Projet personnel réalisé en cours de formation — Application Web

Projet personnel réalisé en cours de formation :

Développement d'un client léger de location d’habitations de vacances — Neige & Soleil

avec PHP, MySQL et le pattern MVC

Sommaire

I. Introduction

Neige & Soleil est une application web de gestion de location de logements de vacances, elle met en relation trois types d'acteurs : les clients qui recherchent et réservent des hébergements, les propriétaires qui publient et gèrent leurs annonces, et les administrateurs qui supervisent l'ensemble de la plateforme.

L'application couvre l'intégralité du cycle de vie d'une location : création de compte, dépôt d'annonce, recherche et réservation d'habitation, génération de contrats PDF, et gestion des mots de passe. Elle illustre les principes fondamentaux du développement web côté serveur :

- Architecture MVC (Modèle – Vue – Contrôleur) en PHP
- Conception et exploitation d'une base de données relationnelle MySQL
- Gestion des sessions, des rôles et du contrôle d'accès
- Validation des données côté serveur par expressions régulières
- Upload de fichiers et gestion de photos
- Génération de documents PDF via la bibliothèque FPDF
- Envoi d'emails transactionnels avec PHPMailer

> *Objectif : Développer une plateforme complète de location de vacances permettant aux clients de réserver des logements, aux propriétaires de gérer leurs annonces, et aux administrateurs de piloter l'ensemble des données via une interface dédiée.*

II. Analyse du besoin

2.1 Contexte et objectifs

L'agence Neige & Soleil souhaite digitaliser son activité de location saisonnière. L'application doit permettre à des propriétaires de mettre en ligne leurs habitations (maisons ou appartements) et à des clients de les consulter, filtrer et réserver en ligne. Un administrateur centralise la gestion des comptes, des habitations et des réservations.

2.2 Fonctionnalités attendues

Acteur	Fonctionnalité
Client, Propriétaire	Créer un compte, se connecter, consulter les annonces
Client	Réserver une habitation, visualiser et annuler ses réservations
Client	Télécharger son contrat de réservation (PDF)
Client, Propriétaire	Modifier ses informations personnelles
Propriétaire	Déposer une annonce (maison ou appartement) avec photos
Propriétaire	Modifier les tarifs et les informations de son habitation
Propriétaire	Supprimer une annonce, télécharger son contrat de mandat
Administrateur	Gérer les clients, propriétaires, habitations et réservations (CRUD complet)
Tous	Réinitialiser son mot de passe par email avec code de vérification
Tous	Renouveler obligatoirement son mot de passe tous les 90 jours

III. Architecture et conception

3.1 Structure du projet (MVC)

L'application suit strictement le patron de conception MVC. L'ensemble des requêtes transite par un point d'entrée unique (index.php) qui joue le rôle de routeur. Selon le paramètre « URL ?page=N », le contrôleur approprié est chargé, qui à son tour appelle la vue correspondante.

Couche	Fichier(s)	Rôle
Routeur	index.php	Point d'entrée unique, gestion de la navigation et des droits
Contrôleur	controleur_class.php	Intermédiaire entre les vues et le modèle
Modèle	modele_class.php	Requêtes SQL et accès à la base de données
Vue	vue_*.php	Affichage HTML des données
Gestion	gestion_*.php	Traitement des formulaires et logique métier
Config	parametres.php	Constantes BASE_URL et BASE_PATH

Le routeur protège les pages sensibles grâce à un tableau « $pageProtege » associant chaque numéro de page aux rôles autorisés. Toute tentative d'accès non autorisé redirige vers la page d'accueil.

```php
<?php
//protection accés pages sensibles
$pageProtege = [
    2 => ['admin'],
    3 => ['admin'],
    4 => ['admin'],
    5 => ['admin'],
    6 => ['client'],
    7 => ['proprietaire'],
    11 => ['client'],
    12 => ['client'],
    13 => ['proprietaire'],
    14 => ['proprietaire'],
    20 => ['client'],
    21 => ['proprietaire'],
    22 => ['proprietaire']
];

$page = (isset($_GET['page'])) ? intval($_GET['page']) : 1;

if(isset($pageProtege[$page])){
    if(!isset($_SESSION['email']) || !isset($_SESSION['role'])){
        header("Location: index.php?page=1");
        exit;
    }
    if(!in_array($_SESSION['role'], $pageProtege[$page])){
        header("Location: index.php?page=1");
        exit;
    }
}
```

3.2 Base de données

La base de données neigeetsoleil est organisée autour de plusieurs tables principales. Le schéma utilise l'héritage par délégation : la table utilisateur centralise les données communes, tandis que les tables client, proprietaire et admin y font référence par clé étrangère.

Table	Description
Utilisateur	Données communes à tous les utilisateurs (nom, email, mdp, rôle)
Client	Données spécifiques au client (adresse, RIB, nb_resa)
Proprietaire	Données spécifiques au propriétaire (adresse, RIB, nb_contrat)
Habitation	Table centrale des logements (adresse, tarifs, capacité)
Maison	Sous-table des maisons (caractéristiques spécifiques)
Appartement	Sous-table des appartements (étage, type)
Reservation	Réservations (dates, nb personnes, état, prix)
Contrat	Contrats de mandat liés aux habitations
Photos	Photos des habitations (URL, indicateur principal)
reset_mdp	Codes de réinitialisation de mot de passe (expiration 15 min)
archiveReservation	Historique des réservations annulées ou expirées

```sql
/**********************************Création de la bdd ******************************/
drop database if exists neigeetsoleil;
create database neigeetsoleil;
use neigeetsoleil;

/********************************** Création des tables ******************************/
drop table if exists utilisateur;
CREATE TABLE utilisateur (
    id_user INT AUTO_INCREMENT,
    nom VARCHAR(50) NOT NULL,
    prenom VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    mdp VARCHAR(255) NOT NULL,
    tel VARCHAR(15) NOT NULL,
    role ENUM('client', 'proprietaire', 'admin') NOT NULL DEFAULT 'client',
    PRIMARY KEY (id_user)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;

drop table if exists admin;
create table admin (
    id_a int not null,
    primary key(id_a),
    constraint fk_admin_user foreign key (id_a) references utilisateur(id_user) on delete cascade
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;

drop table if exists client;
create table client(
    id_c int not null,
    adresse varchar(100),
    cp varchar(10),
    ville varchar(50),
    RIB varchar(50),
    primary key(id_c),
    constraint fk_client_user foreign key(id_c) references utilisateur(id_user) on delete cascade
)ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;
```

```sql
drop table if exists proprietaire;
create table proprietaire(
    id_p int not null,
    adresse varchar(100),
    cp varchar(10),
    ville varchar(50),
    RIB varchar(50),
    nb_contrat int default 0,
    primary key(id_p),
    constraint fk_proprietaire_user foreign key(id_p) references utilisateur(id_user) on delete cascade
)ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;

create table habitation(
    ref_hab int(5) not null auto_increment,
    type_hab varchar(20) not null,
    adr_hab varchar(120) not null,
    cp_hab varchar(5) not null,
    ville_hab varchar(50) not null,
    tarif_hab_bas float(5) not null,
    tarif_hab_moy float(5) not null,
    tarif_hab_hau float(5) not null,
    surface varchar(10) not null,
    id_p int(5) not null,
    description_hab text,
    titre_hab varchar(60) not null,
    capacite_hab int(2) not null,
    primary key (ref_hab),
    foreign key (id_p) references proprietaire(id_p)
)ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;
```

```sql
create table appartement(
    ref_hab int(5) not null auto_increment,
    type_hab varchar(20) not null,
    adr_hab varchar(120) not null,
    cp_hab varchar(5) not null,
    ville_hab varchar(50) not null,
    tarif_hab_bas float(5) not null,
    tarif_hab_moy float(5) not null,
    tarif_hab_hau float(5) not null,
    surface varchar(10) not null,
    id_p int(5) not null,
    description_hab text,
    titre_hab varchar(60) not null,
    capacite_hab int(2) not null,
    etage_ap int(2) not null,
    type_ap varchar(3),
    primary key (ref_hab)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;

create table maison(
    ref_hab int(5) not null auto_increment,
    type_hab varchar(20) not null,
    adr_hab varchar(120) not null,
    cp_hab varchar(5) not null,
    ville_hab varchar(50) not null,
    tarif_hab_bas float(5) not null,
    tarif_hab_moy float(5) not null,
    tarif_hab_hau float(5) not null,
    surface varchar(10) not null,
    id_p int(5) not null,
    description_hab text,
    titre_hab varchar(60) not null,
    capacite_hab int(2) not null,
    carac_m varchar(50) not null,
    primary key (ref_hab)
) ENGINE = InnoDB CHARSET = utf8mb4;
```

```sql
create table reservation(
    ref_res int(5) not null auto_increment,
    date_res date not null,
    nb_perso int(2) not null,
    date_debut date not null,
    date_fin date not null,
    etat_res enum("Validee","En attente","Annulee"),
    id_c int(5) not null,
    ref_hab int(5) not null,
    prix_a_payer float not null default 0,
    primary key (ref_res),
    foreign key (id_c) references client(id_c),
    foreign key (ref_hab) references habitation(ref_hab)
);

create table contrat(
    ref_c int(20) not null auto_increment,
    status_c enum("En validation","En cours","Annule","Resilie"),
    annee_signature date,
    annee_fin date,
    id_p int(5) not null,
    ref_hab int(5) not null,
    primary key (ref_c),
    foreign key (id_p) references proprietaire(id_p),
    foreign key (ref_hab) references habitation(ref_hab)
)ENGINE = InnoDB CHARSET = utf8mb4;

create table if not exists photos(
    id_photo int not null auto_increment,
    ref_hab int not null,
    url_photo varchar(255) not null,
    is_principal boolean default false,
    primary key(id_photo),
    foreign key(ref_hab) references habitation(ref_hab) on delete cascade on update cascade
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;
```

```sql
drop table if exists reset_mdp;
create table reset_mdp(
    email varchar(100) not null,
    code varchar(6) not null,
    created_at datetime default now(),
    primary key(email)
) ENGINE = InnoDB DEFAULT CHARSET = utf8mb4;

create table archiveReservation like reservation;
alter table archivereservation add column datehisto date;

create table archiveContrat like contrat;
alter table archiveContrat add column datehisto date;
```

