Système de Gestion d'Hôtel

Rapport de projet

UE Projet BD & Réseau - L3 Informatique – S5 - 2025-2026

CY Cergy Paris Université

Équipe projet – Groupe D4:

Lisa ISSAD

NOUHA ELYAMANY

Ouardia ACHAB

Responsable de formation : Marc Lemaire

Notre objectif principal : Concevoir et développer un système complet de gestion hôtelière intégrant une base de données relationnelle optimisée et une architecture réseau client-serveur et créer une solution scalable et performante qui garantit une expérience utilisateur fluide pour le personnel hôtelier et les clients

Date : 30 Novembre 2025

URL du site : https://nouhaelyamany.alwaysdata.net/Gest_hotel/

Dépôt Git : https://github.com/Ouardia2003/systeme-gestion-hotel

Sommaire

1 Contexte Général

Le projet vise à concevoir un système de gestion hôtelière centralisée, destiné à automatiser les opérations clés. Grâce à une architecture client-serveur fonctionnant sur réseau, les utilisateurs peuvent consulter les disponibilités des chambres, effectuer ou annuler des réservations, ainsi que gérer les factures. Les échanges de données, formatés en JSON et transmis via le TCP, assurent à la fois la fiabilité et la clarté des informations échangées.

1.1 Architecture Technique Retenue

Conformément aux exigences du projet, notre solution s'articulera autour de quatre composants principaux :

- **Base de données PostgreSQL** : stockage centralisé de toutes les données (clients, chambres, réservations, factures, services)
- **Interface Web PHP** : destinée aux clients pour les réservations en ligne, avec authentification sécurisée
- **Serveur applicatif réseau** : développé en **Java** pour la communication avec les clients distants. Il utilise **JDBC** pour assurer la connexion avec la base de données PostgreSQL et gérer les transactions de manière fiable.
- **Client réseau** : développé en **Python**, c'est une application sur tablette permettant au personnel de scanner les QR codes des chambres pour consulter et mettre à jour en temps réel les statuts d'occupation, de maintenance et de nettoyage. Le choix de ce langage, distinct du serveur, démontre l'interopérabilité du protocole et facilite l'intégration tierce grâce à ses bibliothèques natives de gestion réseau et JSON

1.2 Architecture Réseau :

La partie réseau s'appuie sur une architecture client-serveur robuste utilisant le protocole TCP avec des échanges au format JSON. Cette approche permet aux systèmes externes (agences de voyage, partenaires, plateformes de réservation) d'interagir directement avec notre système pour effectuer des réservations, consulter les disponibilités et gérer les annulations, tout en préservant l'intégrité et la cohérence des données centralisées.

2 Réflexion générale sur les données échangées via le réseau

Dans le cadre du projet de gestion d'hôtel, la partie réseau vise à assurer la communication entre un client réseau et un serveur applicatif connecté à la base de données.
L'objectif est de permettre aux clients distants d'effectuer des actions (réservations, annulations, consultations) tout en garantissant la fiabilité et la clarté des échanges.

2.1 Types de données échangées

- **Requêtes de consultation** : demander des informations à la base (ex. liste des chambres disponibles, détails d'une réservation).
- **Requêtes de mise à jour** : insérer, modifier ou supprimer des informations (ex. enregistrer une réservation, annuler une chambre, mettre à jour une facture).
- **Messages de contrôle** : gérer la session réseau (identification simple du client, confirmation de réception, fermeture de connexion).

2.2 Format des échanges

- Les échanges seront réalisés au format **JSON**, léger et lisible.
- Le protocole de transport retenu est **TCP**, car il garantit la livraison et l'ordre des messages.

Exemple de requête client (création de réservation) :

```json
{
  "action": "create_reservation",
  "client_id": 102,
  "room_id": 305,
  "date_start": "2025-11-12",
  "date_end": "2025-11-15"
}
```

Exemple de réponse serveur :

```json
{
  "status": "ok",
  "reservation_id": 789,
  "message": "Réservation confirmée"
}
```

2.3 Scénarios typiques d’échanges

Consulter les disponibilités

a. Le client demande les chambres libres pour une période donnée.

b. Le serveur renvoie la liste correspondante.

Créer une réservation

a. Le client envoie une demande avec les dates et l’identifiant de la chambre.

b. Le serveur met à jour la base et renvoie une confirmation ou un message d’erreur.

Annuler une réservation

a. Le client envoie l’identifiant de la réservation.

b. Le serveur supprime l’entrée en base et confirme l’annulation.

Clôturer la session

a. Le client envoie une requête de fermeture.

b. Le serveur confirme puis termine la connexion proprement.

2.4 Principes retenus

- **Simplicité** : limiter les messages à ce qui est strictement nécessaire.
- **Fiabilité** : chaque requête reçoit une réponse claire (succès ou erreur).
- **Clarté** : messages structurés avec champs explicites (action, status, message).
- **Robustesse** : gestion des erreurs réseau (ex. requête invalide, client déconnecté).

3 Diagramme de GANTT

Notre planification suit le calendrier imposé par le cahier des charges, avec des jalons intermédiaires pour assurer un suivi régulier :

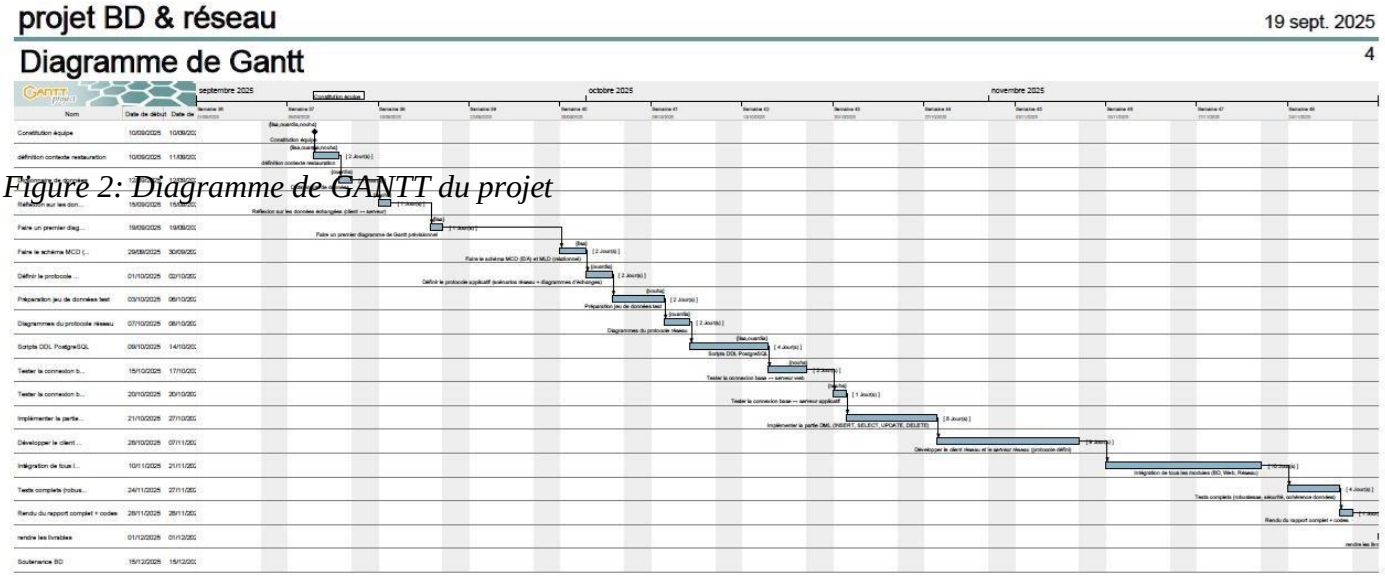

Figure 2: Diagramme de GANTT du projet

Figure 1:Digramme de GANTT du projet

4 Dictionnaire de données :

Le dictionnaire de données de notre système de gestion hôtelière comprend 10 entités principales couvrant l'ensemble des processus métier : de la gestion client (Client, Reservation, Facture) à l'administration opérationnelle (Employe, Maintenance) en passant par la gestion des ressources (Chambre, Type_chambre, Service). Chaque entité est rigoureusement définie avec ses contraintes spécifiques (formats email, cohérence temporelle, statuts métier) pour assurer l'intégrité des données et refléter fidèlement les règles de fonctionnement d'un établissement hôtelier moderne.

