Mini-Projet d'Ingénierie de bases de données

Sujet Choisi: Sujet 7

Application JAVA de gestion de réservations de chambres d’Hôtel

Moula Mariem & Slaoui Yassine

Enseignante: FEKIH Tayssir

Faculté des Sciences de Tunis

Université de Tunis El Manar

# Note de l'auteur

Premier paragraphe : La Répartition des tâches.

Deuxième paragraphe : Le Diagramme des cas d’utilisations.

Troisième paragraphe : Le Modèle Entité-Association.

Quatrième paragraphe : Les Imprimes écran.

Cinquième paragraphe : Divulgation et remerciements.

# Description du Projet

Vu la complexité des activités d’un Hôtel, l'outil informatique est un élément qui lui facilite une bonne gestion pour une meilleure prise des décisions dans la vie.

Dans ce contexte, ce projet s’est focalisé sur une des problématiques de la gestion des Hôtel qui est la gestion des réservations des chambres car dans le domaine des Hôtels, la vitesse de traitement des réservations est fondamentale, c’est pour cela que l’automatisation et l’informatisation de la gestion d’un Hôtel est devenue indispensable.

## La répartition des tâches.

Moula Mariem : Frontend et lien

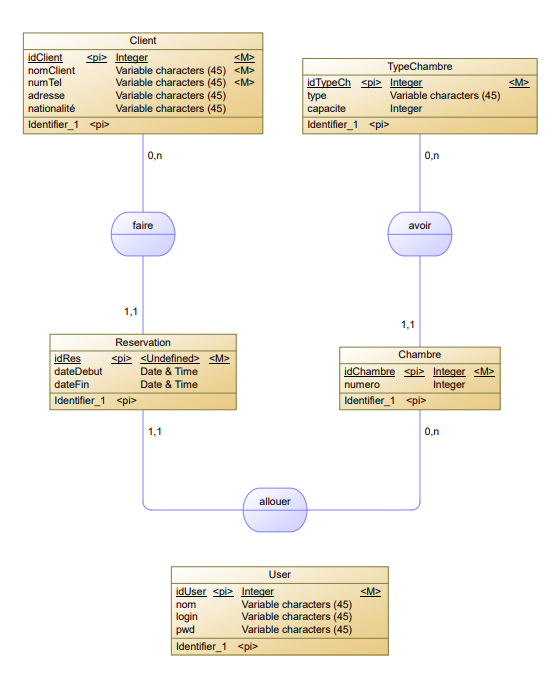
Slaoui Yassine : Backend et lien

# Méthode

## Le diagramme des Cas d'utilisation

# 

## Le Modèle Entité-Association



## Les Imprimes écran

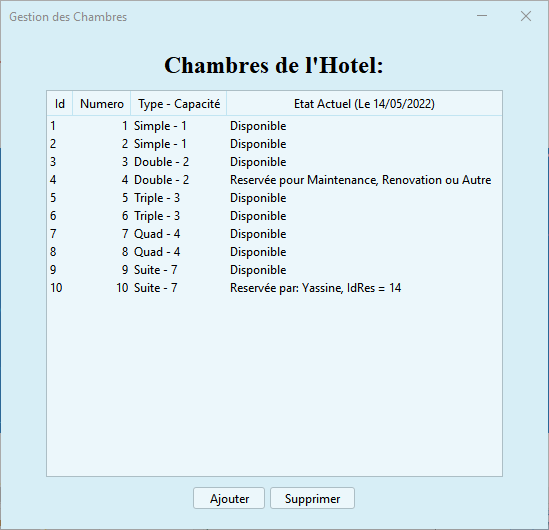
## 

Interface d’authentification

Si aucun utilisateur n’est enregistré sur la base de données un utilisateur de départ se crée automatiquement pour pouvoir gérer l'hôtel provisoirement et/ou ajouter des utilisateurs a la base de données.



Interface principale

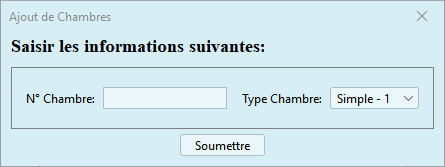


Interface de gestion des chambres

On peut ici gérer les chambres, modifier les numéros, les types et voir si une chambre est réservée ou disponible ou en maintenance.