Des triggers MySQL assurent la cohérence automatique des données : insertion synchronisée entre les tables maison/appartement et habitation, mise à jour des compteurs (nb_contrat, nb_resa), formatage automatique des noms/prénoms, et historisation des réservations annulées.

```
/************************************ Triggers ************************************/
drop trigger if exists insert_appart;
delimiter //
create trigger insert_appart
before insert on appartement for each row BEGIN
    if new.ref_hab is null or new.ref_hab in (select ref_hab from habitation) or new.ref_hab = 0
    then
   set new.ref_hab = ifnull((select ref_hab from habitation where ref_hab >= all
    (select ref_hab from habitation)), 0) +1 ;
end if;
insert into habitation values(new.ref_hab,new.type_hab,new.adr_hab,new.cp_hab,new.ville_hab,
                              new.tarif_hab_bas,new.tarif_hab_moy,new.tarif_hab_hau,new.surface,
                              new.id_p,new.description_hab,new.titre_hab,new.capacite_hab
                             );
end //
delimiter ;

drop trigger if exists update_appart;
delimiter //
create trigger update_appart
before update on appartement for each row BEGIN
    update habitation set ref_hab=new.ref_hab,type_hab=new.type_hab,adr_hab=new.adr_hab,
                          cp_hab=new.cp_hab,ville_hab=new.ville_hab,tarif_hab_bas=new.tarif_hab_bas,
                          tarif_hab_moy=new.tarif_hab_moy,tarif_hab_hau=new.tarif_hab_hau,
                          surface=new.surface,id_p=new.id_p,description_hab=new.description_hab,
                          titre_hab=new.titre_hab,capacite_hab=new.capacite_hab
    where habitation.ref_hab=new.ref_hab;
end //
delimiter ;
```

```
drop trigger if exists delete_appart;
delimiter //
create trigger delete_appart
before delete on appartement for each row BEGIN
    delete from habitation where habitation.ref_hab=old.ref_hab;
end //
delimiter ;

drop trigger if exists insert_maison;
delimiter //
create trigger insert_maison
before insert on maison for each row BEGIN
if new.ref_hab is null or new.ref_hab in (select ref_hab from habitation) or new.ref_hab = 0
    then
set new.ref_hab = ifnull((select ref_hab from habitation where ref_hab >= all
    (select ref_hab from habitation)), 0) +1 ;
end if;
insert into habitation values(new.ref_hab,new.type_hab,new.adr_hab,new.cp_hab,new.ville_hab,
                              new.tarif_hab_bas,new.tarif_hab_moy,new.tarif_hab_hau,new.surface,
                              new.id_p,new.description_hab,new.titre_hab,new.capacite_hab);
end //
delimiter ;

drop trigger if exists update_maison;
delimiter //
create trigger update_maison
before update on maison for each row BEGIN
    update habitation set ref_hab=new.ref_hab,type_hab=new.type_hab,adr_hab=new.adr_hab,
                          cp_hab=new.cp_hab,ville_hab=new.ville_hab,tarif_hab_bas=new.tarif_hab_bas,
                          tarif_hab_moy=new.tarif_hab_moy,tarif_hab_hau=new.tarif_hab_hau,
                          surface=new.surface,id_p=new.id_p,description_hab=new.description_hab,
                          titre_hab=new.titre_hab,capacite_hab=new.capacite_hab
    where habitation.ref_hab=new.ref_hab;
end //
delimiter ;
```

```
drop trigger if exists delete_maison;
delimiter //
create trigger delete_maison
before delete on maison for each row BEGIN
    delete from habitation where habitation.ref_hab=old.ref_hab;
end //
delimiter ;

drop trigger if exists insert_contrat;
delimiter //
create trigger insert_contrat
after insert on habitation
for each row
begin
insert into contrat values (null,'En validation',null,null,new.id_p,new.ref_hab);
end //
delimiter ;

drop trigger if exists delete_contrat;
delimiter //
create trigger delete_contrat
after delete on habitation
for each row
begin
delete from contrat where contrat.ref_hab = old.ref_hab;
end //
delimiter ;
```

```
drop trigger if exists updateNbContratInsert;
delimiter //
create trigger updateNbContratInsert
after insert on contrat
for each row
begin
update proprietaire set nb_contrat = nb_contrat + 1
where id_p = new.id_p;
end //
delimiter ;

drop trigger if exists updateNbContratDelete;
delimiter //
create trigger updateNbContratDelete
after delete on contrat
for each row
begin
update proprietaire set nb_contrat = nb_contrat - 1
where id_p = old.id_p;
end //
delimiter ;

drop trigger if exists formeNomsPrenomsUtilisateurInsert;
delimiter //
create trigger formeNomsPrenomsUtilisateurInsert
before insert on utilisateur
for each row
begin
set new.nom = upper(new.nom), new.prenom = capitalisation(new.prenom);
end//
delimiter ;
```

```
drop trigger if exists formeNomsPrenomsUtilisateurUpdate;
delimiter //
create trigger formeNomsPrenomsUtilisateurUpdate
before insert on utilisateur
for each row
begin
set new.nom = upper(new.nom), new.prenom = capitalisation(new.prenom);
end//
delimiter ;

drop trigger if exists tr_insertAdmin;
delimiter //
create trigger tr_insertAdmin
after insert on utilisateur
for each row
begin
    if
        new.role = 'admin'
    then
        insert into admin values (new.id_user);
    end if;
end //
delimiter ;
```

```
/* historisation contrats */
drop trigger if exists histoContratResilieAnnule;
delimiter //
create trigger histoContratResilieAnnule
after update on contrat
for each row
BEGIN
if new.status_c = 'Resilie' or new.status_c = 'Annule'
then
    insert into archivecontrat
    values(new.ref_c,new.status_c,new.annee_signature,new.annee_fin,new.id_p,
           new.ref_hab, curdate()
          );
end if;
end //
delimiter ;

/* historisation reservations*/
drop trigger if exists histoReservationAnnulee;
delimiter //
create trigger histoReservationAnnulee
after update on reservation
for each row
begin
if new.etat_res = 'Annulee'
then insert into archivereservation
       values(new.ref_res, new.date_res, new.nb_perso, new.date_debut, new.date_fin,
              new.etat_res, new.id_c, new.ref_hab, new.prix_a_payer, curdate()
             );
end if;
end //
delimiter ;
```

3.3 Sécurité et contrôle d'accès

Plusieurs mécanismes de sécurité sont mis en place tout au long de l'application :

- Contrôle de rôle à chaque page via les sessions PHP ($_SESSION['role'])
- Validation côté serveur de tous les champs de formulaire par expressions régulières
- Renouvellement obligatoire du mot de passe après 90 jours d'inactivité
- Vérification du format et de la taille des photos uploadées (3 photos max, 4 Mo, jpg/png/avif)
- Utilisation de htmlspecialchars() dans les vues pour prévenir les injections XSS
- Code de réinitialisation de mot de passe à 6 chiffres avec durée de validité de 15 minutes

```php
<?php
//recup infos formulaire
    if(isset($_POST['Connexion'])){
        $email = $_POST['email'];
        $mdp = $_POST['mdp'];

        $unUtilisateur = $unControleur->selectWhereUtilisateur($email,$mdp);

        if(!$unUtilisateur){
            $_SESSION['msg-erreur-connexion'] = "Identifiants incorrects";
        }else{
            $dateMdp = $unUtilisateur['date_mdp'];
            $dateAjd = date('Y-m-d');
            $dateMdp = new DateTime($dateMdp);
            $dateAjd = new DateTime($dateAjd);
            $interval = $dateAjd->diff($dateMdp);
            $ecart = $interval->days;

            if($ecart >= 90){
                $_SESSION['email'] = $unUtilisateur['email'];
                $_SESSION['role'] = "changement mdp";
                header("Location:index.php?page=30");
                exit;
            }else{
                $_SESSION['email'] = $unUtilisateur['email'];
                $_SESSION['prenom'] = $unUtilisateur['prenom'];
                $_SESSION['nom'] = $unUtilisateur['nom'];
                $_SESSION['tel'] = $unUtilisateur['tel'];
                $_SESSION['role'] = $unUtilisateur['role'];
                $_SESSION['id'] = $unUtilisateur['id_user'];
                //on recharge la page
                header("Location: index.php?page=1");
                exit;
            }
        }
    }
```

IV. Développement

4.1 Gestion des utilisateurs

L'application propose trois profils distincts. La création de compte est accessible depuis la page de connexion ; l'utilisateur choisit son type (client ou propriétaire) et remplit un formulaire complet. Chaque champ est validé côté serveur avant insertion en base.