Table	Attribut	Type	NULL NOT NULL	Autre Contrainte Comme	Contrainte De domaine	Particularité	Défault	Exemple
Reservation	id_reservation	int	NN	PK	>0	Auto-incremeter		1
	date_debut	DATE	NN		>= CURRENT_DATE			2025-10-01
	date_fin	DATE	NN		>date_debut			2025-10-05
	nombre_personnes	int	NN					3
	statut	varchar(20)	NN		IN('en attente', 'confirmée', 'annulée', 'terminée')			confirmée

Client	id_client	int	NN	PK	>0	Auto-incremeter		101
	nom	varchar(50)	NN		len(nom)>=2			Dupont
	prenom	varchar(40)	NN		Len(prenom)>=2			Alice
	mail	varchar(100)	NN		FORMAT MAIL			alice.dupont@gmail.com
	tel	char(10)	N	UNIQUE	NE CONTIENT QUE DES CHIFFRES			0611345078
	date_naiss	DATE	NN					1990-04-15
	date_inscription	TIMESTAMP	NN					2025-09-20 14:35:00
	adresse	varchar(256)	NN					5 Avenue Anatole, 75007 Paris
Chambre	id_chambre	int	NN	PK	>0	Auto-incremeter		201
	statut	varchar(20)	NN		IN('libre', 'occupée', 'maintenance', 'nettoyage')			libre
	etage	int	NN		BETWEEN 0 and 20	0=RDC		2
	prix_base	DECIMAL(8, 2)	NN		>0	Prix par nuit		120.00
	Superficie	DECIMAL(5, 2)	NN		>0	En m²		25.50
Type_chambre	Id_type	int	NN	PK	>0	Auto-incremeter		1
	nom	varchar(30)	NN	UNIQUE				Suite
	capacite_max	int	NN		BETWEEN 1 AND 10			4
	surface_type	DECIMAL(5, 2)	NN		>0	En m²		30.00
	description	TEXT	N			Description détaillée	NULL	Suite spacieuse avec vue sur mer
Service	Id_service	int	NN	PK	>0	Auto-incremeter		301
	nom	varchar(50)	NN		Len(nom)>=2			Massage
	description	TEXT	N				NULL	Massage relaxant d'une heure

	prix_unitaire	DECIMAL(8, 2)	NN		>0			60.00
	actif	BOOLEAN	NN					true
	categorie	varchar(30)	NN		IN('restauration', 'wellness', 'trasport', 'autre')		'autre'	wellness
Facture	id_facture	int	NN	PK	>0	Auto-incrementer		501
	date_emission	DATE	NN					2025-09-22
	montant_total	DECIMAL(10 ,2)	NN		>0			350.00
	tva	DECIMAL(8, 2)	NN		>=0			70.00
	statut_paiement	varchar(20)	NN		IN('impayes', 'partiellemnt_payee', 'payée', 'remboursé')			payée
Paiement	id_paiement	int	NN	PK	>0	Auto-incrementer		603
	mode_paiement	varchar(20)	NN		IN('espaces', 'carte', 'virement', 'cheque')			carte
	date_paiement	TIMESTAMP	NN					2025-09-22 15:00:00
	numero_transaction	varchar(50)	N					TRX987654321
Employe	id_employe	int	NN	PK	>0	Auto_increme nte		601
	nom	varchar(50)	NN		Len(nom)>=2			Martin
	Prenom	varchar(40)	NN		Len(prenom)>=2			Lucas
	poste	varchar(30)	NN					Réceptionniste
	salaire	DECIMAL(8, 2)	N		>=0	Confidentiel	NULL	1800.00
	date_embauche	DATE	NN		<=CURRENT_DATE			2023-03-01
	actif	BOOLEAN	NN					true
	mail_pro	varchar(100)	NN	UNIQUE	FORMAT MAIL			Lucas.martin@gmail.com
	id_maintenance	int	NN	PK	>0	Auto_increme nter		801
	type_intervention	varchar(50)	NN					Climatisation

Maintenance	description	TEXT	NN					Verification du système
	priorite	int	NN		BETWEEN 1 AND 5	1=urgent 5=routine	3	2
	statut	varchar(20)	NN		IN('planifiee', 'en_cours', 'terminee', 'rep ortée')			en_cours
	date_prevue	TIMESTAMP	NN		>=CURRENT_TIM ESTAMP			2025-09-30 09:00:00
	date_realisatio n	TIMESTAMP	N		>=date_prevue			2025-09-30 11:00:00

5 Modélisation Conceptuelle - MCD

5.1 Schéma Entité-Association

Le Modèle Conceptuel de Données (MCD) représente graphiquement la structure logique de notre base de données avec les entités, leurs attributs et relations. Notre schéma illustre les flux métier principaux : Client → Réservation → Chambre → Facture → Paiement, ainsi que la gestion opérationnelle (Employé, Maintenance). Les cardinalités (1:1, 1:n, 0:n) définissent les règles métier et garantissent une représentation fidèle du fonctionnement hôtelier.