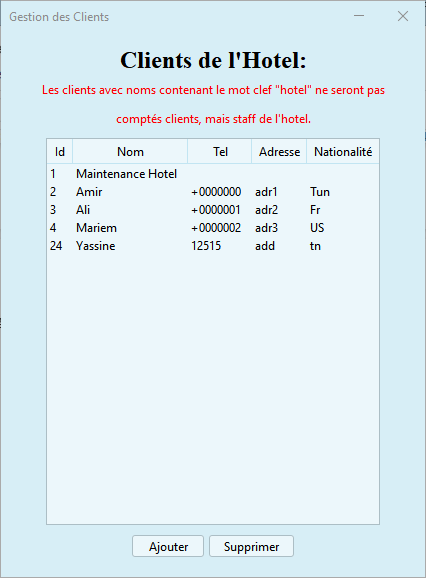
Si on modifie le numéro, un message d’erreur s'affiche si le numéro est dupliqué.

La suppression se fait après sélection des chambres à supprimer et clique sur le bouton supprimer.



Interface d’ajout d’une chambre

Le numéro de chambre à ajouter ne doit pas exister déjà si c’est le cas un message d’erreur indiquant cela s’afficher.

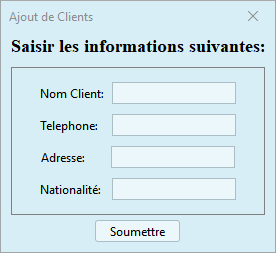


Interface de gestion des clients

On peut ici gérer les clients, modifier leurs données personnelles inscrites déjà, la seule contrainte est que le nom ne soit pas vide, ceci engendra un message d’erreur.

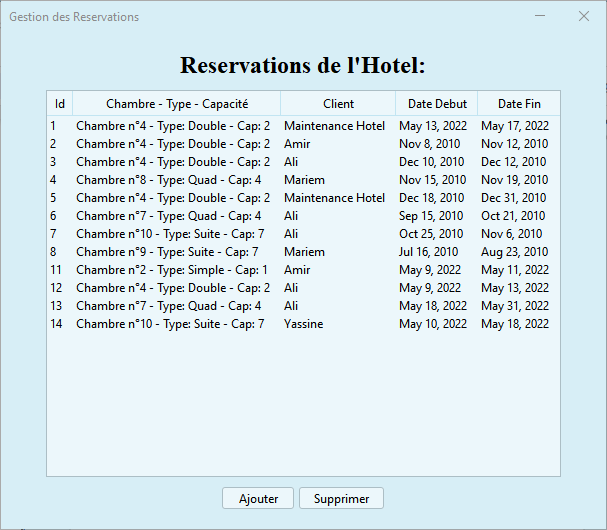
Les clients avec des noms contenant le mot clef « hôtel » seront considérés comme staff de l’hôtel et ne seront pas comptés dans les statistiques.

La suppression se fait après sélection des clients à supprimer et clique sur le bouton supprimer.



Interface d’ajout de Clients

Ici les noms des clients à ajouter ne doivent pas être vides.

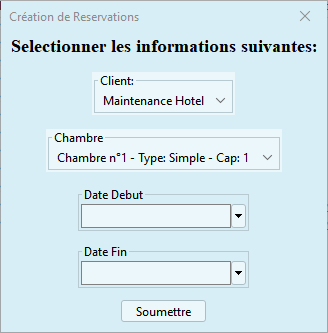


Interface de gestion des réservations

On peut ici gérer les réservations, modifier la période de la réservation ou la chambre si possible (car la chambre peut ne pas être disponible), et on peut aussi modifier les clients qui louent la chambre a partir d’une liste de clients qui permet de choisir un client sans faire d’erreurs de nom ou id.

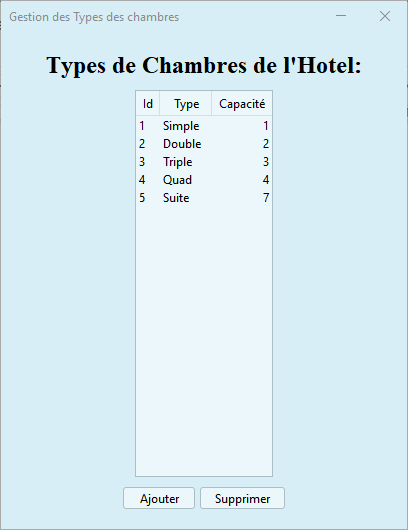
Si on tombe sur une période d’une chambre ou si elle n’est pas disponible un message d’erreur s'affiche et la date précédente se rétablit.

La suppression se fait après sélection des chambres à supprimer et clique sur le bouton supprimer.

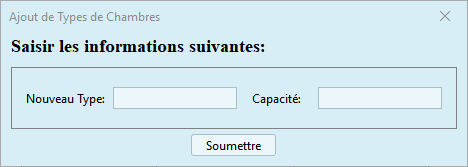


Interface de création de réservation

La chambre choisie doit être disponible dans l’intervalle choisi et la date de début doit être inférieure à la date de fin, sinon un message d’erreur s’affichera.