Champ	Regex de validation
Nom / Prénom	Lettres, espaces, tirets, apostrophes (min. 2 car.)
Email	Format standard user@domaine.ext
Mot de passe	Tout caractère sauf espace (min. 3 car.)
Téléphone	Exactement 10 chiffres
Code postal	Exactement 5 chiffres
RIB	27 caractères alphanumériques majuscules

```php
<?php

    if(isset($_POST['ajouter'])){

        $erreurs = [];

        $type = $_POST['type'];
        $nom = $_POST['nom'];
        $prenom = $_POST['prenom'];
        $email = $_POST['email'];
        $mdp = $_POST['mdp'];
        $adresse = $_POST['adresse'];
        $cp = $_POST['cp'];
        $ville = $_POST['ville'];
        $tel = $_POST['tel'];
        $rib = $_POST['rib'];

        $regexNom = '/^[A-Za-zÀ-ÖØ-öø-ÿ\' -]{2,}$/u';
        $regexPrenom = '/^[A-Za-zÀ-ÖØ-öø-ÿ\' -]{2,}$/u';
        $regexEmail = '/^[A-Za-zÀ-ÖØ-öø-ÿ0-9._-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}/u';
        $regexMdp = '/^[^ ]{3,}$/u';
        $regexAdresse = '/^[0-9]{1,5} [A-Za-zÀ-ÖØ-öø-ÿ\' .-]{3,}$/u';
        $regexCp = '/^[0-9]{5}$/';
        $regexVille = '/^[A-Za-zÀ-ÖØ-öø-ÿ\' -]{2,}$/u';
        $regexTel = '/^[0-9]{10}$/';
        $regexRib = '/^[0-9A-Z]{27}$/';


        $champs = [$type,$nom,$prenom,$email,$mdp,$adresse,$cp,$ville,$tel,$rib];

        foreach($champs as $champ){
                if($champ == ""){
                        $erreurs[] = "Veuillez remplir touts les champs";
                        break;
                }
        }
```

```php
$regles = [
                "nom" => [$regexNom,     "Veuillez rentrer un nom valide"],
                "prenom" => [$regexPrenom,  "Veuillez rentrer un prénom valide"],
                "email" => [$regexEmail,   "Veuillez rentrer un email valide"],
                "mdp" => [$regexMdp,     "Veuillez rentrer un mot de passe valide"],
                "adresse" => [$regexAdresse, "Veuillez rentrer une adresse valide"],
                "cp" => [$regexCp,      "Veuillez rentrer un code postal valide"],
                "ville" => [$regexVille,   "Veuillez rentrer un nom de ville valide"],
                "tel" => [$regexTel,     "Veuillez rentrer un numéro de téléphone valide"],
                "rib" => [$regexRib,     "Veuillez rentrer un RIB valide"]
];

foreach($regles as $champ => [$regex, $msg]){
        if(!preg_match($regex, trim($_POST[$champ]))){
                $erreurs[] = $msg;
                break;
        }
}
```

```php
if(!empty($erreurs)){
        $_SESSION['msg-erreurs'] = $erreurs;
}else{
        if($type == 'client'){
                $unControleur->insertClient([
                        "nom" => $nom,
                        "prenom" => $prenom,
                        "email" => $email,
                        "mdp" => $mdp,
                        "tel" => $tel,
                        "adresse" => $adresse,
                        "cp" => $cp,
                        "ville" => $ville,
                        "rib" => $rib
                        ]);

                header("Location: index.php?page=16");
                exit;

        }
        if($type == 'proprietaire'){
                $unControleur->insertProprietaire([
                        "nom" => $nom,
                        "prenom" => $prenom,
                        "email" => $email,
                        "mdp" => $mdp,
                        "tel" => $tel,
                        "adresse" => $adresse,
                        "cp" => $cp,
                        "ville" => $ville,
                        "rib" => $rib
                        ]);
                header("Location: index.php?page=16");
                exit;
        }
}
```

```php
}

if(isset($_POST['annuler'])){
        header("Location: index.php?page=8");
        exit;
}

        require_once("vue/vue_creation_compte.php");
?>
```

Le processus de connexion vérifie l'existence du compte en base, puis contrôle la date du dernier changement de mot de passe. Si elle dépasse 90 jours, l'utilisateur est redirigé obligatoirement vers la page de renouvellement avant d'accéder à son espace.

4.2 Gestion des habitations

Deux types d'habitations sont supportés : les maisons et les appartements. Chaque type possède ses propres champs spécifiques (caractéristiques pour la maison, étage et type pour l'appartement). La création déclenche automatiquement via un trigger l'insertion dans la table habitation et la création d'un contrat de mandat en état 'En validation'.

Lors de la création ou modification d'une annonce, exactement 3 photos doivent être fournies. La première est automatiquement marquée comme photo principale et s'affiche sur la page d'accueil. Les fichiers sont renommés avec uniqid() pour éviter les collisions.

