Université Cadi Ayyad
École supérieure de technologie
Département informatique
Filière génie informatique

Rapport de Tp 2

Java Avancées

Réalisé par :

El markhi Imane

Encadré par :

Leila Elkhrof

Année universitaire : 2024 /2025

Table des matières

1. Architecture du Projet (MVC + DAO)

Le projet de gestion des congés est organisé en plusieurs packages respectant le design pattern **MVC (Model-View-Controller)**, complété par un module **DAO (Data Access Object)** pour gérer les interactions avec la base de données. Voici un aperçu des différents packages et leur contenu :

1.1. Package `Model`

Le package **`Model`** contient les classes métiers représentant les données et la logique métier du système.

- **Classe `Holiday`** :
  - Représente un congé avec les attributs suivants :
    - `id` : Identifiant unique du congé.
    - `startDate` et `endDate` : Dates de début et de fin du congé.
    - `holidayType` : Type de congé (ex : **CONGE_PAYEE**, **CONGE_NON_PAYEE**, **CONGE_MALADIE**).
    - `employeeId` et `employeeNom` : Identifiants et noms des employés liés au congé.
  - Les constructeurs permettent de créer des objets `Holiday` pour différentes opérations (affichage, ajout, modification).

Classe Employe :

- Représente un employé avec les attributs suivants :
  - `id` : Identifiant unique de l'employé.
  - `nom` et `prenom` : Nom et prénom de l'employé.
  - `email` et `telephone` : Coordonnées de l'employé.
  - `salaire` : Salaire de l'employé.
  - `role` et `poste` : Rôle et poste de l'employé.
- Les constructeurs permettent de créer des objets Employe pour différentes opérations (affichage, ajout, modification).

- **Classe `HolidayModel`** : Contient la logique métier et interagit avec la base de données via le DAO. Principales méthodes :
  - `ajouterHoliday` : Ajoute un congé après validation des dates et du solde de l'employé.
  - `modifierHoliday` : Met à jour un congé existant.
  - `supprimerHoliday` : Supprime un congé en fonction de son identifiant.

- `afficherHolidays` : Récupère tous les congés pour affichage.
- `isValidDateRange` : Vérifie la validité de la plage de dates (début <= fin).

Classe EmployeModel :

- Contient la logique métier et interagit avec la base de données via le DAO.
- Principales méthodes :
  - `ajouter` : Ajoute un employé après validation des données.
  - `modifier` : Met à jour un employé existant.
  - `supprimer` : Supprime un employé en fonction de son identifiant.
  - `afficher` : Récupère tous les employés pour affichage.

1.2. Package `View`

Le package **`View`** contient les interfaces graphiques (GUI) permettant l'interaction utilisateur.

Classe EmployeView :

- Fournit une interface utilisateur pour gérer les employés :
  - Champs pour les données de l'employé : Nom, prénom, email, téléphone, salaire, rôle, poste.
  - Tableau `JTable` : Affiche les employés existants. Permet de sélectionner un employé pour modification ou suppression via un double-clic.
  - Boutons d'action : Ajouter, Modifier, Supprimer, Afficher.

- **Classe `HolidayView`** : ○ Fournit une interface utilisateur pour gérer les congés :
  - Champs pour les données de congé :
  - **Nom de l'employé** : Sélection via un `JComboBox`.
  - **Dates de début et fin** : Sélection via des `JSpinner`.
  - **Type de congé** : Choix via un `JComboBox`.
  - Tableau `JTable` :
  - Affiche les congés existants.
  - Permet de sélectionner un congé pour modification ou suppression via un double-clic.
  - Boutons d'action :
  - **Ajouter** : Ajoute un nouveau congé.
  - **Modifier** : Met à jour un congé sélectionné.
  - **Supprimer** : Supprime un congé sélectionné.

- **Afficher** : Recharge les congés depuis la base.

- **Comportement additionnel** :
  - Lorsqu'une ligne du tableau est double-cliquée, les champs sont automatiquement remplis avec les valeurs existantes pour permettre une modification.

1.3. Package `Controller`

Le package **`Controller`** contient la logique qui connecte la vue au modèle.