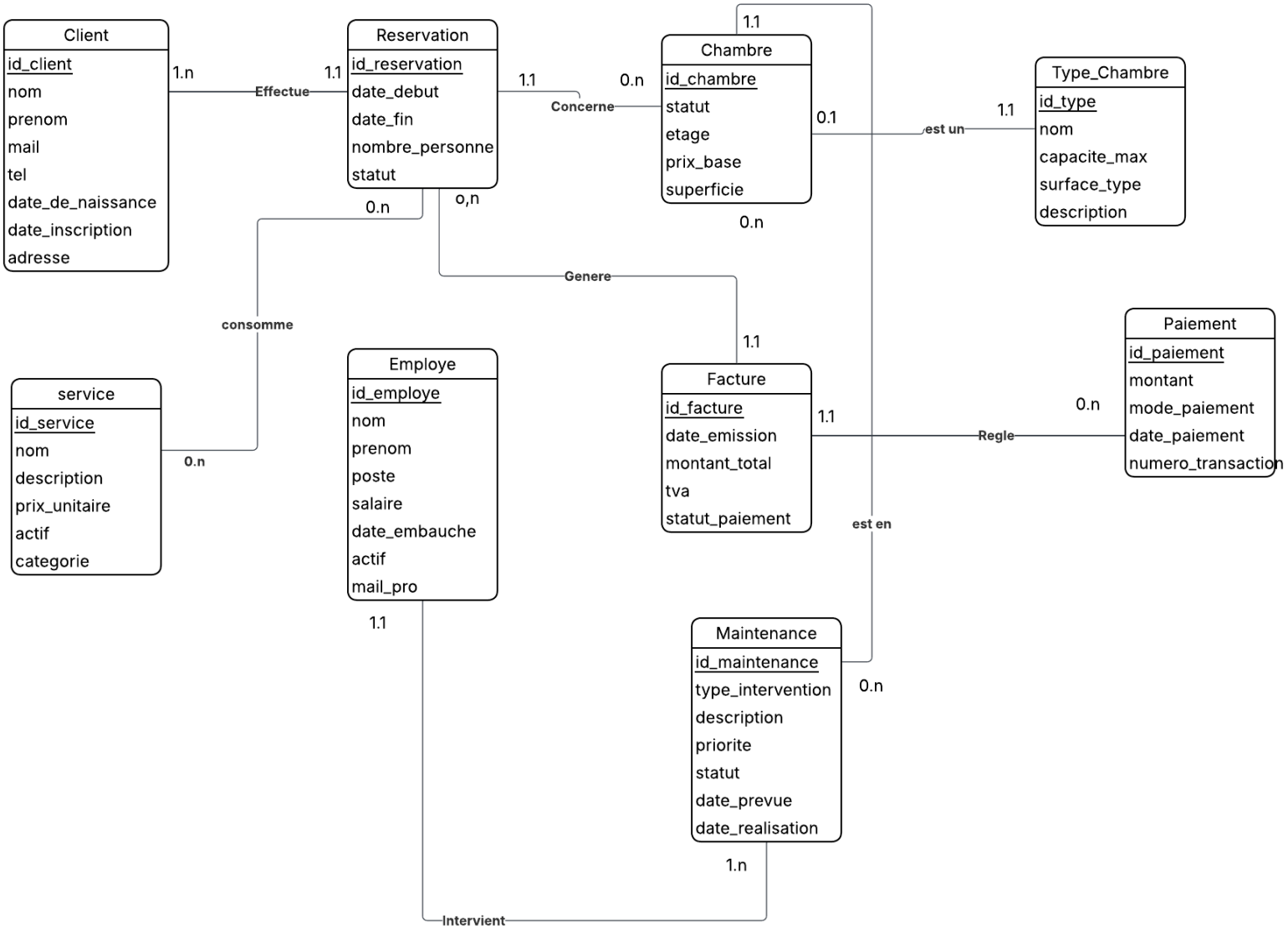

Figure 2: shéma E/A

6 Modélisation logique - MLD

6.1 Schéma Relationnel

Ce modèle transforme le MCD en structure relationnelle implémentable en base de données. Notre schéma comprend 10 tables principales avec leurs clés primaires et 1 table de liaison (Consomme) pour gérer la relation many-to-many. Les clés étrangères (préfixées par #) matérialisent les liens entre entités et assurent l'intégrité référentielle. Cette structure optimise les performances tout en respectant les formes normales et les contraintes métiers définis dans le MCD :

Client(id_client, nom, prenom, mail, tel, date_de_naissance, date_inscription, adresse)

Chambre(id_chambre, statut, etage, prix_base, superficie, #id_maintenance , #id_type)

Type_Chambre(id_type, nom, capacite_max, surface_type, description)

Reservation(id_reservation, date_debut, date_fin, nombre_personne, statut, #id_client, #id_chambre)

Service(id_service, nom, description, prix_unitaire, actif, categorie)

Facture(id_facture, date_emission, montant_total, tva, statut_paiement, #id_reservation, #id_paiement)

Paiement(id_paiement, mode_paiement, date_paiement, numero_transaction)

Employe(id_employe, nom, prenom, poste, salaire, date_embauche, actif, mail_pro, #id_maintenance)

Maintenance(id_maintenance, type_intervention, description, priorite, statut, date_prevue, date_realisation)

Consomme(#id_reservation, #id_service ,type_consommation)

7 Diagrammes applicatifs du protocole réseau

Cette partie présente les diagrammes applicatifs du protocole réseau d'un système de gestion hôtelière. L'architecture repose sur un client, un serveur applicatif et une base **PostgreSQL**, communiquant via **TCP**. Les échanges sont fiables **(ordre & livraison garantis par TCP)** et lisibles (messages **JSON**). Chaque scénario montre qui parle à qui, quels messages, dans quel ordre, et quelles actions BD (**SELECT/INSERT/UPDATE/DELETE**).

.

7.1 Scénario 1 : Vérification de disponibilité d'une chambre par scan de code

Objectif : Le client arrive, on marque la chambre comme occupée

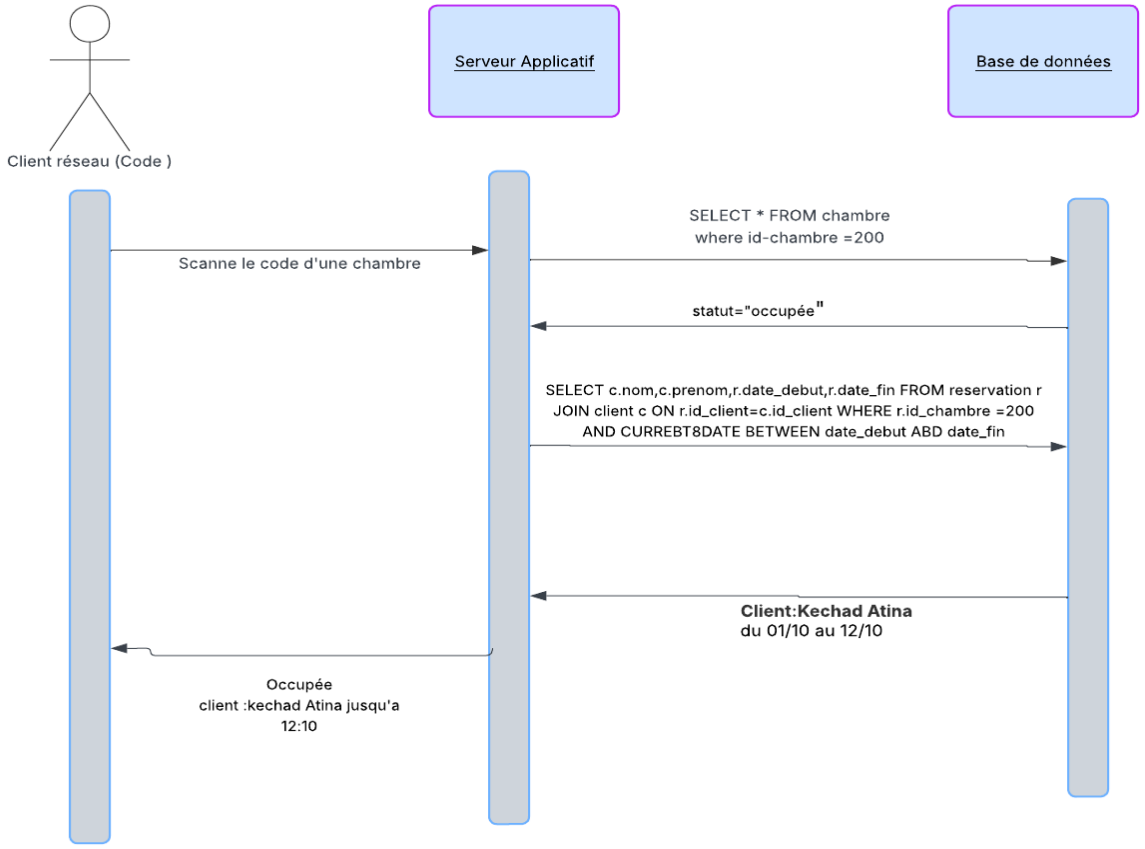

Figure 3 : Ce diagramme illustre le processus de consultation d'une chambre d'hôtel par scan de code QR

7.2 Scénario 2 : Gestion d'une maintenance urgente

Objectif : Illustrer le flux de communication entre le client, le serveur applicatif et la base de données **lors de la gestion d'une maintenance d'urgence**

