Le dossier ci -joint est une documentation destinée aux développeurs. Il présente l'aspect technique des composantes du code source Du Site web ROILLE

Documentation Technique

ROILLE

Table des matières

Cette documentation technique est destinée aux développeurs souhaitant comprendre et/ou apporter des modifications/améliorations à l'application web.

I. Site ROILLE

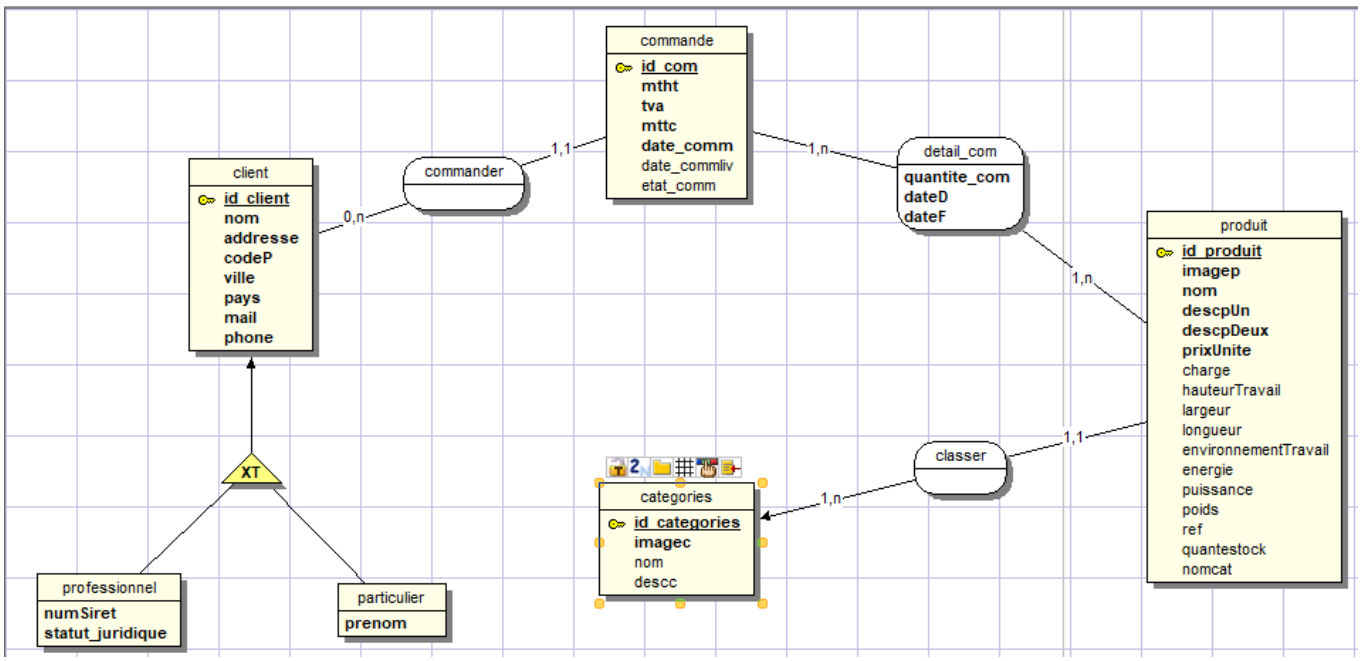
1. Présentation

L'application web demandée par le directeur de l'entreprise ROILLE permet (une fois réalisée et mis en ligne) aux client de s'inscrir/se connecter, avoir accés au catalogue des matériaux à louer et pouvoir réserver un matériel.

