

# Rapport d'itération n°4

## 1. Documentation utilisateur

### 1.1. Mode opératoire

- **Nom du livrable** : Projet JAVA modération
- **Version** : 0.4
- **Date de livraison** : 31/03/2025
- **Description** : Les propriétaires du site r3st0.fr souhaitent se doter d'une application Java qui sera installée sur les postes de travail des modérateurs, permettant de contrôler les avis déposés et de supprimer ou de bloquer la publication des avis ne respectant pas la charte d'utilisation du site ; chaque modérateur devra s'authentifier pour pouvoir utiliser l'application.

#### Installation :

- Pré-requis :
  - java <= 18
  - un IDE (netBeans, vs code)
  - wamp/xamp avec la BDD importée
- Procédure d'installation :
  1. Récupérer le projet depuis le dépôt gitLab (git clone <https://gitlab.com/APeixoto/modo-modo.git>)
  2. Importer la BDD fournit avec le projet dans wamp/xamp
  3. Le projet est prêt à être exécuté

#### Choix de la technologie d'accès aux données :

Nous avons choisi d'utiliser une **couche DAO** dans notre projet pour plusieurs raisons :

- **Simplicité et contrôle** : Le DAO nous permet de gérer directement les requêtes SQL et d'avoir un contrôle total sur l'accès aux données, sans complexité supplémentaire.
- **Architecture adaptée** : L'application étant locale et monolithique, une **API REST** serait inutilement lourde car il n'y a pas de communication entre services ou clients distants.
- **Alternative à JPA** : JPA apporte de la facilité pour des projets complexes, mais il impose un certain cadre et génère des surcharges inutiles pour un projet simple. Le DAO reste plus léger et adapté ici.

- **Bonne pratique** : Le DAO respecte la séparation des responsabilités et facilite la maintenance du code.