Classe EmployeController :

- Interagit entre la vue (EmployeView) et le modèle (EmployeModel).
- Actions principales :
  - Ajout d'un employé : Récupère les données depuis la vue, valide les données et appelle `ajouter` du modèle.
  - Affichage des employés : Appelle `afficher` du modèle et met à jour le tableau.
  - Modification d'un employé : Récupère les nouvelles données via la vue, valide les données et appelle `modifier`.
  - Suppression d'un employé : Supprime un employé sélectionné via `supprimer`.

- **Classe `HolidayController`** :
  - Interagit entre la vue (`HolidayView`) et le modèle (`HolidayModel`).
  - Actions principales :
    - **Ajout d'un congé** :
    - Récupère les données depuis la vue.
    - Valide les données et appelle `ajouterHoliday` du modèle. ▪ **Affichage des congés** :
    - Appelle `afficherHolidays` du modèle et met à jour le tableau.
    - **Modification d'un congé** :
    - Récupère les nouvelles données via la vue.
    - Valide les dates et appelle `modifierHoliday`. ▪ **Suppression d'un congé** :
    - Supprime un congé sélectionné via `supprimerHoliday`.

1.4. Package `DAO`

Le package **`DAO`** gère toutes les interactions avec la base de données.

- **Classe `HolidayDAOImpl`** :
  - Implémente les méthodes pour manipuler les données dans la table `holiday` :
    - `ajouter` : Ajoute un congé avec gestion du solde.
    - `afficher` : Récupère tous les congés avec les informations liées aux employés et types.
    - `modifier` : Met à jour un congé sélectionné.
    - `supprimer` : Supprime un congé en fonction de son ID.
  - Utilise `DBConnection` pour établir la connexion à la base.

Classe EmployeDAOImpl :

- Implémente les méthodes pour manipuler les données dans la table `employes` :
  - `ajouter` : Ajoute un employé avec gestion des rôles et postes.
  - `afficher` : Récupère tous les employés avec les informations liées aux rôles et postes.
  - `modifier` : Met à jour un employé sélectionné.
  - `supprimer` : Supprime un employé en fonction de son ID.
- Utilise `Connexion` pour établir la connexion à la base.

2. Fonctionnalités Analytiques

a) Ajouter un Congé (ajouterHoliday)

- **Étapes détaillées** :
  1. **Récupération des données** :
     - Nom de l'employé, dates de début et fin, type de congé via la vue.
  2. **Validation métier** :
     - Vérifie que la date de début est antérieure ou égale à la date de fin.
     - Vérifie que le solde de congé est suffisant pour la durée demandée.
  3. **Insertion dans la base** :
     - Exécution d'une requête `INSERT INTO` avec calcul du solde restant.
  4. **Retour utilisateur** :
     - Affiche un message de succès ou d'échec dans la vue.

b) Afficher les Congés (afficherHolidays)

- **Étapes détaillées** :
  1. Connexion à la base et exécution d'une requête SQL pour récupérer les congés avec jointures sur les employés et types de congés.

2. Mapping des résultats en objets `Holiday`.
3. Mise à jour du tableau dans la vue via un `DefaultTableModel`.

c) Modifier un Congé (modifierHoliday)

- **Étapes détaillées** :
  1. Sélection dans le tableau :
     - Double-cliquer remplit les champs de la vue avec les données du congé sélectionné.
  2. Modification des données :
     - L'utilisateur peut changer les dates ou le type de congé.
  3. Validation :
     - Vérifie la validité des nouvelles dates.
  4. Mise à jour :
     - Exécution d'une requête `UPDATE` pour modifier l'enregistrement.

d) Supprimer un Congé (supprimerHoliday)

- **Étapes détaillées** :
  1. Sélection d'une ligne dans le tableau.
  2. Exécution d'une requête `DELETE` pour supprimer le congé de la base.

3. Fonctionnalités Supplémentaires

- **Chargement des noms d'employés** :
  - Les noms des employés sont récupérés via une requête SQL dans `chargerNomsEmployes`.
  - Les données sont affichées dans un `JComboBox` pour simplifier la sélection.
- **Messages utilisateurs** :
  - La vue affiche des messages clairs en cas d'erreur (dates invalides, solde insuffisant, etc.) ou de succès.

