

# Projet AP 3

## Création d'une application de modération

User story : En tant qu'utilisateur, je souhaite, à partir du site restaurant avoir une interface de modération

### Table des matières

Création d'une application de modération.....	1
<b>Daily scrum debut (10/03/2025) :</b> .....	2
<b>TÂCHE 01 : Afficher la liste des avis publiés, de manière ordonnée afin de pouvoir les traités .....</b>	3
<b>TÂCHE 02 : Suppression d'un avis.....</b>	7
<b>TÂCHE 03 : Possibilité de masqué une critique .....</b>	11
<b>TÂCHE 04 : Connexion.....</b>	13
<b>Daily scrum Fin (17/03/2025) :</b> .....	16

## Daily scrum debut (10/03/2025) :

GitLab : [https://gitlab.com/Crouan/projet\\_java\\_moderation.git](https://gitlab.com/Crouan/projet_java_moderation.git)

### Pendant la réunion :

- Création des Ticket de l'IT2
- Attribution des tickets de l'IT2

### Planification de l'itération :

Dans un premier temps, nous déciderons tous ensemble des taches qui serait à effectuer dans cette deuxième itération, c'est à dire celle qui nous semble nécessaire pour avancer plus loin sur le projet. Puis se les attribués.

Au niveau de la répartition des taches il a été décider qu'on s'occuperait tous de nos tâches attribuées.

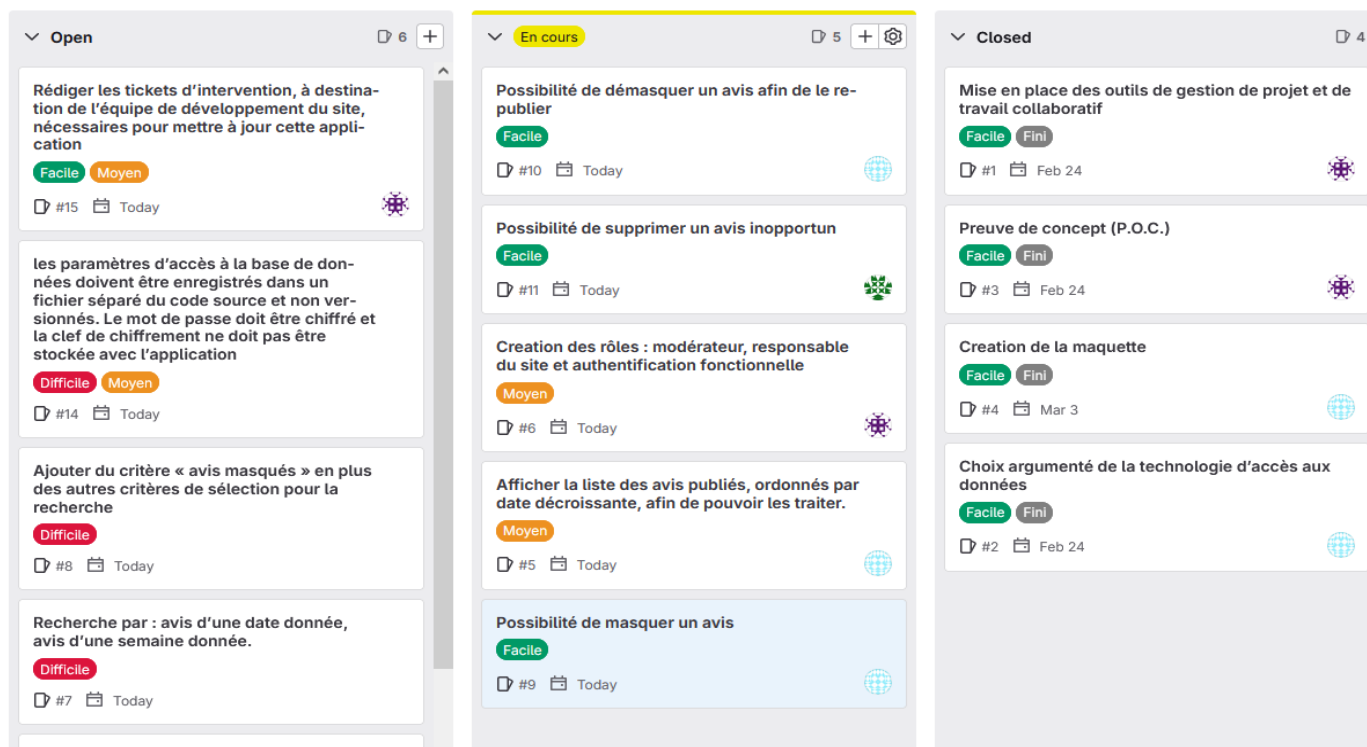
### Répartition des tâches du début :

**Alexis** → Tâche 1, 3

**Pierre** → Tâche 2

**Thomas** → Tâche 4

### Tableau de Kanban du début :



## TÂCHE 01 : Afficher la liste des avis publiés, de manière ordonnée afin de pouvoir les traités

Pour cela dans un premier temps, pour gagner en sécurité j'ai ajouté un utilisateur pour la base de données qui nous servira par la suite :

```

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0138 seconds.)

-- Création de l'utilisateur CREATE USER 'resto_user'@'localhost' IDENTIFIED BY 'secret';

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0062 seconds.)

-- Attribution de tous les privilèges sur la base de données resto3 GRANT ALL PRIVILEGES ON resto3.* TO 'resto_user'@'localhost';

[ Edit inline ] [ Edit ] [ Create PHP code ]

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0018 seconds.)

-- Appliquer les changements de privilèges FLUSH PRIVILEGES;

[ Edit inline ] [ Edit ] [ Create PHP code ]