## 2. Documentation technique

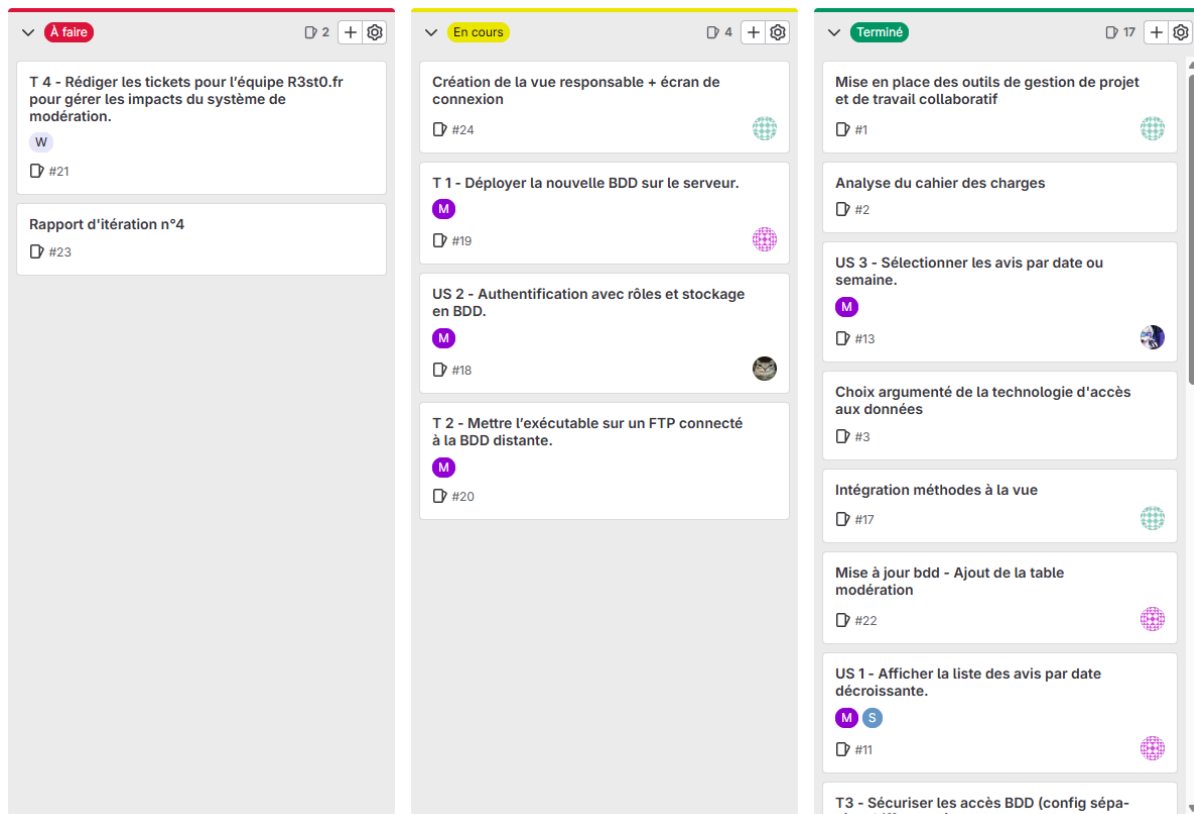
### 2.1. Gestion de projet

#### Comptes-rendus des Daily Scrums :

Date	Participants	Avancement	Problèmes rencontrés	Prochaines actions
24/02/2025	Loric Anthony Titouan Stefen	Argumentation du choix de la technologie et P.O.C effectué	Problème de synchronisation avec le fichier restoJdbc dans le dossier « src » et « build »	Maquette à terminer, déterminer les prochains objectifs du projet
03/03/2025	Loric Anthony Titouan Stefen	Avancement sur la détermination de la priorisation des tâches	x	Commencer <b>US 1</b> et <b>T 3</b>
10/03/2025	Loric Anthony Titouan Stefen	Réorganisation des tâches prioritaires	x	Commencer <b>US 3</b> , <b>US 5</b> , <b>US 6</b> , <b>US 4</b> et <b>US 7</b>
17/03/2025	Loric Anthony Titouan Stefen	Organisation des tâches prioritaires	x	Commencer <b>US2</b> , <b>T 1</b> , <b>T 2</b> et <b>T 4</b>
17/03/2025	Loric Anthony Titouan	Organisation des tâches prioritaires	x	

	Stefen			
--	--------	--	--	--

## Répartition des tâches :



## Dépôt de code :

- Adresse du dépôt : [https://gitlab.com/APeixoto/modo-modo]
- Branche de travail de l'itération : [Main]

## 2.2. Code source

- **Branche Git associée :**  
[https://gitlab.com/APeixoto/modo-modo/-/tree/main/ModerationResto?ref\_type=heads]
- **Évolutions du code :**

- L'objectif de l'**US2** était de créer un système d'authentification selon le rôle d'un utilisateur avec une nouvelle méthode, cette dernière a donc été ajoutée dans **UtilisateurDAO** :  
Tout d'abord, nous avons ajouté une méthode **getRole()** qui récupère le rôle selon le mail et mdp d'un utilisateur, celle-ci va nous être utile pour la prochaine méthode.

```

public static String getRole(String mail, String mdp) throws SQLException, IOException, Exception {
    Connection cnx = ConnexionBDD.getConnexion();
    PreparedStatement pstmt = cnx.prepareStatement("SELECT role FROM utilisateur WHERE mailU = ? AND mdpU = ?");
    pstmt.setString(1, mail);
    pstmt.setString(2, mdp); // Attention : dans un vrai projet, hache le mot de passe (ex: BCrypt)

    ResultSet rs = pstmt.executeQuery();
    if (rs.next()) {
        return rs.getString("role"); // Retourne le rôle si l'utilisateur est trouvé
    }

    return null; // Aucun utilisateur trouvé ou mauvais mot de passe
}

```

- Comme pour chacune de nos méthodes créées, un test de celles-ci a également été ajoutée :

```

// Test 8 : getRole UtilisateurDAO.getRole
System.out.println("\n Test 8 : UtilisateurDAO.getRole");
try {
    String role = UtilisateurDAO.getRole("alex.garat@gmail.com", "$1$zvn5hY$QSQDFUIQ$dufUQ$DFznHF5osT.");
    if (role != null) {
        System.out.println("Rôle : " + role);
    } else {
        System.out.println("Aucun rôle trouvé pour cet utilisateur.");
    }
} catch (SQLException | IOException ex) {
    System.out.println("TestRestoDAO - échec getRole : " + ex.getMessage());
}

```

- Ensuite, l'objectif de l'**US2** étant de créer un système d'authentification, une nouvelle méthode a été ajoutée : **connexion()**. Cette méthode permet la connexion à un compte utilisateur de la base de données grâce aux paramètres avec le mail et le mdp du compte. Un appel à la méthode créée précédemment, le rôle de cet utilisateur est récupéré et selon ce rôle, l'application envoie vers un cas différent.

```

public static void connexion(String mail, String mdp) throws Exception {
    try {
        String role = UtilisateurDAO.getRole(mail, mdp);

        if (role == null) {
            System.out.println("Échec de la connexion : identifiants incorrects.");
            return;
        }

        switch (role) {
            case "responsable":
                afficherVueResponsable();
                break;
            case "modérateur":
                afficherVueModerateur();
                break;
            default:
                System.out.println("Accès refusé : rôle inconnu.");
                break;
        }
    } catch (SQLException | IOException e) {
        System.out.println("Erreur lors de l'authentification : " + e.getMessage());
    }
}

