

**Kevin
Hawed
Yanis
Miloud**

PROJET



Gestion du matériel et des incidents matériels

Sommaire :

<https://prod.liveshare.vsengsaas.visualstudio.com/join?B55BBBDBE47429D60537A2317BE886C6F5F3>

1. Structure BD

1.1. MLD

1.2. MCD

2. Schéma de classes

3. Screens interface

4. Description de chaque clic

5. Manuel utilisateur

6. Script sauvegarde BD + tuto génération set-up

7. Diagramme

8. Gantt

8.1. Gantt prévisionnel - Gantt effectif avec affectation des ressources

8.2. Gantt réel

Structure BD

MLD :

utilisateur = (id_utilisateur VARCHAR(50), nom VARCHAR(50), prenom VARCHAR(50), email VARCHAR(50), mot_de_passe VARCHAR(50), type_utilisateur VARCHAR(50));

technicien = (id_technicien VARCHAR(50), niveau_d_intervention VARCHAR(50), formation VARCHAR(50), nom VARCHAR(50), prenom VARCHAR(50), competences_technicien VARCHAR(50));

materiel = (id_materiel VARCHAR(50), nomMat VARCHAR(50), type VARCHAR(50), carcteristique VARCHAR(50), date_achat VARCHAR(50), garantie VARCHAR(50), #id_utilisateur);

incident = (id_incident VARCHAR(50), description VARCHAR(50), date_demande VARCHAR(50), statut VARCHAR(50), date_resolution VARCHAR(50), commentaire VARCHAR(50), #id_materiel, #id_utilisateur);

interventions = (id_intervention VARCHAR(50), date_interventiun VARCHAR(50), duree VARCHAR(50), description_travail VARCHAR(50), #id_incident, #id_technicien);

MCD :

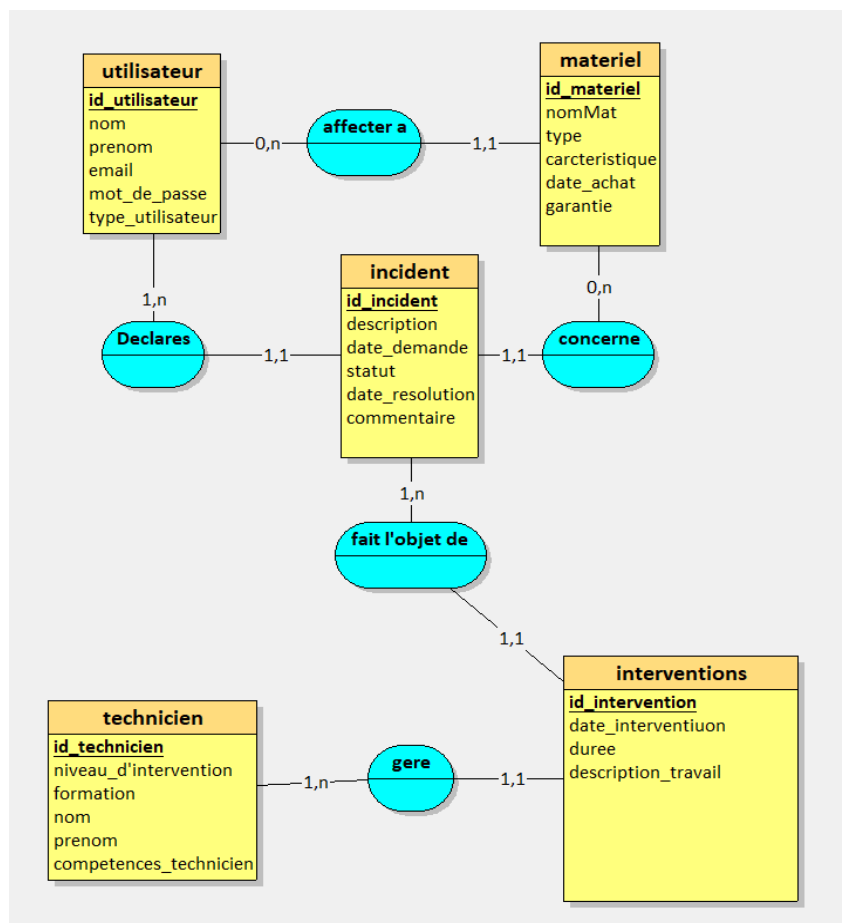


Schéma de classes

RÉALISATION DES CLASSES MÉTIERS (classes issues des tables en BD)

UTILISATEUR :

```
public class Utilisateur
{
    private int Id_utilisateurs;
    private string nom;
    private string prenom;
    private string email;
    private string mot_de_passe;
    private string type_utilisateur;

    public Utilisateur(int Unid_utilisateurs, string Unnom, string Unprenom, string Unemail,
string Unmot_de_passe, string Untype_utilisateur)
    {
        this.Id_utilisateurs = Unid_utilisateurs;
        this.nom = Unnom;
        this.prenom = Unprenom;
        this.email = Unemail;
        this.mot_de_passe = Unmot_de_passe;
        this.type_utilisateur = Untype_utilisateur;
    }
    public Utilisateur(string Unnom, string Unprenom, string Unemail, string
Unmot_de_passe, string Untype_utilisateur)
    {
        this.nom = Unnom;
        this.prenom = Unprenom;
        this.email = Unemail;
        this.mot_de_passe = Unmot_de_passe;
        this.type_utilisateur = Untype_utilisateur;
    }
    public int getId()
    {
        return Id_utilisateurs;
    }
    public string getUnnom()
    {
        return nom;
    }
    public string getUnprenom()
    {
        return prenom;
    }
    public string getUnemail()
    {
```

```

        return email;
    }
    public string getUnmot_de_passe()
    {
        return mot_de_passe;
    }
    public string getUntype_utilisateur()
    {
        return type_utilisateur;
    }
}
}

```

La classe Utilisateur représente un utilisateur dans un système avec des informations essentielles telles que son identifiant (ID), son nom, son prénom, son email, son mot de passe, et son type d'utilisateur (par exemple, administrateur, utilisateur standard, etc.).

MATERIEL :

```

public class Materiel
{
    private int Id_Materiel;
    private string nomMat;
    private string type;
    private string carcteristique;
    private string date_achat;
    private string garantie;

    public int Id_Materiel1 { get => Id_Materiel; set => Id_Materiel = value; }
    public string NomMat { get => nomMat; set => nomMat = value; }
    public string Type { get => type; set => type = value; }
    public string Carcteristique { get => carcteristique; set => carcteristique = value; }
    public string Date_achat { get => date_achat; set => date_achat = value; }
    public string Garantie { get => garantie; set => garantie = value; }

    public Materiel(int unId_Materiel, string UnnomMat, string Untype, string
Uncarcteristique, string Undate_achat, string Ungarantie)
    {
        this.Id_Materiel1 = unId_Materiel;
        this.NomMat = UnnomMat;
        this.Type = Untype;
        this.Carcteristique = Uncarcteristique;
        this.Date_achat = Undate_achat;
    }
}

```

```

        this.Garantie = Ungarantie;
    }
    public Materiel(string UnnomMat, string Untype, string Uncarcteristique, string
Undate_achat, string Ungarantie)
    {
        this.NomMat = UnnomMat;
        this.Type = Untype;
        this.Carcteristique = Uncarcteristique;
        this.Date_achat = Undate_achat;
        this.Garantie = Ungarantie;
    }
}
}

```

Cette classe modélise un matériel avec ses caractéristiques importantes comme son nom, son type, ses spécifications techniques, et sa garantie.

TECHNICIEN :

```

public class Technicien
{
    private int id_technicien;
    private string niveau_Dintervention;
    private string mot_de_passe;
    private string formation;
    private string competences_technicien;

    public Technicien( int unId_technicien, string UnNiveau_Dintervention, string
UnMot_de_passe, string UneFormation, string UneCompetences_technicien)
    {
        this.id_technicien = unId_technicien;
        this.niveau_Dintervention = UnNiveau_Dintervention;
        this.mot_de_passe = UnMot_de_passe;
        this.formation = UneFormation;
        this.competences_technicien = UneCompetences_technicien;
    }
    public Technicien( string UnNiveau_Dintervention, string UnMot_de_passe, string
UneFormation, string UneCompetences_technicien)
    {
        this.niveau_Dintervention = UnNiveau_Dintervention;
        this.mot_de_passe = UnMot_de_passe;
        this.formation = UneFormation;
        this.competences_technicien = UneCompetences_technicien;
    }
}

```

```

    }
    public int getId()
    {
        return id_technicien;
    }

    public string getNiveau_Dintervention()
    {
        return niveau_Dintervention;
    }

    public string getMot_de_passe()
    {
        return mot_de_passe;
    }

    public string getFormation()
    {
        return formation;
    }

    public string getCompetences_technicien()
    {
        return competences_technicien;
    }
}

```

La classe Technicien représente les techniciens, avec des informations spécifiques à leur rôle telles que leur niveau d'intervention et leurs compétences.

INTERVENTION :

```

public class Intervention
{
    private int Id_intervention;
    private string date_intervention;
    private int duree;
    private string description_travail;

    public Intervention(int Unid_intervention, string Undate_intervention, int Unduree, string Undescription_travail)
    {
        this.Id_intervention = Unid_intervention;
        this.date_intervention = Undate_intervention;
        this.duree = Unduree;
        this.description_travail = Undescription_travail;
    }
}

```

```

    }
    public Intervention(string Undate_intervention, int Unduree, string Undescription_travail)
    {
        this.date_intervention = Undate_intervention;
        this.duree = Unduree;
        this.description_travail = Undescription_travail;
    }
    public int getId()
    {
        return Id_intervention;
    }
    public string getDateIntervention()
    {
        return date_intervention;
    }
    public int getDuree()
    {
        return duree;
    }
    public string getDescription()
    {
        return description_travail;
    }
}
}

```

Cette classe modélise une intervention technique avec des détails comme la date, la durée, et la description des travaux effectués.