```

Ensuite, j'ai dû créer "l'état" du commentaire c'est à dire masquer ou publier ce qui n'était pas précisé avant, j'ai décidé pour cela de créer une nouvelle table `etat_avis` :

```

CREATE TABLE IF NOT EXISTS `etat_avis` (
  `idEA` INT NOT NULL AUTO_INCREMENT,
  `libelleEA` VARCHAR(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  PRIMARY KEY (`idEA`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

```

J'ai ensuite ajouté les libelles

```

✓ 2 rows inserted.
Inserted row id: 2 (Query took 0.0070 seconds.)

INSERT INTO `etat_avis` (`libelleEA`) VALUES ('masquer'), ('publier');

```

Ajouté une colonne dans la table `critiquer` pour la lier à la nouvelle table

```

ALTER TABLE `critiquer`
ADD COLUMN `idEA` INT DEFAULT NULL,
ADD CONSTRAINT `fk_etat_avis` FOREIGN KEY (`idEA`) REFERENCES `etat_avis` (`idEA`) ON DELETE SET NULL;

```

Et définit les commentaires actuels comme publier :

```

UPDATE `critiquer`
SET `idEA` = 2 -- L'état "publier"
WHERE `idEA` IS NULL;

```

J'ai ensuite créé une API dans le même style de notre getAllResto.php pour récupérer tous les avis de la base de données :

```
<?php
require 'connexion.php';

try {
    // Suppression de la condition de l'idR, pour récupérer tous les restaurants
    $reponse = $connexion->prepare("SELECT * FROM critiquer c INNER JOIN etat_avis e ON c.idEA = e.idEA");

    // Exécuter la requête
    if ($reponse->execute()) {
        $avis = $reponse->fetchAll(PDO::FETCH_ASSOC);
        echo json_encode(["avis" => $avis]);
    } else {
        echo json_encode(['error' => 'Erreur lors de la sélection de données']);
    }
} catch (PDOException $e) {
    echo json_encode(["error" => $e->getMessage()]);
}
?>
```

Puis une classe correspondante dans NetBeans :

```
public class GetAllAvis {

    public static void main(String[] args) {
        // URL de l'API qui récupère les avis des restaurants
        String apiUrl = "http://localhost/ApiResto/getAllAvis.php";

        try {
            // Créer une URL à partir de l'URL de l'API
            URL url = new URL(apiUrl);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("GET"); // méthode GET pour récupérer des données

            // Lire la réponse de l'API
            BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
                response.append(inputLine);
            }
            in.close();

            // Convertir la réponse en objet JSON
            JSONObject jsonResponse = new JSONObject(response.toString());

            // Vérifier si la clé "avis" existe dans la réponse
            if (jsonResponse.has("avis")) {
                JSONArray avisArray = jsonResponse.getJSONArray("avis");

                // Afficher les informations des avis
                for (int i = 0; i < avisArray.length(); i++) {
                    JSONObject currentAvis = avisArray.getJSONObject(i);

                    // Extraire les informations de l'avis
                    int idResto = currentAvis.optInt("idR", -1); // idR pour le restaurant
                    double note = currentAvis.optDouble("note", 0.0); // note de l'avis
                    String commentaire = currentAvis.optString("commentaire", "Commentaire inconnu");
                    int idUtilisateur = currentAvis.optInt("idU", -1); // id de l'utilisateur ayant laissé l'avis
                    int idEtatAvis = currentAvis.optInt("idEA", -1); // id de l'état de l'avis
                    String libelleEtatAvis = currentAvis.optString("libelleEA", "État inconnu");

                    // Affichage des informations de l'avis
                    System.out.println("Restaurant ID: " + idResto);
                    System.out.println("Note: " + note);
                    System.out.println("Commentaire: " + commentaire);
                    System.out.println("Utilisateur ID: " + idUtilisateur);
                    System.out.println("État de l'avis ID: " + idEtatAvis);
                    System.out.println("Libellé de l'état: " + libelleEtatAvis);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

On teste dans le terminal :

```

Restaurant ID: 1
Note: 3.0
Commentaire: moyen
Utilisateur ID: 2
État de l'avis ID: 2
Libellé de l'état: publier
-----
Restaurant ID: 1
Note: 3.0
Commentaire: Très bonne entrecôte, les frites sont maisons et délicieuses.
Utilisateur ID: 3
État de l'avis ID: 2
Libellé de l'état: publier
-----
Restaurant ID: 1
Note: 4.0
Commentaire: Très bon accueil.
Utilisateur ID: 5
État de l'avis ID: 2
Libellé de l'état: publier
-----
Restaurant ID: 1
Note: 4.0
Commentaire: 5/5 parce que j'aime les entrecôtes
Utilisateur ID: 6
État de l'avis ID: 2
Libellé de l'état: publier
-----

```

On constate que cela fonctionne

On modifie ensuite le code de notre JFrame pour l'afficher dans notre tableau :

