Documentation sur un projet mené durant ma seconde année de BTS SIO :

Projet mené en groupe avec : MEZAOUI Alex Sofiane, PETRONIS Nicolas et YOUCEFI Hugo

Projet actuellement mené, et ayant débuté le 02/10/2023

Projet « Personnel »

Présentation du projet :

Cette application est réalisée dans le contexte d'une situation professionnelle pour la maison des ligues de Lorraine (M2L), s'occupant de ligues sportives. Cette application permet de gérer les ligues, les administrateurs des ligues ainsi que les employés de ses ligues.

Ce projet a été réparti en plusieurs missions, dans cette documentation nous spécifierons les modifications effectuées, au code initialement fournis par notre professeur, lors de nos missions.

Initialisation du projet :

Ressources fournies :

- Couche métier
- Les missions exprimant les nouveaux besoins

Classes déjà présentes dans le projet :

Classe Ligue :

→Constructeurs et variables ayant une visibilité private :

```
public class Ligue implements Serializable, Comparable<Ligue>
{
    private static final long serialVersionUID = 1L;
    private int id = -1;
    private String nom;
    private SortedSet<Employe> employes;
    private Employe administrateur;
    private GestionPersonnel gestionPersonnel;

    /**
     * Crée une ligue.
     * @param nom le nom de la ligue.
     */

    Ligue(GestionPersonnel gestionPersonnel, String nom) throws SauvegardeImpossible
    {
        this(gestionPersonnel, -1, nom);
        this.id = gestionPersonnel.insert(this);
    }

    Ligue(GestionPersonnel gestionPersonnel, int id, String nom)
    {
        this.nom = nom;
        employes = new TreeSet<>();
        this.gestionPersonnel = gestionPersonnel;
        administrateur = gestionPersonnel.getRoot();
        this.id = id;
    }
```

La classe ***Ligue*** représente une ligue avec des informations telles que son ID, son nom, ses employés, son administrateur, et elle est associée à une instance de la classe ***GestionPersonnel.***

Classe Employe

→Constructeurs et variables ayant une visibilité private

```
public class Employe implements Serializable, Comparable<Employe>
{
    private static final long serialVersionUID = 4795721718037994734L;
    private String nom, prenom, password, mail;
    private Ligue ligue;
    private GestionPersonnel gestionPersonnel;
    private LocalDate dateArrivee = LocalDate.of(0000, 01, 01);
    private LocalDate dateDepart = LocalDate.of(0000, 01, 01);
    private int id;

    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;
        try {
        this.id = gestionPersonnel.insert(this);
        }
        catch(SauvegardeImpossible e)
        {
            e.printStackTrace();
        }
    }
```

La classe Employe regroupe les informations liées à un employé dans un système de gestion du personnel, avec la possibilité de les créés, sauvegarder et modifier un employé.

Classe GestionPersonnelle

→Une seule instance de GestionPersonnel

```
public class GestionPersonnel implements Serializable
{
    private static final long serialVersionUID = -105283113987886425L;
    private static GestionPersonnel gestionPersonnel = null;
    private SortedSet<Ligue> ligues;
    private Employe root = new Employe(this, null, "root", "", "", "toor");
    public final static int SERIALIZATION = 1, JDBC = 2,
            TYPE_PASSERELLE = JDBC;

    private static Passerelle passerelle = TYPE_PASSERELLE == JDBC ? new jdbc.JDBC() : new serialisation.Serialization();

    /**
     * Retourne l'unique instance de cette classe.
     * Crée cet objet s'il n'existe déjà.
     * @return l'unique objet de type {@link GestionPersonnel}.
     */

    public static GestionPersonnel getGestionPersonnel()
    {
        if (gestionPersonnel == null)
        {
            gestionPersonnel = passerelle.getGestionPersonnel();
            if (gestionPersonnel == null)
                gestionPersonnel = new GestionPersonnel();
        }
        return gestionPersonnel;
    }

    public GestionPersonnel()
    {
        if (gestionPersonnel != null)
            throw new RuntimeException("Vous ne pouvez créer qu'une seuls instance de cet objet.");
        ligues = new TreeSet<>();
        gestionPersonnel = this;
    }
```