INCIDENT :

```

public class Incident
{
    private int id_incident;
    private string description;
    private string dateDemande;
    private string statut;
    private string dateResolution;
    private string commentaire;
    private string poste;

    public int Id_incident { get => id_incident; set => id_incident = value; }
    public string Description { get => description; set => description = value; }
    public string DateDemande { get => dateDemande; set => dateDemande = value; }
    public string Statut { get => statut; set => statut = value; }
}

```



```

public string DateResolution { get => dateResolution; set => dateResolution = value; }
public string Commentaire { get => commentaire; set => commentaire = value; }
public string Poste { get => poste; set => poste = value; }

public Incident(int Uneid_incident, string Unedescription, string UnedateDemande, string
Unestatut, string UnedateResolution, string Uncommentaire, string Unposte)
{
    this.Id_incident = Uneid_incident;
    this.Description = Unedescription;
    this.DateDemande = UnedateDemande;
    this.Statut = Unestatut;
    this.DateResolution = UnedateResolution;
    this.Commentaire = Uncommentaire;
    this.Poste = Unposte;
}
public Incident( string Unedescription, string UnedateDemande, string Unestatut, string
UnedateResolution, string Uncommentaire, string Unposte)
{
    this.Description = Unedescription;
    this.DateDemande = UnedateDemande;
    this.Statut = Unestatut;
    this.DateResolution = UnedateResolution;
    this.Commentaire = Uncommentaire;
    this.Poste = Unposte;
}
public Incident ( string Unedescription, string Unposte)
{
    this.description = Unedescription;
    this.poste = Unposte;
}
}

```

Cette classe modélise un incident signalé avec ses détails tels que la description, les dates importantes, le statut et les commentaires.

Screens interface

Connexion	Utilisateur1	Utilisateur2	Technicien1	Technicien2	Admin	tabPage7	tabPage8	Technicien3	Utilisateur3
-----------	--------------	--------------	-------------	-------------	-------	----------	----------	-------------	--------------

E-mail

Mot De Passe

Connexion	Utilisateur1	Utilisateur2	Technicien1	Technicien2	Admin	tabPage7	tabPage8	Technicien3	Utilisateur3
-----------	--------------	--------------	-------------	-------------	-------	----------	----------	-------------	--------------

Connexion	Utilisateur1	Utilisateur2	Technicien1	Technicien2	Admin	tabPage7	tabPage8	Technicien3	Utilisateur3
-----------	--------------	--------------	-------------	-------------	-------	----------	----------	-------------	--------------

description

date_demande

urgence

id materiel

id utilisateur

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin tabPage7 tabPage8 Technicien3 Utilisateur3

Consulter materiel

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin tabPage7 tabPage8 Technicien3 Utilisateur3

id utilisateur 1

nom mat

type

caracteristique

date achat

garantie

ajouter

actualiser supprimer

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

gerer les techniciens gerer les utilisateur

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

ID

Nom

Prenom

formation

compétence

niveau d'intervention

Créer

Actualiser

Retour

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

ID

Nom

Prenom

Email

Type

Créer

Actualiser

Retour

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

description

date_demande

urgence

id_materiel

id_utilisateur

Sousmettre


Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

Actualiser

Id utilisateur

Retour

RÉPARTITION DES TÂCHES:

Tâches	Description	État
Réalisation de la structure de la BD	Hawed, Miloud	Terminé ▾
Prototype gantt prévisionnel (tâches + ressources)  ProjetGanttLabo.gan	Kevin, Yanis	Terminé ▾
Mise en place BD	Hawed	Terminé ▾
Réalisation des classes métiers (classes issues des tables en BD)	Tous	Terminé ▾
Conception de l'interface	Tous	Terminé ▾
Réalisation de l'interface	Kevin	Terminé ▾
Réalisation de la classe BD	Kevin	Terminé ▾
Réalisation du programme principal	Kevin	Terminé ▾
Débug du programme	Kevin	Terminé ▾
Générer un set-up	Kevin	Terminé ▾
Gantt réel	Yanis, Miloud	Terminé ▾
Guide Utilisateur	Yanis, Kevin	Terminé ▾
diagramme de classe	Yanis	Terminé ▾
Commentaire format XML	Kevin	Terminé ▾
Guide d'utilisation	Yanis	Terminé ▾
Rédaction rapport de projet(voir doc attendue)	Tous	Terminé ▾

- séance 1 :

structure BD (**Hawed**, **Miloud** (2h)) +

prototype gantt prévisionnel (tâches + ressources) (**Yanis**, **Kevin** (1h))

- séance 2 : mise en place BD (**Yanis** (2h)) +

réalisation des classes métiers (classes issues des tables en BD) (**Hawed**, **Kevin**, **Miloud**, **Yanis** (2h)) + Réalisation de l'interface (**Kevin** (1h))

- séance 3 : réalisation de la classe BD (**Tous** 2h) +

réalisation du programme principal (**Tous** 2h)

- séance 4 : debug du programme (**Kevin**)

+ rédaction rapport de projet(voir doc attendue) + générer un set-up (**Kevin**)

+ gantt réel(**Yanis**,**Miloud**)+ diagramme de classe(**Yanis**)+ guide de l'application (**Tous** 2h)

Manuel utilisateur :

Kevin
Yanis
Miloud
Hawed

Guide d'Utilisation de l'Application KYMH

1. Connexion

- Entrez votre adresse mail et votre mot de passe.
- Cliquez sur **Se connecter** pour accéder à l'application.

The screenshot shows a web application interface with a top navigation bar containing the following tabs: Connexion, Utilisateur1, Utilisateur2, Technicien1, Technicien2, Admin, tabPage7, tabPage8, Technicien3, and Utilisateur3. The 'Connexion' tab is active. The main content area is a light gray rectangle. In the center, there is a login form with two input fields. The first field is labeled 'E-mail' and contains the text 'utilisateur@example.com'. The second field is labeled 'Mot De Passe' and contains the text '123'. Below these fields is a button labeled 'Se Connecter'.

2. Page Utilisateur

- Si vous êtes habilité en tant qu'Utilisateur, vous serez redirigé vers une page dédiée à la déclaration d'un problème.

The screenshot shows the same web application interface as the previous one, but the 'Utilisateur1' tab is now active in the top navigation bar. The main content area is a light gray rectangle. In the center, there are two buttons: 'Suivre' and 'Déclarer'.

3. Déclaration d'un Problème

- Sur cette page :
 1. **Décrivez votre problème** dans le champ prévu à cet effet.
 2. **Indiquez la date** où le problème est survenu.
 3. **Sélectionnez l'urgence** du problème : Faible, Moyenne ou Grande.
 4. Entrez votre **ID utilisateur** et l'**ID de votre matériel**.
- Une fois rempli, cliquez sur **Soumettre** pour transmettre votre problème au support technique.

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin tabPage7 tabPage8 Technicien3 Utilisateur3

description

date_demande

urgence Faible ▾

id materiel 1 ▾

id utilisateur 1 ▾

Soumettre

4. Consultation par le Technicien

- Les techniciens peuvent consulter les problèmes des utilisateurs ainsi que les problèmes liés aux matériels via une interface dédiée.

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin tabPage7 tabPage8 Technicien3 Utilisateur3

Consulter materiel

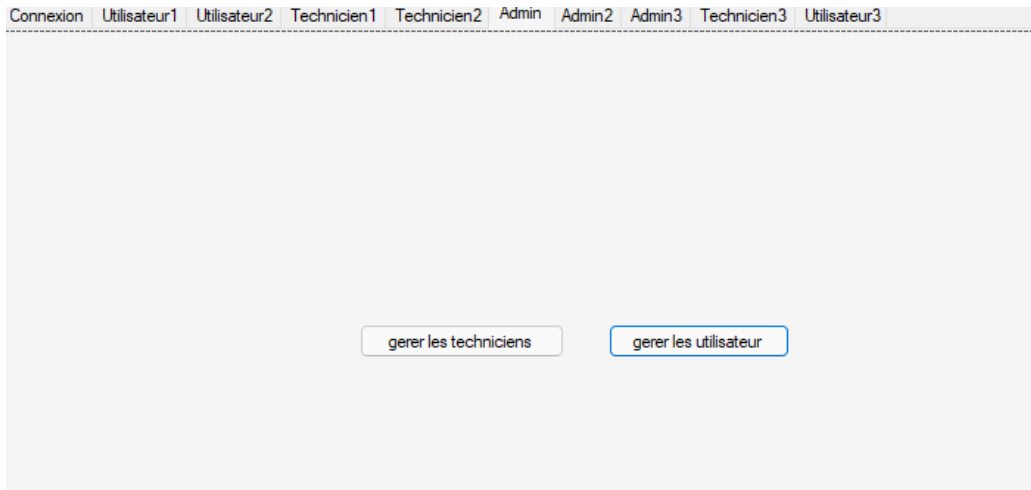
5. Gestion des Matériels (Technicien)

- Si vous êtes **Technicien**, cliquez sur le bouton **Matériel** depuis la page d'accueil.
- Sur la page suivante, renseignez les informations suivantes :
 - **ID utilisateur**
 - **Nom du matériel**
 - **Type de matériel**
 - **Caractéristiques du matériel**
 - **Date d'achat**
 - **Garantie**
- Utilisez les boutons suivants :
 - **Soumettre** : pour enregistrer les données.
 - **Actualiser** : pour mettre à jour les informations.
 - **Supprimer** : Cliquez sur la case à gauche de la première colonne pour sélectionner le matériel que vous voulez supprimer puis sur le bouton
 -

The screenshot shows a web application interface for a Technician user. At the top, there is a navigation bar with tabs: 'Connexion', 'Utilisateur1', 'Utilisateur2', 'Technicien1' (selected), 'Technicien2', 'Admin', 'tabPage7', 'tabPage8', 'Technicien3', and 'Utilisateur3'. Below the navigation bar, the main content area is divided into two sections. On the left, there is a form with the following fields: 'id utilisateur' (a dropdown menu showing '1'), 'nom mat' (a text input), 'type' (a text input), 'caractéristique' (a text input), 'date achat' (a text input), and 'garantie' (a text input). Below these fields is an 'ajouter' button. On the right, there is a large gray rectangular area representing a table. Below this area are two buttons: 'actualiser' and 'supprimer'.