```

- Les méthodes du switch case sont les suivantes, pour l'instant, pour montrer l'efficacité de la méthode précédente :

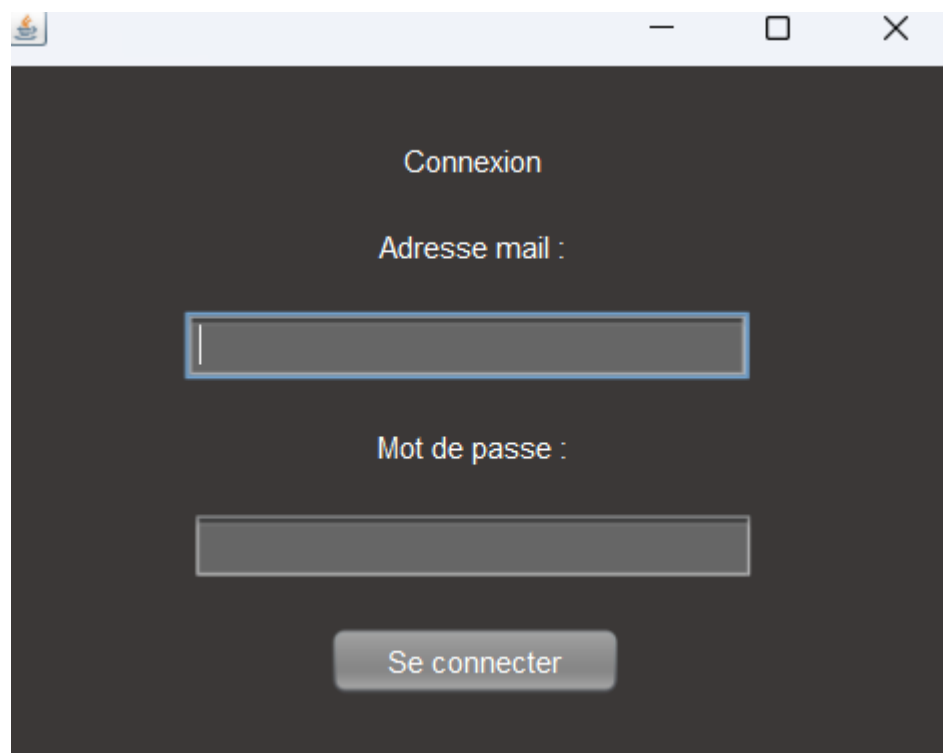
```
// Méthodes pour afficher les vues
public static void afficherVueResponsable() {
    System.out.println("Bienvenue, Responsable !");
    // Ici, ouvre une nouvelle fenêtre
}

public static void afficherVueModerateur() {
    System.out.println("Bienvenue, Modérateur !");
    // Idem, ouverture de la vue adaptée
}
```

- Comme la méthode précédente, un test a été créé pour voir le fonctionnement de cette dernière :

```
// Test 9 : Connexion UtilisateurDAO.connexion
System.out.println("\n Test 9 : UtilisateurDAO.connexion");
try {
    UtilisateurDAO.connexion("alex.garat@gmail.com", "$l$zvN5hYSQSQDFUIQSdufUQSDfznHF5osT.");
} catch (Exception ex) {
    System.out.println("TestUtilisateurDAO - échec de la connexion : " + ex.getMessage());
}
```

- Nous avons ensuite continué à créer et améliorer les interfaces graphiques de l'application : Nous avons ici l'interface de connexion, pour l'instant non fonctionnelle.




L'interface du modérateur, nous y avons intégré deux filtre, un qui recherche par date précisément (qui sera certainement remanié avec un jCalendar pour plus de sécurité et ergonomie et également un spinner à droite pour filtrer par semaine passée



The screenshot shows a web application window titled "Commentaire :". It has a search bar, a date spinner set to "0", and a "Rechercher" button. Below these is a list of comments, each with a date and a text description. The comments are as follows:

Date	Comment
2024-03-18	Très bon accueil :)
2024-03-15	Excellent.
2024-03-14	Bon accueil.
2024-03-13	Cuisine de qualité.
2024-03-11	Rapide.
2024-03-09	Cuisine correcte.
2024-03-08	Cuisine tres moyenne.
2024-03-07	À éviter...
2024-03-06	bof.
2024-03-04	5/5 parce que j'aime les entrecotes
2024-03-03	Très bon accueil.
2024-03-02	Très bonne entrecote, les frites sont maisons et delicieu
2024-03-01	moyen

—□×

## Commentaire :

▲▼

Rechercher

2024-03-11

Rapide.

