {
            List<String> condition = new List<String>();
            int parenthese = 0;
            String inner_temp = "Recettes r";
            String requete = "SELECT r.codeRecette FROM ";
            if(categories.Count > 0)
            {
                parenthese++;
                inner_temp += " INNER JOIN CatégoriesRecette d ON r.codeRecette =
d.codeRecette)";
                String cond= "d.codeCategorie IN (";
                for (int i = 0; i < categories.Count; i++)
                {
                    cond += categories[i];
                    if (i < categories.Count - 1)
                    {
                        cond += ", ";
                    }
                }
                cond += ")";
                condition.Add(cond);
            }
            if(ingredients.Count > 0)
            {
                parenthese++;
                inner_temp += " INNER JOIN IngrédientsRecette i ON r.codeRecette
= i.codeRecette)";
                String cond = "i.codeIngredient = " + ingredient;
                condition.Add(cond);
            }
            if(echelle_prix != 0)
            {
                condition.Add("r.categPrix <= " + echelle_prix);
            }
            if(temps_max != 0)
            {
                condition.Add("r.tempsCuisson <= " + temps_max);
            }

            for(int i= 0; i<parenthese;i++)
            {
                requete += "(";
            }
            requete += inner_temp;
            if(condition.Count > 0)
            {
                requete += " WHERE " + String.Join(" AND ", condition);
            }
            return requete;
        }
        private String contruction_requette()
        {
            List<String> condition = new List<String>();
            int parenthese = 0;
            String inner_temp = "Recettes r";
            String requete = "SELECT r.codeRecette FROM ";
```

```
            if (categories.Count > 0)
            {
                parenthese++;
                inner_temp += " INNER JOIN CatégoriesRecette d ON r.codeRecette =
d.codeRecette)";
                String cond = "d.codeCategorie IN (";
                for (int i = 0; i < categories.Count; i++)
                {
                    cond += categories[i];
                    if (i < categories.Count - 1)
                    {
                        cond += ", ";
                    }
                }
                cond += ")";
                condition.Add(cond);
            }
            if (ingredients.Count > 0)
            {
                parenthese++;
                inner_temp += " INNER JOIN IngrédientsRecette i ON r.codeRecette
= i.codeRecette)";
                String cond = "i.codeIngredient IN (";
                for (int i = 0; i < ingredients.Count; i++)
                {
                    cond += ingredients[i];
                    if (i < ingredients.Count - 1)
                    {
                        cond += ", ";
                    }
                }
                cond += ")";
                condition.Add(cond);
            }
            if (echelle_prix != 0)
            {
                condition.Add("r.categPrix <= " + echelle_prix);
            }
            if (temps_max != 0)
            {
                condition.Add("r.tempsCuisson <= " + temps_max);
            }

            for (int i = 0; i < parenthese; i++)
            {
                requete += "(";
            }
            requete += inner_temp;
            if (condition.Count > 0)
            {
                requete += " WHERE " + String.Join(" AND ", condition);
            }
            return requete;
        }
```

Code complet de UserControl UCrecette :

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Fridgeo
{
    public partial class UCRecette : UserControl
    {
        public UCRecette()
        {
            InitializeComponent();
        }

        string NomRecette;
        int Temps;
        int Prix;
        int note;
        int etoile;
        string cheminImage;

        public UCRecette(string CNomRecette, int CTemps, int CPrix, string
CcheminImage, int etoilesnb, int notesnb, System.EventHandler h, int recette)
        {
            InitializeComponent();
            NomRecette = CNomRecette;
            Temps = CTemps;
            Prix = CPrix;
            cheminImage = CcheminImage;
            etoile = etoilesnb;
            note = notesnb;
            lbl_details.Click += h;
            pbc_details.Click += h;
            lbl_details.Tag = recette;
            pbc_details.Tag = recette;
        }

        private void UCRecette_Load(object sender, EventArgs e)
        {
            Etoiles etoiles = new Etoiles(etoile,note);
            etoiles.Left = 150;
            etoiles.Top = 115;
            etoiles.Width = 314;
            etoiles.Height = 38;
            this.Controls.Add(etoiles);
            etoiles.BringToFront();
            lblRecette.Text = NomRecette;
            lblTemp.Text = "Temps de préparation " + Temps + " minutes";
            if(Prix == 1)
            {
                lblPrix.Text = "€";
            }
            else if (Prix == 2)
            {
                lblPrix.Text = "€€";
            }
            else
            {
                lblPrix.Text = "€€€";
```

```
            }
            Image image = Image.FromFile(cheminImage);
            pcbImageRecette.Image = image;


        }
    }
}
```