4. Conclusion

Le projet de gestion des congés suit une architecture propre et modulaire. Chaque couche est bien séparée et respecte les principes du MVC et DAO, facilitant la maintenance et l'ajout de nouvelles fonctionnalités.

5.Le code source :

Le package Controller :

Le fichier EmployeController.java :

```
package Controller;


import Model.EmployeModel;


import java.sql.SQLException;
import java.util.List;


import Model.Employe.Poste;
import Model.Employe.Role;
import View.EmployeView;


public class EmployeController {
	private EmployeView view;
	private EmployeModel model;

	public EmployeController(EmployeView view, EmployeModel model) {
		this.model=model;
		this.view=view;

		this.view.ajouterButton.addActionListener(e-> ajouter());
```

```
            this.view.afficherButton.addActionListener(e ->
view.remplirTable(model.afficher()));
            this.view.supprimerButton.addActionListener(e-> supprimer());
            this.view.modifierButton.addActionListener(e-> modifierEmploye());
    }


    public void modifierEmploye() {
            try {
                    int id = view.getId()  ;
        String nom = view.getNom();
        String prenom = view.getPrenom();
        String email = view.getEmail();
        String telephone = view.getTelephone();
        double salaire = view.getSalaire();
        Role role = view.getRole();
        Poste poste = view.getPoste();

        boolean modified =model.modifier(id, nom, prenom, email, telephone, salaire, role,
poste);
        if(modified) {
            view.afficherMessageSucces("Employe a été modifié !");
           view.remplirTable(model.afficher());
        }else {
            view.afficherMessageErreur("Echec de modification");
        }


            }catch(Exception e) {
```

```
                e.printStackTrace();
        }

}


public void supprimer() {
        try {
        int id =view.getId();
        if(model.supprimer(id)) {
                view.afficherMessageSucces("Employe supprimé avec succes .");
                view.remplirTable(model.afficher());

        }else {
                view.afficherMessageErreur("cannot delete employe");
        }
        }catch(Exception e) {
                e.printStackTrace();
        }

}


public void ajouter() {
        int id = 0;
        String nom = view.getNom();
        String prenom = view.getPrenom();
        String email = view.getEmail();
        String telephone = view.getTelephone();
```

```
                double salaire=view.getSalaire();
                Role role = view.getRole();
                Poste poste = view.getPoste();
                boolean added=model.ajouter(id, nom, prenom, email, telephone, salaire, role,
poste);
                if(added) {
                        view.afficherMessageSucces("Employe ajoute avec succes");
                }else {
                        view.afficherMessageErreur("Echec de l'ajout");
                }
        }


}
```

Le package Controller :