6. Interface Administrateur : Gestion des Techniciens et Utilisateurs

- En tant qu'**Administrateur**, vous êtes redirigé vers une page avec deux boutons :
 1. **Gérer les Techniciens**
 2. **Gérer les Utilisateurs**



6.1. Gestion des Techniciens

Une fois redirigé sur cette page, vous avez la possibilité d'ajouter ou de gérer les techniciens.

Champs à compléter :

- **ID** : Saisissez l'identifiant unique du technicien.
- **Nom** : Entrez le nom de famille du technicien.
- **Prénom** : Indiquez le prénom du technicien.
- **Formation** : Décrivez la formation suivie par le technicien.
- **Compétence** : Mentionnez les compétences principales du technicien.
- **Niveau d'intervention** : Spécifiez son niveau d'intervention (débutant, intermédiaire, expert).

Actions disponibles :

- **Créer** : Cliquez pour enregistrer un nouveau technicien avec les informations saisies.
- **Actualiser** : Met à jour les données affichées à l'écran.
- **Supprimer** : Cliquez sur la case à gauche de la première colonne pour sélectionner le technicien que vous voulez supprimer puis sur le bouton
-
- **Retour** : Revient à l'interface précédente.

Zone d'affichage :

- Un tableau est dédié à l'affichage des informations ou des techniciens existants.

Connexion Utilisateur1 Utilisateur2 Technicien1 Technicien2 Admin Admin2 Admin3 Technicien3 Utilisateur3

ID

Nom

Prenom

formation

compétence

niveau d'intervention

Créer

Actualiser Supprimer

Retour

6.3. Gestion des Utilisateurs

Une fois redirigé sur cette page, vous avez la possibilité d'ajouter ou de gérer les utilisateurs.

Champs à compléter :

- **ID** : Saisissez l'identifiant unique de l'utilisateur.
- **Nom** : Entrez le nom de famille de l'utilisateur.
- **Prénom** : Indiquez le prénom de l'utilisateur.
- **Email** : Fournissez l'adresse email de l'utilisateur.
- **Mot de passe** : entrez le mot de passe voulu
- **Type** : Sélectionnez le type d'utilisateur dans la liste déroulante (par exemple : Employé, Manager, etc.).

Actions disponibles :

- **Créer** : Cliquez pour enregistrer un nouvel utilisateur avec les informations saisies.
- **Actualiser** : Met à jour les données affichées à l'écran (par exemple, la liste des utilisateurs).
- **Supprimer** : Cliquez sur la case à gauche de la première colonne pour sélectionner l'utilisateur que vous voulez supprimer puis sur le bouton
- **Retour** : Revient à l'interface précédente.

Zone d'affichage :

- Une zone (à droite) est dédiée à l'affichage des utilisateurs existants

The screenshot displays a web application interface for user management. At the top, a navigation bar contains links: Connexion, Utilisateur1, Utilisateur2, Technicien1, Technicien2, Admin, Admin2, Admin3, Technicien3, and Utilisateur3. The main content area is divided into two sections. On the left, there is a form for creating a new user with the following fields: ID (text input), Nom (text input), Prenom (text input), Email (text input), mot de passe (text input), and Type (dropdown menu). Below these fields are two buttons: 'Créer' and 'Retour'. On the right, there is a large grey rectangular box representing the area for displaying existing users. Below this box are two buttons: 'Actualiser' and 'Supprimer'.

6.4. Gestion des Incidents

Cette page permet de gérer les incidents signalés par les utilisateurs.

Champs et fonctionnalités disponibles :

- **ID Incident :**
 - Sélectionnez l'incident à gérer via une liste déroulante.
- **Statut :**
 - Indiquez ou mettez à jour le statut de l'incident (par exemple : Ouvert, En cours, Résolu).
- **Modifier :**
 - Cliquez sur ce bouton pour enregistrer les modifications apportées à l'incident sélectionné.

Actions disponibles :

- **Actualiser :**
 - Met à jour la liste des incidents affichés dans la grande zone d'affichage en haut de la page.
- **Retour :**
 - Revenez à l'interface précédente.

Zone d'affichage :

- Une large zone (en haut) affiche les détails des incidents enregistrés ou une liste d'incidents à gérer.

The screenshot shows a web application interface for incident management. At the top, there is a navigation bar with the following links: Connexion, Utilisateur1, Utilisateur2, Technicien1, Technicien2, Admin, Admin2, Admin3, Technicien3, and Utilisateur3. Below the navigation bar is a large, empty rectangular area, likely intended for displaying a list of incidents or their details. In the center of the page, there is a form with the following fields: a text input for 'description', a text input for 'date_demande', a dropdown menu for 'urgence', a dropdown menu for 'id materiel', and a dropdown menu for 'id utilisateur'. Below these fields is a button labeled 'Soumettre'.

6.5. Consultation des Incidents par Utilisateur

Cette page permet de consulter les incidents signalés par un utilisateur spécifique.

Champs et fonctionnalités disponibles :

- **ID Utilisateur :**
 - Saisissez l'identifiant de l'utilisateur dont vous souhaitez afficher les incidents.

Actions disponibles :

- **Actualiser :**
 - Met à jour les données affichées dans la grande zone de texte en fonction de l'utilisateur
- **Retour :**
 - Revenez à l'interface précédente.

Zone d'affichage :

- **Un tableau** est dédié à l'affichage des incidents signalés par l'utilisateur.

The screenshot shows a web application interface with a navigation bar at the top containing the following links: Connexion, Utilisateur1, Utilisateur2, Technicien1, Technicien2, Admin, Admin2, Admin3, Technicien3, and Utilisateur3. The main content area is light gray and contains a large, empty rectangular box with a black border, intended for displaying incident data. Below this box is a button labeled 'Actualiser'. At the bottom of the interface, there is a label 'Id utilisateur' on the left and a button labeled 'Retour' on the right.

**L'application KYMH vous permet de
gérer efficacement les incidents,
utilisateurs, techniciens et matériels via
des interfaces adaptées à chaque rôle.
Ce guide vise à simplifier votre prise en
main et à garantir une utilisation
optimale des fonctionnalités proposées.**

Pour toute assistance complémentaire, contactez le support technique.

Script pour Importer la BD:

```
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1:3306
-- Generation Time: Nov 21, 2024 at 08:48 AM
-- Server version: 8.3.0
-- PHP Version: 8.2.18

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `projetc1`
--

--
-- -----
--
-- Table structure for table `incident`
--

DROP TABLE IF EXISTS `incident`;
CREATE TABLE IF NOT EXISTS `incident` (
  `id_incident` varchar(50) NOT NULL,
  `description` varchar(50) DEFAULT NULL,
  `date_demande` varchar(50) DEFAULT NULL,
  `urgence` varchar(50) DEFAULT NULL,
  `statut` varchar(50) DEFAULT NULL,
  `date_resolution` varchar(50) DEFAULT NULL,
  `commentaire` varchar(50) DEFAULT NULL,
  `id_materiel` varchar(50) NOT NULL,
  `id_utilisateur` varchar(50) NOT NULL,
  PRIMARY KEY (`id_incident`),
```

```

    KEY `id_materiel` (`id_materiel`),
    KEY `id_utilisateur` (`id_utilisateur`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `incident`
--

INSERT INTO `incident` (`id_incident`, `description`, `date_demande`,
`urgence`, `statut`, `date_resolution`, `commentaire`, `id_materiel`,
`id_utilisateur`) VALUES
('0', 'a', 'a', 'Moyenne', 'en cours', NULL, NULL, '2', '2');

--
-----

--
-- Table structure for table `interventions`
--

DROP TABLE IF EXISTS `interventions`;
CREATE TABLE IF NOT EXISTS `interventions` (
  `id_intervention` varchar(50) NOT NULL,
  `date_intervention` varchar(50) DEFAULT NULL,
  `duree` varchar(50) DEFAULT NULL,
  `description_travail` varchar(50) DEFAULT NULL,
  `id_incident` varchar(50) NOT NULL,
  `id_technicien` varchar(50) NOT NULL,
  PRIMARY KEY (`id_intervention`),
  KEY `id_incident` (`id_incident`),
  KEY `id_technicien` (`id_technicien`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-----

--
-- Table structure for table `materiel`
--

DROP TABLE IF EXISTS `materiel`;
CREATE TABLE IF NOT EXISTS `materiel` (
  `id_materiel` varchar(50) NOT NULL,
  `nomMat` varchar(50) DEFAULT NULL,
  `type` varchar(50) DEFAULT NULL,

```

```

`carcteristique` varchar(50) DEFAULT NULL,
`date_achat` varchar(50) DEFAULT NULL,
`garantie` varchar(50) DEFAULT NULL,
`id_utilisateur` varchar(50) NOT NULL,
PRIMARY KEY (`id_materiel`),
KEY `id_utilisateur` (`id_utilisateur`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `materiel`
--

INSERT INTO `materiel` (`id_materiel`, `nomMat`, `type`,
`carcteristique`, `date_achat`, `garantie`, `id_utilisateur`) VALUES
('1', 'Ordinateur Portable HP', 'Informatique', 'Processeur i5, 8Go
RAM, SSD 256Go', '2023-05-10', '2', '1'),
('2', 'Imprimante Epson', 'Périphérique', 'Jet d\'encre couleur, WiFi
intégré', '2022-11-20', '3', '2'),
('3', 'Scanner Canon', 'Périphérique', 'Scanner à plat, Résolution 1200
dpi', '2021-07-15', '1', '3');

-- -----
--
-- Table structure for table `technicien`
--

DROP TABLE IF EXISTS `technicien`;
CREATE TABLE IF NOT EXISTS `technicien` (
  `id_technicien` varchar(50) NOT NULL,
  `nom` varchar(50) DEFAULT NULL,
  `prenom` varchar(50) DEFAULT NULL,
  `niveau_d_intervention` varchar(50) DEFAULT NULL,
  `mot_de_passe` varchar(50) DEFAULT NULL,
  `formation` varchar(50) DEFAULT NULL,
  `competences_technicien` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`id_technicien`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

--
-- Dumping data for table `technicien`
--