```

public class GetAllAvis {

    public static List<Object>[] getAvis() {
        // URL de l'API qui récupère les avis des restaurants
        String apiUrl = "http://localhost/ApiResto/getAllAvis.php";
        List<Object>[] avisList = new ArrayList<>();

        try {
            // Créer une URL à partir de l'URL de l'API
            URL url = new URL(apiUrl);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("GET"); // méthode GET pour récupérer des données

            // Lire la réponse de l'API
            BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
                response.append(inputLine);
            }
            in.close();

            // Convertir la réponse en objet JSON
            JSONObject jsonResponse = new JSONObject(response.toString());

            // Vérifier si la clé "avis" existe dans la réponse
            if (jsonResponse.has("avis")) {
                JSONArray avisArray = jsonResponse.getJSONArray("avis");

                // Extraire les informations de chaque avis et les ajouter dans la liste
                for (int i = 0; i < avisArray.length(); i++) {
                    JSONObject currentAvis = avisArray.getJSONObject(i);

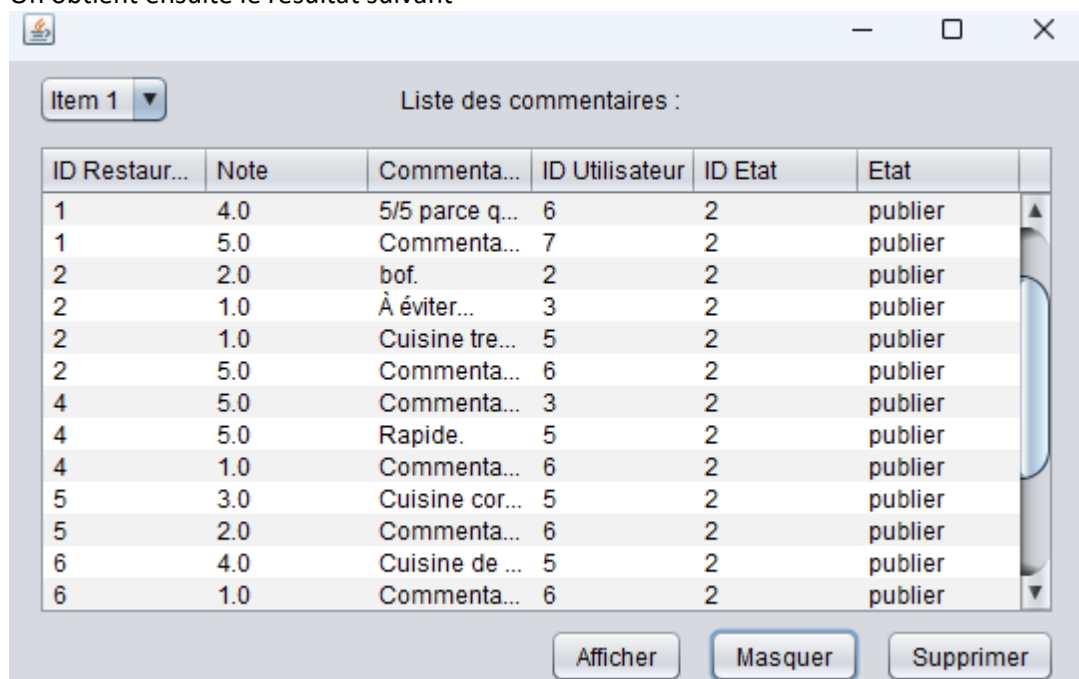
                    // Extraire les informations de l'avis
                    int idResto = currentAvis.optInt("idR", -1); // idR pour le restaurant
                    double note = currentAvis.optDouble("note", 0.0); // note de l'avis
                    String commentaire = currentAvis.optString("commentaire", "Commentaire inconnu");
                    int idUtilisateur = currentAvis.optInt("idU", -1); // id de l'utilisateur ayant laissé l'avis
                    int idEtatAvis = currentAvis.optInt("idEA", -1); // id de l'état de l'avis
                    String libelleEtatAvis = currentAvis.optString("libelleEA", "État inconnu");

                    // Ajouter l'avis sous forme de ligne dans le tableau
                    Object[] row = {idResto, note, commentaire, idUtilisateur, idEtatAvis, libelleEtatAvis};
                    avisList.add(row);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

```
private void loadDataFromAPI() {  
    // Récupérer les avis depuis la classe GetAllAvis  
    List<Object[]> avisList = GetAllAvis.getAvis();  
  
    // Définir les colonnes du tableau  
    String[] columns = {"ID Restaurant", "Note", "Commentaire", "ID Utilisateur", "ID Etat", "Etat"};  
  
    // Créer un modèle de tableau avec les colonnes définies  
    DefaultTableModel tableModel = new DefaultTableModel(columns, 0);  
  
    // Ajouter chaque avis récupéré dans le tableau  
    for (Object[] avis : avisList) {  
        tableModel.addRow(avis);  
    }  
    jTableListeCommentaires.setModel(tableModel);  
}
```

On obtient ensuite le résultat suivant



ID Restaur...	Note	Commenta...	ID Utilisateur	ID Etat	Etat
1	4.0	5/5 parce q...	6	2	publier
1	5.0	Commenta...	7	2	publier
2	2.0	bof.	2	2	publier
2	1.0	À éviter...	3	2	publier
2	1.0	Cuisine tre...	5	2	publier
2	5.0	Commenta...	6	2	publier
4	5.0	Commenta...	3	2	publier
4	5.0	Rapide.	5	2	publier
4	1.0	Commenta...	6	2	publier
5	3.0	Cuisine cor...	5	2	publier
5	2.0	Commenta...	6	2	publier
6	4.0	Cuisine de ...	5	2	publier
6	1.0	Commenta...	6	2	publier