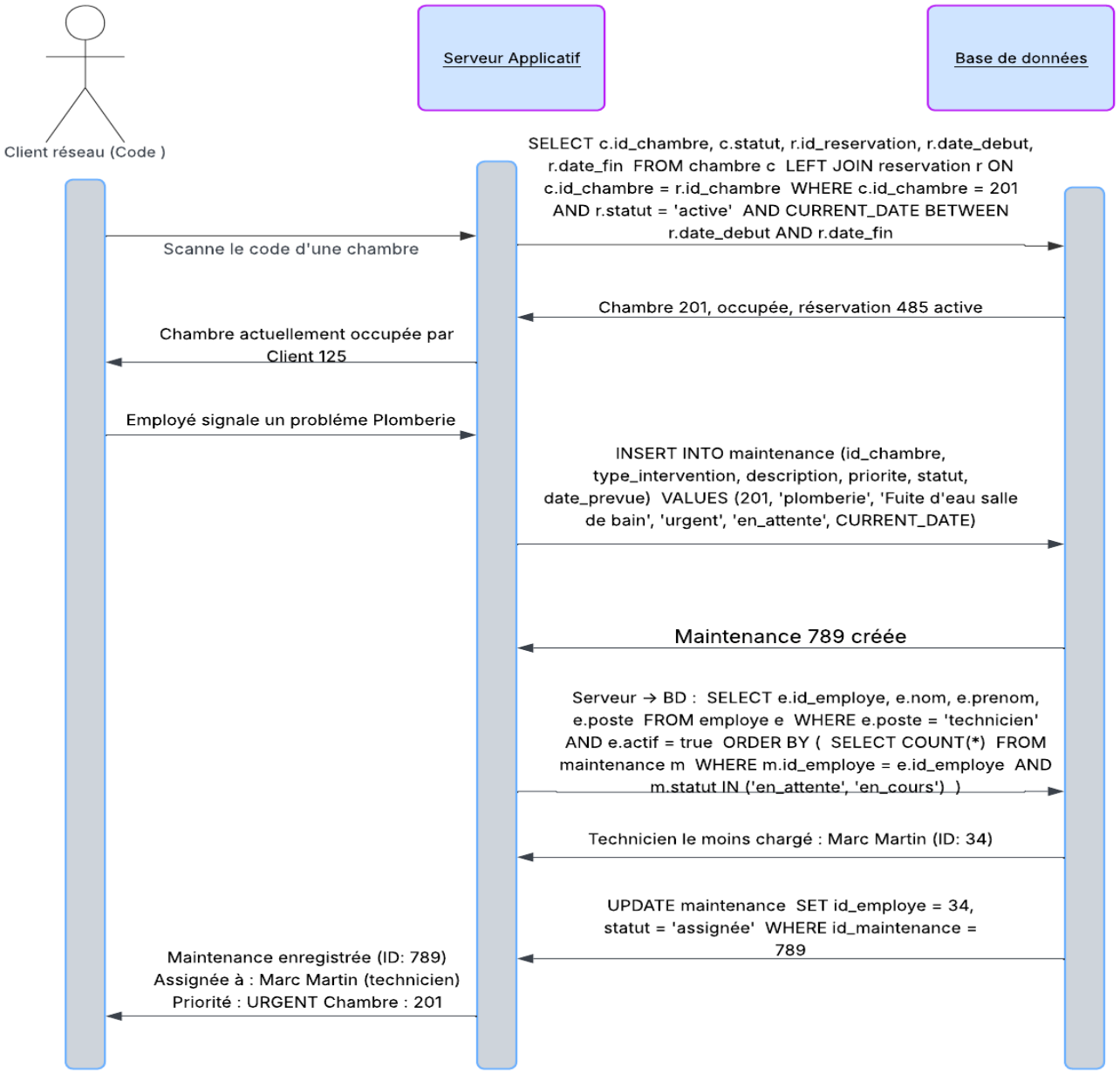

Figure 4: Processus de signalement et d'attribution automatique d'une maintenance d'urgence pour une chambre d'hôtel

7.3 Scénario 3 : Check-out et facturation complète

Objectif : Automatiser le processus de départ d'un client en calculant la facture totale, enregistrant le paiement et libérant la chambre pour nettoyage

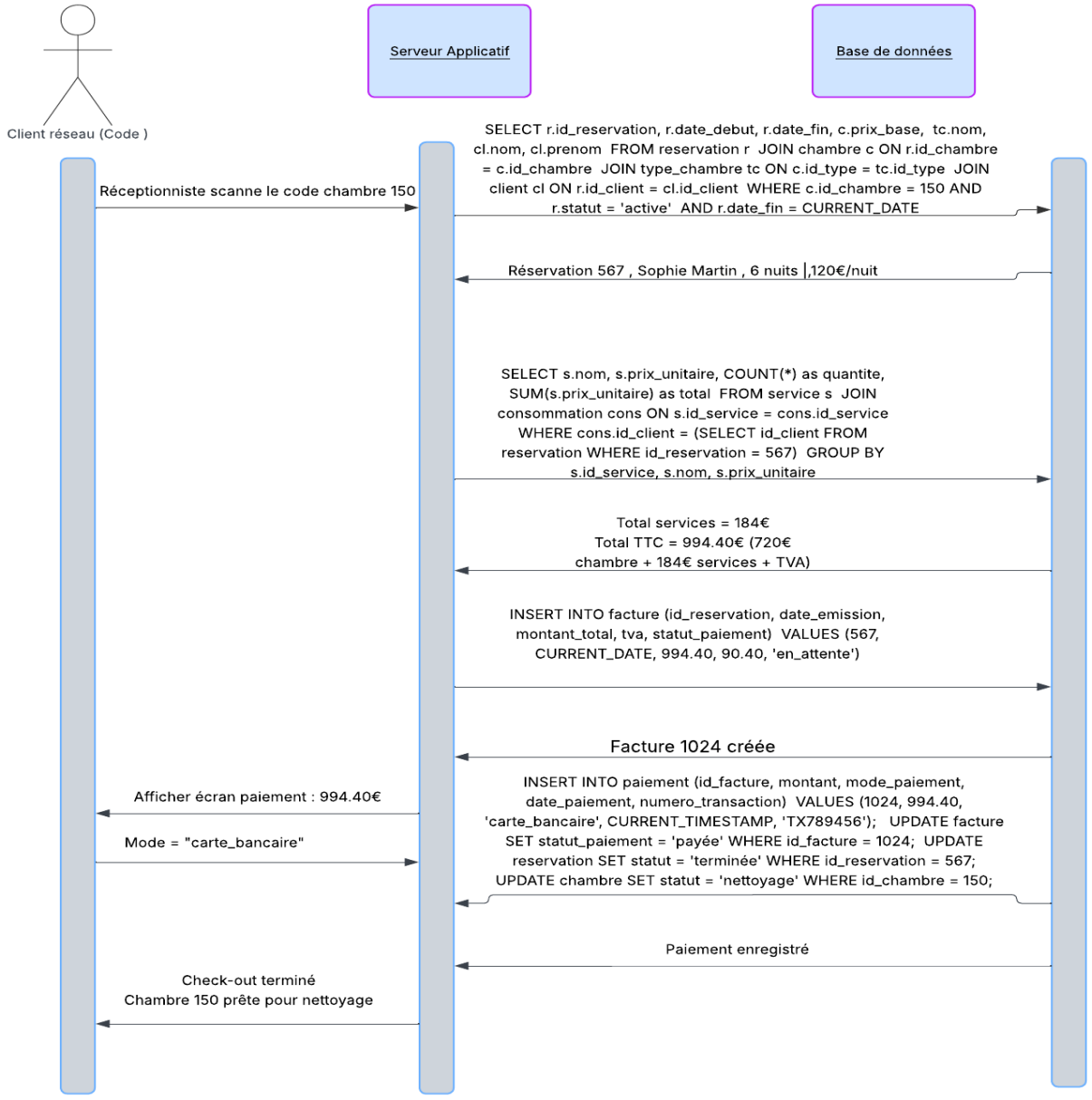

Figure 5: Processus de check-out d'un client avec génération de facture, paiement par carte bancaire et libération de la chambre

7.4 Scénario 4 : Modification d'une réservation existante

Objectif : Permettre la modification des dates d'une réservation existante tout en vérifiant automatiquement la disponibilité de la chambre pour la nouvelle période.