```

```

INSERT INTO `technicien` (`id_technicien`, `nom`, `prenom`,
`niveau_d_intervention`, `mot_de_passe`, `formation`,
`competences_technicien`) VALUES
('0', 'a', 'a', '4', NULL, 'a', 'a'),
('1', NULL, NULL, 'Niveau 1', 'tech1234', 'Diplôme en Informatique',
'Reseaux, Dépannage matériel'),
('2', NULL, NULL, 'Niveau 2', 'securePass2024', 'Formation avancée en
systèmes', 'Virtualisation, Sécurité réseau');

```

```

-- -----

```

```

--
-- Table structure for table `utilisateur`
--

```

```

DROP TABLE IF EXISTS `utilisateur`;
CREATE TABLE IF NOT EXISTS `utilisateur` (
  `id_utilisateur` varchar(50) NOT NULL,
  `nom` varchar(50) DEFAULT NULL,
  `prenom` varchar(50) DEFAULT NULL,
  `email` varchar(50) DEFAULT NULL,
  `mot_de_passe` varchar(50) DEFAULT NULL,
  `type_utilisateur` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`id_utilisateur`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

```

--
-- Dumping data for table `utilisateur`
--

```

```

INSERT INTO `utilisateur` (`id_utilisateur`, `nom`, `prenom`, `email`,
`mot_de_passe`, `type_utilisateur`) VALUES
('1', 'Dupont', 'Marie', 'admin', '123', 'A'),
('2', 'Durand', 'Pierre', 'utilisateur', '123', 'U'),
('3', 'Martin', 'Clara', 'technicien', '123', 'T');

```

```

--
-- Constraints for dumped tables
--

```

```

--
-- Constraints for table `incident`
--

```

```

ALTER TABLE `incident`
  ADD CONSTRAINT `incident_ibfk_1` FOREIGN KEY (`id_materiel`)
REFERENCES `materiel` (`id_materiel`) ON DELETE RESTRICT ON UPDATE
RESTRICT,
  ADD CONSTRAINT `incident_ibfk_2` FOREIGN KEY (`id_utilisateur`)
REFERENCES `utilisateur` (`id_utilisateur`) ON DELETE RESTRICT ON
UPDATE RESTRICT;

--
-- Constraints for table `interventions`
--
ALTER TABLE `interventions`
  ADD CONSTRAINT `interventions_ibfk_1` FOREIGN KEY (`id_incident`)
REFERENCES `incident` (`id_incident`) ON DELETE RESTRICT ON UPDATE
RESTRICT,
  ADD CONSTRAINT `interventions_ibfk_2` FOREIGN KEY (`id_technicien`)
REFERENCES `technicien` (`id_technicien`) ON DELETE RESTRICT ON UPDATE
RESTRICT;

--
-- Constraints for table `materiel`
--
ALTER TABLE `materiel`
  ADD CONSTRAINT `materiel_ibfk_1` FOREIGN KEY (`id_utilisateur`)
REFERENCES `utilisateur` (`id_utilisateur`) ON DELETE RESTRICT ON
UPDATE RESTRICT;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Diagramme :

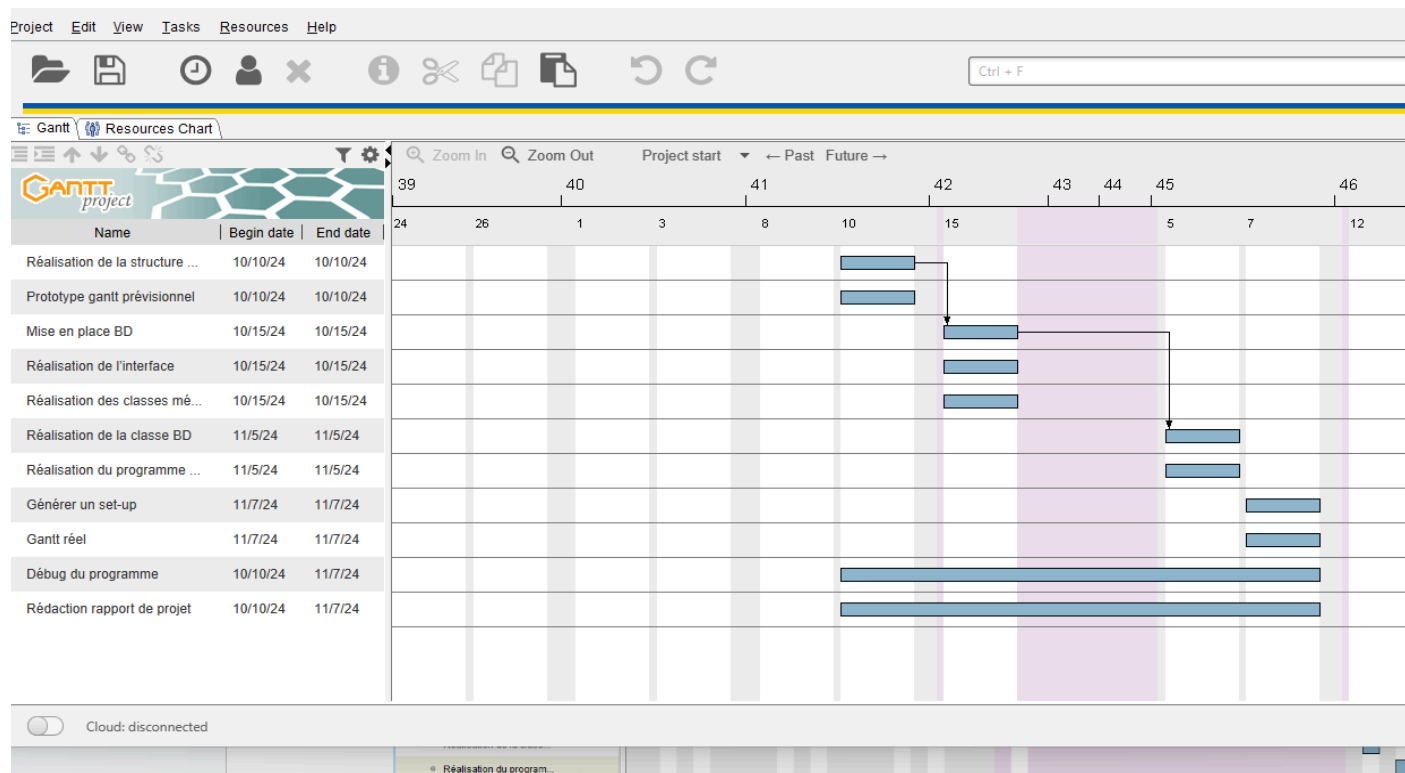
CLASSE+PROPRIETE+METHODES

	UTILISATEUR	MATERIEL	TECHNICIEN	INTERVENTION	INCIDENT
PROPRIETE	<ul style="list-style-type: none"> • Id_utilisateurs : Identifiant unique (int) • nom : Nom de l'utilisateur (string) • prenom : Prénom de l'utilisateur (string) • email : Adresse email de l'utilisateur (string) • mot_de_passe : Mot de passe de l'utilisateur (string) • type_utilisateur : Type d'utilisateur (string) 	<ul style="list-style-type: none"> • Id_Materiel : Identifiant du matériel (int) • NomMat : Nom du matériel (string) • Type : Type de matériel (string) • Carcteristique : Caractéristiques techniques (string) • Date_achat : Date d'achat du matériel (string) • Garantie : Détails sur la garantie (string) 	<ul style="list-style-type: none"> • id_techicien : Identifiant du technicien (int) • niveau_Dintervention : Niveau d'intervention (string) • mot_de_passe : Mot de passe (string) • formation : Formation suivie (string) • competences_tec hnicien : • Compétences techniques (string) 	<ul style="list-style-type: none"> • Id_intervention : Identifiant de l'intervention (int) • date_intervention : Date de l'intervention (string) • duree : Durée de l'intervention (int) • description_travail : Description des travaux (string) 	<ul style="list-style-type: none"> • id_incident : Identifiant de l'incident (int) • description : Description de l'incident (string) • dateDemande : Date de la demande (string) • statut : Statut actuel de l'incident (string) • dateResolution : Date de résolution (string) • commentaire : Commentaire sur l'incident (string) • poste : Poste concerné (string)
METHODES	<ul style="list-style-type: none"> • getId() : Retourne l'identifiant • getUnnom() : Retourne le nom • getUnprenom() : Retourne le prénom • getUnemail() : Retourne l'email • getUnmot_de_passe() : Retourne le mot de passe • getUntype_utilisateur() : Retourne le type d'utilisateur 	<ul style="list-style-type: none"> • Getters et setters pour chaque propriété via : • Id_Materiel1 • NomMat • Type • Carcteristique • Date_achat • Garantie 	<ul style="list-style-type: none"> • getId() : Retourne l'identifiant • getNiveau_Dintervention() : Retourne le niveau d'intervention • getMot_de_passe() : Retourne le mot de passe • getFormation() : Retourne la formation • getCompetences_technicien() : Retourne les compétences 	<ul style="list-style-type: none"> • getId() : Retourne l'identifiant • getDateIntervention() : Retourne la date de l'intervention • getDuree() : Retourne la durée • getDescription() : Retourne la description des travaux 	<ul style="list-style-type: none"> • GETTERS ET SETTERS POUR CHAQUE PROPRIÉTÉ VIA : • ID_INCIDENT • DESCRIPTION • DATEDEMANDE • STATUT • DATERESOLUTION • COMMENTAIRE • POSTE

