

MSPR n°4 - Projet Hôtel Neptune



l'école d'ingénierie
informatique



l'école [tech]
de l'expertise digitale

Membre du groupe :

Antoine BERTHE ; Clément SALINGUE ; Tom CLEMENT ; Annalia PRIEUR

Sommaire :

- Rappel du sujet.
- Organisation.
- Descriptions fonctionnelles.
- Maquette.
- Code.
- Amélioration.
- Conclusion.
- Démonstration du site.

Rappel du sujet :

Contexte :

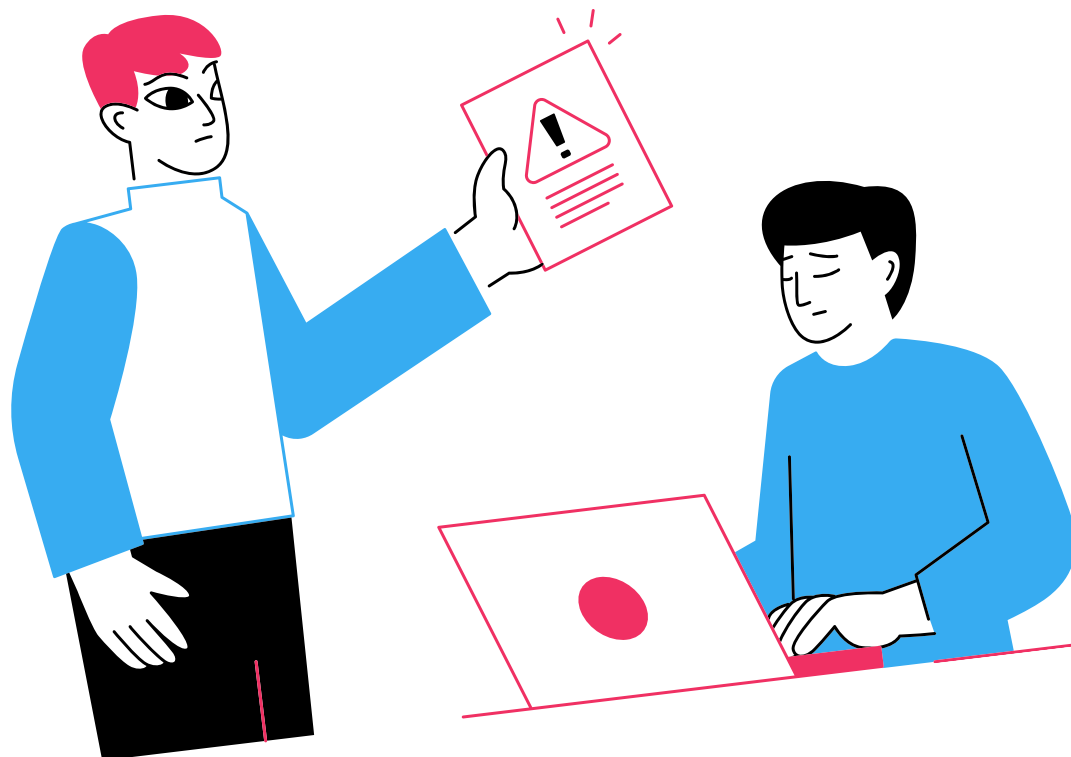
Le projet porte sur la digitalisation des réservations de l'Hôtel Neptune qui est un établissement situé à Arras. En développant un site web dynamique en PHP avec une base de données MariaDB.

Ce site comprend une interface Admin pour gérer les chambres, les clients ou les réservations. Il comprend également une interface Clients pour rechercher, réserver ou gérer leurs profils.

Le but est de remplacer les méthodes manuelles de gestion actuelles par une solution numérique plus rapide, fiable et intuitive.

Problématique :

Comment créer un site web qui facilite la gestion des chambres, clients et réservations, tout en garantissant une expérience utilisateur fluide pour les clients ?



Les sous problèmes :

N°1 : La Gestion des chambres.

N°2 : La Gestion des clients.

N°3 : La Gestion des réservations.

N°4 : Les Fonctionnalités avancées.

N°5 : La Sécurité et performance.