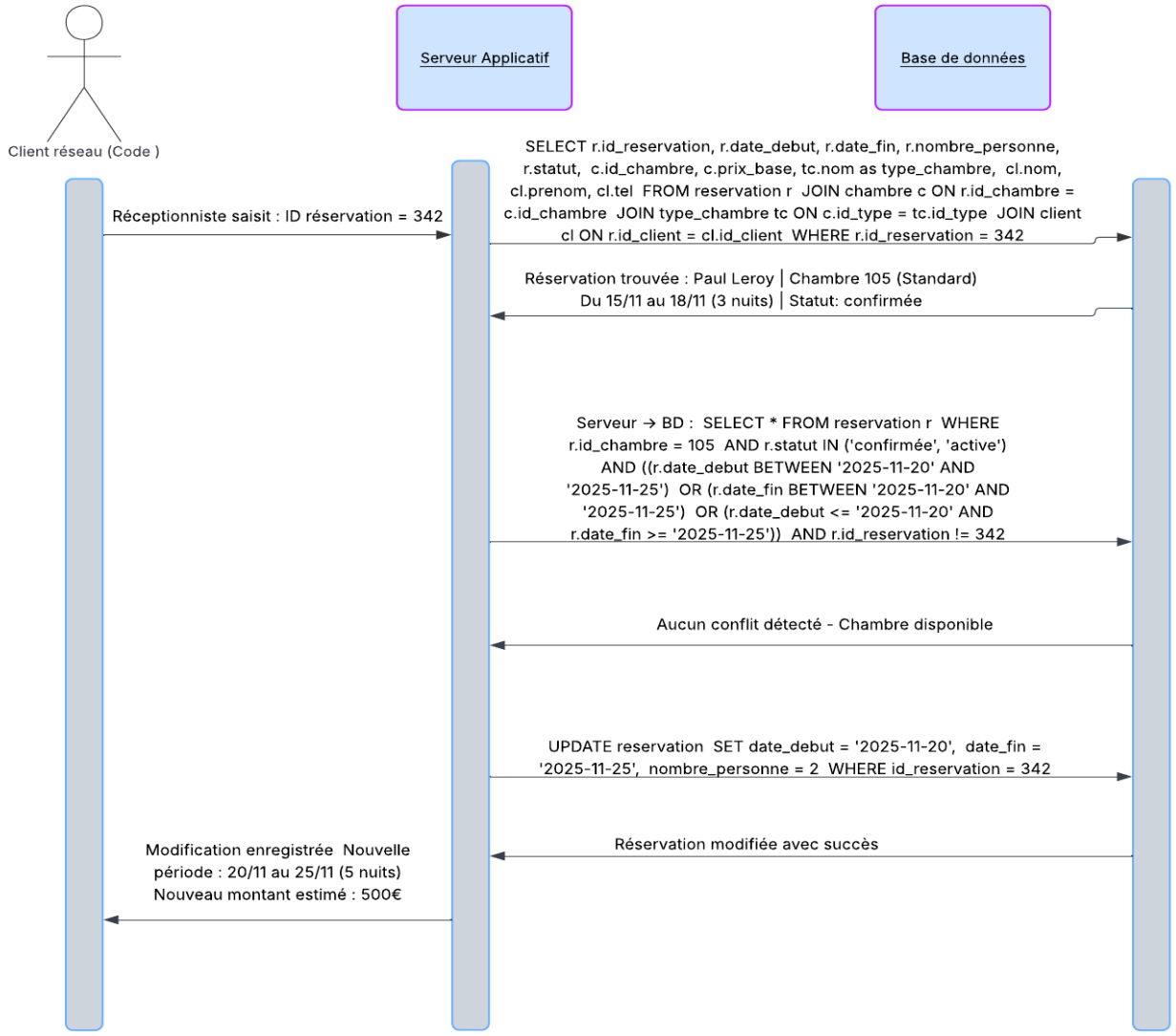

Figure 6: Processus de modification d'une réservation avec vérification de disponibilité et mise à jour des dates

"Grâce à ce mécanisme, le protocole reste robuste, prévisible et surtout facile à déboguer côté client.".

8 Exemples Représentatifs du Jeu de Données

Cette section présente un jeu de données complet et cohérent qui sera utilisé pour les tests du système de gestion hôtelière. Ces données respectent toutes les contraintes définies dans le dictionnaire de données et illustrent les différents scénarios d'usage du système.

Table Client

id-client	nom	prenom	mail	tel	date_naiss	date_inscription	adresse
101	Dupont	Alice	alice.dupont@gmail.com	0611345078	1990-04-15	2025-09-20 14:35:00	5 Avenue Anatole, 75007 Paris
102	Martin	Bruno	Bruno.martin@yahoo.fr	0712456789	1985-11-22	2025-09-21 10:20:00	12 Rue de la Paix, 69001 Lyon
103	Bernard	Claire	Claire.bernard@outlook.fr	0626789456	1992-07-30	2025-09-22 16 :45 :00	8 Boulvard Victor Hugo,13001 Marseille
104	Petit	David	David.petit@free.fr	0734567890	1988-03-12	2025-09-23 09:15:00	15 Rue Gambetta, 33000 Bordeaux
105	Rousseau	Emma	Emma.rousseau@laposte.net	0645123789	1995-09-05	2025-09-24 11:30:00	22 Avenue Des Champs, 31000 Toulouse
106	Robert	François	Francois.robert@sfr.fr	0756234901	1982-12-18	2025-09-25 14:00:00	7 Place de la République, 5900 Lille
107	Simon	Gabrielle	Gabrielle.simon@orange.fr	0667890123	1993-06-21	2025-09-26 10:45:00	30 Rue Saint-Jacques, 44000 Nantes

Table Chambre

id-chambre	statut	etage	prix_base	superficie	id_mainteance	Id_type
201	libre	2	120 .00	25.50	801	201
202	occupée	1	90 .00	20.00	802	202
203	maintenance	3	150.00	30.00	803	205
204	libre	1	85.00	18.00	804	203
205	libre	3	180.00	35.00	805	201
206	occupée	1	95.00	22.00	806	202
207	nettoyage	3	130.00	28.00	801	204
208	libre	2	110.00	24.00	802	205

Table Type_chambre

id_type	nom	capacite_max	surface_type	description
201	Suite	4	30.00	Suite spacieuse avec vue sur mer
202	Double	2	20.00	Chambre double standard
203	Simple	1	15.00	Chambre simple confortable
204	Familiale	5	40.00	Chambre familiale avec deux lits doubles
205	Deluxe	3	35.00	Chambre de luxe avec balcon privé

Table Réservation

id_reservation	date_debut	date_fin	nombre_personnes	statut	id_client	id_chambre
1	2025-10-01	2025-10-05	3	confirmée	101	201
2	2025-10-03	2025-10-07	2	en attente	102	202
3	2025-10-10	2025-10-15	1	Confirmée	103	204
4	2025-10-12	2025-10-18	4	Confirmée	104	205
5	2025-10-05	2025-10-08	2	Annulée	105	208
6	2025-10-20	2025-10-25	2	Confirmée	106	206
7	2025-10-15	2025-10-20	3	En_attente	107	207

Table Service

id_service	nom	description	prix_unitaire	actif	categorie
301	Massage	Massage relaxant d'une heure	60.00	true	wellness
302	Petit-déjeuner	Service en chambre	15.00	true	restauration
303	Navette aéroport	Transport aller-retour	45.00	true	transport
304	Spa	Accès spa et jacuzzi	35.00	true	wellness
305	Dîner gastronomique	Menu 3 plats au restaurant	55.00	true	restauration
306	Blanchisserie	Service pressing 24h	20.00	true	autre

307	Location vélo	Vélo pour la journée	12.00	true	transport

Table Facture

id_facture	date_emission	montant_total	tva	statut_paiement	mode_paiement	id-reservation	id_paiement
501	2025-09-22	350.50	70.00	payee	carte	1	601
502	2025-09-23	180.00	36.00	impayee	virement	2	
503	2025-10-10	125.00	25.00	payee	especes	3	602
504	2025-10-12	890.00	178.00	partiellement_payee	carte	4	603
505	2025-10-08	50.00	10.00	remboursee	carte	5	604
506	2025-10-20	525.00	105.00	payee	virement	6	605