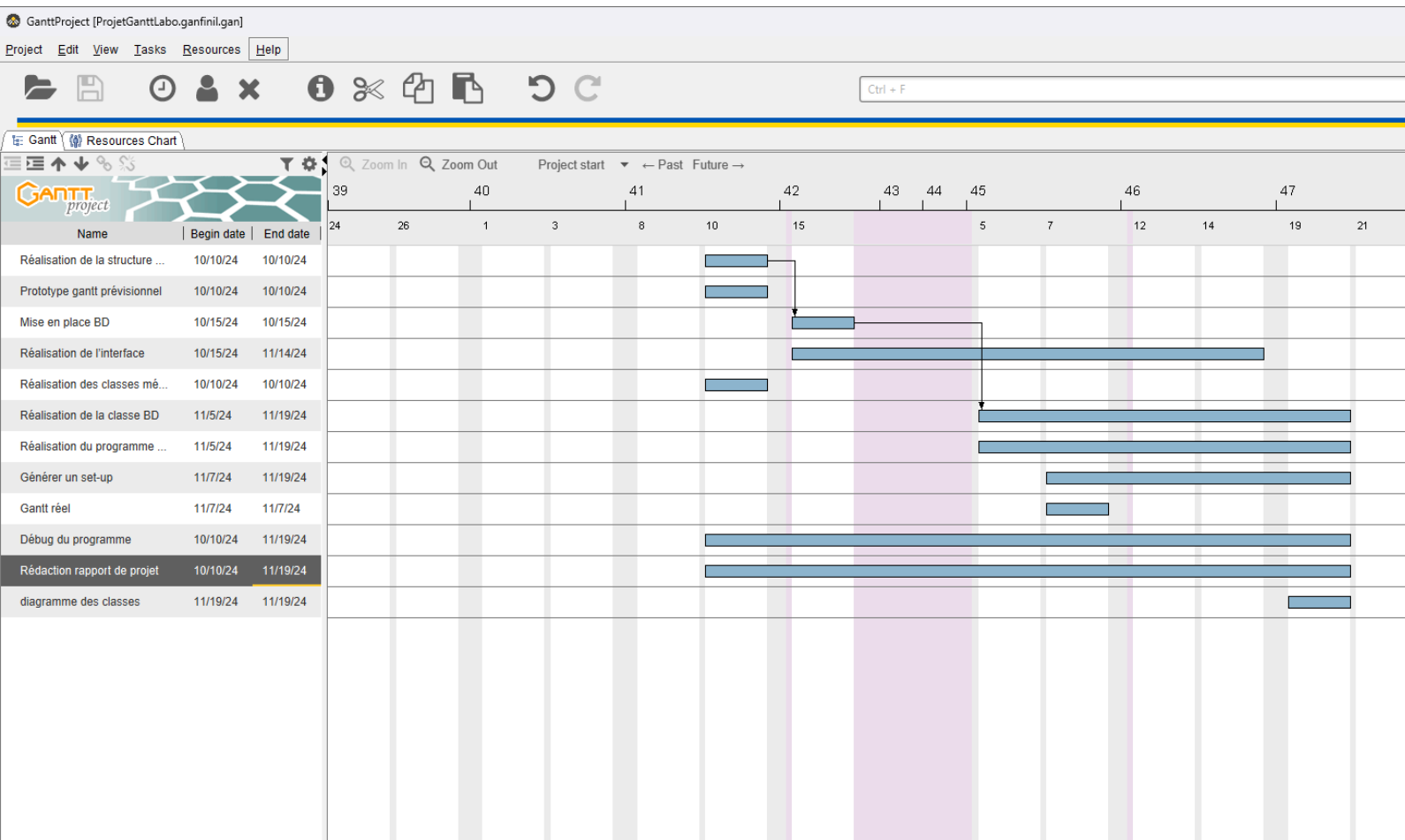
Gantt :

Gantt prévisionnel - Gantt effectif avec affectation des ressources :



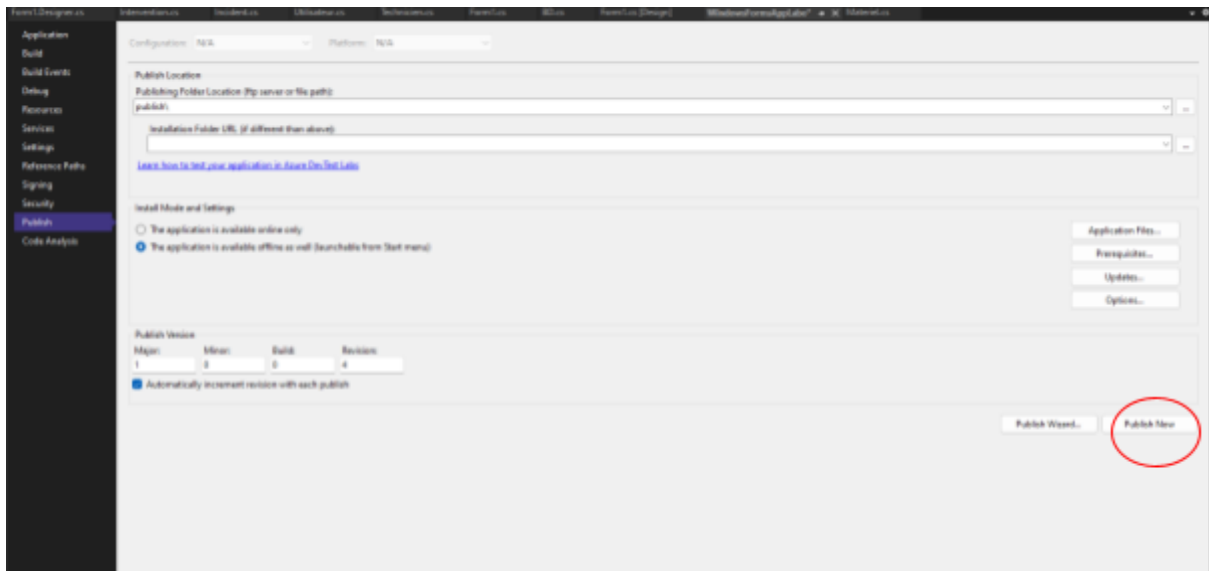


Gantt réel :

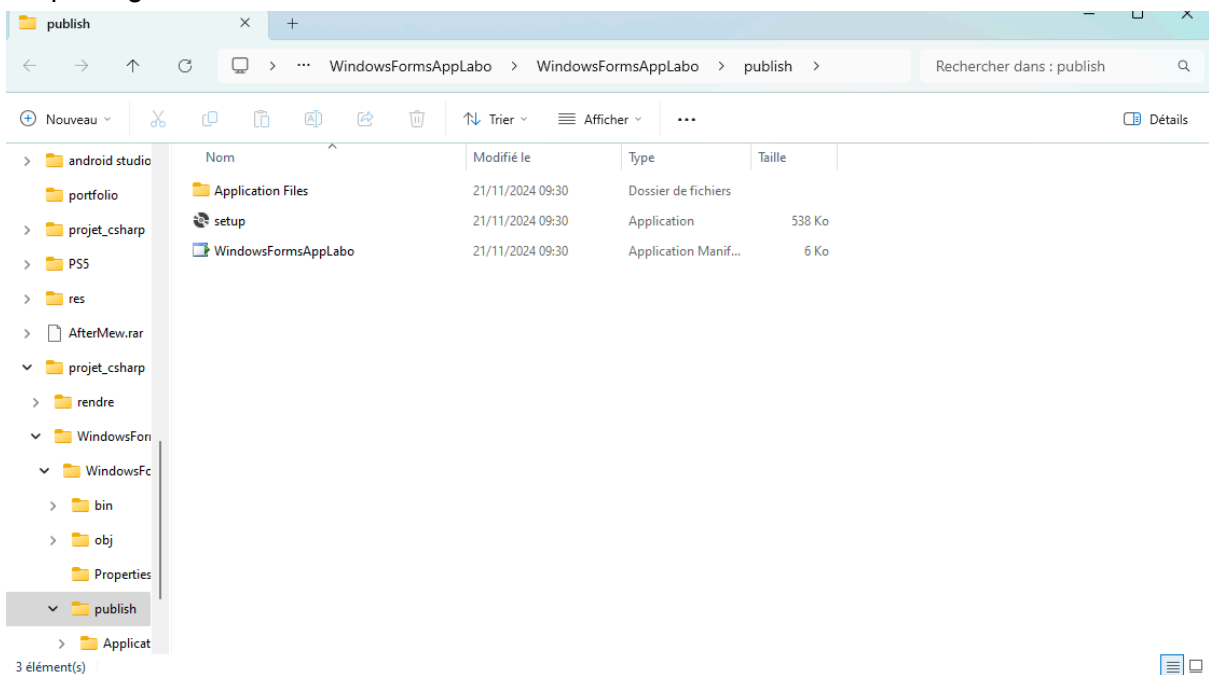


Générer un set-up:

pour permettre le déploiement de l'application il va falloir que les utilisateur puissent l'utiliser sans visual studio pour cela on va publier le projet en cliquant sur le bouton "publish now"



ce qui va générer ces trois fichier :