Organisation :

Utilisation
Discord

The screenshot shows the GitHub repository page for 'NeptuneHostel' by user 'mevyona'. The repository is public and has 1 branch (main) and 0 tags. It has 1 watch, 0 forks, and 0 stars. The repository contains a folder 'public' and 'src', and files '.env', '.gitignore', 'composer.json', and 'db.sql'. The 'About' section states 'No description, website, or topics provided.' The 'Releases' section states 'No releases published' with a link to 'Create a new release'. The 'Packages' section states 'No packages published' with a link to 'Publish your first package'. The 'Contributors' section shows 4 contributors.

NeptuneHostel Public

Watch 1 Fork 0 Star 0

main 1 Branch 0 Tags

Go to file Add file Code

mevyona v1 a2c8b2f · 3 months ago 99 Commits

public	v1	3 months ago
src	v1	3 months ago
templates	v1	3 months ago
.env	db	3 months ago
.gitignore	CSS Comments & Payments	3 months ago
composer.json	Errors Styling	3 months ago
db.sql	DB Update	3 months ago

README

About

No description, website, or topics provided.

Activity

0 stars

1 watching

0 forks

Report repository

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

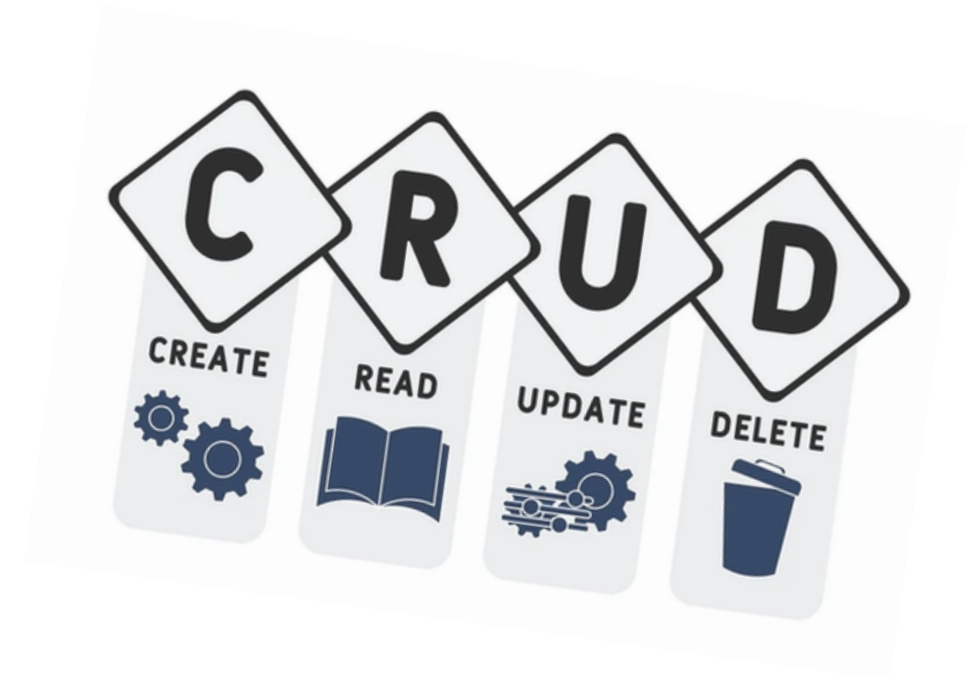
Contributors 4

<https://github.com/mevyona/NeptuneHostel>

Description fonctionnelles :

Fonctionnalités principales :

- Gestion des chambres (CRUD + gestion de photos associées).
- Gestion des clients (CRUD).
- Réservations (visualisation, filtrage, annulation).
- Améliorations (e-mail de confirmation, factures PDF).