## TÂCHE 02 : Suppression d'un avis

Création de la classe DeleteAvis

```
public class DeleteAvis {

    // Méthode pour supprimer un avis via une requête HTTP POST
    public static boolean deleteAvis(int idR, int idU) {
        String apiUrl = "http://localhost/ApiResto/deleteAvis.php"; // URL de l'API

        try {
            // Création de l'URL et ouverture de la connexion
            URL url = new URL(apiUrl);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("POST");

            // Configuration de la requête
            connection.setRequestProperty("Content-Type", "application/json");
            connection.setDoOutput(true); // Activation de l'envoi de données

            // Création de l'objet JSON avec les paramètres
            JSONObject jsonObject = new JSONObject();
            jsonObject.put("idR", idR);
            jsonObject.put("idU", idU);

            // Envoi des données
            OutputStream os = connection.getOutputStream();
            byte[] input = jsonObject.toString().getBytes("utf-8");
            os.write(input, 0, input.length);
            os.close();

            // Lecture de la réponse du serveur
            BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
                response.append(inputLine);
            }
            in.close();

            // Affichage de la réponse pour le débogage
            System.out.println("Response from server: " + response.toString());

            // Vérification de la réponse de l'API
            if (response.toString().contains("success")) {
                System.out.println("L'avis a été supprimé avec succès.");
                return true; // Suppression réussie
            } else {
                System.out.println("Erreur lors de la suppression de l'avis.");
                return false; // Échec de la suppression
            }
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Une erreur s'est produite lors de la suppression de l'avis.");
            return false;
        }
    }
}
```

## Ajout du bouton supprimer

Item 1 ▾

Liste des commentaires :

Title 1	Title 2	Title 3	Title 4
---------	---------	---------	---------

Afficher

Masquer

Supprimer

## Et du code du bouton

```
private void jButtonSupprimerActionPerformed(java.awt.event.ActionEvent evt) {  
    int selectedRow = jTableListeCommentaires.getSelectedRow();  
  
    if (selectedRow != -1) {  
        // Récupérer l'ID de l'avis sélectionné (colonne 0)  
        int idR = (int) jTableListeCommentaires.getValueAt(selectedRow, 0);  
  
        // Récupérer l'ID de l'utilisateur ayant écrit l'avis (colonne 3)  
        int idU = (int) jTableListeCommentaires.getValueAt(selectedRow, 3);  
  
        // Demander confirmation  
        int confirm = JOptionPane.showConfirmDialog(this, "Voulez-vous vraiment supprimer cet avis ?", "Confirmation", JOptionPane.YES  
        if (confirm == JOptionPane.YES_OPTION) {  
            // Appeler la méthode pour supprimer l'avis via l'API  
            boolean deleteSuccessful = DeleteAvis.deleteAvis(idR, idU); // CORRECTION ICI  
  
            if (deleteSuccessful) {  
                // Supprimer la ligne du tableau  
                DefaultTableModel model = (DefaultTableModel) jTableListeCommentaires.getModel();  
                model.removeRow(selectedRow);  
                JOptionPane.showMessageDialog(this, "Avis supprimé avec succès !");  
            } else {  
                JOptionPane.showMessageDialog(this, "Erreur lors de la suppression de l'avis.");  
            }  
        } else {  
            JOptionPane.showMessageDialog(this, "Veuillez sélectionner un avis à supprimer.");  
        }  
    }  
}
```

## Et ensuite création de l'API deleteAvis

```
deleteAvis.php  
C:\wamp64> www\Aplicto > deleteAvis.php  
1 <?php  
2 require 'connexion.php';  
3  
4 try {  
5     // Récupérer les données envoyées dans la requête POST  
6     $data = json_decode(file_get_contents("php://input"));  
7  
8     // Vérifier si les paramètres sont présents  
9     if (isset($data->idR) && isset($data->idU)) {  
10        // Préparer la requête SQL pour supprimer l'avis  
11        $query = "DELETE FROM critique WHERE idR = :idR AND idU = :idU";  
12        $stmt = $connexion->prepare($query);  
13  
14        // Lier les paramètres  
15        $stmt->bindParam(':idR', $data->idR, PDO::PARAM_INT);  
16        $stmt->bindParam(':idU', $data->idU, PDO::PARAM_INT);  
17  
18        // Exécuter la requête  
19        if ($stmt->execute()) {  
20            // Vérifier si une ligne a été supprimée  
21            if ($stmt->rowCount() > 0) {  
22                echo json_encode(["status" => "success", "message" => "Avis supprimé avec succès"]);  
23            } else {  
24                echo json_encode(["error" => "Aucun avis trouvé avec ces identifiants"]);  
25            }  
26        } else {  
27            echo json_encode(["error" => "Erreur lors de la suppression de l'avis"]);  
28        }  
29    } else {  
30        // Si les paramètres sont manquants  
31        echo json_encode(["error" => "Paramètres manquants"]);  
32    }  
}
```



Puis Test

Sélection d'un avis

Item 1

Liste des commentaires :