il faudrait juste lancer le fichier "setup" pour que cela marche

screen de la classe BD

```
1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Linq;
5  using System.Security.Cryptography;
6  using System.Text;
7  using System.Threading.Tasks;
8  using MySql.Data.MySqlClient;
9  using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;
10
11 namespace WindowsFormsAppLabo
12 {
13     18 references
14     public class BD
15     {
16         //
17         static string connectionString = @"Server=localhost; Database=projetcl; Uid=root; Password=";
18
19         /// <summary>
20         /// Ajoute un technicien dans la base de données.
21         /// </summary>
22         /// <param name="unTechnicien">Instance de la classe Technicien.</param>
23         1 reference
24         public static void ajouteTechnicien(Technicien unTechnicien)
25         {
26             MySqlConnection conn = new MySqlConnection(connectionString);
27             conn.Open();
28             MySqlCommand cmd = conn.CreateCommand();
29             cmd.CommandText = "INSERT INTO technicien (id_technicien, nom, prenom, niveau_d_intervention, formation, competences_technicien) " +
30                 "VALUES (@id,@nom, @prenom, @niveau_d_intervention, @formation, @competences_technicien)";
31             cmd.Parameters.AddWithValue("@id", unTechnicien.getId());
32             cmd.Parameters.AddWithValue("@nom", unTechnicien.getNom());
33             cmd.Parameters.AddWithValue("@prenom", unTechnicien.getPrenom());
34             cmd.Parameters.AddWithValue("@niveau_d_intervention", unTechnicien.getMiveau_Dintervention());
35             cmd.Parameters.AddWithValue("@formation", unTechnicien.getFormation());
36             cmd.Parameters.AddWithValue("@competences_technicien", unTechnicien.getCompetences_technicien());
37             cmd.ExecuteNonQuery();
38             conn.Close();
39         }
40         /// <summary>
41         /// Ajoute un utilisateur dans la base de données.
42         /// </summary>
43         /// <param name="unUtilisateur">Instance de la classe Technicien.</param>
44         1 reference
45         public static void ajouteUtil(Utilisateur unUtilisateur)
46         {
47             MySqlConnection conn = new MySqlConnection(connectionString);
48             conn.Open();
49             MySqlCommand cmd = conn.CreateCommand();
50             cmd.CommandText = "INSERT INTO utilisateur (id_utilisateur, nom, prenom, email, mot_de_passe, type) " +
51                 "VALUES (@id,@nom, @prenom, @email, @mot_de_passe, @type)";
```

```

50
51 cmd.Parameters.AddWithValue("@id", unUtilisateur.getId());
52 cmd.Parameters.AddWithValue("@nom", unUtilisateur.getUnnom());
53 cmd.Parameters.AddWithValue("@prenom", unUtilisateur.getUnprenom());
54 cmd.Parameters.AddWithValue("@email", unUtilisateur.getUnemail());
55 cmd.Parameters.AddWithValue("@mot_de_passe", unUtilisateur.getUnmot_de_passe());
56 cmd.Parameters.AddWithValue("@type", unUtilisateur.GetType());
57
58 cmd.ExecuteNonQuery();
59 conn.Close();
60 }
61
62 /// <summary>
63 /// Ajoute un incident dans la base de données.
64 /// </summary>
65 /// <param name="unIncident">Instance de la classe Incident.</param>
66 1 reference
67 public static void ajouteIncident(Incident unIncident)
68 {
69     MySqlConnection conn = new MySqlConnection(connectionString);
70     conn.Open();
71     MySqlCommand cmd = conn.CreateCommand();
72     cmd.CommandText = "INSERT INTO INCIDENT (id_incident, description, date_demande, urgence, statut, id_materiel, id_utilisateur) " +
73         "VALUES (@id_incident, @description, @date_demande, @urgence, @statut, @id_materiel, @id_utilisateur)";
74     cmd.Parameters.AddWithValue("@id_incident", unIncident.Id_incident);
75     cmd.Parameters.AddWithValue("@description", unIncident.Description);
76     cmd.Parameters.AddWithValue("@date_demande", unIncident.DateDemande);
77     cmd.Parameters.AddWithValue("@urgence", unIncident.Urgence);
78     cmd.Parameters.AddWithValue("@statut", unIncident.Statut);
79     cmd.Parameters.AddWithValue("@id_materiel", unIncident.Id_materiel);
80     cmd.Parameters.AddWithValue("@id_utilisateur", unIncident.Id_utilisateur);
81     cmd.ExecuteNonQuery();
82     conn.Close();
83 }
84
85 /// <summary>
86 /// Ajoute un matériel dans la base de données.
87 /// </summary>
88 /// <param name="unMatériel">Instance de la classe Matériel.</param>
89 1 reference
90 public static void ajouteMat(Matériel unMatériel)
91 {
92     MySqlConnection conn = new MySqlConnection(connectionString);
93     conn.Open();
94     MySqlCommand cmd = conn.CreateCommand();
95     cmd.CommandText = "INSERT INTO materiel (id_materiel, nomMat, type, caractéristique, date_achat, garantie, id_utilisateur) " +
96         "VALUES (@id_materiel, @nomMat, @type, @caractéristique, @date_achat, @garantie, @id_utilisateur)";
97     cmd.Parameters.AddWithValue("@id_materiel", unMatériel.Id_Materiel);
98     cmd.Parameters.AddWithValue("@nomMat", unMatériel.NomMat);
99     cmd.Parameters.AddWithValue("@type", unMatériel.Type);
100     cmd.Parameters.AddWithValue("@caractéristique", unMatériel.Caractéristique);
101     cmd.Parameters.AddWithValue("@date_achat", unMatériel.Date_achat);
102     cmd.Parameters.AddWithValue("@garantie", unMatériel.Garantie);

```

```

99         cmd.Parameters.AddWithValue("@date_achat", unMateriel.Date_achat);
100         cmd.Parameters.AddWithValue("@garantie", unMateriel.Garantie);
101         cmd.Parameters.AddWithValue("@id_utilisateur", unMateriel.Id_utilisateur);
102         cmd.ExecuteNonQuery();
103         conn.Close();
104     }
105
106     /// <summary>
107     /// Modifie les compétences d'un technicien dans la base de données.
108     /// </summary>
109     /// <param name="idTechnicien">ID du technicien à modifier.</param>
110     0 references
111     public static void upTech(int idTechnicien)
112     {
113         MySqlConnection conn = new MySqlConnection(connectionString);
114         conn.Open();
115         MySqlCommand cmd = conn.CreateCommand();
116         cmd.CommandText = "UPDATE technicien SET competences_technicien WHERE id_technicien=@id";
117         cmd.Parameters.AddWithValue("@id", idTechnicien);
118         cmd.ExecuteNonQuery();
119         conn.Close();
120     }
121
122     /// <summary>
123     /// Supprime un matériel de la base de données.
124     /// </summary>
125     /// <param name="idMateriel">ID du matériel à supprimer.</param>
126     1 reference
127     public static void delMat(int idMateriel)
128     {
129         MySqlConnection conn = new MySqlConnection(connectionString);
130         conn.Open();
131         MySqlCommand cmd = conn.CreateCommand();
132         cmd.CommandText = "DELETE FROM materiel WHERE id_materiel=@id";
133         cmd.Parameters.AddWithValue("@id", idMateriel);
134         cmd.ExecuteNonQuery();
135         conn.Close();
136     }
137
138     /// <summary>
139     /// Modifie un matériel dans la base de données.
140     /// </summary>
141     /// <param name="idMateriel">ID du matériel à modifier.</param>
142     0 references
143     public static void upMat(int idMateriel)
144     {
145         MySqlConnection conn = new MySqlConnection(connectionString);
146         conn.Open();
147         MySqlCommand cmd = conn.CreateCommand();
148         cmd.CommandText = "UPDATE materiel SET carcteristique WHERE id_materiel=@id";
149         cmd.Parameters.AddWithValue("@id", idMateriel);
150         cmd.ExecuteNonQuery();
151         conn.Close();

```

```

148         conn.Close();
149     }
150
151     /// <summary>
152     /// Récupère les IDs des matériels.
153     /// </summary>
154     /// <returns>Liste des IDs des matériels.</returns>
155     1 reference
156     public static List<string> getIdmat()
157     {
158         List<string> listId = new List<string>();
159         MySqlConnection conn = new MySqlConnection(connectionString);
160         conn.Open();
161         MySqlCommand cmd = conn.CreateCommand();
162         cmd.CommandText = "SELECT id_materiel FROM materiel;";
163         MySqlDataReader reader = cmd.ExecuteReader();
164         while (reader.Read())
165         {
166             string id = reader["id_materiel"].ToString();
167             listId.Add(id);
168         }
169         conn.Close();
170         return listId;
171     }
172
173     /// <summary>
174     /// Récupère tous les matériels.
175     /// </summary>
176     /// <returns>Un objet MySqlCommand permettant de récupérer les matériels.</returns>
177     1 reference
178     public static MySqlCommand getLesmat()
179     {
180         MySqlConnection conn = new MySqlConnection(connectionString);
181         MySqlCommand cmd = conn.CreateCommand();
182         cmd.CommandText = "SELECT id_materiel, nomMat, type, carcteristique FROM materiel;";
183         conn.Close();
184         return cmd;
185     }
186
187     /// <summary>
188     /// Récupère les incidents pour un utilisateur donné.
189     /// </summary>
190     /// <param name="id">ID de l'utilisateur.</param>
191     /// <returns>Un objet MySqlCommand permettant de récupérer les incidents de l'utilisateur.</returns>
192     1 reference
193     public static MySqlCommand getLesIncident(int id)
194     {
195         MySqlConnection conn = new MySqlConnection(connectionString);
196         MySqlCommand cmd = conn.CreateCommand();
197         cmd.CommandText = "SELECT description, date_demande, statut FROM incident WHERE id_utilisateur = " + id;
198         conn.Close();
199         return cmd;
200     }

```

```

197     }
198
199     /// <summary>
200     /// Récupère tous les incidents.
201     /// </summary>
202     /// <returns>Un objet MySqlCommand permettant de récupérer tous les incidents.</returns>
203     1 reference
204     public static MySqlCommand getLesIncidents()
205     {
206         MySqlConnection conn = new MySqlConnection(connectionString);
207         MySqlCommand cmd = conn.CreateCommand();
208         cmd.CommandText = "SELECT * FROM incident";
209         conn.Close();
210         return cmd;
211     }
212
213     /// <summary>
214     /// Récupère tous les techniciens.
215     /// </summary>
216     /// <returns>Un objet MySqlCommand permettant de récupérer tous les techniciens.</returns>
217     1 reference
218     public static MySqlCommand getLesTech()
219     {
220         MySqlConnection conn = new MySqlConnection(connectionString);
221         MySqlCommand cmd = conn.CreateCommand();
222         cmd.CommandText = "SELECT * FROM technicien";
223         conn.Close();
224         return cmd;
225     }
226
227     /// <summary>
228     /// Récupère tous les utilisateurs.
229     /// </summary>
230     /// <returns>Un objet MySqlCommand permettant de récupérer tous les utilisateurs.</returns>
231     1 reference
232     public static MySqlCommand getLesUtil()
233     {
234         MySqlConnection conn = new MySqlConnection(connectionString);
235         MySqlCommand cmd = conn.CreateCommand();
236         cmd.CommandText = "SELECT * FROM utilisateur";
237         conn.Close();
238         return cmd;
239     }
240
241     /// <summary>
242     /// Vérifie le type d'utilisateur en fonction de l'email et du mot de passe.
243     /// </summary>
244     /// <param name="email">Email de l'utilisateur.</param>
245     /// <param name="mdp">Mot de passe de l'utilisateur.</param>
246     /// <returns>Le type d'utilisateur (ex. : administrateur, technicien).</returns>
247     1 reference
248     public static string verifUtilisateur(string email, string mdp)
249     {