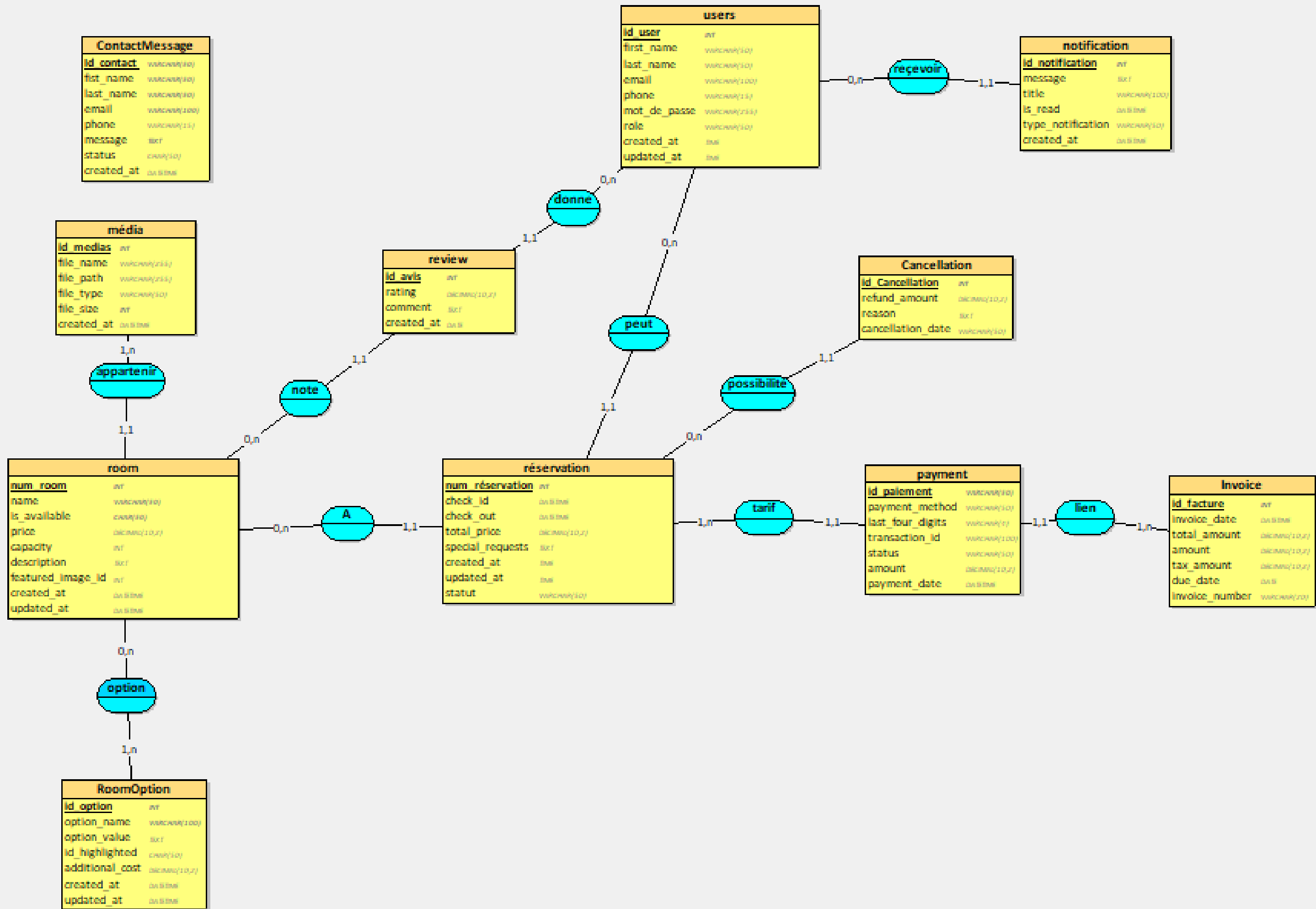


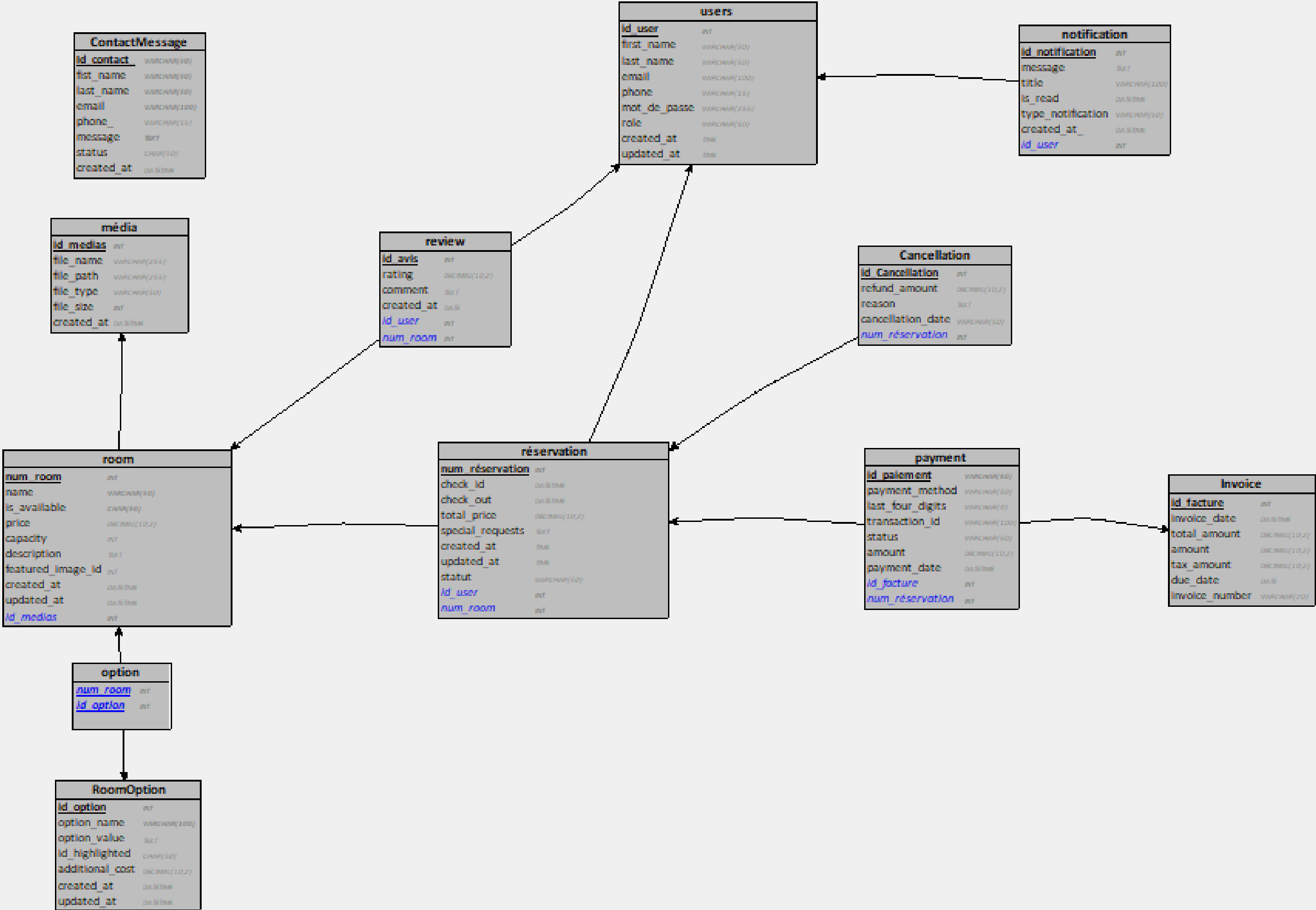
Solutions proposées :

On à du réaliser un site web qui permet de gérer efficacement les réservations d'un hôtel. On à développer des modules distincts pour chaque fonctionnalité en adoptant une méthodologie modulaire. Pour pouvoir réaliser ce site web, le code se fera sur un environnement local avec un serveur PHP et MySQL.

Ce projet à été découpe en plusieurs étapes. On à réaliser ce projet pour permettre l'automatisation des tâches de gestions pour pouvoir gagner en efficacité et en fiabilité. Ce groupe est composé de 2 devs ainsi que de 2 réseaux.