Table Paiement

id_paiement	mode_paiement	date_paiement	numero_transaction
601	carte	2025-09-22 15:00:00	TRX987654321
602	virement	2025-10-10 16:30:00	TRX123456789
603	carte	2025-10-12 14:20:00	TRX456789123
604	cheque	2025-10-08 11:45:00	TRX789123456
605	virement	2025-10-20 18:00:00	TRX321654987

Table Employé

id_employe	nom	prenom	poste	salaire	date_embauche	actif	mail_pro	id_maintenace
601	Martin	Lucas	Réceptionniste	1800.00	2023-03-01	true	Lucas.martin@gmail.com	
602	Leroy	Sophie	Femme de ménage	1600.00	2024-01-15	true	Sophie.leroy@gmail.com	
603	Dubois	Pierre	Chef cuisinier	2500.00	2022-06-10	true	Pierre.dubois@gmail.com	
604	Moreau	Julie	Réception niste	1850.00	2023-09-20	true	Julie.moreau@gmail.com	
605	Lambert	Thomas	Technicien	2100.00	2021-11-05	true	Thomas.lambert@gmail.com	805
606	Garnier	Marie	Femme de ménage	1650.00	2024-03-12	true	Marie.garnier@gmail.com	
607	Roux	Antoine	Serveur	1750.00	2023-07-18	true	Antoine.roux@gmail.com	

Table Maintenance

id_maintenance	Type_intervention	description	priorite	statut	date_ prevue	date_ realisation
801	Climatisation	Vérification du système	2	en_cours	2025-09-30 09:00:00	NULL
802	Plomberie	Fuite robinet salle bain	1	terminne	2025-09-28 10:00:00	2025-09-28 15:30:00
803	Electricité	Remplacement ampoules	4	planifiee	2025-10-05 14:00:00	NULL
804	Peinture	Retouche murs	3	planifiee	2025-10-08 08:00:00	NULL
805	Plomberie	Débouchage canalisation	1	terminee	2025-09-29 11:00:00	2025-09-29 16:00:00
806	Climatisation	Changement filtres	3	reportee	2025-10-01 10:00:00	NULL

Table Consomme

id_reservation	id_service	type_consommation
1	301	Massage relaxant
2	302	Petit-déjeuner en chambre
3	303	Navette aéroport
4	307	Location vélo journée
5	304	Accès spa et jacuzzi
6	305	Menu dégustation

9 Analyse des Données – 10 Requêtes SQL sur la Base de l'Hôtel :

9.1 Questions Statistiques

1. Combien de clients distincts ont effectué au moins une réservation ?

```
34
35   SELECT COUNT(DISTINCT id_client) AS nombre_clients_avec_reservation
36  FROM Reservation;
37
38
39
```

Data Output | Messages | Notifications

Showing rows: 1 to 1

	nombre_clients_avec_reservation bigint
1	7

2. Combien de réservations ont été faites pour la chambre n°499 (ou pour une chambre donnée) ?

```sql
SELECT COUNT(id_reservation) AS nombre_reservations_chambre_499
FROM Reservation
WHERE id_chambre = 449;
```

Data Output | Messages | Notifications

	nombre_reservations_chambre_499 bigint
1	0

3. Quel est le montant total facturé par jour (sur une période donnée) ?

```sql
SELECT
   date_emission::date AS date_facturation_jour,
   SUM(montant_total) AS total_facture_jour
FROM Facture
GROUP BY date_emission::date
ORDER BY date_emission::date;
```

Data Output | Messages | Notifications

	date_facturation_jour date	total_facture_jour numeric
1	2025-09-22	350.50
2	2025-09-25	180.00
3	2025-10-08	50.00
4	2025-10-10	125.00
5	2025-10-12	890.00
6	2025-10-20	525.00

4. Quel est le prix moyen des chambres par étage ?

```sql
SELECT etage, ROUND(AVG(prix_base), 2) AS prix_moyen
FROM Chambre
GROUP BY etage
ORDER BY etage;
```

Data Output | Messages | Notifications — Showing rows: 1

	etage integer	prix_moyen numeric
1	1	90.00
2	2	115.00
3	3	153.33

9.2 Questions de Sélection

1.Quels clients ont une réservation qui inclut la date 3 octobre 2025 ?

```sql
SELECT C.nom, C.prenom, C.mail
FROM Client C
JOIN Reservation R ON C.id_client = R.id_client
WHERE '2025-10-03' BETWEEN R.date_debut AND R.date_fin;
```

Data Output | Messages | Notifications

	nom character varying (50)	prenom character varying (40)	mail character varying (100)
1	Dupont	Alice	alice.dupont@gmail.com
2	Martin	Bruno	Bruno.martin@yahoo.fr

2.Quelles sont les chambres actuellement disponibles (statut = libre) et leur prix ?

```sql
SELECT id_chambre, etage, prix_base, superficie
FROM Chambre
WHERE statut = 'libre';
```

Data Output | Messages | Notifications

	id_chambre [PK] integer	etage integer	prix_base numeric (8,2)	superficie numeric (5,2)
1	201	2	120.00	25.50
2	204	1	85.00	18.00
3	205	3	180.00	35.00
4	208	2	110.00	24.00

3.Quelles factures sont encore impayées, avec le nom du client et le montant ?

Hypothèse : Le statut d'une facture impayée est `'impayee'`.

```sql
SELECT F.id_facture, C.nom, C.prenom, F.montant_total
FROM Facture F
JOIN Reservation R ON F.id_reservation = R.id_reservation
JOIN Client C ON R.id_client = C.id_client
WHERE F.statut_paiement = 'impayee';
```

Data Output | Messages | Notifications

Showing rows: 1 to 1

	id_facture integer	nom character varying (50)	prenom character varying (40)	montant_total numeric (8,2)
1	502	Martin	Bruno	180.00

4. Quelles interventions de maintenance prioritaires n'ont pas encore été réalisées, et quel employé y est affecté ?

```sql
SELECT
   M.id_maintenance,
   M.type_intervention,
   M.description,
   E.nom AS nom_employe,
   E.prenom AS prenom_employe
FROM Maintenance M
INNER JOIN Employe E ON E.id_maintenance = M.id_maintenance
WHERE M.statut IN ('en_cours', 'planifiee')
  AND M.priorite = 5;
```

Data Output | Messages | Notifications

id_maintenance integer	type_intervention character varying (50)	description text	nom_employe character varying (50)	prenom_employe character varying (40)

9.3 Questions "Réseau / Jeu de données"

1.Scénario "Check-out" : Demande pour la réservation X (ex. : ID 501)

```sql
-- DML 1 : Mettre à jour la chambre associée
UPDATE Chambre
SET statut = 'libre'
WHERE id_chambre = (SELECT id_chambre FROM Reservation WHERE id_reservation = 501);

-- DML 2 : Mettre à jour le statut de la Réservation
UPDATE Reservation
SET statut = 'terminee'
WHERE id_reservation = 501;

SELECT 'CHECKOUT_SUCCESS' AS code_reponse, 'Le check-out pour la réservation 501 est terminé.' AS message;
```

Data Output | Messages | Notifications

Showing rows: 1 to 1 Page No: 1

	code_reponse text	message text
1	CHECKOUT_SUCCESS	Le check-out pour la réservation 501 est terminé.

2.Scénario "Liste des réservations en cours" pour un client (ex. : ID 201)

```sql
SELECT
  R.id_reservation,
  R.date_debut,
  R.date_fin,
  R.statut,
  Ch.id_chambre AS chambre_id
FROM Reservation R
JOIN Chambre Ch ON R.id_chambre = Ch.id_chambre
WHERE R.id_client = 101
  AND R.statut = 'confirmée';
```

Data Output | Messages | Notifications

Showing rows: 1 to 1

	id_reservation integer	date_debut date	date_fin date	statut character varying (20)	chambre_id integer
1	1	2025-10-01	2025-10-...	confirmée	201

10 Résultats des tests Client/Réseau :

Cette section présente les trois tests de validation du protocole réseau, démontrant l'interopérabilité entre les composants du système.