```

```

245
246     MySqlConnection conn = new MySqlConnection(connectionString);
247     conn.Open();
248     MySqlCommand cmd = conn.CreateCommand();
249     cmd.CommandText = "SELECT type_utilisateur FROM utilisateur WHERE email = @email AND mot_de_passe = @mdp";
250     cmd.Parameters.AddWithValue("@email", email);
251     cmd.Parameters.AddWithValue("@mdp", mdp);
252     MySqlDataReader reader = cmd.ExecuteReader();
253     reader.Read();
254     string type = reader["type_utilisateur"].ToString();
255     conn.Close();
256     return type;
257 }
258
259 /// <summary>
260 /// Récupère l'ID de l'utilisateur en fonction de l'email et du mot de passe.
261 /// </summary>
262 /// <param name="email">Email de l'utilisateur.</param>
263 /// <param name="mdp">Mot de passe de l'utilisateur.</param>
264 /// <returns>ID de l'utilisateur.</returns>
265 1 reference
266 public static int verifIdUtil(string email, string mdp)
267 {
268     MySqlConnection conn = new MySqlConnection(connectionString);
269     conn.Open();
270     MySqlCommand cmd = conn.CreateCommand();
271     cmd.CommandText = "SELECT id_utilisateur FROM utilisateur WHERE email = @email AND mot_de_passe = @mdp";
272     cmd.Parameters.AddWithValue("@email", email);
273     cmd.Parameters.AddWithValue("@mdp", mdp);
274     MySqlDataReader reader = cmd.ExecuteReader();
275     reader.Read();
276     int id = Convert.ToInt16(reader["id_utilisateur"]);
277     conn.Close();
278     return id;
279 }
280
281 /// <summary>
282 /// Supprime un utilisateur par son ID.
283 /// </summary>
284 /// <param name="idUtilisateur">ID de l'utilisateur à supprimer.</param>
285 1 reference
286 public static void delUtil(int idUtilisateur)
287 {
288     MySqlConnection conn = new MySqlConnection(connectionString);
289     conn.Open();
290     MySqlCommand cmd = conn.CreateCommand();
291     cmd.CommandText = "DELETE FROM utilisateur WHERE id_utilisateur = @id";
292     cmd.Parameters.AddWithValue("@id", idUtilisateur);
293     cmd.ExecuteNonQuery();
294     conn.Close();
295 }

```



```

294
295     /// <summary>
296     /// Supprime un technicien par son ID.
297     /// </summary>
298     /// <param name="id">ID du technicien à supprimer.</param>
299     1 reference
300     public static void delTech(int id)
301     {
302         MySqlConnection conn = new MySqlConnection(connectionString);
303         conn.Open();
304         MySqlCommand cmd = conn.CreateCommand();
305         cmd.CommandText = "DELETE FROM technicien WHERE id_technicien = @id";
306         cmd.Parameters.AddWithValue("@id", id);
307         cmd.ExecuteNonQuery();
308         conn.Close();
309     }
310
311     /// <summary>
312     /// Récupère les IDs de tous les utilisateurs.
313     /// </summary>
314     /// <returns>Liste des IDs des utilisateurs.</returns>
315
316     public static List<string> getIdutil()
317     {
318         List<string> listId = new List<string>();
319         MySqlConnection conn = new MySqlConnection(connectionString);
320         conn.Open();
321         MySqlCommand cmd = conn.CreateCommand();
322         cmd.CommandText = "SELECT id_utilisateur FROM utilisateur;";
323         MySqlDataReader reader = cmd.ExecuteReader();
324         while (reader.Read())
325         {
326             listId.Add(reader["id_utilisateur"].ToString());
327         }
328         conn.Close();
329         return listId;
330     }
331
332     /// <summary>
333     /// Récupère les IDs de tous les matériels.
334     /// </summary>
335     /// <returns>Liste des IDs des matériels.</returns>
336
337     public static List<string> getIdMat()
338     {
339         List<string> listId = new List<string>();
340         MySqlConnection conn = new MySqlConnection(connectionString);
341         conn.Open();
342         MySqlCommand cmd = conn.CreateCommand();
343         cmd.CommandText = "SELECT id_materiel FROM materiel;";
344         MySqlDataReader reader = cmd.ExecuteReader();
345         while (reader.Read())
346         {

```

```

343     {
344         listId.Add(reader["id_materiel"].ToString());
345     }
346     conn.Close();
347     return listId;
348 }
349
350 /// <summary>
351 /// Récupère les IDs de tous les incidents.
352 /// </summary>
353 /// <returns>Liste des IDs des incidents.</returns>
354 0 references
355 public static List<string> getIdIncident()
356 {
357     List<string> listId = new List<string>();
358     MySqlConnection conn = new MySqlConnection(connectionString);
359     conn.Open();
360     MySqlCommand cmd = conn.CreateCommand();
361     cmd.CommandText = "SELECT id_incident FROM incident;";
362     MySqlDataReader reader = cmd.ExecuteReader();
363     while (reader.Read())
364     {
365         listId.Add(reader["id_incident"].ToString());
366     }
367     conn.Close();
368     return listId;
369 }
370
371 /// <summary>
372 /// Modifie le statut d'un incident.
373 /// </summary>
374 /// <param name="idIncid">ID de l'incident à modifier.</param>
375 /// <param name="statut">Nouveau statut de l'incident.</param>
376 public static void upIncid(string idIncid, string statut)
377 {
378     MySqlConnection conn = new MySqlConnection(connectionString);
379     conn.Open();
380     MySqlCommand cmd = conn.CreateCommand();
381     cmd.CommandText = "UPDATE incident SET statut = @statut WHERE id_incident = @id";
382     cmd.Parameters.AddWithValue("@statut", statut);
383     cmd.Parameters.AddWithValue("@id", idIncid);
384     cmd.ExecuteNonQuery();
385     conn.Close();
386 }
387 }

```

(local variable) List<string> listId

Screens de form1.cs:

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using MySql.Data.MySqlClient;
11 using static System.Windows.Forms.VisualStyles.VisualStyleElement;
12
13 namespace WindowsFormsAppLabo
14 {
15     3 references
16     public partial class Form1 : Form
17     {
18         1 reference
19         public Form1()
20         {
21             InitializeComponent();
22         }
23
24         int idconn;
25
26         /// <summary>
27         /// Au lancement met dans remplit les combo box
28         /// </summary>
29         /// <param name="sender"></param>
30         /// <param name="e"></param>
31         1 reference
32         private void Form1_Load_1(object sender, EventArgs e)
33         {
34
35             comboBoxIdUtil.Items.Clear();
36             foreach (string i in BD.getIdutil())
37                 comboBoxIdUtil.Items.Add(i);
38
39             comboBoxIdUtil2.Items.Clear();
40             foreach (string i in BD.getIdutil())
41                 comboBoxIdUtil2.Items.Add(i);
42
43             comboBoxIdMat.Items.Clear();
44             foreach (string i in BD.getIdmat())
45                 comboBoxIdMat.Items.Add(i);
46
47
48             comboBoxUrgence.Items.Clear();
49             comboBoxUrgence.Items.Add("Faible");
50             comboBoxUrgence.Items.Add("Moyenne");
51             comboBoxUrgence.Items.Add("Forte");
52

```

```

50     comboBoxUrgence.Items.Add("Moyenne");
51     comboBoxUrgence.Items.Add("Forte");
52
53     comboBoxTypeUtil.Items.Clear();
54     comboBoxTypeUtil.Items.Add("U");
55     comboBoxTypeUtil.Items.Add("T");
56     comboBoxTypeUtil.Items.Add("A");
57
58     comboBoxStatut.Items.Clear();
59     comboBoxStatut.Items.Add("non traité");
60     comboBoxStatut.Items.Add("en cours");
61     comboBoxStatut.Items.Add("traiter");
62
63 }
64
65 0 references
66 private void Form1_Load(object sender, EventArgs e)
67 {
68 }
69 /// <summary>
70 /// bouton permettant d'etre rediriger vers l'onglet correspondant au niveau d'habilitation
71 /// </summary>
72 /// <param name="sender"></param>
73 /// <param name="e"></param>
74 1 reference
75 private void btConn_Click(object sender, EventArgs e)
76 {
77     string email = tbMail.Text;
78     string mdp = tbMdp.Text;
79     string type = BD.verifUtilisateur(email, mdp);
80
81     if (type == "U")
82     {
83         tabControl1.SelectedTab = tabPage2;
84     }
85     else if (type == "T")
86     {
87         tabControl1.SelectedTab = tabPage4;
88     }
89     else
90     {
91         tabControl1.SelectedTab = tabPage6;
92     }
93     idconn = BD.verifIdUtil(email, mdp);
94 }
95 /// <summary>
96 /// redirection vers la page de suivi
97 /// </summary>
98 /// <param name="sender"></param>
99 /// <param name="e"></param>
100 1 reference
101 private void button1_Click(object sender, EventArgs e)
102 {

```

```

109 // <param name="e"></param>
110 //reference
111 private void button1_Click(object sender, EventArgs e)
112 {
113     tabControl1.SelectedTab = tabPage10;
114 }
115 // <summary>
116 // bouton de redirection vers la page de décaration
117 // </summary>
118 // <param name="sender"></param>
119 // <param name="e"></param>
120 //reference
121 private void btDecl_Click(object sender, EventArgs e)
122 {
123     tabControl1.SelectedTab = tabPage3;
124 }
125 // <summary>
126 // bouton pour ajouter un incident
127 // </summary>
128 // <param name="sender"></param>
129 // <param name="e"></param>
130 //reference
131 private void btSousmettre_Click(object sender, EventArgs e)
132 {
133     Incident unIncident = new Incident(tbDesc.Text, tbDate_Demande.Text, comboBoxUrgence.SelectedItem.ToString(), comboBoxIdMat.SelectedItem.ToString(), comboBoxIdUtil.SelectedItem.ToString() );
134     BD.ajouteIncident(unIncident);
135 }
136 // <summary>
137 // bouton pour ajouter un materiel
138 // </summary>
139 // <param name="sender"></param>
140 // <param name="e"></param>
141 //reference
142 private void button2_Click(object sender, EventArgs e)
143 {
144     Materiel unMateriel = new Materiel(tbIdMat.Text, tbNomMat.Text, tbType.Text, tbCaract.Text, tbDate.Text, tbGarantie.Text, comboBoxIdUtil.SelectedItem.ToString());
145     BD.ajouteMat(unMateriel);
146 }
147 // <summary>
148 // supprimer un materiel selectionné
149 // </summary>
150 // <param name="sender"></param>
151 // <param name="e"></param>
152 //reference
153 private void button5_Click(object sender, EventArgs e)
154 {
155     int id = Convert.ToInt16(dataGridView1.SelectedRows[0].Cells[0].Value);
156     BD.delMat(id);
157 }
158 // <summary>
159 // actualiser les suivis
160 // </summary>
161 // <param name="sender"></param>
162 // <param name="e"></param>

```