MCD








Figma

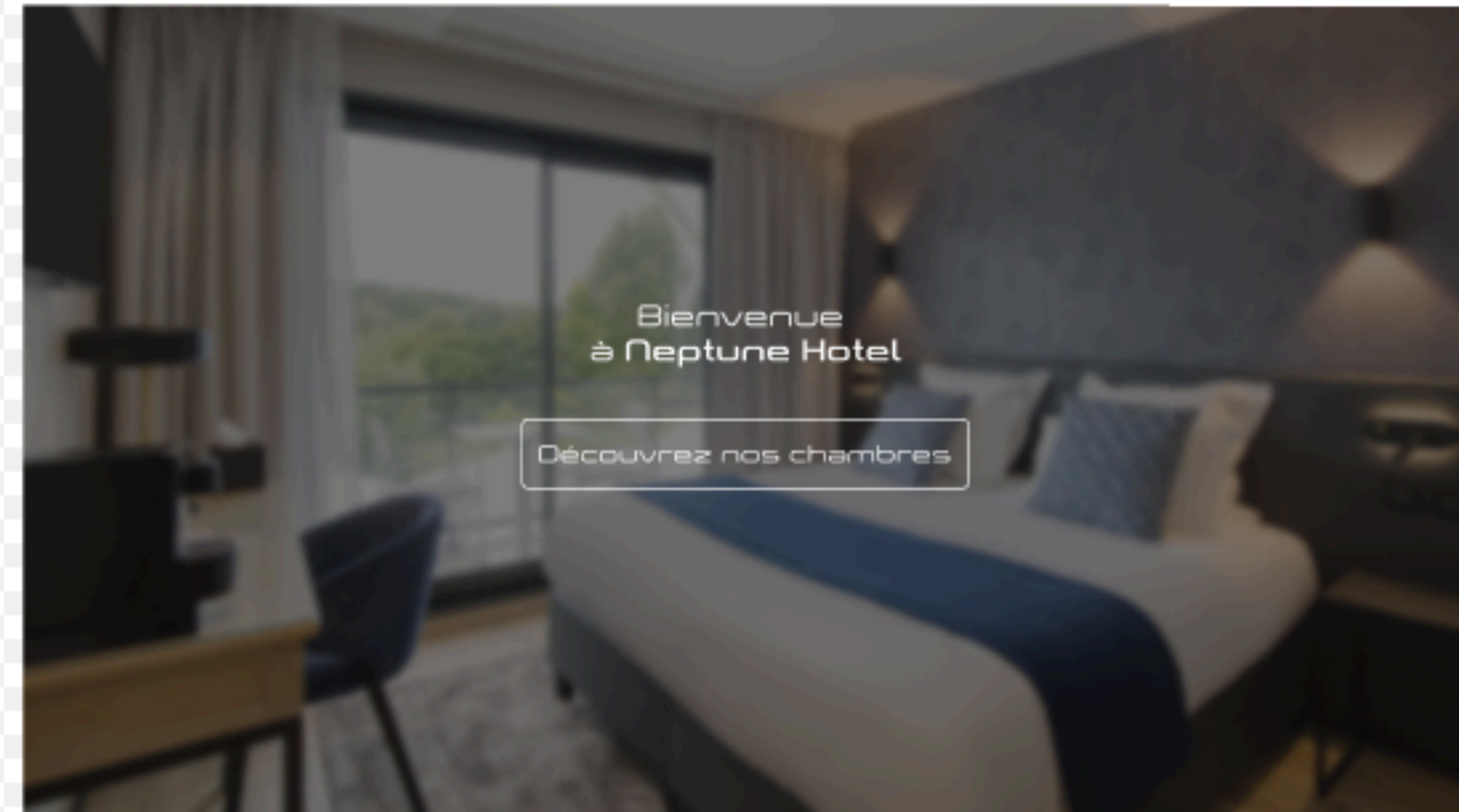
MAQUETTES :

Page d'accueil :

Page d'Accueil



Réserver 



L'Hôtel Neptune

L'Hôtel Neptune vous invite à découvrir un havre de sérénité alliant confort, élégance et service personnalisé. Que vous soyez en voyage d'affaires ou en quête de détente, notre établissement vous propose des chambres spacieuses et chaleureusement décorées, dotées de toutes les commodités modernes.

Profitez de notre atmosphère conviviale et laissez-vous séduire par notre emplacement privilégié à proximité des principales attractions de la région. Chaque séjour à l'Hôtel Neptune est une expérience unique où bien-être et hospitalité se rencontrent.