Deux interfaces de gestion coexistent : l'interface propriétaire (modification des tarifs, description, capacité, photos) et l'interface administrateur (CRUD complet avec accès à tous les champs).

```php
<?php

$idProprietaire = $_SESSION['id'];
$leProprietaire = $unControleur->selectWhereIdProprietaire($idProprietaire);
$erreurs = [];

if (isset($_POST['ajouter'])){


        $photos = $_FILES['photos'];
        $formats = array("jpg","jpeg","png","avif");
        $tailleMax = 4 * 1024 * 1024;

        $id_p = $_POST['id_p'];
        $adr_hab = $_POST['adr_hab'];
        $cp_hab = $_POST['cp_hab'];
        $ville_hab = $_POST['ville_hab'];
        $tarif_hab_bas = $_POST['tarif_hab_bas'];
        $tarif_hab_moyen = $_POST['tarif_hab_moy'];
        $tarif_hab_haut = $_POST['tarif_hab_hau'];
        $surface = $_POST['surface'];
        $description_hab = $_POST['description_hab'];
        $titre_hab = $_POST['titre_hab'];
        $capacite_hab = $_POST['capacite_hab'];
        $etage_ap = $_POST['etage_ap'];
        $type_ap = $_POST['type_ap'];

        $regexAdresse = '/^[0-9]{1,5} [A-Za-zÀ-ÖØ-öø-ÿ\' .-]{3,}$/u';
        $regexCp = '/^[0-9]{5}$/';
        $regexVille = '/^[A-Za-zÀ-ÖØ-öø-ÿ\' -]{2,}$/u';
        $regexTarifs = '/^[0-9]{1,5}([.,][0-9]{0,2})?$/';
        $regexSurface = '/^[1-9][0-9]{0,2}$/';
        $regexCapacite = '/^[1-9][0-9]{0,1}$/';
```

```php
    $champs = [$adr_hab, $cp_hab, $ville_hab,
               $tarif_hab_bas, $tarif_hab_moyen, $tarif_hab_haut,
               $surface, $description_hab, $titre_hab, $capacite_hab,
               $etage_ap, $type_ap
              ];

    foreach ($champs as $champ) {
            if ($champ === "") {
                    $erreurs[] = "Veuillez remplir tous les champs";break;
            }
    }

    $regles = [
                    "adr_hab" => [$regexAdresse, "Veuillez rentrer une adresse valide"],
                    "cp_hab" => [$regexCp,      "Veuillez rentrer un code postal valide"],
                    "ville_hab" => [$regexVille,   "Veuillez rentrer un nom de ville valide"],
                    "tarif_hab_bas" => [$regexTarifs, "Veuillez rentrer un tarif bas valide"],
                    "tarif_hab_moy" => [$regexTarifs, "Veuillez rentrer un tarif moyenvalide"],
                    "tarif_hab_hau" => [$regexTarifs, "Veuillez rentrer un tarif haut valide"],
                    "surface"  => [$regexSurface, "Veuillez rentrer une surface valide"],
                    "capacite_hab"  => [$regexCapacite, "Veuillez rentrer une capacité valide"],
    ];

    foreach($regles as $champ => [$regex, $msg]){
            if(!preg_match($regex, trim($_POST[$champ]))){
                    $erreurs[] = $msg;
                    break;
            }
    }
```

```php
$nbPhotos = count(array_filter($photos['name']));

if ($nbPhotos !== 3) {
        $erreurs[] = "Veuillez sélectionner exactement 3 photos.";
}else{
        foreach ($photos['name'] as $i => $name) {
                $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));
                if (!in_array($extension, $formats)) {
                        $erreurs[] = "Format non autorisé pour la photo : $name";
                }
                if($photos['size'][$i] > $tailleMax){
                        $erreurs[] = "Taille de la photo non autorisé : $name";
                }
                if ($photos['error'][$i] !== UPLOAD_ERR_OK) {
                        $erreurs[] = "Erreur lors de l'upload de la photo : $name";
                }
        }
}
```

```php
if (!empty($erreurs)) {
        $_SESSION['erreurs'] = $erreurs;
        header("Location:index.php?page=13");
        exit;
}else{
        //ajouter les insert dans photos et habitations
        $refHab = $unControleur->insertAppartement([
                "adr_hab" => $adr_hab,
                "cp_hab" => $cp_hab,
                "ville_hab" => $ville_hab,
                "tarif_hab_bas" => $tarif_hab_bas,
                "tarif_hab_moy" => $tarif_hab_moyen,
                "tarif_hab_hau" => $tarif_hab_haut,
                "surface" => $surface,
                "id_p" => $id_p,
                "description_hab" => $description_hab,
                "titre_hab" => $titre_hab,
                "capacite_hab" => $capacite_hab,
                "etage_ap" => $etage_ap,
                "type_ap" => $type_ap
        ]);
```

```php
                foreach ($photos['name'] as $i => $name) {

                $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));
                $newName = uniqid("photo_") . "." . $extension;

                move_uploaded_file($photos['tmp_name'][$i], "images/habitations/" . $newName);

                $isPrincipal = ($i === 0) ? 1 : 0;

                $unControleur->insertPhoto([
                        "ref_hab" => $refHab,
                        "url_photo" => $newName,
                        "is_principal" => $isPrincipal
                        ]);
                }

                header("Location: index.php?page=14");
                exit;
        }

}

if(isset($_POST['annuler'])){
        header("Location: index.php?page=7");
        exit;
}

        require_once("vue/vue_creation_annonce_appartement.php");

?>
```

```php
<?php

$lesAppartements = $unControleur->selectAllAppartement();
$lesProprietaires = $unControleur->selectAllProprietaire();
$appartement = null;
$erreurs = [];



if(isset($_SESSION['role']) && $_SESSION['role'] == 'admin'){

    if(isset($_GET['action']) && isset($_GET['ref_hab'])){
        $action = $_GET['action'];
        $ref_hab = $_GET['ref_hab'];

        switch($action){
            case "sup" : $unControleur->deleteAppartement($ref_hab);
                                    header("Location:index.php?page=29");
                                    exit;
                                    break;
            case "edit" : $appartement = $unControleur->selectWhereAppartement($ref_hab);break;
        }
    }
}
```

```php
if(isset($_POST['valider']) || isset($_POST['modifier'])){

    $photos = $_FILES['photos'];
    $formats = array("jpg","jpeg","png","avif");
    $tailleMax = 4 * 1024 * 1024;

    $ref_hab = $_POST['ref_hab'];
    $id_p = $_POST['id_p'];
    $adr_hab = $_POST['adr_hab'];
    $cp_hab = $_POST['cp_hab'];
    $ville_hab = $_POST['ville_hab'];
    $tarif_hab_bas = $_POST['tarif_hab_bas'];
    $tarif_hab_moyen = $_POST['tarif_hab_moy'];
    $tarif_hab_haut = $_POST['tarif_hab_hau'];
    $surface = $_POST['surface'];
    $description_hab = $_POST['description_hab'];
    $titre_hab = $_POST['titre_hab'];
    $capacite_hab = $_POST['capacite_hab'];
    $etage_ap = $_POST['etage_ap'];
    $type_ap = $_POST['type_ap'];

    $regexAdresse = '/^[0-9]{1,5} [A-Za-zÀ-ÖØ-öø-ÿ\' .-]{3,}$/u';
    $regexCp = '/^[0-9]{5}$/';
    $regexVille = '/^[A-Za-zÀ-ÖØ-öø-ÿ\' -]{2,}$/u';
    $regexTarifs = '/^[0-9]{1,5}([.,][0-9]{0,2})?$/';
    $regexSurface = '/^[1-9][0-9]{0,2}$/';
    $regexCapacite = '/^[1-9][0-9]{0,1}$/';

    $champs = [$adr_hab, $cp_hab, $ville_hab,
                $tarif_hab_bas, $tarif_hab_moyen, $tarif_hab_haut,
                $surface, $description_hab, $titre_hab, $capacite_hab,
                $etage_ap,$type_ap
            ];
}
```

```php
if(isset($_POST['valider'])){
    foreach ($champs as $champ) {
        if ($champ === "") {
            $erreurs[] = "Veuillez remplir tous les champs";break;
        }
    }

    $regles = [
                "adr_hab" => [$regexAdresse, "Veuillez rentrer une adresse valide"],
                "cp_hab" => [$regexCp,      "Veuillez rentrer un code postal valide"],
                "ville_hab" => [$regexVille,   "Veuillez rentrer un nom de ville valide"],
                "tarif_hab_bas" => [$regexTarifs, "Veuillez rentrer un tarif bas valide"],
                "tarif_hab_moy" => [$regexTarifs, "Veuillez rentrer un tarif moyenvalide"],
                "tarif_hab_hau" => [$regexTarifs, "Veuillez rentrer un tarif haut valide"],
                "surface"  => [$regexSurface, "Veuillez rentrer une surface valide"],
                "capacite_hab"  => [$regexCapacite, "Veuillez rentrer une capacité valide"]
            ];
    foreach($regles as $champ => [$regex, $msg]){
        if(!preg_match($regex, trim($_POST[$champ]))){
            $erreurs[] = $msg;break;
        }
    }
```

```php
    $nbPhotos = count(array_filter($photos['name']));
    if ($nbPhotos !== 3) {
        $erreurs[] = "Veuillez sélectionner exactement 3 photos.";
    }else{
        foreach ($photos['name'] as $i => $name) {
            $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));

            if (!in_array($extension, $formats)) {
                $erreurs[] = "Format non autorisé pour la photo : $name";
            }
            if($photos['size'][$i] > $tailleMax){
                $erreurs[] = "Taille de la photo non autorisé : $name";
            }
            if ($photos['error'][$i] !== UPLOAD_ERR_OK) {
                $erreurs[] = "Erreur lors de l'upload de la photo : $name";
            }
        }
    }
```

```php
if (!empty($erreurs)) {
            $_SESSION['erreurs'] = $erreurs;
            header("Location:index.php?page=29");
            exit;
    }else{
        $refHab = $unControleur->insertAppartement([
            "adr_hab" => $adr_hab,
            "cp_hab" => $cp_hab,
            "ville_hab" => $ville_hab,
            "tarif_hab_bas" => $tarif_hab_bas,
            "tarif_hab_moy" => $tarif_hab_moyen,
            "tarif_hab_hau" => $tarif_hab_haut,
            "surface" => $surface,
            "id_p" => $id_p,
            "description_hab" => $description_hab,
            "titre_hab" => $titre_hab,
            "capacite_hab" => $capacite_hab,
            "etage_ap" => $etage_ap,
            "type_ap" => $type_ap
        ]);

    foreach ($photos['name'] as $i => $name) {
        $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));
        $newName = uniqid("photo_") . "." . $extension;

        move_uploaded_file($photos['tmp_name'][$i], "images/habitations/" . $newName);

        $isPrincipal = ($i === 0) ? 1 : 0;

        $unControleur->insertPhoto([
            "ref_hab" => $refHab,
            "url_photo" => $newName,
            "is_principal" => $isPrincipal
        ]);
    }
```

```php
        header("Location: index.php?page=29");
        exit;
    }
}



if(isset($_POST['modifier'])){
    foreach ($champs as $champ) {
        if ($champ === "") {
            $erreurs[] = "Veuillez remplir tous les champs";break;
        }
    }

    $regles = [
                "adr_hab" => [$regexAdresse, "Veuillez rentrer une adresse valide"],
                "cp_hab" => [$regexCp,      "Veuillez rentrer un code postal valide"],
                "ville_hab" => [$regexVille,   "Veuillez rentrer un nom de ville valide"],
                "tarif_hab_bas" => [$regexTarifs, "Veuillez rentrer un tarif bas valide"],
                "tarif_hab_moy" => [$regexTarifs, "Veuillez rentrer un tarif moyenvalide"],
                "tarif_hab_hau" => [$regexTarifs, "Veuillez rentrer un tarif haut valide"],
                "surface"  => [$regexSurface, "Veuillez rentrer une surface valide"],
                "capacite_hab"  => [$regexCapacite, "Veuillez rentrer une capacité valide"],
            ];
    foreach($regles as $champ => [$regex, $msg]){
        if(!preg_match($regex, trim($_POST[$champ]))){
            $erreurs[] = $msg;break;
        }
    }
```

```php
$nbPhotos = count(array_filter($photos['name']));
if (($nbPhotos > 0 && $nbPhotos < 3) || $nbPhotos > 3) {
    $erreurs[] = "Veuillez sélectionner exactement 3 photos si vous souhaitez les modifier.";
}else{
    if($nbPhotos == 3){
        foreach ($photos['name'] as $i => $name) {
            $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));
            if (!in_array($extension, $formats)) {
                $erreurs[] = "Format non autorisé pour la photo : $name";
            }
            if($photos['size'][$i] > $tailleMax){
                $erreurs[] = "Taille de la photo non autorisé : $name";
            }
            if ($photos['error'][$i] !== UPLOAD_ERR_OK) {
                $erreurs[] = "Erreur lors de l'upload de la photo : $name";
            }
        }
    }
}
```

```php
if (!empty($erreurs)) {
    $_SESSION['erreurs'] = $erreurs;
    header("Location:index.php?page=29");
    exit;
}else{
    $refHab = $unControleur->updateAppartement([
        "adr_hab" => $adr_hab,
        "cp_hab" => $cp_hab,
        "ville_hab" => $ville_hab,
        "tarif_hab_bas" => $tarif_hab_bas,
        "tarif_hab_moy" => $tarif_hab_moyen,
        "tarif_hab_hau" => $tarif_hab_haut,
        "surface" => $surface,
        "id_p" => $id_p,
        "description_hab" => $description_hab,
        "titre_hab" => $titre_hab,
        "capacite_hab" => $capacite_hab,
        "etage_ap" => $etage_ap,
        "type_ap" => $type_ap,
        "ref_hab" => $ref_hab
    ]);
```

```php
        if($nbPhotos == 3){
            $unControleur->deletePhotos($refHab);

            foreach ($photos['name'] as $i => $name) {
                $extension = strtolower(pathinfo($name, PATHINFO_EXTENSION));
                $newName = uniqid("photo_") . "." . $extension;

                move_uploaded_file($photos['tmp_name'][$i], "images/habitations/" . $newName);

                $isPrincipal = ($i === 0) ? 1 : 0;

                $unControleur->insertPhoto([
                    "ref_hab" => $refHab,
                    "url_photo" => $newName,
                    "is_principal" => $isPrincipal
                ]);
            }
        }

        header("Location:index.php?page=29");
        exit;
    }
}

if(isset($_POST['annuler']) || isset($_POST['effacer'])){
    header("Location:index.php?page=29");
    exit;
}

if(isset($_POST['filtrer'])){
    $filtre = $_POST['filtre'];
    $lesAppartements = $unControleur->selectLikeAppartement($filtre);
}else{
    $lesAppartements = $unControleur->selectAllAppartement();
}
require_once ("vue/vue_appartement.php");
?>
```