Code permettant de générer le pdf :

```
public void GenererPDF()
        {
            string fichierSortie = Environment.CurrentDirectory + "/recette.pdf";
            // Création du document
            Document doc = new Document();
            PdfWriter.GetInstance(doc, new FileStream(fichierSortie,
FileMode.Create));

            doc.Open();

            // Palette de couleurs
            BaseColor noir = new BaseColor(27, 27, 27);
            BaseColor orange = new BaseColor(255, 163, 26);
            BaseColor grisclair = new BaseColor(128, 128, 128);
            BaseColor grisfonce = new BaseColor(41, 41, 41);
            BaseColor blanc = BaseColor.WHITE;

            // Polices d'écriture
            iTextSharp.text.Font policeRecette = new
iTextSharp.text.Font(iTextSharp.text.Font.FontFamily.HELVETICA, 30f,
iTextSharp.text.Font.BOLD, orange);
            iTextSharp.text.Font policeTitre = new
iTextSharp.text.Font(iTextSharp.text.Font.FontFamily.HELVETICA, 20f,
iTextSharp.text.Font.BOLD, orange);
            iTextSharp.text.Font policeTexte = new
iTextSharp.text.Font(iTextSharp.text.Font.FontFamily.HELVETICA, 16f,
iTextSharp.text.Font.BOLD, blanc);

            // Page PDF
            // Couleur de fond du pdf
            iTextSharp.text.Rectangle pageSize = doc.PageSize;
            iTextSharp.text.Image background =
iTextSharp.text.Image.GetInstance(CreateBackgroundImage(pageSize.Width,
pageSize.Height, new BaseColor(41, 41, 41)));
            background.ScaleToFit(pageSize.Width, pageSize.Height);
            background.SetAbsolutePosition(0, 0);
            doc.Add(background);

            // Création de paragraphes

            // Logo de fridgeo
            string cheminImage = "..\\..\\Resources\\fridgeo_logo_scaled.png";
            iTextSharp.text.Image img =
iTextSharp.text.Image.GetInstance(cheminImage);
            img.ScaleToFit(150, 80);
            img.Alignment = Element.ALIGN_CENTER;
            doc.Add(img);
```

```
            // Nom de la recette
            Paragraph NomRecette = new Paragraph(lblTitre.Text, policeRecette);
            NomRecette.Alignment = Element.ALIGN_LEFT;
            doc.Add(NomRecette);

            //Prix et personnes
            Paragraph InfosRecette = new Paragraph(lbl_personnes_prix.Text +
"\n\n", policeTexte);
            InfosRecette.Alignment = Element.ALIGN_LEFT;
            doc.Add(InfosRecette);

            //Image de la recette
            string cheminRecette = "..\\..\\Resources\\" +
recette.Rows[0]["imageDesc"].ToString();
            iTextSharp.text.Image imgrec =
iTextSharp.text.Image.GetInstance(cheminRecette);
            imgrec.ScaleAbsolute(80, 80);
            imgrec.Alignment = Element.ALIGN_LEFT;
            doc.Add(imgrec);

            // Description de la recette
            Paragraph Description = new Paragraph(lbl_description.Text + "\n\n",
policeTexte);
            Description.Alignment = Element.ALIGN_LEFT;
            doc.Add(Description);

            // Ingrédients de la recette
            StringBuilder ingredientsText = new StringBuilder();
            foreach (Control control in gbp_ingredients.Controls)
            {
                if (control is Label label)
                {
                    ingredientsText.AppendLine(label.Text);
                }
            }
            Paragraph TitreIngredients = new Paragraph("Ingrédients : \n\n",
policeTitre);
            Paragraph Ingredients = new Paragraph(ingredientsText.ToString(),
policeTexte);
            Ingredients.Alignment = Element.ALIGN_LEFT;
            doc.Add(TitreIngredients);
            doc.Add(Ingredients);

            //Espace entre les ingredients et les etapes
            Paragraph Espace = new Paragraph("\n\n");
            doc.Add(Espace);

            //Titre Etape
            Paragraph Etape = new Paragraph("Etapes : \n\n", policeTitre);
            doc.Add(Etape);

            // Création du tableau
            PdfPTable table = new PdfPTable(2);
            table.WidthPercentage = 100;

            // Définir les largeurs des colonnes
            float[] columnWidths = { 30f, 70f }; // Largeur de chaque colonne en
pourcentage
            table.SetWidths(columnWidths);



            int i = 1;
```

```
            foreach (DataRow row in etape.Rows)
            {
                // Colonne de gauche : Image des étapes
                string cheminEtape = "..\\..\\Resources\\" +
row["imageEtape"].ToString();
                iTextSharp.text.Image imgEtape;

                if (string.IsNullOrEmpty(row["imageEtape"].ToString()) ||
!File.Exists(cheminEtape))
                {
                    // Utiliser l'image par défaut si le chemin est vide ou si le
fichier n'existe pas
                    cheminEtape = "..\\..\\Resources\\fridgeo_logo.png";
                }

                iTextSharp.text.Image image =
iTextSharp.text.Image.GetInstance(cheminEtape);
                imgEtape = image;
                PdfPCell cellImage = new PdfPCell(imgEtape);
                cellImage.HorizontalAlignment = PdfPCell.ALIGN_CENTER;
                cellImage.VerticalAlignment = PdfPCell.ALIGN_MIDDLE;
                imgEtape.ScaleAbsolute(50, 50);
                cellImage.FixedHeight = 50;

                // Définir la couleur des bordures de la cellule de gauche
                cellImage.BorderColor = new BaseColor(255, 163, 26); // Couleur
des bordures

                table.AddCell(cellImage);

                // Colonne de droite : Description des étapes
                PdfPCell cellDescription = new PdfPCell();
                cellDescription.AddElement(new Phrase("Etape " + i + " : " +
row["texteEtape"].ToString(), new
iTextSharp.text.Font(iTextSharp.text.Font.FontFamily.HELVETICA, 12,
iTextSharp.text.Font.NORMAL, new BaseColor(255, 163, 26))));
                cellDescription.VerticalAlignment = PdfPCell.ALIGN_MIDDLE;

                // Définir la couleur des bordures de la cellule de droite
                cellDescription.BorderColor = new BaseColor(255, 163, 26); //
Couleur des bordures

                table.AddCell(cellDescription);

                i++;
            }

            doc.Add(table);

            // Couleur de fond de la 2nd page du pdf
            background =
iTextSharp.text.Image.GetInstance(CreateBackgroundImage(pageSize.Width,
pageSize.Height, new BaseColor(41, 41, 41)));
            background.ScaleToFit(pageSize.Width, pageSize.Height);
            background.SetAbsolutePosition(0, 0);
            doc.Add(background);


            // Liste de courses
            Paragraph ListeCourses = new Paragraph("Liste de courses :\n\n",
policeTitre);
            doc.Add(ListeCourses);
```

```
            // Texte des ingrédients
            string texteIngredients = ingredientsText.ToString();

            // Séparer les ingrédients par des sauts de ligne
            string[] ingredients = texteIngredients.Split(new[] { "\r\n", "\r",
"\n" }, StringSplitOptions.RemoveEmptyEntries);

            // Créer une liste non ordonnée pour les ingrédients
            List liste = new List(List.UNORDERED);
            liste.SetListSymbol("\u2022"); // Utiliser le symbole • pour chaque
élément de la liste

            // Ajouter chaque ingrédient à la liste
            foreach (string ingredient in ingredients)
            {
                liste.Add(new ListItem(ingredient, policeTexte));
            }

            // Ajouter la liste de courses au document
            doc.Add(liste);


            // Couleur de fond de la page actuelle du pdf
            background =
iTextSharp.text.Image.GetInstance(CreateBackgroundImage(pageSize.Width,
pageSize.Height, new BaseColor(41, 41, 41)));
            background.ScaleToFit(pageSize.Width, pageSize.Height);
            background.SetAbsolutePosition(0, 0);
            doc.Add(background);


            // Fermer le document
            doc.Close();
            Process.Start(fichierSortie);
        }

        private iTextSharp.text.Image CreateBackgroundImage(float width, float
height, BaseColor color)
        {
            Bitmap bitmap = new Bitmap((int)width, (int)height);
            using (Graphics g = Graphics.FromImage(bitmap))
            {
                g.Clear(Color.FromArgb(color.R, color.G, color.B));
            }
            return iTextSharp.text.Image.GetInstance(bitmap, ImageFormat.Png);
        }
```