HÔTEL NEPTUNE

2 RUE DU DEPOT
62120 ARRAS
FRANCE

05 00 00 00 00



Nom

Prenom

Tel

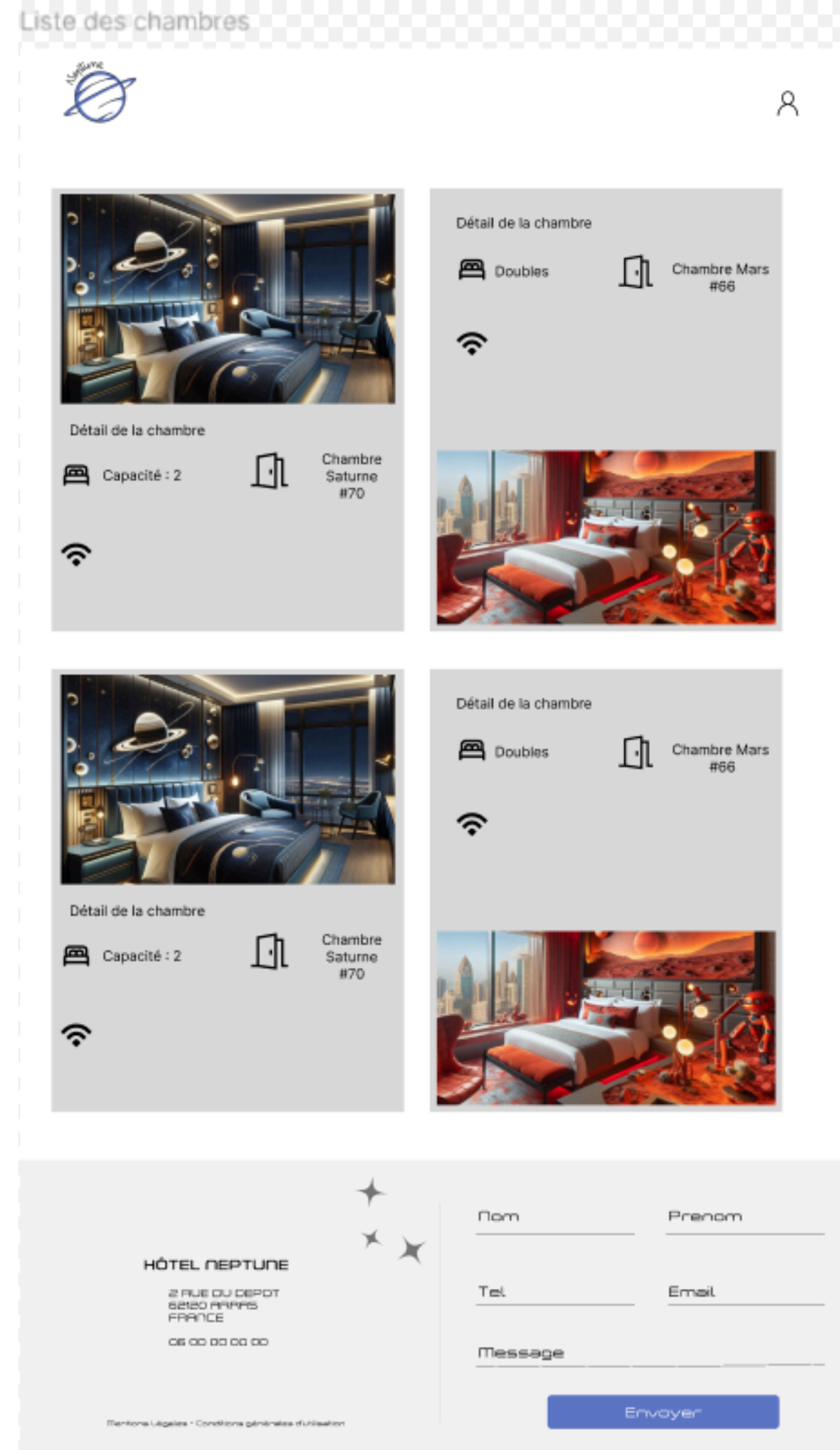
Email

Message

Envoyer

Mentions Légales - Conditions générales d'utilisation

Page pour la liste des chambres :



Formulaire de réservation :

Formulaire de réservation

Formulaire de réservation

Completez le formulaire

Nom complet

Nom

Prenom

Numero de tél:

(000) 000 - 000

Heures/minutes
HH:MM

Arrivée: date-heures

MM - DD - YYYY

Heures/minutes
HH:MM

Départ: dates-heures

MM - DD - YYYY

Options

Nombres d'adultes

Nombres d'enfants

Envoyer

Réinitialiser

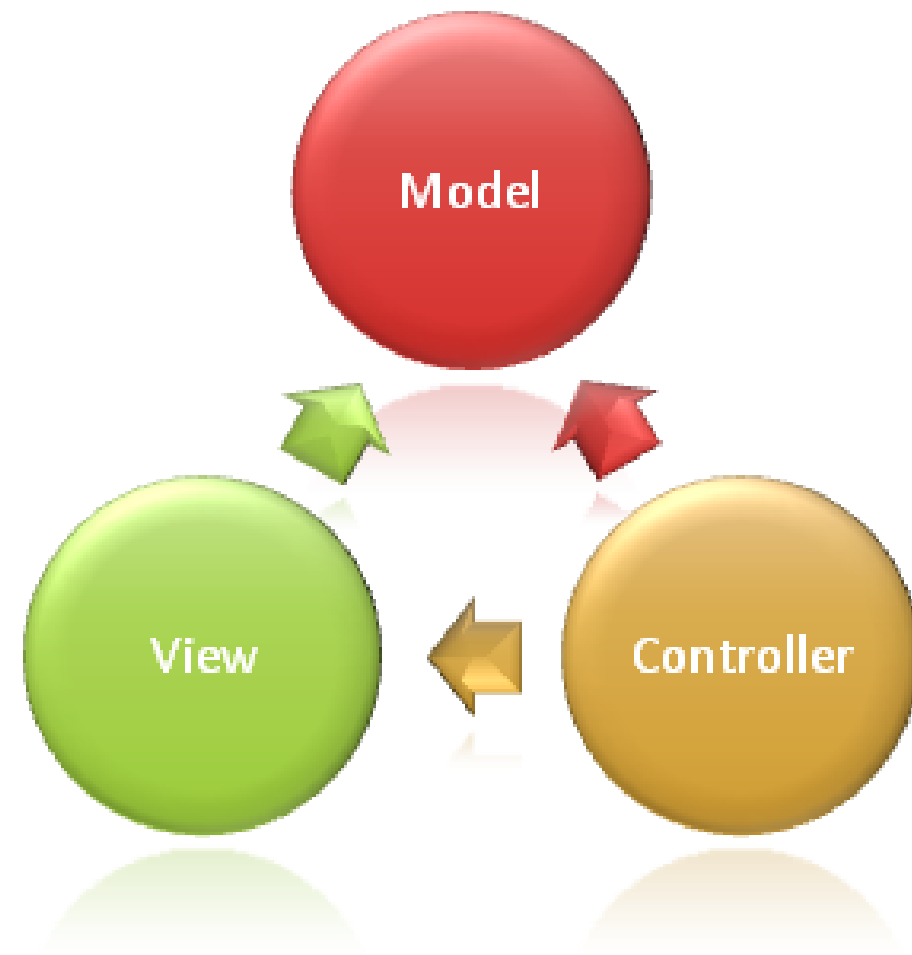
Organisation du groupe :

- . **Annalia** : Réalisation des maquettes ; Développement backend ; HTML/CSS.
- . **Clément** : Réalisation des maquettes ; Conception BDD ; Développement backend ; HTML/CSS.
- . **Tom** : Conception BDD ; Développement backend ; HTML/ CSS.
- . **Antoine** : Conception BDD ; Développement backend ; HTML/ CSS.



Code :

Le projet repose sur une architecture MVC (**Modele-vue-controlleur**)
claire et organisé.



Le routeur :

C'est lui qui regarde ce que l'utilisateur veut faire en fonction de (l'url) et envoie la demande au bon contrôleur.

- `$dependencyContainer` : Pour instancier les contrôleurs ou services automatiquement.
- `$pageMappings` : Une liste de toutes les routes du site (clé = nom de la page ; valeur = [contrôleur, méthode, rôle requis]).
- `$defaultPage`, `$errorPage`, `$unauthorizedPage` : Les pages de secours (Ex : 404, erreur, page par défaut...).

```
private $dependencyContainer;  
private $pageMappings;  
private $defaultPage;  
private $errorPage;  
private $unauthorizedPage;
```

Les contrôleurs :

- . Il traite la demande.
- . Reçoit la requête de l'utilisateur.
- . Appellent les modèles pour récupérer ou modifier les données.
 - . Utilisent les entités pour structurer ces données.
 - . Transmettent ensuite les résultats à une vue **Twig**.

```
use MyApp\Model\InvoiceModel;  
use MyApp\Model\ReservationModel;  
use MyApp\Entity\Invoice;  
use MyApp\Entity\Reservation;  
use MyApp\Service\DependencyContainer;  
use Twig\Environment;
```

Dans le projet on utilise plusieurs contrôleurs :

- . DefaultController.php
- . PaymentController.php
- . ReservationController.php

Les modèles :

Ils contiennent la logique d'accès à la base de données : requêtes SQL, insertion, modification, suppression, lecture.