Le fichier HolidayController.java :

```
package Controller;

import java.util.Date;
import java.util.List;

import DAO.EmployeDAOImpl;
import Model.HolidayModel;
import View.HolidayView;
import Model.Employe;
import Model.Holiday.HolidayType;
```

```
public class HolidayController {

	private HolidayView view;
	private HolidayModel model;

	public HolidayController(HolidayView view, HolidayModel model){
		this.model=model;
		this.view=view;
		view.ajouterButton.addActionListener(e -> ajouterHoliday());
		view.afficherButton.addActionListener(e->view.remplirTable(model.afficher()));
		view.supprimerButton.addActionListener(e->supprimer());
		view.modifierButton.addActionListener(e->modifierHoliday());
		loadEmployees();

	}
	//pour naviger les employes
	  private void loadEmployees() {
	      EmployeDAOImpl employeDAO = new EmployeDAOImpl();
	      List<Employe> employees = employeDAO.afficher();
	      view.populateEmployeeList(employees);
	    }

	public void ajouterHoliday() {
		int id=0;
		int Employeid=view.getSelectedEmployeeId();
		Date startDate=view.getStartDate(), endDate=view.getEndDate();
		HolidayType holidayType=view.getHolidayType();
```

```
            boolean ajoutResult=model.ajouterHoliday(id,Employeid, startDate, endDate,
holidayType);

            if (ajoutResult) {
                view.afficherMessageSucces("Ajout reussi :)");
            }else {
                view.afficherMessageErreur("Solde Insuffisant!!");
            }
        }


        public void supprimer() {
            try {
            int id =view.getId();
            if(model.supprimer(id)) {
                view.afficherMessageSucces("Holiday supprimé avec succes .");
                view.remplirTable(model.afficher());

            }else {
                view.afficherMessageErreur("cannot delete Holiday");
            }
            }catch(Exception e) {
                e.printStackTrace();
            }

        }

        public void modifierHoliday() {
           try {
```

```
                int id = view.getId();
                int employeId = view.getSelectedEmployeeId();
                Date startDate = view.getStartDate();
                Date endDate = view.getEndDate();
                HolidayType holidayType = view.getHolidayType();

                boolean modified = model.modifier(id, employeId, startDate, endDate,
holidayType);

                if (modified) {
                    view.afficherMessageSucces("Le congé a été modifié avec succès !");
                    view.remplirTable(model.afficher());
                } else {
                    view.afficherMessageErreur("Échec de la modification du congé !");
                }
            } catch (Exception e) {
                e.printStackTrace();
                view.afficherMessageErreur("Une erreur inattendue s'est produite !");
            }
        }

}
```

Le package DAO :

Le fichier Connexion.java :

```
package DAO;

import java.sql.*;

public class Connexion {

	private static final String URL
="jdbc:mysql://localhost:3306/employe_management";
	private static final String USER = "root";
    private static final String PASSWORD = "";
	static Connection conn = null;

	public static Connection getConnection() {
		if (conn != null) {
			System.out.println("Database connection established.");
		}
		try {

			conn = DriverManager.getConnection(URL , USER ,
PASSWORD);

		}catch(Exception e) {
			e.printStackTrace();
		}
		return conn;
	}

}
```

Le fichier EmployeDAOImpl.java :

```
package DAO;

import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import Model.Employe;
import Model.Employe.Poste;
import Model.Employe.Role;

public class EmployeDAOImpl implements GenericDAOI <Employe>{

	@Override
	public int ajouter(Employe employe) {
		String sql= "INSERT INTO employes (nom , prenom , email , telephone , salaire , role_id , poste_id) VALUES "
				+ "(?,?,?,?,?,?,"
				+ "(SELECT id FROM roles WHERE nom = ?),"
				+ "(SELECT id FROM postes WHERE nom = ?))";
		try(PreparedStatement stmt = Connexion.getConnection().prepareStatement(sql)){