2024-03-13

Cuisine de qualité.

2024-03-14

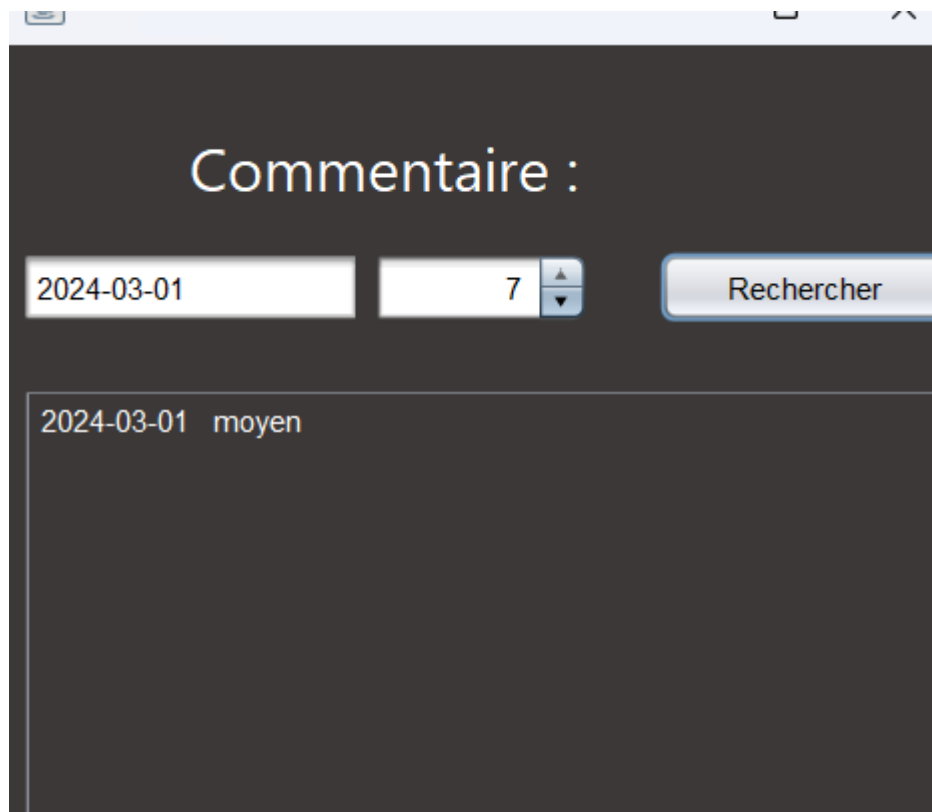
Bon accueil.

2024-03-15

Excellent.

2024-03-18

Très bon accueil :)



Commentaire :

2024-03-01 7 Rechercher

2024-03-01 moyen

Nous avons également fait la même interface pour le responsable mais celui-ci affiche uniquement les commentaires déjà masqués par le modérateur et peut donc après le signaler, le supprimer etc, fonctionnalité à rajouter lors de la prochaine séance.

## 2.4. Tests unitaires et d'intégration

### Jeux d'essai :

- Test de la méthode **getRole** issue de la classe **UtilisateurDAO** :

Pour les prochains tests, l'utilisateur enregistré **alex.garat@gmail.com** va recevoir une modification depuis la base de données afin de lui attribuer manuellement le rôle modérateur entre temps.

Premiers tests avec le rôle modérateur : **getRole()** :

```
Test 8 : UtilisateurDAO.getRole  
getConnexion : jdbc:mysql://localhost:3306/resto2  
Rôle : modérateur
```

Test de **connexion()** :

```
Test 9 : UtilisateurDAO.connexion  
Bienvenue, Modérateur !
```



Nouveaux tests avec aucun rôle : **getRole()** :

```
Test 8 : UtilisateurDAO.getRole  
getConnexion : jdbc:mysql://localhost:3306/resto2  
Aucun rôle trouvé pour cet utilisateur.
```

Test de **connexion()** :

```
Test 9 : UtilisateurDAO.connexion  
Échec de la connexion : identifiants incorrects.
```

## 2.4. Priorisation des tâches

Estimation des complexités manquantes :

ID	Description	Criticité	Complexité (estimée)
US 1	Afficher la liste des avis par date décroissante.	M	S
US 2	Authentification avec rôles et stockage en BDD.	M	L
US 3	Sélectionner les avis par date ou semaine.	M	M
US 4	Filtrer par « avis masqués ».	W	S
US 5	Masquer un avis.	S	S
US 6	Démasquer un avis pour le republier.	S	S