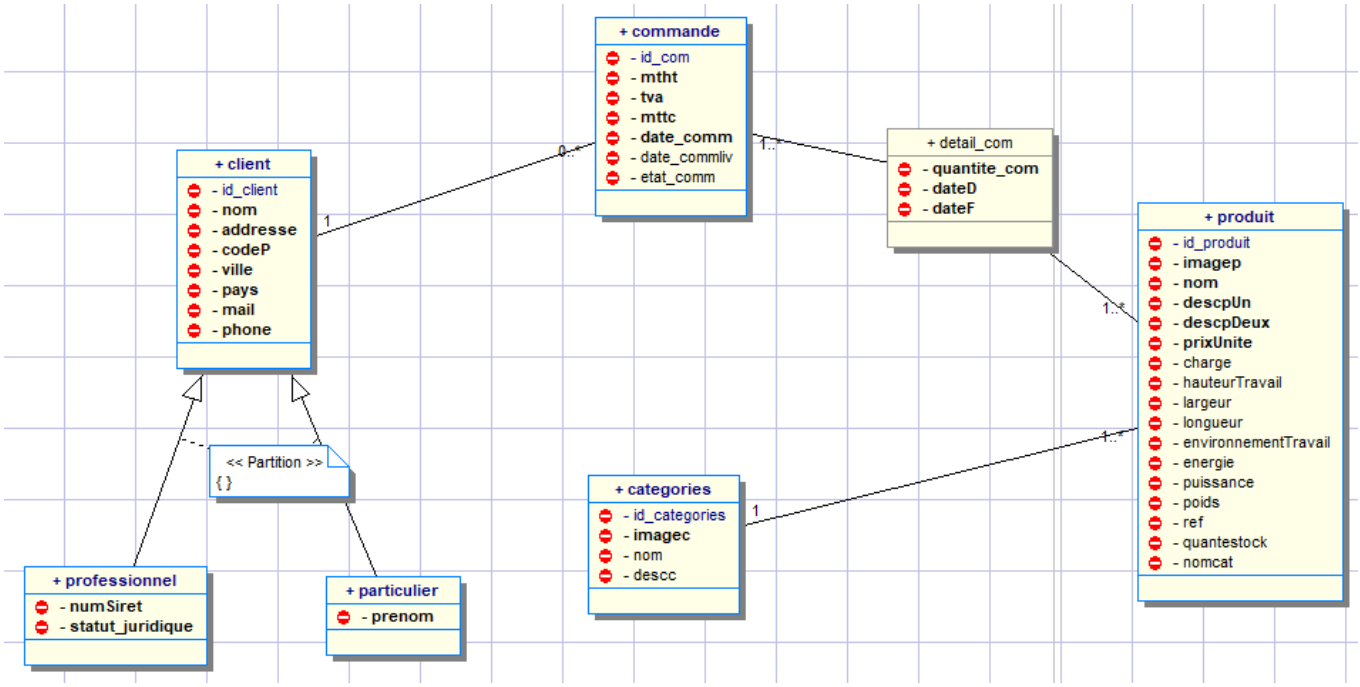
Pour plus d'informations, voir le fichier Cahier des charges.

2. Structure du site

2.1. Modéle conceptuel de données :



2.2. Diagramme de classes:

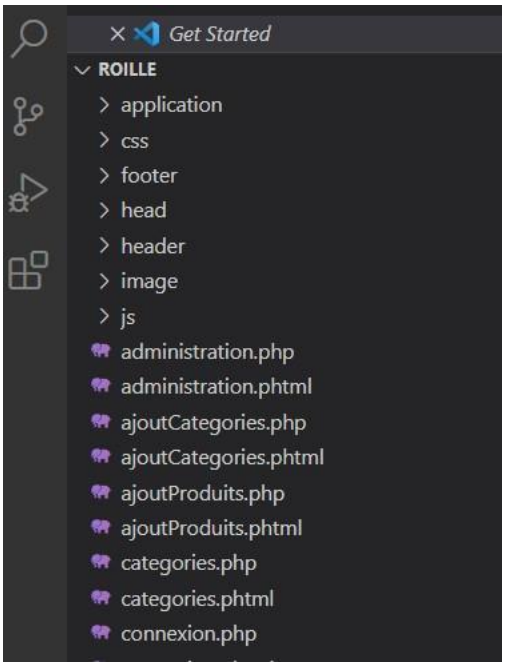


2. Technologies utilisées

Les languages utilisées pour la conception du site sont : PHP,JAVASCRIPT,JQUERY.

II. Développement

1. Structure du code



- Fichier database qui contient la function qui permet de se connecter à la base de données et les requétes utilisées (insert/update/delete) ainsi que les triggers, les fonctions et Procedures stockées.
- Fichier layout qui contient le page principale du site.
- Le dossier js: Contient les requête ajax et les code javascript.

JDBC (Java Data
Base Connection).

- Le dossier Image: Contient les images et logo de l'application.

2. La connexion à la base de données

```php
<?php

function createconnection(){
    $pdo=new PDO('mysql:host=localhost;dbname=roille;charset=utf8','root','');
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    return $pdo;
}
```

Connexion à la base de données en utilisant l'API (Application Programming Interface) PDO (PHP Data Objects).

3. Inscription d'un nouveau client

Particulier:

```php
function registerParticulier($nom,$prenom,$mdp,$addresse,$codeP,$ville,$pays,$phone,$mail){
    $pdo=createconnection();
    $req=$pdo->prepare("INSERT INTO particulier (avatar,nom,prenom,mdp,addresse,codeP,ville,
                        pays,phone,mail)
                        values (null,:nom,:prenom,:mdp,:addresse,:codeP,:ville,:pays,:phone,
                        :mail)");
    $req->execute(array(
        ':nom'=>$nom,
        ':prenom'=>$prenom,
        ':mdp'=>$mdp,
        ':addresse'=>$addresse,
        ':codeP'=>$codeP,
        ':ville'=>$ville,
        ':pays'=>$pays,
        ':phone'=>$phone,
        ':mail'=>$mail
    ));
}
```

```sql
/*
 trigger pour l'ajout d'un particulier
*/
drop trigger if exists ajout_particulier ;
 delimiter //
 create trigger ajout_particulier
 before insert on particulier
 for each row
 begin
declare a,x,e int ;
select max(id_client) into x
from professionnel;
if x =0
  then
      set  x= 1;
      else
       set new.id_client= x+1;
end if;
```