			stmt.setString(1, employe.getNom());
			stmt.setString(2, employe.getPrenom());
```

```
                    stmt.setString(3, employe.getEmail());
                    stmt.setString(4, employe.getTelephone());
                    stmt.setDouble(5, employe.getSalaire());
            stmt.setString(6, employe.getRole().name());
            stmt.setString(7, employe.getPoste().name());
            int isInserted=stmt.executeUpdate();
            if(isInserted==0) {
               System.out.println("Échec de l'insertion : aucun poste ou rôle correspondant trouvé.");
               return 0;
            }

                }catch(SQLException e) {
                        e.printStackTrace();
                }
                return 1;

        }
        @Override
        public List<Employe> afficher(){
                List <Employe> employes = new ArrayList<>();
                String sql="SELECT e.id,e.nom,e.prenom,e.email,e.telephone,e.salaire,r.nom AS role_nom,p.nom AS poste_nom FROM employes e JOIN roles r ON e.role_id=r.id JOIN postes p ON e.poste_id=p.id";
                try(PreparedStatement stmt =Connexion.getConnection().prepareStatement(sql)){
                        ResultSet res=stmt.executeQuery();
                        while(res.next()) {
                                int id =res.getInt("id");
```

```
                String nom =res.getString("nom");
                String prenom = res.getString("prenom");
                String email = res.getString("email");
                String telephone =res.getString("telephone");
                Double salaire = res.getDouble("salaire");
                String role = res.getString("role_nom");
                String  poste = res.getString("poste_nom");
                Employe nvEmploye = new
Employe(id,nom,prenom,email,telephone,salaire,Role.valueOf(role),Poste.valueOf(poste));
                employes.add(nvEmploye);
            }
        }catch(SQLException e) {
            e.printStackTrace();
        }
        return employes;
    }
    @Override
    public boolean supprimer(int id) {
        String sql="DELETE FROM employes WHERE id=? ";
        try(PreparedStatement stmt
=Connexion.getConnection().prepareStatement(sql)){
            stmt.setInt(1, id);
            int rowsAffected=stmt.executeUpdate();
            return rowsAffected>0;
        }catch(SQLException e) {
            e.printStackTrace();
            return false;
        }
```

```
        }
        @Override
        public void modifier(int id, Employe employe) {
                String sql = "UPDATE employes SET nom = ?, prenom = ?, email = ?, telephone =
?, salaire = ?, "
                + "role_id = (SELECT id FROM roles WHERE nom = ?), "
                + "poste_id = (SELECT id FROM postes WHERE nom = ?) "
                + "WHERE id = ?";

               try(PreparedStatement stmt =
Connexion.getConnection().prepareStatement(sql)){
                        if(Connexion.getConnection()==null) {
                                System.out.println("Echec de connecter à la bae de donnée.");
                        }
                        stmt.setString(1,employe.getNom());
                        stmt.setString(2,employe.getPrenom());
                        stmt.setString(3,employe.getEmail());
                        stmt.setString(4,employe.getTelephone());
                        stmt.setDouble(5,employe.getSalaire());
                        stmt.setString(6,employe.getRole().name());
                        stmt.setString(7,employe.getPoste().name());
                        stmt.setInt(8,id);

                        int rest=stmt.executeUpdate();
                        if(rest>0) {
                                System.out.println("Modification was succesful.");
                        }else {
                                System.out.println("aucune modification a effectuée");
```

```
                }


            }catch(SQLException e) {
                e.printStackTrace();
            }


        }


}
```