ID Restaur...	Note	Commentaire	ID Utilisateur	ID Etat	Etat
1	3.0	Très bonn...	3	2	publier
1	4.0	Très bon a...	5	2	publier
1	4.0	5/5 parce ...	6	2	publier
1	5.0	Commenta...	7	2	publier
2	1.0	À éviter...	3	2	publier
2	1.0	tres moyenne.	5	2	publier
4	5.0	Commenta...	3	2	publier
4	5.0	Rapide.	5	2	publier
4	1.0	Commenta...	6	2	publier
5	3.0	Cuisine co...	5	2	publier
5	2.0	Commenta...	6	2	publier
6	4.0	Cuisine de...	5	2	publier
6	1.0	Commenta...	6	2	publier

Afficher

Masquer

Supprimer

Confirmation de la suppression

Item 1

Liste des commentaires :

ID Restaur...	Note	Commentaire	ID Utilisateur	ID Etat	Etat
1	3				publier
1	4				publier
1	4				publier
1	5				publier
2	1				publier
2	1				publier
4	5				publier
4	5				publier
4	1				publier
5	3.0	Cuisine co...	5	2	publier
5	2.0	Commenta...	6	2	publier
6	4.0	Cuisine de...	5	2	publier
6	1.0	Commenta...	6	2	publier

Afficher

Masquer

Supprimer

Confirmation

?

Voulez-vous vraiment supprimer cet avis ?

Yes

No

BDD avant confirmation

<input type="checkbox"/>		Éditer		Copier		Supprimer	2	1	À éviter...	3	2
<input type="checkbox"/>		Éditer		Copier		Supprimer	2	1	Cuisine tres moyenne.	5	2
<input type="checkbox"/>		Éditer		Copier		Supprimer	2	5	NULL	6	1

## Validation de la suppression

ID Restaur...	Note	Commentaire	ID Utilisateur	ID Etat	Etat
1	3.0				publier
1	4.0				publier
1	4.0				publier
1	5.0				publier
2	1.0				publier
4	5.0				publier
4	5.0				publier
4	1.0				publier
5	3.0				publier
5	2.0	Commenta...	6	2	publier
6	4.0	Cuisine de...	5	2	publier
6	1.0	Commenta...	6	2	publier
7	4.0	Bon accueil.	1	2	publier

## BDD Après suppression

<input type="checkbox"/>	Éditer	Copier	Supprimer	2	1 À éviter...	3	2
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	5 NULL	6	1
<input type="checkbox"/>	Éditer	Copier	Supprimer	4	5 NULL	3	2

## TÂCHE 03 : Possibilité de masqué une critique

Pour ce faire j'ai dû ajouter le code suivant dans le `jButtonMasquerActionPerformed` de mon `JFrame`

```
private void jButtonMasquerActionPerformed(java.awt.event.ActionEvent evt) {
    int selectedRow = jTableListeCommentaires.getSelectedRow();

    if (selectedRow != -1) {
        // Récupérer l'ID de l'avis sélectionné (l'ID de l'avis est dans la première colonne)
        int idR = (int) jTableListeCommentaires.getValueAt(selectedRow, 0); // ID de l'avis dans la colonne 0

        // Récupérer l'ID de l'utilisateur ayant écrit l'avis (en supposant qu'il soit dans la colonne 3)
        int idU = (int) jTableListeCommentaires.getValueAt(selectedRow, 3); // ID de l'utilisateur dans la colonne 3

        // Vérifier l'état actuel de l'avis (l'état est dans la 5e colonne - état en texte)
        String currentStateLabel = (String) jTableListeCommentaires.getValueAt(selectedRow, 5); // Libellé de l'état dans la colonne 5

        // Si l'état n'est pas déjà "Masqué", procéder à la mise à jour
        if (!"masquer".equalsIgnoreCase(currentStateLabel)) {
            // Mettre à jour l'état de l'avis dans la base de données via l'API
            boolean updateSuccessful = UpdateAvis.updateAvisState(idR, 1, idU); // 1 correspond à l'état "Masqué" (idEA = 1)

            if (updateSuccessful) {
                // Mettre à jour le tableau pour refléter ce changement
                jTableListeCommentaires.setValueAt(2, selectedRow, 4); // Mettre à jour l'ID Etat (colonne 4)
                jTableListeCommentaires.setValueAt("masquer", selectedRow, 5); // Mettre à jour l'état visuel (colonne 5)
            } else {
                JOptionPane.showMessageDialog(this, "Erreur lors de la mise à jour de l'état de l'avis.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "L'avis est déjà masqué.");
        }
    } else {
        JOptionPane.showMessageDialog(this, "Veuillez sélectionner un avis.");
    }
}
```