10.1 TEST 1 : Client Python → Netcat en mode Serveur

Objectif : Valider l'envoi de messages JSON par le client Python.

Procédure : Netcat lancé en serveur (`nc -l -p 8080`), puis exécution du client Python.

Résultat : Message {"type": "countClients"} correctement reçu par Netcat. La réponse {"resultat":4,"message":"Succès : 4 clients trouvés.","status":"ok"} est bien reçue par le client.

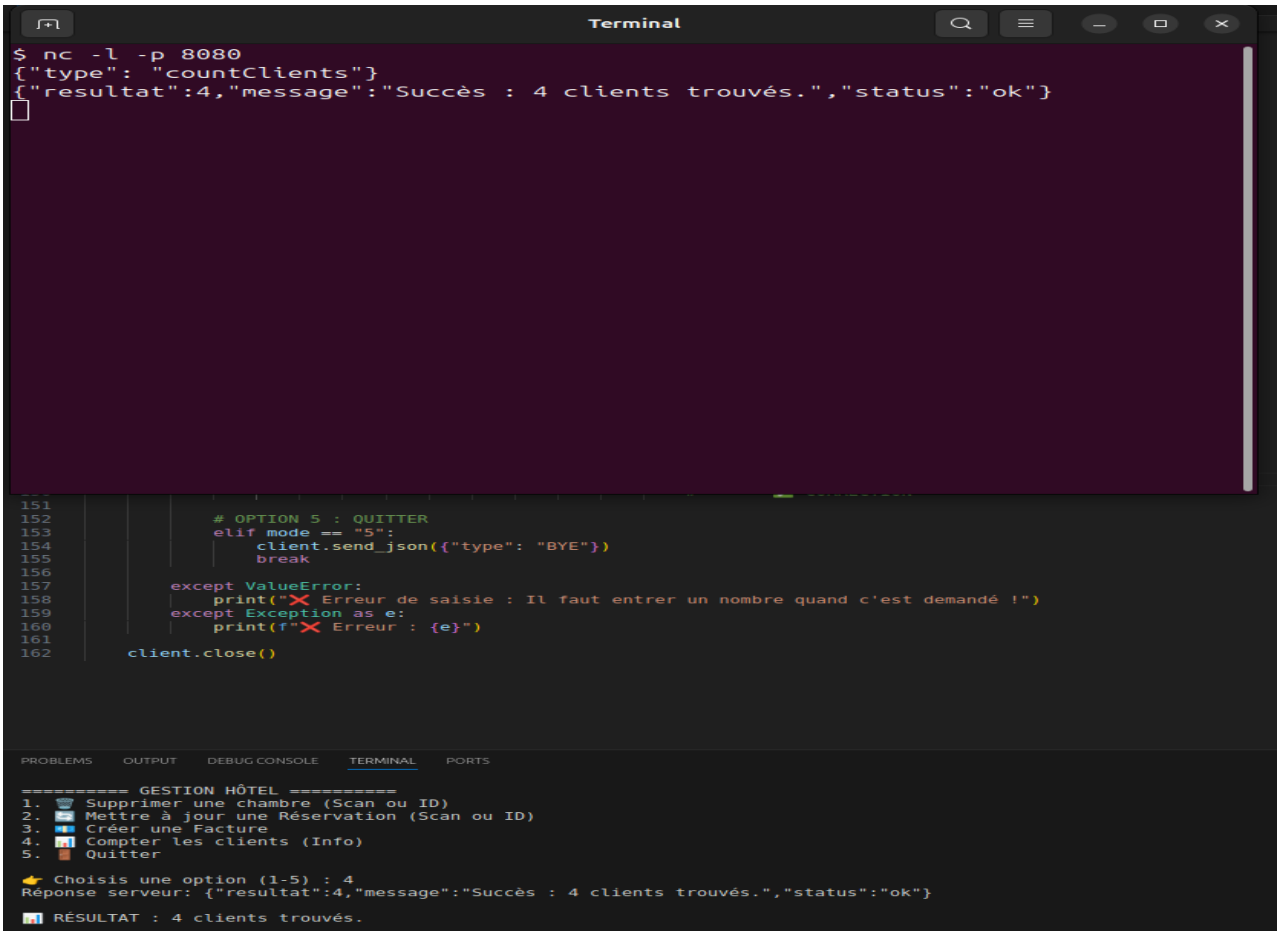

Figure 7 : Capture TEST 1 - Terminal Netcat recevant le message du client Python

10.2 TEST 2 : Netcat en mode Client → Serveur Java

Objectif : Valider le traitement des requêtes par le serveur Java et l'interaction avec PostgreSQL.

Procédure : Serveur Java lancé (`java -cp .:json-20240303.jar:postgresql-42.7.1.jar server.ServerTCP`), connexion via Netcat (`nc 127.0.0.1 8080`), envoi de `{"type":"countClients"}`.

Résultat : Requête SQL exécutée, réponse `{"status":"ok"}` renvoyée, 4 clients comptés.

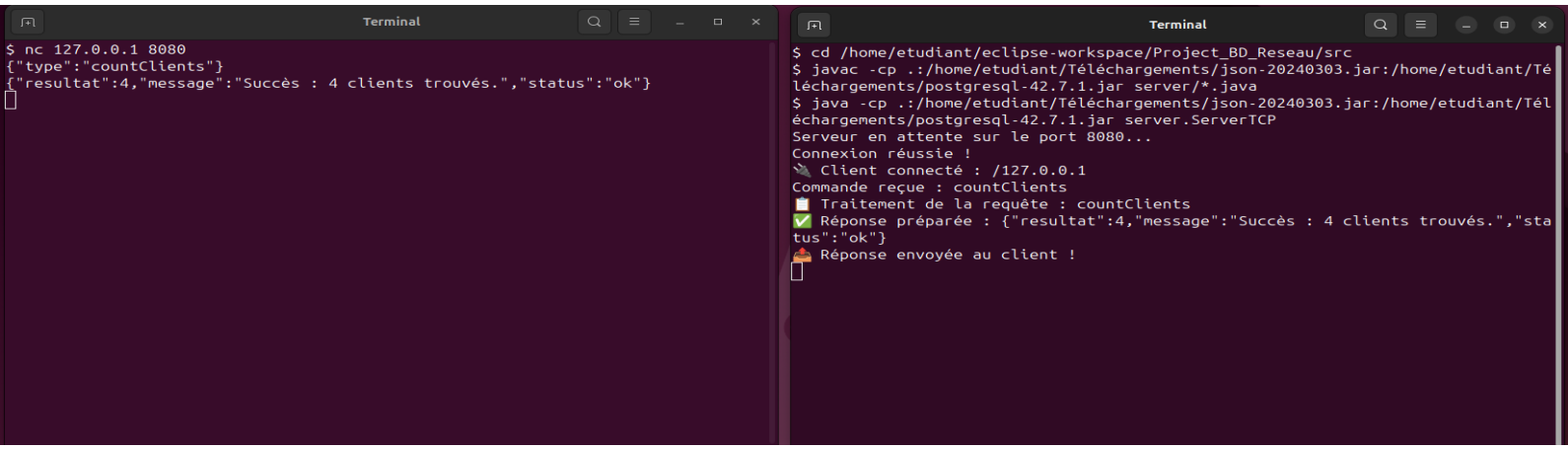

Figure 8:Capture TEST 2 - Serveur Java traitant la requête Netcat

10.3 TEST 3 : Client Python ↔ Serveur Java (Communication complète)

Objectif : Valider la communication bidirectionnelle complète entre le client Python et le serveur Java avec accès à la base de données.