4.3 Gestion des réservations

Le parcours de réservation d'un client suit plusieurs étapes :

- Consultation de la fiche habitation avec galerie photos et formulaire de réservation
- Calcul dynamique du prix via JavaScript (tarif moyen × nombre de nuits)
- Page de récapitulatif permettant de vérifier et confirmer la réservation
- Confirmation avec message de succès et accès à l'espace personnel

Les données de réservation transitent par la session PHP entre les étapes, évitant la re-soumission accidentelle. La capacité maximale de l'habitation est respectée côté serveur. Un client peut annuler une réservation depuis son espace tant que la date de début n'est pas passée.

```php
<?php
    if(isset($_GET['ref_hab'])){
        $refHab = $_GET['ref_hab'];

        $habitation = $unControleur->selectWhereHabitation($refHab);
        $photoPrincipal = $unControleur->selectPhotoPrincipalHabitation($refHab);
        $photosSecondaires = $unControleur->selectAllPhotosWhere($refHab);
    }
?>
```

```javascript
<script>
  function calculPrix() {
      const prixParNuit = <?=$habitation['tarif_hab_moy']?>;
      const debut = new Date(document.getElementById('arrivee').value);
      const fin = new Date(document.getElementById('depart').value);

      let prix = prixParNuit;
      let nbJours = 1;

      if (!isNaN(debut.getTime()) && !isNaN(fin.getTime()) && fin > debut) {
          const diffTime = fin - debut;
          nbJours = diffTime / (1000 * 60 * 60 * 24);
          prix = nbJours * prixParNuit;

          document.getElementById('prixTotal').textContent = `${nbJours} nuits – ${prix} €`;
      } else {
          document.getElementById('prixTotal').textContent = '1 nuit –'+prixParNuit+'€';
      }

      document.getElementById('prixTotalHidden').value = prix;
      document.getElementById('nbJours').value = nbJours;
      document.getElementById('prixParNuit').value = prixParNuit;
  }

  document.addEventListener("DOMContentLoaded", () => {
      const arrivee = document.getElementById('arrivee');
      const depart = document.getElementById('depart');
      const ajd = new Date().toISOString().split('T')[0];
      const demain = new Date(ajd);
      demain.setDate(demain.getDate() + 1);
      arrivee.min = ajd;
      arrivee.value = ajd;
      depart.min = demain.toISOString().split('T')[0];
      depart.value = demain.toISOString().split('T')[0];


      arrivee.addEventListener('change', () => {
          depart.min = arrivee.value;
          calculPrix();
      });

      depart.addEventListener('change', calculPrix);

      calculPrix();
  });
</script>
```

```php
<?php
    if(isset($_POST['reserver'])){
        $arrivee = $_POST['arrivee'];
        $depart = $_POST['depart'];
        $voyageurs = $_POST['voyageurs'];

        if(isset($_SESSION['email']) && $_SESSION['role'] == 'client'){
            if($depart > $arrivee){
                if($voyageurs > 0 && $voyageurs <= $habitation['capacite_hab']){
                    //gérer la resa ici
                    $_SESSION['reservation'] = $_POST;
                    header("Location: index.php?page=11");
                    exit;
                }else{
                    echo "<p style='color:red'>";
                    echo "Veuillez choisir un nombre de voyageurs respectant les limites";
                    echo "</p>";
                }

            }else{
                echo "<p style='color:red'>";
                echo "Veuillez choisir une date depart supérieur a la date d'arrivée";
                echo "</p>";
            }

        }else{
            $_SESSION['msg-login-resa'] = "Veuillez vous connecter ou créer un compte pour pouvoir réserver
                                           une habitation !";
            header("Location: index.php?page=8");
        }
    }

    require_once("vue/vue_reservation_habitation.php");
?>
```

Un événement MySQL (archiveReservationExpiree) s'exécute quotidiennement pour archiver automatiquement les réservations dont la date de fin est dépassée.

```sql
/************************************** events **************************************/
set global event_scheduler = on;

delimiter //

create event archiveReservationExpiree
on schedule every 1 day
starts curdate()
on completion preserve
enable
do
begin
insert into archivereservation
select *, curdate() from reservation where date_fin < curdate();
delete from reservation where date_fin < curdate();

end //

delimiter ;
```

4.4 Génération de contrats PDF

La bibliothèque FPDF génère à la volée deux types de contrats :

Contrat	Destinataire	Contenu
Contrat de réservation	Client	Identité du client, détail de la réservation (dates, habitation, nb personnes), bloc de signatures
Contrat de mandat locatif	Propriétaire	Identité du propriétaire, caractéristiques complètes de l'habitation (tarifs, surface, type), bloc de signatures

Les PDFs sont générés dynamiquement à la demande (lien depuis l'espace client/propriétaire) et s'ouvrent dans un nouvel onglet sans stockage serveur.

```php
<?php
session_start();
require_once("./fpdf/fpdf.php");
require_once("controleur/controleur.class.php");


class PDF extends FPDF{

    function header(){
        $this->SetFont("Arial","B",12);
        $this->Cell(0,10,"CONTRAT DE RESERVATION",1,1,"C");
    }

    function body(){
        $unControleur = new Controleur();

        $id = $_SESSION['id'];
        $leClient = $unControleur->selectWhereIdClient($id);
        $refRes = isset($_GET['ref_res']) ? $_GET['ref_res'] : null;
        $reservationClient = $unControleur->selectWhereReservation($refRes);

        $this->SetFont("Arial","B",12);
        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Identité du client (ou de son représentant légal)"),0,1,"
        $this->MultiCell(0,10,iconv('UTF-8', 'ISO-8859-1',"Nom : ".($leClient['nom']).
                              "\nPrenom : ".($leClient['prenom']).
                              "\nAdresse : ".($leClient['adresse']).
                              "\nCode postal : ".($leClient['cp']).
                              "\nVille : ".($leClient['ville']).
                              "\nTéléphone : ".($leClient['tel']).
                              "\nRIB : ".($leClient['RIB'])),1,"L");

        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Réservation"),0,1,"C");
```

```php
        if (!empty($reservationClient)) {
            $this->MultiCell(0,10,iconv('UTF-8', 'ISO-8859-1',"Référence : ".($reservationClient['ref_res'])
                                  "\nDate réservation : ".($reservationClient['date_res']).
                                  "\nNombre personnes : ".($reservationClient['nb_perso']).
                                  "\nDate début : ".($reservationClient['date_debut']).
                                  "\nDate fin : ".($reservationClient['date_fin']).
                                  "\nIdentifant habitation : ".($reservationClient['ref_hab'])),1,"L");
        }else {
            $this->Cell(0, 10,iconv('UTF-8','ISO-8859-1', "Aucune information trouvée pour la réservation :
        }

    }

    function footer(){
        $this->SetFont("Arial","B",12);
        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Signature des deux parties"),0,1,"C");
        $this->Cell(0,30,iconv('UTF-8', 'ISO-8859-1',"Le Client"),1,1,"L");
        $this->Cell(0,30,iconv('UTF-8', 'ISO-8859-1',"Neige & Soleil"),1,1,"L");
    }

}

$pdf = new PDF();
$pdf->AddPage();
$pdf->body();
$pdf->Output();

?>
```

```php
<?php
session_start();
require_once("./fpdf/fpdf.php");
require_once("controleur/controleur.class.php");



class PDF extends FPDF{

    function header(){
        $this->SetFont("Arial","B",12);
        $this->Cell(0,10,"CONTRAT DE MANDAT LOCATIF",1,1,"C");
    }

    function body(){
        $unControleur = new Controleur();

        $refHab = isset($_GET['ref_hab']) ? $_GET['ref_hab'] : null;

        $maison = $unControleur->selectWhereMaison($refHab);
        $appartement = $unControleur->selectWhereAppartement($refHab);
        $id = $_SESSION['id'];
        $leProprietaire = $unControleur->selectWhereIdProprietaire($id);

        $this->SetFont("Arial","B",12);
        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Identité du propriétaire (ou de son représentant légal)"),0,1,"C");
        $this->MultiCell(0,10,iconv('UTF-8', 'ISO-8859-1',"Nom : ".($leProprietaire['nom']).
                                "\nPrenom : ".($leProprietaire['prenom']).
                                "\nAdresse : ".($leProprietaire['adresse']).
                                "\nCode postal : ".($leProprietaire['cp']).
                                "\nVille : ".($leProprietaire['ville']).
                                "\nTéléphone : ".($leProprietaire['tel']).
                                "\nRIB : ".($leProprietaire['RIB'])
                                ),1,"L");

        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Informations habitation"),0,1,"C");
```

```php
    if (!empty($maison)) {
            $this->MultiCell(0,10,iconv('UTF-8', 'ISO-8859-1',"Type : ".($maison['type_hab']).
                                "\nAdresse : ".($maison['adr_hab']).
                                "\nCode postal : ".($maison['cp_hab']).
                                "\nVille : ".($maison['ville_hab']).
                                "\nMontant de location hebdomadaire saison basse : ".($maison['tarif_hab_bas']).
                                "\nMontant de location hebdomadaire saison moyenne : ".($maison['tarif_hab_moy']).
                                "\nMontant de location hebdomadaire saison haute : ".($maison['tarif_hab_hau']).
                                "\nSurface : ".($maison['surface'])."m2").
                                "\nCaractéristiques : ".($maison['carac_m'])),1,"L");
    }
    elseif (!empty($appartement)) {
            $this->MultiCell(0,10,iconv('UTF-8', 'ISO-8859-1',"Type : ".($appartement['type_hab']).
                                "\nAdresse : ".($appartement['adr_hab']).
                                "\nCode postal : ".($appartement['cp_hab']).
                                "\nVille : ".($appartement['ville_hab']).
                                "\nMontant de location hebdomadaire saison basse : ".($appartement['tarif_hab_bas']).
                                "\nMontant de location hebdomadaire saison moyenne : ".($appartement['tarif_hab_moy']).
                                "\nMontant de location hebdomadaire saison haute : ".($appartement['tarif_hab_hau']).
                                "\nSurface : ".($appartement['surface'])."m2").
                                "\nEtage : ".($appartement['etage_ap']).
                                "\nType d'appartement : ".($appartement['type_ap'])),1,"L");
    }
    else {
        $this->Cell(0, 10, "Aucune information trouvee pour l'habitation ref : " . $refHab, 1, 1);
    }
}
```

```php
    function footer(){
        $this->SetFont("Arial","B",12);
        $this->Cell(0,15,iconv('UTF-8', 'ISO-8859-1',"Signature des deux parties"),0,1,"C");
        $this->Cell(0,30,iconv('UTF-8', 'ISO-8859-1',"Le propriétaire"),1,1,"L");
        $this->Cell(0,30,iconv('UTF-8', 'ISO-8859-1',"Neige & Soleil"),1,1,"L");
    }

}

$pdf = new PDF();
$pdf->AddPage();
$pdf->body();
$pdf->Output();
?>
```