Et crée une classe java update pour appeler l'api et lui donner les valeurs de la ligne qui a été sélectionnée

```

public class UpdateAvis {

    // Méthode pour mettre à jour l'état de l'avis via une requête HTTP
    public static boolean updateAvisState(int idR, int idEA, int idU) {
        String apiUrl = "http://localhost/ApiResto/updateAvisState.php";

        try {
            // Créer une URL à partir de l'URL de l'API
            URL url = new URL(apiUrl);
            HttpURLConnection connection = (HttpURLConnection) url.openConnection();
            connection.setRequestMethod("POST");

            // Configurer la requête pour accepter une réponse JSON
            connection.setRequestProperty("Content-Type", "application/json");
            connection.setDoOutput(true); // Activer l'envoi de données

            // Créer le corps de la requête en JSON
            JSONObject jsonObject = new JSONObject();
            jsonObject.put("idR", idR);
            jsonObject.put("idEA", idEA);
            jsonObject.put("idU", idU);

            // Envoyer les paramètres via la requête POST
            OutputStream os = connection.getOutputStream();
            byte[] input = jsonObject.toString().getBytes("utf-8");
            os.write(input, 0, input.length);

            // Lire la réponse du serveur
            BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));
            String inputLine;
            StringBuilder response = new StringBuilder();

            while ((inputLine = in.readLine()) != null) {
                response.append(inputLine);
            }
            in.close();

            // Vérification de la réponse
            System.out.println("Response from server: " + response.toString()); // Ajout du log pour débogage

            if (response.toString().contains("success")) {
                System.out.println("L'état de l'avis a été mis à jour.");
                return true; // Mise à jour réussie
            } else {
                System.out.println("Erreur lors de la mise à jour de l'avis.");
                return false; // Erreur dans la mise à jour
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

Puis une api pour exécuter la requête :

```

<?php
require 'connexion.php';

try {
    // Récupérer les données envoyées dans la requête POST
    $data = json_decode(file_get_contents("php://input"));

    // Vérifier si les paramètres sont présents
    if (isset($data->idR) && isset($data->idEA) && isset($data->idU)) {
        // Préparer la requête SQL pour mettre à jour l'état de l'avis
        $query = "UPDATE critiquer SET idEA = :idEA WHERE idR = :idR AND idU = :idU";
        $stmt = $connexion->prepare($query);

        // Lier les paramètres
        $stmt->bindParam(':idR', $data->idR, PDO::PARAM_INT);
        $stmt->bindParam(':idEA', $data->idEA, PDO::PARAM_INT);
        $stmt->bindParam(':idU', $data->idU, PDO::PARAM_INT);

        // Exécuter la requête
        if ($stmt->execute()) {
            // Si la mise à jour réussit
            echo json_encode(["status" => "success", "message" => "État de l'avis mis à jour"]);
        } else {
            // Si la mise à jour échoue
            echo json_encode(["error" => 'Erreur lors de la mise à jour']);
        }
    } else {
        // Si les paramètres sont manquants
        echo json_encode(["error" => 'Paramètres manquants']);
    }
} catch (PDOException $e) {
    // En cas d'exception, afficher l'erreur
    echo json_encode(["error" => $e->getMessage()]);
}
>>

```

Ensuite, pour tester on sélectionne une ligne et clique sur le bouton

Item 1 ▾ Liste des commentaires :

ID Restaur...	Note	Commenta...	ID Utilisateur	ID Etat	Etat
1	3.0	moyen	2	2	masquer
1	3.0	Très bonne...	3	2	publier
1	4.0	Très bon a...	5	2	publier
1	4.0	5/5 parce q...	6	2	publier
1	5.0	Commenta...	7	2	publier
2	2.0	bof.	2	2	publier
2	1.0	À éviter...	3	2	publier
2	1.0	Cuisine tre...	5	2	publier
2	5.0	Commenta...	6	2	publier
4	5.0	Commenta...	3	2	publier
4	5.0	Rapide.	5	2	publier
4	1.0	Commenta...	6	2	publier
5	3.0	Cuisine cor...	5	2	publier

Afficher Masquer Supprimer

On constate ensuite les modifications dans la bdd

			idR	note	commentaire	idU	1	idEA
<input type="checkbox"/>	Edit	Copy	Delete	7	4	Bon accueil.	1	2
<input type="checkbox"/>	Edit	Copy	Delete	1	3	moyen	2	1
<input type="checkbox"/>	Edit	Copy	Delete	2	2	bof.	2	2
<input type="checkbox"/>	Edit	Copy	Delete	1	3	Très bonne entrecote, les frites sont maisons et d...	3	2
<input type="checkbox"/>	Edit	Copy	Delete	2	1	À éviter	3	2

Quand on relance la JFrame on constate que la critique est bien modifiée.

## TÂCHE 04 : Connexion

Le JFrameConnexion était déjà créé par Alexis lors de l'itération 1. Voici le JFrameConnexion :

Connexion :

Login :

Mot de passe :

Connexion

J'ai créé l'API afin de comparer avec les logins et les mdps dans la bdd :