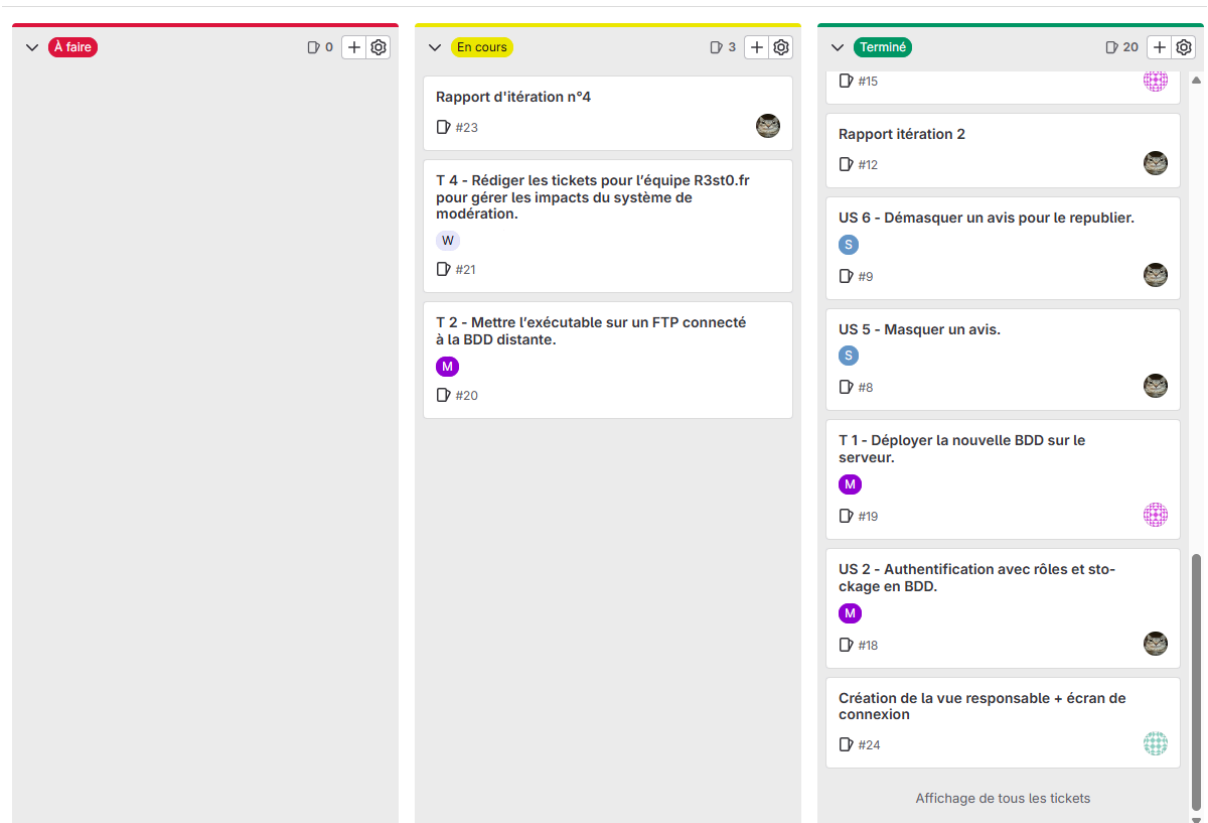
US 7	Supprimer un avis inopportun.	C	S
T 1	Déployer la nouvelle BDD sur le serveur.	M	S
T 2	Mettre l'exécutable sur un FTP connecté à la BDD distante.	M	M
T 3	Sécuriser les accès BDD (config séparée, chiffrement).	S	M
T 4	Rédiger les tickets pour l'équipe R3st0.fr pour gérer les impacts du système de modération.	W	M

A faire :

- **Restant : T 2 et T 4.**

## 2.5. Bilan de l'itération

- **Travail réalisé :**



- L'**US2**, l'ajout des interfaces ainsi que le déploiement de la base de données sur le serveur ont tous été terminés durant cette itération. Cependant, il faut encore ajouter le **T2** et puis finalement rédiger les tickets (**T4**).

### CAPTURE TABLEAUX TICKETS FIN ITÉRATION

- **Travail restant à faire :**

- Le travail de cette itération est fait, il nous faut donc poursuivre sur la prochaine.

- **Difficultés rencontrées :**

- Aucun problème pertinent retenu durant cette itération.