Code permettant la navigation en liaison de données :

```
DataTable etape = new DataTable();
        System.Drawing.Image[] etape_bd;
        BindingSource bd;
        BindingSource bd2;

    chargement_etape_bd();

     bd = new BindingSource();
            bd2 = new BindingSource();
```

```
            bd.DataSource = etape;
            bd2.DataSource = etape_bd;
            lbl_etape_bd.DataBindings.Add("Text", bd, "texteEtape");
            pcb_etape_bd.DataBindings.Add("Image",bd2,"");

private void chargement_etape_bd()
        {
            etape_bd = new System.Drawing.Image[etape.Rows.Count];

            for (int i = 0; i < etape.Rows.Count; i++)
            {
                string relativePath = etape.Rows[i]["imageEtape"].ToString();
                string fullPath = Path.Combine(Application.StartupPath,
"..\\..\\Resources\\", relativePath);
                System.Drawing.Image image =
System.Drawing.Image.FromFile(fullPath);
                etape_bd[i] = image;
            }
        }

private void pcb_fleche_debut_Click(object sender, EventArgs e)
        {
            bd.MoveFirst();
            bd2.MoveFirst();
        }

        private void pcb_debut_etape_Click(object sender, EventArgs e)
        {
            bd.MovePrevious();
            bd2.MovePrevious();
        }

        private void pcb_fleche_suivante_Click(object sender, EventArgs e)
        {
            bd.MoveNext();
            bd2.MoveNext();
        }

        private void pcb_fleche_fin_Click(object sender, EventArgs e)
        {
            bd.MoveLast();
            bd2.MoveLast();
        }
```