Le fichier GenericDAOI :

```
package DAO;

import java.util.List;

import Model.Employe;

public interface GenericDAOI<T> {
            public int ajouter(T obj);
            public List<T> afficher();
            public boolean supprimer(int id);
            public void modifier(int id ,T obj);

}
```

Le fichier HolidayDAOImpl :

```
package DAO;

import java.sql.*;

import java.util.ArrayList;

import java.util.List;


import Model.Holiday;

import Model.Holiday.HolidayType;


public class HolidayDAOImpl implements GenericDAOI<Holiday> {

   @Override

   public int ajouter(Holiday holiday) {


      ResultSet rs = null;

      int solde = 0;


      try {

          String queryDays = "select datediff(?, ?) as days_diff";

         PreparedStatement stmt = Connexion.getConnection().prepareStatement(queryDays);

         stmt.setDate(1, new java.sql.Date(holiday.getEndDate().getTime()));

         stmt.setDate(2, new java.sql.Date(holiday.getStartDate().getTime()));

         rs = stmt.executeQuery();


         int daysDiff = 0;

         if (rs.next()) {

            daysDiff = rs.getInt("days_diff");

         }
```

```
String querySolde = "select solde from employes where id=?";
stmt = Connexion.getConnection().prepareStatement(querySolde);
stmt.setInt(1, holiday.getEmployeeId());
rs = stmt.executeQuery();


if (rs.next()) {
   solde = rs.getInt("solde");
}

if (daysDiff <= solde) {
   String query = "insert into holiday (employe_id, start_date, end_date, conge_id) values
(?, ?, ?,(select id from conges where nom=?))";

   stmt = Connexion.getConnection().prepareStatement(query);

   java.sql.Date sqlStartDate = new java.sql.Date(holiday.getStartDate().getTime());
   java.sql.Date sqlEndDate = new java.sql.Date(holiday.getEndDate().getTime());
   stmt.setInt(1, holiday.getEmployeeId());
   stmt.setDate(2, sqlStartDate);
   stmt.setDate(3, sqlEndDate);
   stmt.setString(4, holiday.getHolidayType().name());

   int rowAffected = stmt.executeUpdate();

   if (rowAffected > 0) {
      String updateSoldeQuery = "update employes set solde=solde-? where id=?";
```

```
                PreparedStatement updateStmt =
Connexion.getConnection().prepareStatement(updateSoldeQuery);
                updateStmt.setInt(1, daysDiff);
                updateStmt.setInt(2, holiday.getEmployeeId());
                updateStmt.executeUpdate();

                System.out.println("Insertion reussi :)");

            } else {
                System.out.println("Echec de l'insertion!!");

            }
        } else {
            System.out.println("Solde insuffisant :(");
            return solde;

        }

    } catch (SQLException e) {
        e.printStackTrace();
    }
    return solde;
  }

        @Override
        public List<Holiday> afficher() {
                List <Holiday> holidays = new ArrayList<>();
```

```
String sql="SELECT h.id AS id,e.id AS employe_id, CONCAT(e.nom, ' ', e.prenom) AS nom_complet, h.start_date AS start_date, h.end_date AS end_date, c.nom AS type FROM Holiday h JOIN Employes e ON h.employe_id = e.id JOIN Conges c ON h.conge_id = c.id ";

try(PreparedStatement stmt=Connexion.getConnection().prepareStatement(sql)){

ResultSet res=stmt.executeQuery();

while(res.next()) {

int id=res.getInt("id");

int employe_id=res.getInt("employe_id");

String nom_complet =res.getString("nom_complet");

String start_date=res.getString("start_date");

String end_date=res.getString("end_date");

String type=res.getString("type");

Date startDate = Date.valueOf(start_date);

Date endDate = Date.valueOf(end_date);

Holiday newHoliday = new Holiday(id,employe_id,nom_complet,startDate,endDate,HolidayType.valueOf(type));

holidays.add(newHoliday);

}

}catch(SQLException e) {

e.printStackTrace();

}


return holidays;

}



@Override

public boolean supprimer(int id) {
```

```
                String sql="DELETE FROM holiday WHERE id=? ";
                try(PreparedStatement stmt
=Connexion.getConnection().prepareStatement(sql)){
                        stmt.setInt(1, id);
                        int rowsAffected=stmt.executeUpdate();
                        return rowsAffected>0;
                }catch(SQLException e) {
                        e.printStackTrace();
                        return false;
                }
        }

        @Override
        public void modifier(int id, Holiday holiday) {
           String sql = "UPDATE holiday SET start_date = ?, end_date = ?, conge_id = (SELECT id
FROM conges WHERE nom = ?) WHERE id = ?";
           try (PreparedStatement stmt = Connexion.getConnection().prepareStatement(sql)) {
              stmt.setDate(1, new java.sql.Date(holiday.getStartDate().getTime()));
              stmt.setDate(2, new java.sql.Date(holiday.getEndDate().getTime()));
              stmt.setString(3, holiday.getHolidayType().name());
              stmt.setInt(4, id);

              int rowsAffected = stmt.executeUpdate();
              if (rowsAffected > 0) {
                 System.out.println("Holiday updated successfully!");
              } else {
                 System.out.println("No holiday found with the specified ID.");
              }
```

```
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }

}
```

Le main.java :

```
package Default;
import Controller.EmployeController;
import Controller.HolidayController;
import DAO.EmployeDAOImpl;
import DAO.HolidayDAOImpl;
import Model.EmployeModel;
import Model.HolidayModel;
import View.EmployeView;
import View.HolidayView;
import View.ManagementInterfaces;
public class Main {
        public static void main(String[] args) {
                EmployeView view = new EmployeView();
                HolidayView view2 =new HolidayView();
                EmployeDAOImpl dao = new EmployeDAOImpl();
                HolidayDAOImpl dao2 = new HolidayDAOImpl();
                EmployeModel model = new EmployeModel(dao);
                HolidayModel model2 = new HolidayModel(dao2);
                new EmployeController(view,model);
```

```
                new HolidayController(view2,model2);

                ManagementInterfaces combinedView = new
ManagementInterfaces(view,view2);

                combinedView.setVisible(true);




        }

}
```