4.5 Réinitialisation du mot de passe

Le processus de réinitialisation utilise PHPMailer (via SMTP Gmail) et se déroule en trois étapes : saisie de l'email, réception d'un code à 6 chiffres valable 15 minutes, puis saisie du code et du nouveau mot de passe. Des validations sont effectuées à chaque étape.

```php
<?php

use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;

require_once __DIR__ . "/../PHPMailer-master/src/Exception.php";
require_once __DIR__ . "/../PHPMailer-master/src/PHPMailer.php";
require_once __DIR__ . "/../PHPMailer-master/src/SMTP.php";


$erreurs = [];

    if(isset($_POST['annuler'])){
        header("Location:index.php?page=8");
        exit;
    }

    if(isset($_POST['verifier'])){
        $email = trim($_POST['email']);
        $compte_exist = $unControleur->selectWhereEmailUtilisateur($email);

        $regexEmail = '/^[A-Za-zÀ-ÖØ-öø-ÿ0-9._-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}/u';

        if($email == ""){
            $erreurs[] = "Veuillez indiquer votre email";
        }elseif(!preg_match($regexEmail,$email)){
            $erreurs[] = "Veuillez saisir un email valide";
        }
        elseif($compte_exist == null){
            $erreurs[] = "Aucun compte associé a cet email";
        }
```

```php
if(!empty($erreurs)){
    $_SESSION['msg-erreur'] = $erreurs;
}
else{
    $code = str_pad(rand(0, 999999), 6, '0', STR_PAD_LEFT);
    $unControleur->resetCode($email,$code);

    $mail = new PHPMailer(true);
    $mail->CharSet = 'UTF-8';

    $mail->SMTPOptions = array('ssl' => array(
                                'verify_peer' => false,
                                'verify_peer_name' => false,
                                'allow_self_signed' => true
                                )
                            );
    try {
        // Paramètres Serveur (Exemple Gmail)
        $mail->isSMTP();
        $mail->Host       = 'smtp.gmail.com';
        $mail->SMTPAuth   = true;
        $mail->Username   = 'neigeetsoleil2026@gmail.com';
        $mail->Password   = 'kgwxwfhtewtlhgce'; // mdp application
        $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
        $mail->Port       = 587;

        // Destinataires
        $mail->setFrom('neigeetsoleil2026@gmail.com', 'Neige et soleil');
        $mail->addAddress($email);

        // Contenu
        $mail->isHTML(true);
        $mail->Subject = 'Votre code de réinitialisation';
        $mail->Body    = "Votre code est : <b>$code</b>. Il est valable 15 minutes.";

        $mail->send();

        // 4. Stocker l'email en session pour la page suivante
        $_SESSION['email_reset'] = $email;
        header("Location: index.php?page=18");
        exit;
```

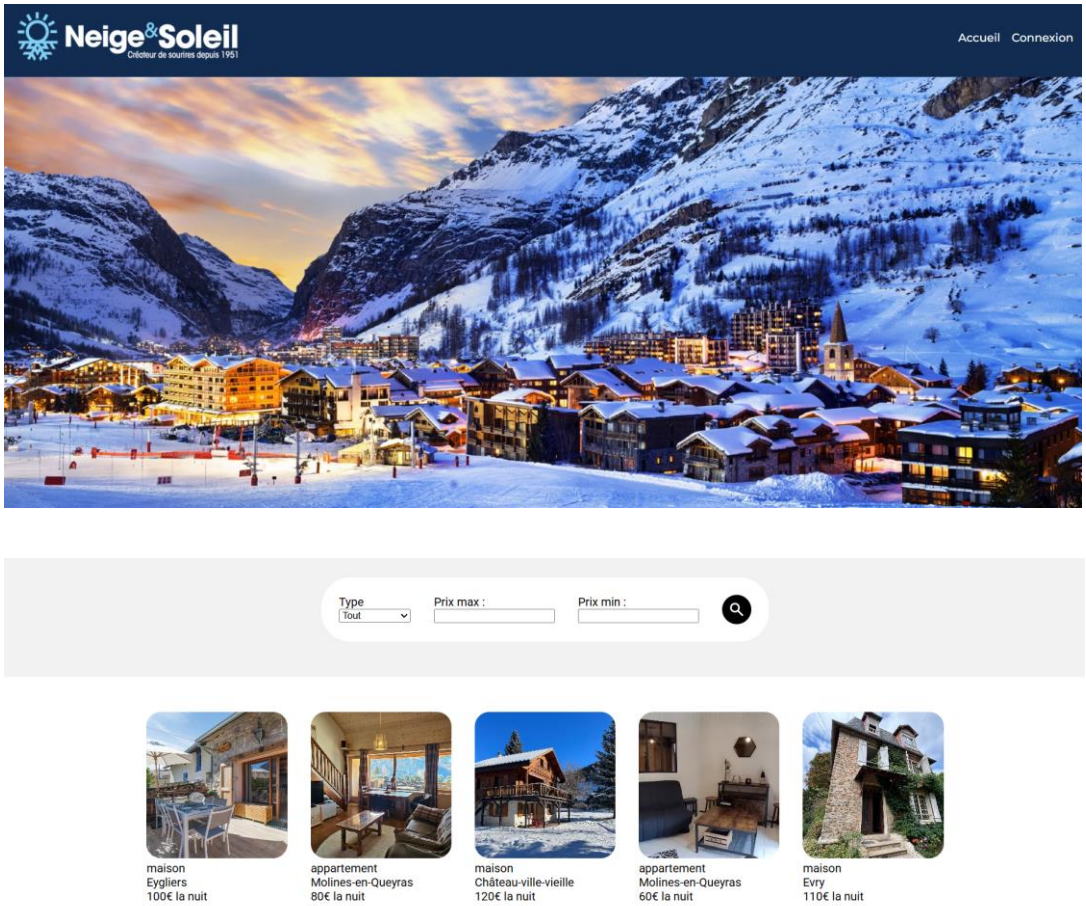
```php
            } catch (Exception $e) {
                $_SESSION['msg-erreur'] = ["L'email n'a pas pu être envoyé. Erreur: {$mail->ErrorInfo}"];
            }
        }
    }

    require_once("vue/vue_mdp_oublie.php");
?>
```