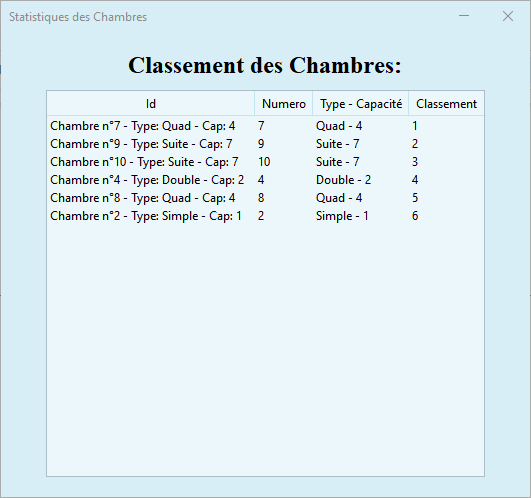
Interface de gestion des types de chambres



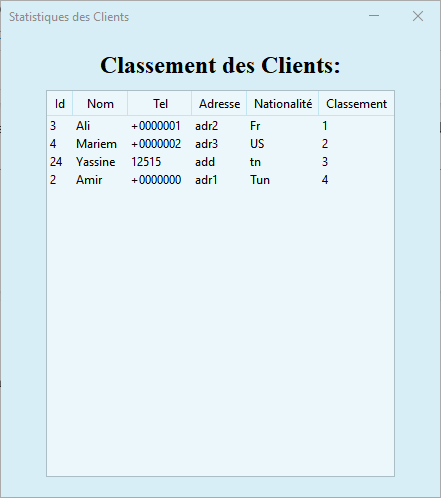
Interface d’ajout de types de chambre



Interface des statistiques de l'hôtel

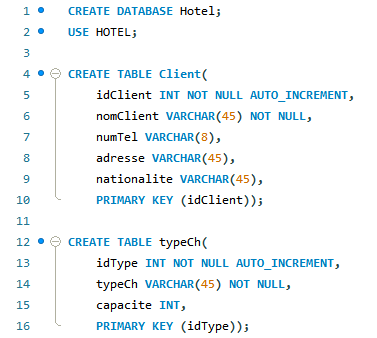


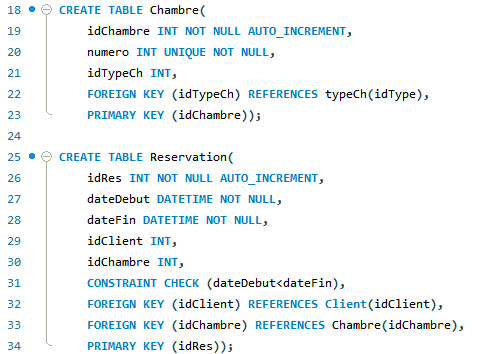
Interface des statistiques sur les chambres (les classements des chambres les plus demandées)

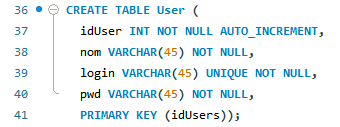


Interface des statistiques des clients (les clients réguliers de l’hotel)

## Les Snippets de requêtes SQL







## Les Snippets de code JAVA

**public** **static** **final** String ***DB\_URL*** = "jdbc:mysql://localhost:3306/hotel";

**public** **static** **final** String ***USER*** = "root";

**public** **static** **final** String ***PASS*** = "toor";

## 

**private** **void** importFromDB() {

**try** (Connection connection = DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***); Statement statement = connection.createStatement();) {

ResultSet rs = statement.executeQuery("SELECT \* FROM Client");

**while** (rs.next()) {

*clients*.add(**new** Client(rs.getInt("idClient"), rs.getString("nomClient"), rs.getString("numTel"), rs.getString("adresse"), rs.getString("nationalite")));

}

rs = statement.executeQuery("SELECT \* FROM TypeCh");

**while** (rs.next()) {

*typeChambres*.add(**new** TypeChambre(rs.getInt("idType"), rs.getString("typeCh"), rs.getInt("capacite")));

}

rs = statement.executeQuery("SELECT \* FROM Chambre");

**while** (rs.next()) {

*chambres*.add(**new** Chambre(rs.getInt("idChambre"), rs.getInt("Numero"), *findTypeChambre*(rs.getInt("idTypeCh"))));

}

rs = statement.executeQuery("SELECT \* FROM Reservation");

**while** (rs.next()) {

*reservations*.add(**new** Reservation(rs.getInt("idRes"), rs.getTimestamp("dateDebut"), rs.getTimestamp("dateFin"), *findClient*(rs.getInt("idClient")), *findChambre*(rs.getInt("idChambre"))));

}

rs = statement.executeQuery("SELECT \* FROM User");