```
updateAvisState.php  verifConnexion.php X
verifConnexion.php
1  <?php
2  require 'connexion.php';
3
4  header('Content-Type: application/json; charset=UTF-8');
5
6  try {
7      // Récupérer les données envoyées dans la requête POST
8      $data = json_decode(file_get_contents("php://input"));
9
10     // Vérifier si les paramètres sont présents
11     if (isset($data->mailU) && isset($data->mdpU)) {
12         // Préparer la requête SQL pour vérifier les informations d'identification de l'utilisateur
13         $query = "SELECT idU, mailU, mdpU, pseudoU, idRole FROM utilisateur WHERE mailU = :mailU";
14         $stmt = $connexion->prepare($query);
15
16         // Lier les paramètres
17         $stmt->bindParam(':mailU', $data->mailU, PDO::PARAM_STR);
18
19         // Exécuter la requête
20         if ($stmt->execute()) {
21             // Récupérer l'utilisateur
22             $utilisateur = $stmt->fetch(PDO::FETCH_ASSOC);
23
24             if ($utilisateur && password_verify($data->mdpU, $utilisateur['mdpU'])) {
25                 // Si les informations d'identification sont correctes
26                 unset($utilisateur['mdpU']); // Ne pas retourner le mot de passe haché
27                 echo json_encode(["status" => "success", "message" => "Connexion réussie", "utilisateur" => $utilisateur]);
28             } else {
29                 // Si les informations d'identification sont incorrectes
30                 echo json_encode(["error" => 'Identifiants incorrects']);
31             }
32         } else {
33             // Si la requête échoue
34             echo json_encode(["error" => 'Erreur lors de la vérification des informations d'identification']);
35         }
36     } else {
37         // Si les paramètres sont manquants
38         echo json_encode(["error" => 'Paramètres manquants']);
39     }
40 } catch (PDOException $e) {
41     // En cas d'exception, afficher l'erreur
42     echo json_encode(["error" => "Erreur de connexion à la base de données : " . $e->getMessage()]);
43 }
44 }
```

En effet, j’ai modifié la bdd pour créer une nouvelle table “rôle” avec une clef étrangère entre cette table et la table “utilisateur” grâce à une nouvelle colonnes.

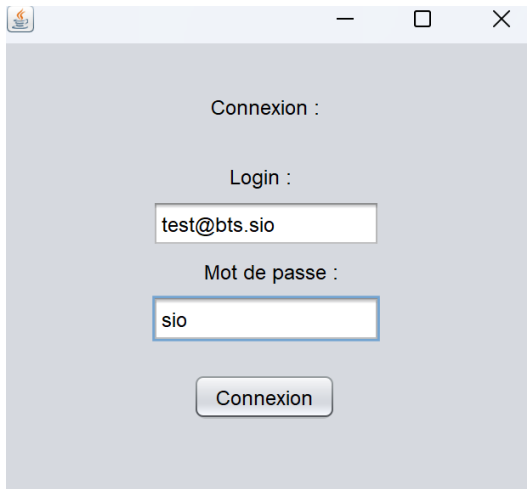


✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0101 seconde(s).)

```
ALTER TABLE utilisateur ADD COLUMN idRole INT;
```

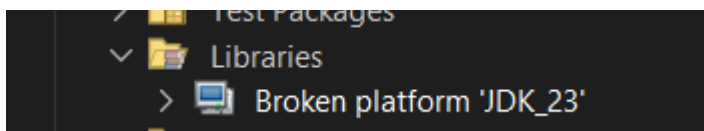
[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]

Et puis j'ai modifié le code du JFrameConnexion pour afficher le JFrameListeCommentaire lors de la connexion :

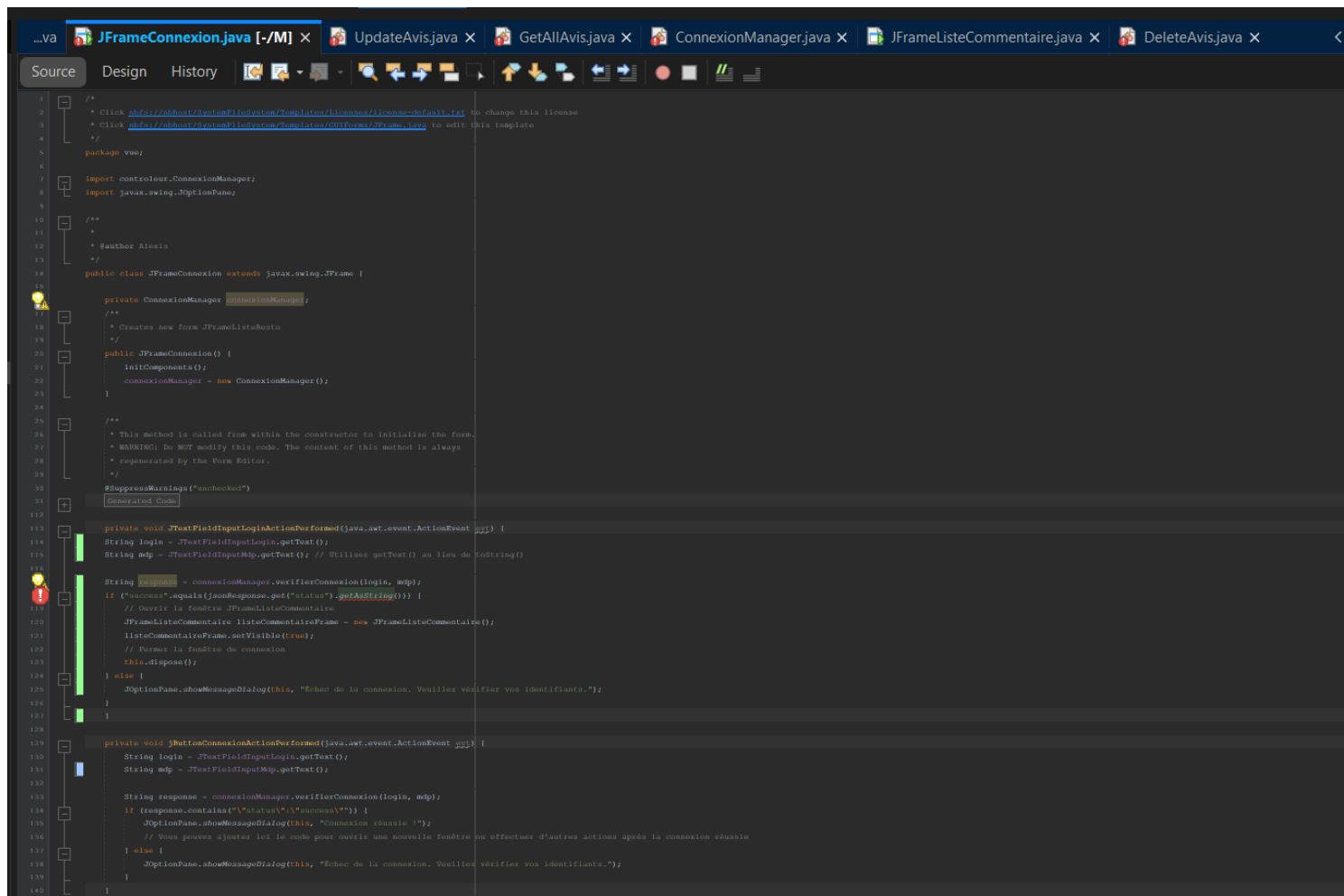


ID Restaur...	Note	Commenta...	ID Utilisateur	ID Etat	Etat
1	3.0	moyen	2	2	masquer
1	3.0	Très bonne...	3	2	publier
1	4.0	Très bon a...	5	2	publier
1	4.0	5/5 parce q...	6	2	publier
1	5.0	Commenta...	7	2	publier
2	2.0	bof.	2	2	publier
2	1.0	À éviter...	3	2	publier
2	1.0	Cuisine tre...	5	2	publier
2	5.0	Commenta...	6	2	publier
4	5.0	Commenta...	3	2	publier
4	5.0	Rapide.	5	2	publier
4	1.0	Commenta...	6	2	publier
5	3.0	Cuisine cor...	5	2	publier

Il y a juste un problème avec mes Jar/Folder qui ne veulent plus s'ajouter au projet ce qui empêche le bon fonctionnement du projet :



Voici le code du JFrameConnexion :



