Cette application Windows ainsi que son service servent à automatiser le passage de l'état d'une fiche de frais un autre. L'application Windows a d'abord servi à réaliser les tests et voir les résultats visuellement. Le service lui permet l'automatisation de ce processus au démarrage de l'ordinateur.

Tout d'abord voici le principal du code de l 'application Windows form.
Cette application utilise un timer, une fois ce timer atteint l'état de la fiche de frais change en fonction de la date du jour :

```
//Si la date du jour est entre le 1 et le 10 du mois alors l'état des fiches de frais du mois précédent change de CR à CL
if (dateVerif >= 1 && dateVerif <= 10)
{

    myConnection.openConnection();

    myCom = myConnection.reqExec("update testfichefrais set idEtat ='CL' where idEtat ='CR' and mois = '" + date.moisPrecedent()+"'");
    myCom.ExecuteNonQuery();

    myDate = new GestionDate();
    date1 = myDate.moisPrecedent();

    myCom = myConnection.reqExec("Select * from testfichefrais where mois='" + date1 + "'");
    dt = new DataTable();
    dt.Load(myCom.ExecuteReader());

    dataGridView1.DataSource = dt;

    myConnection.closeConnection();
}

//Si la date du jour est entre le 10 et le 20 du mois alors l'état des fiches de frais du mois précédent change de RB à MP
else if (dateVerif >= 20 && dateVerif <= 31)
{
    myConnection.openConnection();

    myCom = myConnection.reqExec("update testfichefrais set idEtat ='MP' where idEtat ='RB' and mois = '" + date.moisPrecedent() + "'");
    myCom.ExecuteNonQuery();

    myDate = new GestionDate();
    date1 = myDate.moisPrecedent();

    myCom = myConnection.reqExec("Select * from testfichefrais where mois='" + date1 + "'");
    dt = new DataTable();
    dt.Load(myCom.ExecuteReader());

    dataGridView1.DataSource = dt;

    myConnection.closeConnection();
}
```

Le code du service Windows ressemble au premier code, sans l'affichage :

```
//Si la date du jour est entre 1 et 10 on passe l'état des fiches du mois précédent à CL Si elle est à l'état CR

if (dateVerif >= 1 && dateVerif <= 10)
{

    myConnection.openConnection();

    myCom = myConnection.reqExec("update testfichefrais set idEtat ='CL' where idEtat ='CR' and mois = '" + date.moisPrecedent() + "'");
    myCom.ExecuteNonQuery();




    myConnection.closeConnection();
}

//Si la date du jour est entre 10 et 20 on passe l'état des fiches du mois précédent à MP Si elle est à l'état RB
else if (dateVerif >= 20 && dateVerif <= 31)
{
    myConnection.openConnection();

    myCom = myConnection.reqExec("update testfichefrais set idEtat ='MP' where idEtat ='RB' and mois = '" + date.moisPrecedent() + "'");
    myCom.ExecuteNonQuery();
```

Voici la classe gestion date utilisée pour comparer les dates :

```
DateTime ajd = DateTime.Now;

/// <summary>
/// Retourne la date du jour
/// </summary>

2 références
public String dateJour()
{
    String asString = ajd.ToString("dd/MM/yyyy");
    return asString;
}

/// <summary>
/// Retourne le mois précédent par rapport à la date d'aujourd'hui
/// </summary>

8 références
public String moisPrecedent()
{
    ajd = ajd.AddMonths(-1);
    String asString = ajd.ToString("yyyyMM");

    return asString;
}

/// <summary>
/// Retourne le mois courant
/// </summary>

1 référence
public String moisCourant()
{

    String asString = ajd.ToString("yyyyMM");

    return asString;
}

/// <summary>
/// Retourne le mois suivant en fonction de la date d'aujourd'hui
/// </summary>

1 référence
public String moisSuivant()
{
    ajd = ajd.AddMonths(+1);
    String asString = ajd.ToString("yyyyMM");

    return asString;
}
```