```

144  /// <summary>
145  /// actualiser les suivis
146  /// </summary>
147  /// <param name="sender"></param>
148  /// <param name="e"></param>
149  1 reference
150  private void btActualiserSuivi_Click(object sender, EventArgs e)
151  {
152      MySqlDataAdapter MyAdaptater = new MySqlDataAdapter();
153      DataSet ds = new DataSet();
154      MyAdaptater.SelectCommand = BD.getLesIncident(idconn);
155      MyAdaptater.Fill(ds);
156      dataGridView2.DataSource = ds.Tables[0];
157      dataGridView2.Refresh();
158      label21.Text = idconn.ToString();
159  }
160  /// <summary>
161  /// retour a la page d'accueil des utilisateurs
162  /// </summary>
163  /// <param name="sender"></param>
164  /// <param name="e"></param>
165  1 reference
166  private void button7_Click(object sender, EventArgs e)
167  {
168      tabControl1.SelectedTab = tabPage2;
169  }
170  /// <summary>
171  ///
172  /// </summary>
173  /// <param name="sender"></param>
174  /// <param name="e"></param>
175  //retour a la page d'accueil des utilisateurs
176  1 reference
177  private void button15_Click(object sender, EventArgs e)
178  {
179      tabControl1.SelectedTab = tabPage2;
180  }
181  /// <summary>
182  /// retour a la page d'accueil des techniciens
183  /// </summary>
184  /// <param name="sender"></param>
185  /// <param name="e"></param>
186  0 references
187  private void button8_Click_1(object sender, EventArgs e)
188  {
189      tabControl1.SelectedTab = tabPage4;
190  }
191  /// <summary>
192  /// retour a la pagge d'accueil des techniciens
193  /// </summary>
194  /// <param name="sender"></param>
195  /// <param name="e"></param>

```

```

184 0 references
185 private void button8_Click_1(object sender, EventArgs e)
186 {
187     tabControl1.SelectedTab = tabPage4;
188 }
189 /// <summary>
190 /// retour a la pagge d'accueil des techniciens
191 /// </summary>
192 /// <param name="sender"></param>
193 /// <param name="e"></param>
194 1 reference
195 private void buttonRetourTech_Click(object sender, EventArgs e)
196 {
197     tabControl1.SelectedTab = tabPage4;
198 }
199 /// <summary>
200 /// retour a la page d'accueil des technicien
201 /// </summary>
202 /// <param name="sender"></param>
203 /// <param name="e"></param>
204 /// retour a la page d'accueil des techniciens
205 1 reference
206 private void button7_Click_1(object sender, EventArgs e)
207 {
208     tabControl1.SelectedTab = tabPage4;
209 }
210 /// <summary>
211 /// rediriger vers la page de consultation
212 /// </summary>
213 /// <param name="sender"></param>
214 /// <param name="e"></param>
215 1 reference
216 private void button4_Click(object sender, EventArgs e)
217 {
218     tabControl1.SelectedTab=tabPage9;
219 }
220 /// <summary>
221 /// modifie le statut d'un incident
222 /// </summary>
223 /// <param name="sender"></param>
224 /// <param name="e"></param>
225 1 reference
226 private void button10_Click(object sender, EventArgs e)
227 {
228     string id = Convert.ToString(dataGridView3.SelectedRows[0].Cells[0].Value);
229     BD.upIncid(id, comboBoxStatut.SelectedItem.ToString());
230 }
231
232 /// <summary>
233 /// rediriger vers la page pour gerer les utilisateurs
234 /// </summary>
235 /// <param name="sender"></param>
236 /// <param name="e"></param>
237 1 reference
238 private void buttonGererUtil_Click(object sender, EventArgs e)
239 {
240     tabControl1.SelectedTab = tabPage8;
241 }
242
243 /// <summary>
244 /// rediriger vers la page pour gerer les tech
245 /// </summary>
246 /// <param name="sender"></param>
247 /// <param name="e"></param>
248 1 reference
249 private void buttonGererTech_Click(object sender, EventArgs e)
250 {
251     tabControl1.SelectedTab = tabPage7;
252 }
253
254 /// <summary>
255 /// rediriger vers la page pour gerer les matériaux
256 /// </summary>
257 /// <param name="sender"></param>
258 /// <param name="e"></param>
259 1 reference
260 private void button6_Click(object sender, EventArgs e)
261 {
262     tabControl1.SelectedTab = tabPage5;
263 }
264
265 /// <summary>
266 /// bouton pour creer un technicien
267 /// </summary>
268 /// <param name="sender"></param>
269 /// <param name="e"></param>
270 1 reference
271 private void buttonCreerTech_Click(object sender, EventArgs e)
272 {
273     Technicien unTechnicien = new Technicien(tbIdTech.Text, tbNomTech.Text, tbPrenomTech.Text, tbformation.Text, tbcompetence.Text, tbIntervention.Text);
274     BD.ajouteTechnicien(unTechnicien);
275 }
276
277 /// <summary>
278 /// bouton pour creer un utilisateur
279 /// </summary>
280 /// <param name="sender"></param>
281 /// <param name="e"></param>
282 1 reference
283 private void buttonCreerUtil_Click(object sender, EventArgs e)
284 {
285     Utilisateur unUtilisateur = new Utilisateur(tbIdUtil.Text, tbNomUtil.Text, tbPrenomUtil.Text, tbEmail.Text, tbmdp2.Text, comboBoxTypeUtil.SelectedItem.ToString());
286     BD.ajouteUtil(unUtilisateur);
287 }

```



```

278  /// <summary>
279  /// bouton pour afficher les techniciens
280  /// </summary>
281  /// <param name="sender"></param>
282  /// <param name="e"></param>
283  1 reference
284  private void button11_Click(object sender, EventArgs e)
285  {
286      MySqlDataAdapter MyAdaptater = new MySqlDataAdapter();
287      DataSet ds = new DataSet();
288      MyAdaptater.SelectCommand = BD.getLesTech();
289      MyAdaptater.Fill(ds);
290      dataGridView4.DataSource = ds.Tables[0];
291      dataGridView4.Refresh();
292  }
293  /// <summary>
294  /// bouton pour afficher les incidents coter techniciens
295  /// </summary>
296  /// <param name="sender"></param>
297  /// <param name="e"></param>
298  1 reference
299  private void button9_Click(object sender, EventArgs e)
300  {
301      MySqlDataAdapter MyAdaptater = new MySqlDataAdapter();
302      DataSet ds = new DataSet();
303      MyAdaptater.SelectCommand = BD.getLesIncidents();
304      MyAdaptater.Fill(ds);
305      dataGridView3.DataSource = ds.Tables[0];
306      dataGridView3.Refresh();
307  }
308  /// <summary>
309  /// afficher les materiaux
310  /// </summary>
311  /// <param name="sender"></param>
312  /// <param name="e"></param>
313  1 reference
314  private void button3_Click(object sender, EventArgs e)
315  {
316      MySqlDataAdapter MyAdaptater = new MySqlDataAdapter();
317      DataSet ds = new DataSet();
318      MyAdaptater.SelectCommand = BD.getLesmat();
319      MyAdaptater.Fill(ds);
320      dataGridView1.DataSource = ds.Tables[0];
321      dataGridView1.Refresh();
322  }

```

```

323
324 /// <summary>
325 /// supprimer un technicien
326 /// </summary>
327 /// <param name="sender"></param>
328 /// <param name="e"></param>
329 1 reference
330 private void buttonSupprTech_Click(object sender, EventArgs e)
331 {
332     int id = Convert.ToInt16(dataGridView4.SelectedRows[0].Cells[0].Value);
333     BD.delTech(id);
334 }
335
336 /// <summary>
337 /// retour a la page d'accueil des admins
338 /// </summary>
339 /// <param name="sender"></param>
340 /// <param name="e"></param>
341 1 reference
342 private void buttonRetourAdmin1_Click(object sender, EventArgs e)
343 {
344     tabControl1.SelectedTab = tabPage6;
345 }
346
347 /// <summary>
348 /// retour a la page d'accueil des admins
349 /// </summary>
350 /// <param name="sender"></param>
351 /// <param name="e"></param>
352 1 reference
353 private void buttonRetourAdmin2_Click(object sender, EventArgs e)
354 {
355     tabControl1.SelectedTab = tabPage6;
356 }
357
358 /// <summary>
359 /// affiche les utilisateurs
360 /// </summary>
361 /// <param name="sender"></param>
362 /// <param name="e"></param>
363 1 reference
364 private void buttonActualiserUtilisateur_Click(object sender, EventArgs e)
365 {
366     MySqlDataAdapter MyAdaptater = new MySqlDataAdapter();
367     DataSet ds = new DataSet();
368     MyAdaptater.SelectCommand = BD.getLesUtil();
369     MyAdaptater.Fill(ds);
370     dataGridView5.DataSource = ds.Tables[0];
371     dataGridView5.Refresh();
372 }
373
374 /// <summary>
375 /// supprimer un utilisateur
376 /// </summary>
377 /// <param name="sender"></param>

```

```

378
379 /// <summary>
380 /// supprimer un utilisateur
381 /// </summary>
382 /// <param name="sender"></param>
383 /// <param name="e"></param>
384 1 reference
385 private void buttonSupprUtil_Click(object sender, EventArgs e)
386 {
387     int id = Convert.ToInt16(dataGridView1.SelectedRows[0].Cells[0].Value);
388     BD.delUtil(id);
389 }
390
391 }
392
393 }

```