```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
4  */
5  package voo;
6
7  import controleur.ConnectionManager;
8  import javax.swing.JOptionPane;
9
10
11  /**
12   *
13   * @author Alexis
14   */
15  public class JFrameConnexion extends javax.swing.JFrame {
16
17      private ConnectionManager connexionManager;
18      /**
19       * Creates new form JFrameListeReacts
20       */
21      public JFrameConnexion() {
22          initComponents();
23          connexionManager = new ConnectionManager();
24      }
25
26      /**
27       * This method is called from within the constructor to initialize the form.
28       * WARNING: Do NOT modify this code. The content of this method is always
29       * regenerated by the Form Editor.
30       */
31      @SuppressWarnings("unchecked")
32      Generated Code
33
34
35      private void jTextFieldInputLoginActionPerformed(java.awt.event.ActionEvent evt) {
36          String login = jTextFieldInputLogin.getText();
37          String mdp = jTextFieldInputMdp.getText(); // Utilisez getText() au lieu de toString()
38
39          String response = connexionManager.verifierConnexion(login, mdp);
40          if (!response.equals(javax.swing.JOptionPane.getStackTraceString())) {
41              // Ouvrir la fenetre JFrameListeCommentaire
42              JFrameListeCommentaire listeCommentaireFrame = new JFrameListeCommentaire();
43              listeCommentaireFrame.setVisible(true);
44              // Fermer la fenetre de connexion
45              this.dispose();
46          } else {
47              JOptionPane.showMessageDialog(this, "Echec de la connexion. Veuillez vérifier vos identifiants.");
48          }
49      }
50
51      private void jButtonConnexionActionPerformed(java.awt.event.ActionEvent evt) {
52          String login = jTextFieldInputLogin.getText();
53          String mdp = jTextFieldInputMdp.getText();
54
55          String response = connexionManager.verifierConnexion(login, mdp);
56          if (response.contains("%status%\n%success%")) {
57              JOptionPane.showMessageDialog(this, "Connexion réussie !");
58              // Vous pouvez ajouter ici le code pour ouvrir une nouvelle fenetre ou effectuer d'autres actions après la connexion réussie
59          } else {
60              JOptionPane.showMessageDialog(this, "Echec de la connexion. Veuillez vérifier vos identifiants.");
61          }
62      }
63  }
```

Daily scrum Fin (17/03/2025) :



### Résultat :

Alexis s'est occupé du code Java concernant l'affichage de la liste des critiques et a pu finir son affichage dans le JFrame ainsi que la possibilité de masquer une critique, ces tâches ont pu être finies sans soucis majeur. Les seules erreurs rencontrées étaient concernant la passation des informations à l'API.

Thomas a pu travailler sur la connexion fonctionnelle avec l'aide d'Alexis quand cela était nécessaire notamment avec les erreurs de contrainte concernant la base de données, ce sont les seules erreurs qu'il a rencontrées, la connexion n'a pas pu être finalisée à cause d'un problème de Bibliothèques qui ne se s'ajoute pas au projet.

Pierre lui s'est occupé de la possibilité de suppression d'une critique. La tâche a pu être terminée, les seules erreurs rencontrées étaient à la récupération du projet et de version de NetBeans, ces erreurs ont pu être réglées à l'aide de Monsieur Bourgeois.

Pour la troisième itération nous essayeront donc de finir toutes les tâches qu'il nous reste à faire dans la mesure du possible.

Les tâches effectuées lors de cette deuxième itération sont : US 1, US 2, US 5 et US 7

### Tableau de Kanban de fin :