V. Résultat final

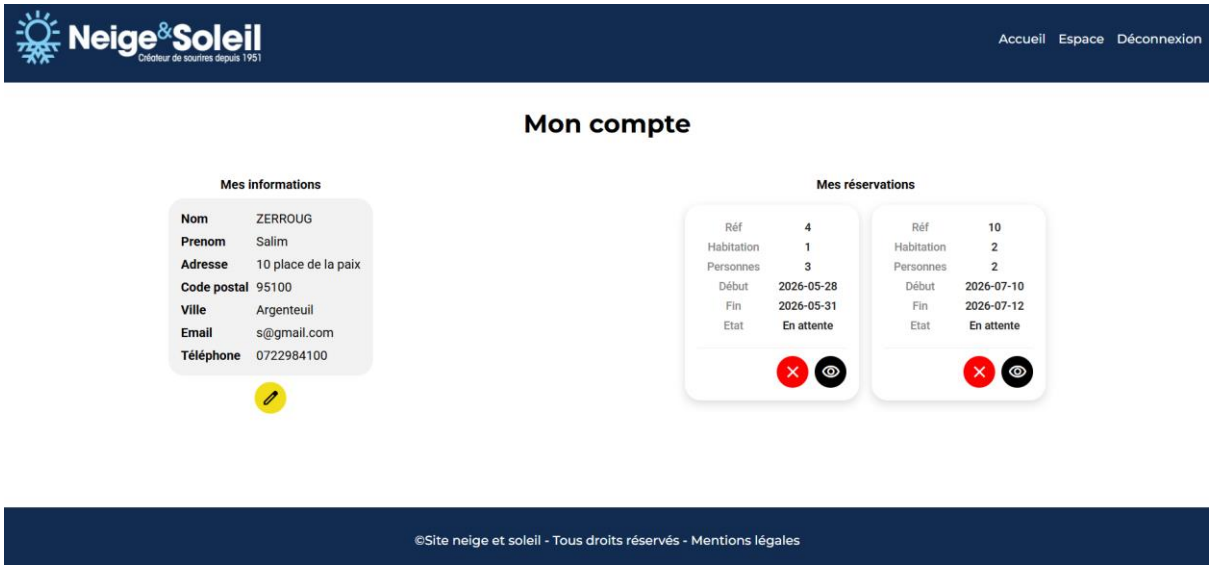
Page d'accueil

La page d'accueil affiche un slider d'images défilant automatiquement (JavaScript, intervalle 3 secondes) suivi d'un moteur de recherche permettant de filtrer les annonces par type d'habitation (maison/appartement) et par fourchette de prix. Les résultats s'affichent sous forme de cartes avec photo principale, type, ville et tarif.



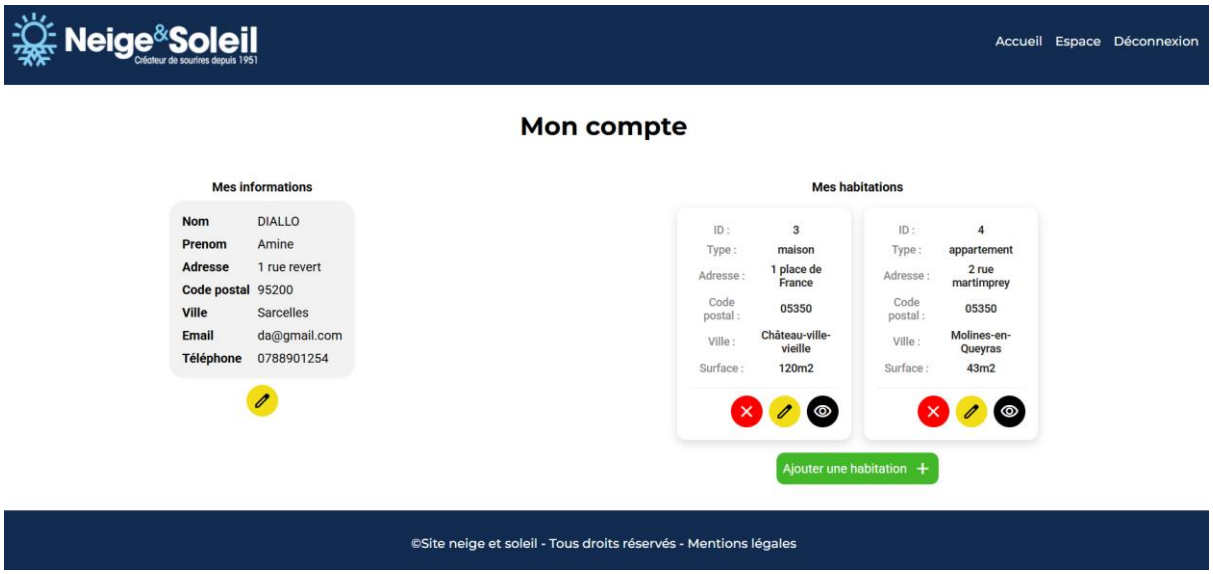
Espace client

L'espace personnel du client affiche ses informations de profil avec un bouton de modification, et la liste de ses réservations sous forme de cartes. Chaque carte indique la référence, l'habitation, les dates, le nombre de voyageurs, le statut et propose deux actions : annuler la réservation ou voir et télécharger le contrat PDF.



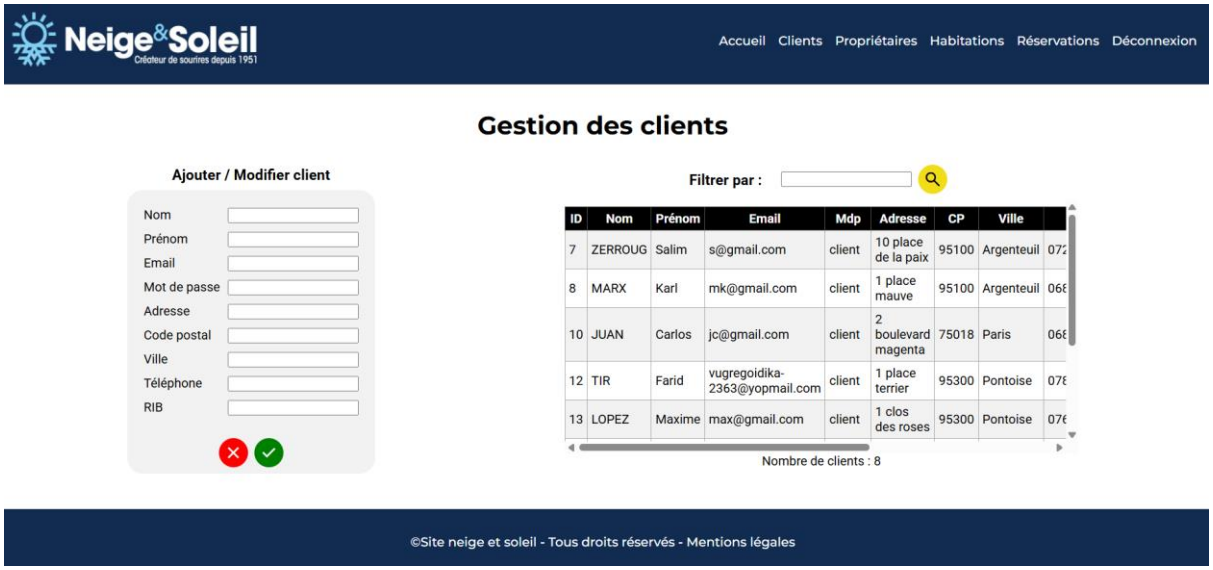
Espace propriétaire

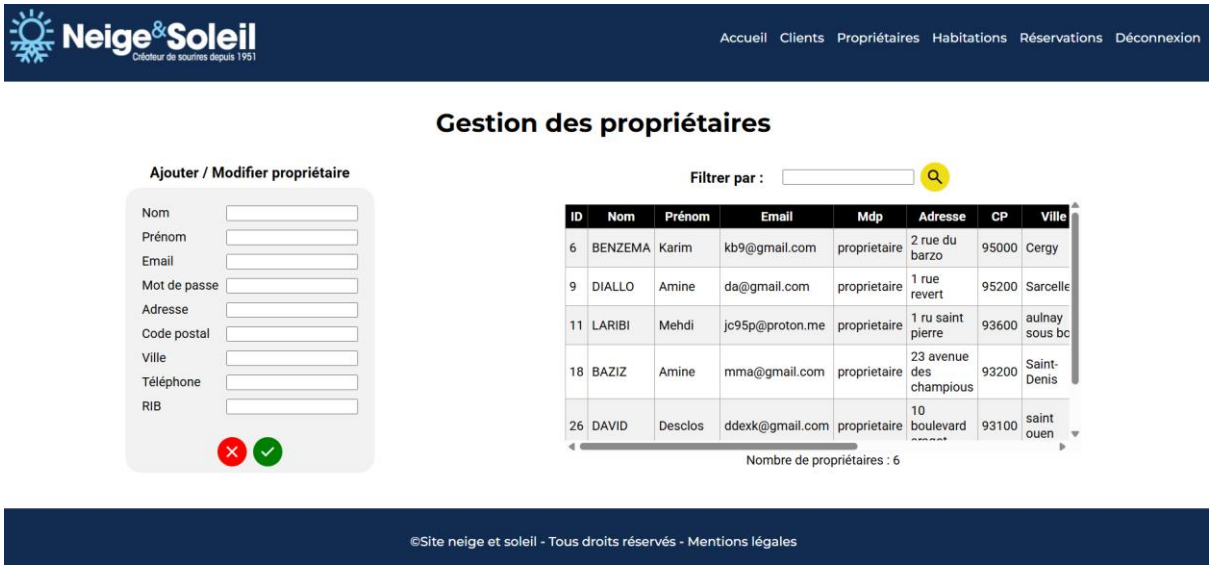
L'espace propriétaire présente les informations du compte et la liste des habitations du propriétaire. Pour chaque habitation, il peut supprimer l'annonce, la modifier (tarifs, description, photos) ou télécharger son contrat de mandat PDF. Un bouton d'ajout permet de créer une nouvelle annonce en choisissant le type (maison ou appartement).