**while** (rs.next()) {

*users*.add(**new** User(rs.getInt(1), rs.getString(2), rs.getString(3), rs.getString(4)));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**static** **public** **int** addClient(String aNom, String aTel, String aAdresse, String aNat) {

**try** (Connection connection = DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***);

PreparedStatement pStatement = connection.prepareStatement("INSERT INTO Client(nomClient, numTel, adresse, nationalite) VALUES (?, ?, ?, ?);", PreparedStatement.***RETURN\_GENERATED\_KEYS***)) {

pStatement.setString(1, aNom);

pStatement.setString(2, aTel);

pStatement.setString(3, aAdresse);

pStatement.setString(4, aNat);

pStatement.executeUpdate();

ResultSet rs = pStatement.getGeneratedKeys();

**if** (rs.next()) {

**int** autoId = rs.getInt(1);

*clients*.add(**new** Client(autoId, aNom, aTel, aAdresse, aNat));

**return** autoId;

}

} **catch** (Exception e) {

**return** -1;

}

**return** 0;

}

**static** **public** **boolean** delClient(**int** aId) {

**try** (Connection connection = DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***); PreparedStatement pStatement = connection.prepareStatement("DELETE FROM Client WHERE idClient = ?;")) {

Client cl = *findClient*(aId);

**for** (**int** i = *reservations*.size() - 1; i >= 0; i--) {// on supprime toutes les réservations

Reservation res = *reservations*.get(i);// faites par ce client

**if** (res.getClient().equals(cl))

*delReservation*(res.getId());

}

pStatement.setInt(1, aId);

pStatement.executeUpdate();

*clients*.remove(cl);

**return** **true**;

} **catch** (Exception e) {

**return** **false**;

}

}

**static** **public** **boolean** updClient(**int** aId, String aNom, String aTel, String aAdresse, String aNat) {

**try** (Connection connection = DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***); PreparedStatement pStatement = connection.prepareStatement("UPDATE Client SET nomClient = ?, numTel = ?, adresse = ?, nationalite = ? WHERE idClient = ?;")) {

pStatement.setString(1, aNom);

pStatement.setString(2, aTel);

pStatement.setString(3, aAdresse);

pStatement.setString(4, aNat);

pStatement.setInt(5, aId);

pStatement.executeUpdate();

*findClient*(aId).update(aNom, aTel, aAdresse, aNat);

**return** **true**;

} **catch** (Exception e) {

**return** **false**;

}

}

**try** (Connection connection = DriverManager.*getConnection*(Launcher.***DB\_URL***, Launcher.***USER***, Launcher.***PASS***);

Statement statement = connection.createStatement();) {

ResultSet rs = statement.executeQuery("SELECT ch.idChambre, sum((r.dateFin - r.dateDebut)) AS Periode\r\n"

+ "FROM Chambre ch, Reservation r, Client cl\r\n"

+ "WHERE ch.idChambre = r.idChambre\r\n"

+ "AND r.idClient = cl.idClient\r\n"

+ "AND UPPER(cl.nomClient) NOT LIKE '%HOTEL%'\r\n"

+ "GROUP BY ch.idChambre\r\n"

+ "ORDER BY periode DESC;");

**int** d=0;

**while** (rs.next()) {

Chambre ch=Launcher.*findChambre*(rs.getInt(1));

tModel.addRow(**new** Object[] { ch, ch.getNum(), ch.getType(), ++d});

}

}**catch** (Exception e) {

e.printStackTrace();

}

**try** (Connection connection = DriverManager.*getConnection*(Launcher.***DB\_URL***, Launcher.***USER***, Launcher.***PASS***);

Statement statement = connection.createStatement();) {

ResultSet rs = statement.executeQuery("SELECT cl.idClient, sum((r.dateFin - r.dateDebut)) AS Periode\r\n"

+ "FROM Reservation r, Client cl\r\n"

+ "WHERE r.idClient = cl.idClient\r\n"

+ "AND UPPER(cl.nomClient) NOT LIKE '%HOTEL%'\r\n"

+ "GROUP BY cl.idClient\r\n"

+ "ORDER BY periode DESC;");

**int** d=0;

**while** (rs.next()) {

Client cl=Launcher.*findClient*(rs.getInt(1));

tModel.addRow(**new** Object[] { cl.getId(),cl.getNom(),cl.getTel(),cl.getAdresse(),cl.getNat(), ++d});

}

}**catch** (Exception e) {

e.printStackTrace();

}

# Divulgation et remerciements

En fin à ce projet nous remercions Allah qui nous a doté d’une grande volonté et qui nous a aidé et nous a donné le courage, la force et la patience d’accomplir ce travail.

On tient à remercier et exprimer notre profonde gratitude à notre enseignante pour les encouragements et les précieux conseils.