# ML2 - Personnel

Nous avons avec mes camarades de groupe, apporté des modifications à une application en java,  des nouvelles fonctionnalités comme :

- l'ajout de date de départ et d'arrivée pour un employé
- connexion à la base de donnée créer pour M2L
- pouvoir sélectionner une ligue existante ou nouvellement créer
- changement de l'admin d'une ligne en ligne de commande

## Les différentes étapes :

1- création des variables en LocalDate des dates d'arrivés et de départs :

```
public class Employe implements Serializable, Comparable<Employe>
{
    private static final long serialVersionUID = 4795721718037994734L;
    private String nom, prenom, password, mail;
    private LocalDate datedarrive = LocalDate.now();
    private LocalDate datedepart = LocalDate.now();
    private Ligue ligue;
    private GestionPersonnel gestionPersonnel;

    Employe(GestionPersonnel gestionPersonnel, Ligue ligue, String nom, String prenom, String mail, String password)
    {
        this.gestionPersonnel = gestionPersonnel;
        this.nom = nom;
        this.prenom = prenom;
        this.password = password;
        this.mail = mail;
        this.ligue = ligue;
    }
```

2- réalisation de setter et getter pour les dates d'arrivées et de départs :

```
public LocalDate getdatedarrive()
{
    return datedarrive;
}

public void setdatedarrive(LocalDate datedarrive)
{
    this.datedarrive = datedarrive;
}

public LocalDate getdatedepart()
{
    return datedepart;
}

public void setdatedepart(LocalDate datedepart)
{
    this.datedepart = datedepart;
}
```

3- ajout de ses options dans le menu de l’application :

```
private Option changerdatedarrive(final Employe employe)
{
        return new Option("Changer la date d'arrivée", "a", () -> {employe.setdatedarrive(LocalDate.parse(getString("Nouveau date d'arriée : ")));});
}

private Option changerdatedepart(final Employe employe)
{
        return new Option("Changer la date de départ", "d", () -> {employe.setdatedepart(LocalDate.parse(getString("Nouveau date de départ : ")));});
}

private Option afficherdates(final Employe employe)
{
        return new Option("Afficher les dates", "f", () -> {System.out.println("Date d'arrivé : " + employe.getdatedarrive() + " date de départ "+ employe.getdatedepart());});
}
```

4- ajout d’exception et de ‘try - catch’ pour les dates afin qu’elles puisse être cohérente :

```
package personnel;

class ExceptionDate extends Exception
{
    public ExceptionDate()
    {
        System.out.println("L'exception a été levée");
    }

    public String toString()
    {
      return "Une erreur dans les dates !";
    }
}
```

```
public LocalDate getdatedarrive()
{
    return datedarrive;
}

public void setdatedarrive(LocalDate datedarrive) throws ExceptionDate
{
    try {
        if ((datedepart != null) && (datedarrive.isBefore(datedepart)))
        {
            throw new ExceptionDate();
        }
        this.datedarrive = datedarrive;
    }
    catch (ExceptionDate e) {
    }
}

public LocalDate getdatedepart()
{
    return datedepart;
}

public void setdatedepart(LocalDate datedepart) throws ExceptionDate
{
    try {
        if ((datedarrive != null) && (datedepart.isAfter(datedarrive)))
        {
            throw new ExceptionDate();
        }
            this.datedepart = datedepart;
        }
    catch (ExceptionDate e) {

        }
}
```

5- Ajout de la sélection de ligue :

```
Menu menuLigues()
{
        Menu menu = new Menu("Gérer les ligues", "l");
        menu.add(afficherLigues());
        menu.add(ajouterLigue());
        menu.add(selectionnerLigue());
        menu.addBack("q");
        return menu;
}
```

6- Connexion à la base de donnée :

```
package jdbc;

public class CredentialsExample
{
    private static String driver = "mysql";
    private static String driverClassName = "com.mysql.cj.jdbc.Driver";
    private static String host = "localhost";
    private static String port = "3306";
    private static String database = "m21";
    private static String user = "user";
    private static String password = "root";

    static String getUrl()
    {
        return "jdbc:" + driver + "://" + host + ":" + port + "/" + database ;
    }

    static String getDriverClassName()
    {
        return driverClassName;
    }

    static String getUser()
    {
        return user;
    }

    static String getPassword()
    {
        return password;
    }
}
```

7- Change l'admin en ligne de commande :

```
private List<Employe> changeradministrateur(final Ligue ligue)
{
        return new List<>("Changer un administrateur", "c",
                        () -> new ArrayList<>(ligue.getEmployes()),
                        (index, element) -> {ligue.setAdministrateur(element);}
                        );
}
```