Interface d'administration

L'administrateur dispose de quatre sections accessibles via le menu de navigation : Clients, Propriétaires, Habitations et Réservations. Chaque section présente un formulaire d'ajout/modification à gauche et un tableau de consultation filtrable à droite, avec des boutons d'action (modifier, supprimer).

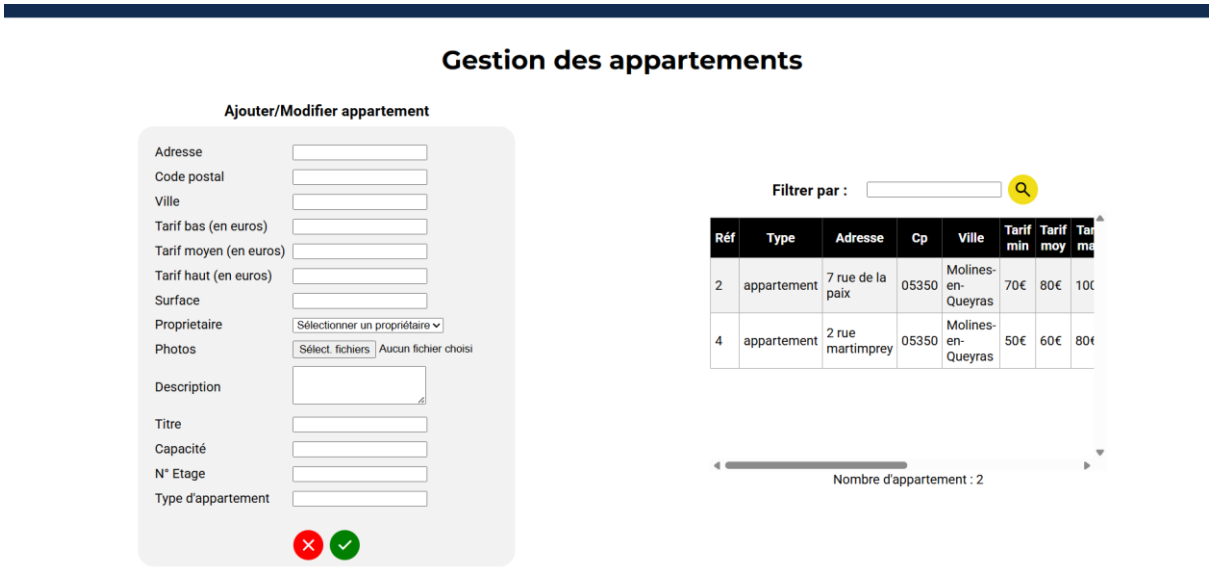


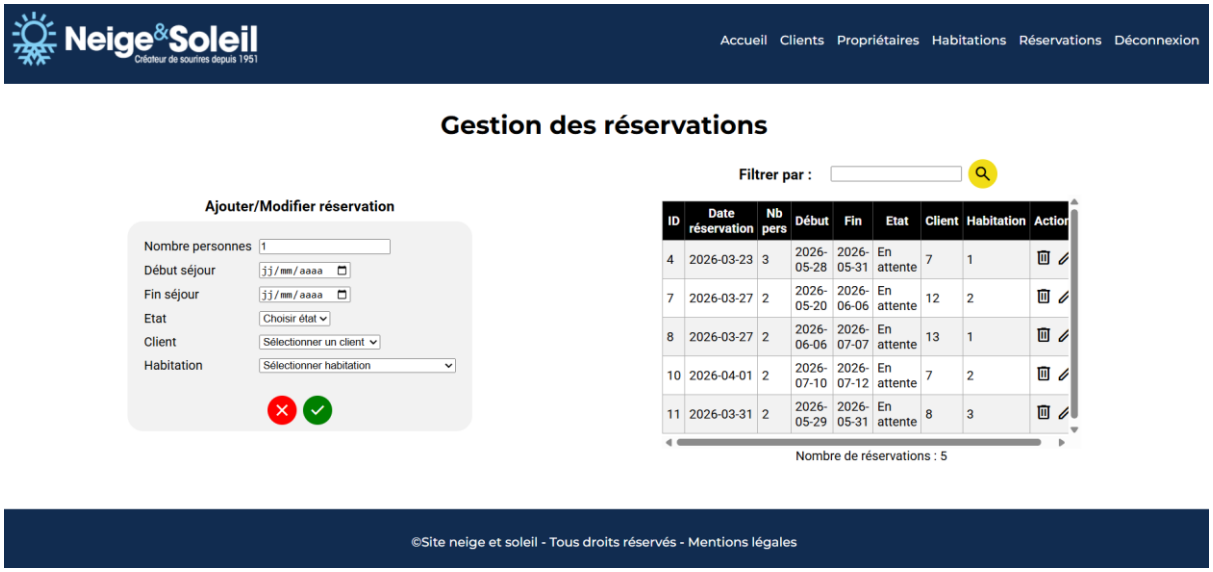



Neige&Soleil
Créateur de sourires depuis 1951
Accueil Clients Propriétaires Habitations Réservations Déconnexion
Gestion des habitations
Maisons
Appartements


©Site neige et soleil - Tous droits réservés - Mentions légales


Neige&Soleil
Créateur de sourires depuis 1951
Accueil Clients Propriétaires Habitations Réservations Déconnexion
Gestion des maisons
Ajouter/Modifier maison
Adresse
Code postal
Ville
Tarif bas (en euros)
Tarif moyen (en euros)
Tarif haut (en euros)
Surface
Proprietaire
Sélectionner un propriétaire
Photos
Sélect. fichiers
Aucun fichier choisi
Description
Titre
Capacité
Caractéristique
Filtrer par :
Réf Type Adresse Cp Ville Tarif min Tarif moy Tarif max
1 maison 12 rue des chauffours 05600 Eygliers 80€ 100€ 120€
3 maison 1 place de France 05350 Château-ville-vieille 100€ 120€ 140€
5 maison 20 square illyre 91042 Evry 100€ 110€ 120€
Nombre de maison : 3


Gestion des appartements
Ajouter/Modifier appartement
Adresse
Code postal
Ville
Tarif bas (en euros)
Tarif moyen (en euros)
Tarif haut (en euros)
Surface
Proprietaire
Sélectionner un propriétaire
Photos
Sélect. fichiers
Aucun fichier choisi
Description
Titre
Capacité
N° Etage
Type d'appartement
Filtrer par :
Réf Type Adresse Cp Ville Tarif min Tarif moy
2 appartement 7 rue de la paix 05350 Molines-en-Queyras 70€ 80€
4 appartement 2 rue martimprey 05350 Molines-en-Queyras 50€ 60€
Nombre d'appartement : 2


Neige&Soleil
Créateur de sourires depuis 1951
Accueil Clients Propriétaires Habitations Réservations Déconnexion
Gestion des réservations
Ajouter/Modifier réservation
Nombre personnes 1
Début séjour jj/mm/aaaa
Fin séjour jj/mm/aaaa
Etat Choisir état
Client Sélectionner un client
Habitation Sélectionner habitation
Filtrer par :
ID | Date réservation | Nb pers | Début | Fin | Etat | Client | Habitation | Actions
4 | 2026-03-23 | 3 | 2026-05-28 | 2026-05-31 | En attente | 7 | 1
7 | 2026-03-27 | 2 | 2026-05-20 | 2026-06-06 | En attente | 12 | 2
8 | 2026-03-27 | 2 | 2026-06-06 | 2026-07-07 | En attente | 13 | 1
10 | 2026-04-01 | 2 | 2026-07-10 | 2026-07-12 | En attente | 7 | 2
11 | 2026-03-31 | 2 | 2026-05-29 | 2026-05-31 | En attente | 8 | 3
Nombre de réservations : 5
©Site neige et soleil - Tous droits réservés - Mentions légales

VI. Conclusion

La réalisation du projet Neige & Soleil a permis de mettre en pratique l'ensemble des compétences attendues en développement web full-stack dans un contexte applicatif concret et cohérent. Le projet couvre aussi bien la conception (modélisation de la base de données, architecture MVC) que l'implémentation complète d'une application multi-utilisateur.

Bilan technique

Le développement de cette application a permis de mettre en pratique :

- La conception d'une base de données relationnelle avec triggers, événements et fonctions MySQL
- L'implémentation du pattern MVC en PHP avec routeur central et contrôle d'accès par rôles
- La validation rigoureuse des données côté serveur par expressions régulières
- La gestion de l'upload de fichiers et des photos associées aux annonces
- La génération dynamique de documents PDF avec FPDF
- L'envoi d'emails transactionnels avec PHPMailer et SMTP
- La gestion des sessions et la sécurité applicative (XSS, contrôle de rôle, expiration de mot de passe)

Perspectives d'évolution

Plusieurs améliorations pourraient enrichir ce projet dans une version future :

- Intégration d'un calendrier de disponibilité pour éviter les chevauchements de réservations
- Système de notation et d'avis des habitations par les clients
- Interface de paiement en ligne sécurisée
- Notifications email automatiques à la confirmation de réservation
- Hachage des mots de passe (bcrypt) pour renforcer la sécurité
- Pagination des listes dans l'interface d'administration

En somme, ce projet constitue une base fonctionnelle solide illustrant les mécanismes essentiels du développement web PHP/MySQL : il produit un résultat utilisable et déployable, tout en démontrant une maîtrise des bonnes pratiques de développement.