Procédure :

- Lancement du serveur Java sur le port 8080
- Exécution du client Python HotelClient.py

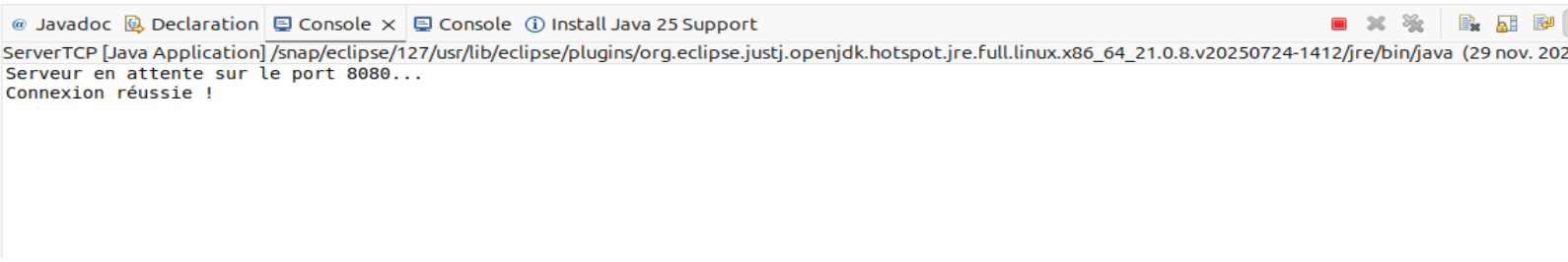

Figure 9 : Capture TEST 3 - Console serveur Java

Figure 10 : Capture TEST 3 - Terminal client Python

Figure 11 : Console serveur Java – traitement des requêtes client et interactions avec PostgreSQL.

11 Gestion du port utilisé par le client et le serveur

On a ajouté une fonctionnalité qui permet d'utiliser soit un **port choisi par l'utilisateur**, soit un **port par défaut** lorsque aucun port n'est indiqué.

Concrètement :

- Quand on lance le programme avec un port, le client et le serveur utilisent ce port pour communiquer.
- Quand on ne précise rien, on reçoit un message qui informe qu'aucun port n'a été donné, et la connexion se fait automatiquement sur le port par défaut (8080).

Les captures ci-dessous montrent les deux situations : la connexion avec un port saisi par l'utilisateur (9000) et la connexion en utilisant le port par défaut.

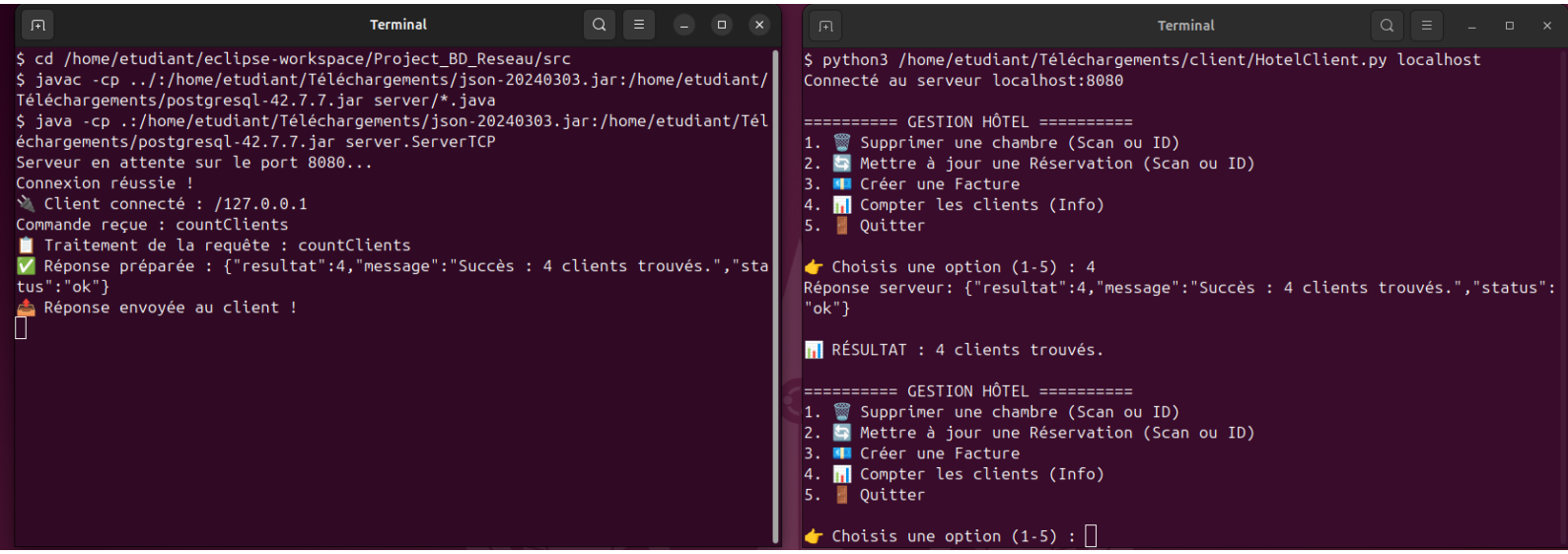

Figure 12 : Connexion client–serveur utilisant automatiquement le port par défaut (8080)

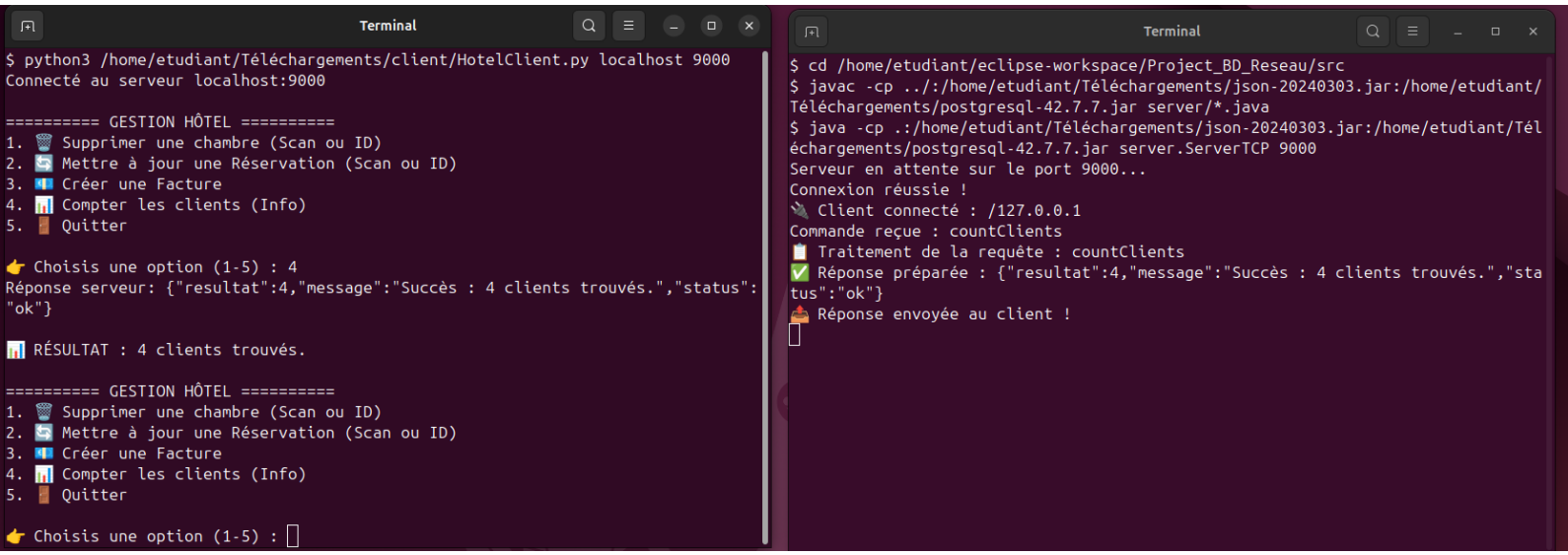

Figure 13 : Connexion client–serveur avec un port saisi par l'utilisateur (9000)