

SCHRAEN
Noah
TS1SIO

Mission 3 AP2 :

1)

Résanet

Réervations

Compte ▾

Ewen PRIGENT (100.00 €)

Se déconnecter

05/02/2024

06/02/2024

07/02/2024

08/02/2024

09/02/2024

12/02/2024

13/02/2024



14/02/2024

15/02/2024

16/02/2024

2)

L'url est la suivante :

  localhost:5000/usager/reservations/lister|

3)

La route associé à la fonction listerReservations() est :

```
@app.route( '/usager/reservations/lister' , methods = [ 'GET' ] )  
def listerReservations() :
```

4)

Cet élément correspond à numeroCarte :

```
@app.route( '/usager/reservations/lister' , methods = [ 'GET' ] )  
def listerReservations() :  
    tarifRepas = modeleResanet.getTarifRepas( session[ 'numeroCarte' ] )  
  
    soldeCarte = modeleResanet.getSolde( session[ 'numeroCarte' ] )
```

5)

Pour listerReservations() :

```
@app.route( '/usager/reservations/lister' , methods = [ 'GET' ] )  
def listerReservations() :  
    print('[START] appResanet::listerReservations()')  
  
    print('[STOP] appResanet::listerReservations()')  
    return render_template( 'vueListeReservations.html' , laSession = ses
```

Pour getTarifRepas() :

```
def getTarifRepas( numeroCarte ) :
    print('[START] modeleResanet::getTarifRepas()')
    try :
        curseur = getConnexionBD().cursor()
        requete = '''
            select tarifRepas
            from Fonction
            inner join Personnel
            on Fonction.idFonction = Personnel.idFonction
            inner join Carte
            on Personnel.matricule = Carte.matricule
            where numeroCarte = %s
        '''

        curseur.execute( requete , ( numeroCarte , ) )

        enregistrement = curseur.fetchone()

        tarif = 'inconnu'
        if enregistrement != None :
            tarif = enregistrement[ 0 ]
            #print type(tarif)

        curseur.close()
        print('[STOP] modeleResanet::getTarifRepas()')
        return tarif

    except :
        return None
```

Pour getSolde() :

```
def getSolde( numeroCarte ) :
    try :
        print('[START] modeleResanet::getSolde()')
        curseur = getConnexionBD().cursor()
        requete = '''
            select solde
            from Carte
            where numeroCarte = %s
        '''

        curseur.execute( requete , ( numeroCarte , ) )

        enregistrement = curseur.fetchone()

        solde = 'inconnu'
        if enregistrement != None :
            solde = enregistrement[ 0 ]
            #print type(solde)

        curseur.close()
        print('[STOP] modeleResanet::getSolde()')
        return solde
```

Pour getReservationsCarte() :

```
def getReservationsCarte( numeroCarte , dateDebut , dateFin ) :
    try:
        print('[START] modeleResanet::getReservationsCarte()')
        curseur = getConnexionBD().cursor()
        requete = '''
            select dateResa
            from Reservation
            where numeroCarte = %s
            and dateResa >= %s
            and dateResa <= %s
        '''

        curseur.execute(requete, ( numeroCarte , dateDebut , dateFin ))

        enregistrements = curseur.fetchall()

        dates = []
        for unEnregistrement in enregistrements:

            uneDate = '%04d-%02d-%02d' % ( unEnregistrement[0].year ,
                                           unEnregistrement[0].month ,
                                           unEnregistrement[0].day )

            dates.append( uneDate )

        curseur.close()
        print('[STOP] modeleResanet::getReservationsCarte()')
        return dates
```

Pour getDateAujourd'huiISO() :

```
def getDateAujourd'huiISO() :
    print('[START] datesResanet::getDateAujourd'huiISO()')
    dateCourante = datetime.datetime.today()
    aujourd'hui = '%04d-%02d-%02d' % ( dateCourante.year ,
                                       dateCourante.month ,
                                       dateCourante.day )
    print('[STOP] datesResanet::getDateAujourd'huiISO()')
    return aujourd'hui
```

Pour getDatesPeriodeCouranteISO() :

```
def getDatesPeriodeCouranteISO() :
    print('[START] datesResanet::getDatesPeriodeCouranteISO()')
    dates = []

    dateAujourd'hui = datetime.datetime.today()
    numJourAujourd'hui = dateAujourd'hui.weekday()

    dateCourante = dateAujourd'hui - datetime.timedelta( numJourAujourd'hui )

    for i in range( 12 ) :
        if i != 5 and i != 6 :
            dateISO = '%04d-%02d-%02d' % ( dateCourante.year ,
                                           dateCourante.month ,
                                           dateCourante.day )
            dates.append( dateISO )

            dateCourante = dateCourante + datetime.timedelta( 1 )

    print('[STOP] datesResanet::getDatesPeriodeCouranteISO()')
    return dates
```

Pour convertir `DateISOversFr()` :

```
def convertirDateISOversFR( dateISO ) :  
    print('[START] datesResanet::convertirDateISOversFR()')  
    annee , mois , jour = dateISO.split( '-' )  
    dateFR = '/' .join( ( jour , mois , annee ) )  
    print('[START] datesResanet::convertirDateISOversFR()')  
    return dateFR
```

6)

L'ordre de l'affichage des sondes est telles quel :

[illegible]