Chaque modèle est lié à une entité du même nom.

```
class InvoiceModel
{
    private PDO $db;

    public function __construct(PDO $db)
    {
        $this->db = $db;
    }

    public function getAllInvoices(): array
    {
        $sql = "SELECT i.*,
                    r.id as reservation_id, r.user_id, r.room_id, r.check_in, r.check_out, r.status as reservation_status, r.total_price, r.special_requests,
                FROM Invoice i
                INNER JOIN Reservation r ON i.reservation_id = r.id
                ORDER BY i.invoice_date DESC";
        $stmt = $this->db->query($sql);
        $invoices = [];

        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            $reservation = new Reservation(
                $row['reservation_id'],
                new User($row['user_id'], '', '', '', null, '', '', ''),
                new Room($row['room_id'], '', true, 0, 0, '', null, '', ''),
                $row['check_in'], $row['check_out'], $row['reservation_status'],
                $row['total_price'], $row['special_requests'], $row['reservation_created'], $row['reservation_updated']
            );

            $invoices[] = new Invoice([
                $row['id'],
                $reservation,
                $row['invoice_number'],
                $row['amount'],
                $row['tax_amount'],
                $row['total_amount'],
                $row['invoice_date'],
                $row['due_date'],
                $row['status'],
                $row['pdf_path']
            ]);
        }
    }
}
```

Dans le projet on utilise plusieurs modèles :

- . CancellationModel.php
- . ContactMessage.php
- . InvoiceModel.php

Les entités :

Les entités représentent les objets métiers du projet. Chaque entité correspond à une table dans la base de données.

```
class Invoice
{
    private ?int $id;
    private Reservation $reservation;
    private string $invoice_number;
    private float $amount;
    private float $tax_amount;
    private float $total_amount;
    private string $invoice_date;
    private string $due_date;
    private string $status;
    private ?string $pdf_path;

    public function __construct(
        ?int $id,
        Reservation $reservation,
        string $invoice_number,
        float $amount,
        float $tax_amount,
        float $total_amount,
        string $invoice_date,
        string $due_date,
        string $status,
        ?string $pdf_path
    ) {
        $this->id = $id;
        $this->reservation = $reservation;
        $this->invoice_number = $invoice_number;
        $this->amount = $amount;
        $this->tax_amount = $tax_amount;
        $this->total_amount = $total_amount;
        $this->invoice_date = $invoice_date;
        $this->due_date = $due_date;
        $this->status = $status;
        $this->pdf_path = $pdf_path;
    }
}
```

Dans le projet on utilise plusieurs entités :

- . ContactMessage.php
- . Notification.php
- . Payment.php

Le twig:

Les vues sont des fichiers Twig qui affichent les données à l'utilisateur. Elles sont appelées par les contrôleurs après traitement.

Cela veut dire que la page hérite d'un template de base (base.html.twig) qui contient le <head>, les balises HTML principales, la structure commune à toutes les pages.

```
{% extends "base.html.twig" %}

{% block title %} {{ parent() }} Inscription {% endblock %}

{% block css %} {{ parent() }} <link rel='stylesheet' href='css/user/auth.css'> {% endblock %}

{% block content %}
<a href="index.php" class="home-link">Accueil</a>
<div class="page-container">
    <div class="login-form">
        <div class="textellogin">
            Inscription
        </div>
    </div>
</div>
```

Amélioration :

- . Possibilité de mettre plus d'image pour chaque chambre.
- . Mode de paiement réel et fonctionnel.
- . Réglage de la notification.
- . Ajout des avoirs.

Conclusion :

Ce projet nous a permis de travailler en groupe en développant un site web pour un hôtel, en utilisant du HTML, du CSS et du PHP. Nous avons également dû interagir avec une base de données.

Grâce à ce projet, nous avons pu acquérir de nouvelles compétences, notamment en maîtrisant différents langages de programmation et en apprenant à gérer l'interaction avec la bases de données. Cette expérience a renforcé notre capacité à collaborer tout en développant des solutions concrètes dans un contexte réel.

Démonstration du site :