Professionnel :

```php
function registerProfessionnel($nom,$mdp,$addresse,$codeP,$ville,$pays,$phone,$mail,$numSiret,
                               $statut_juridique){
    $pdo=createconnection();
    $req=$pdo->prepare("INSERT INTO professionnel
                        (avatar,nom,mdp,addresse,codeP,ville,pays,phone,mail,numSiret,
                        statut_juridique)
                        values (null,:nom,:mdp,:addresse,:codeP,:ville,:pays,:phone,:mail,
                        :numSire,:statut_jurid)");
    $req->execute(array(
        ':nom'=>$nom,
        ':mdp'=>$mdp,
        ':addresse'=>$addresse,
        ':codeP'=>$codeP,
        ':ville'=>$ville,
        ':pays'=>$pays,
        ':phone'=>$phone,
        ':mail'=>$mail,
        ':numSire'=>$numSiret,
        ':statut_jurid'=>$statut_juridique
    ));
}
```

```sql
/*
 trigger pour l'ajout d'un professionnel
*/
drop trigger if exists ajout_professionnel ;
 delimiter //
 create trigger ajout_professionnel
 before insert on professionnel
 for each row
 begin
declare x,e int ;
select max(id_client) into x
from particulier;
if x =0
  then
      set  x= 1;
      else
       set new.id_client= x+1;
end if;
    insert into client values (new.id_client,null,new.nom,new.mdp,new.addresse,
    new.codeP,new.ville,new.pays,new.phone,new.mail,null);
end //
delimiter ;
```

4. Authentification d'un client

```php
<?php
session_start();
if(isset($_POST['ok'])){
    $login=$_POST["mail"];
    $mdp=$_POST["mdp"];
    $id= mysqli_connect("127.0.0.1","root","","roille");
    $connect= mysqli_query($id,"SET NAMES 'utf8'");
    $req="select * from client where mail='$login' and mdp='$mdp'";
    $result=mysqli_query($id,$req);
    if(mysqli_num_rows($result)>0)
    {
        if (isset($_POST['rememberMe'])){
            setcookie('nom',$login,time()+365*24*3600,null,null,false,true);
            setcookie('mdp',$mdp,time()+365*24*3600,null,null,false,true);
        }
        $ligne=mysqli_fetch_assoc($result);
        $_SESSION['id_client']=$ligne['id_client'];$_SESSION['nom']=$ligne['nom'];
        $_SESSION['addresse']=$ligne['addresse'];$_SESSION['codeP']=$ligne['codeP'];
        $_SESSION['ville']=$ligne['ville'];$_SESSION['pays']=$ligne['pays'];
        $_SESSION['phone']=$ligne['phone'];$_SESSION['mail']=$ligne['mail'];
        header('location:index.php?id='. $_SESSION["id_client"]);
    }
    else{$erreur= "connexion impossible, login ou mot de passe incorrect ....";}
}
```

Connexion à la base de données et récupération des données de l'utilisateur qui se connecte en session.

```php
function modifClientParc($avatar,$nom,$prenom,$mdp,$addresse,$codeP,$ville,$pays,$phone,$mail,
                         $id){
    $pdo=createconnection();
        $req=$pdo->prepare('UPDATE particulier SET avatar=?,nom=?,prenom=?,mdp=?,addresse=?,
                            codeP=?,ville=?,pays=?,
                            phone=?,mail=? WHERE id_client=?');
        $req->execute(array($avatar,$nom,$prenom,$mdp,$addresse,$codeP,$ville,$pays,$phone,
                            $mail,$id));
}

function modifClientPro($avatar,$nom,$addresse,$mdp,$codeP,$ville,$pays,$phone,$mail,$statuJ,
                        $numS,$id){
    $pdo=createconnection();
        $req=$pdo->prepare('UPDATE professionnel SET avatar=?,nom=?,mdp=?,addresse=?,codeP=?,
                            ville=?,pays=?,
                            phone=?,mail=?,numSiret=?,statut_juridique=? WHERE id_client=?');
        $req->execute(array($avatar,$nom,$addresse,$mdp,$codeP,$ville,$pays,$phone,$mail,
                            $statuJ,$numS,$id));
}
```

L'utilisateur peut modifier ces données via un formulaire.

5. Affichage des categories

```php
<?php
session_start();
require('application/database.php');

if(isset($_SESSION['id_client'])){
    $userInfo=getIdClient($_SESSION['id_client']);
}

$categories=listCategories();


// Sélection et affichage du template PHTML.
$template = 'categories';
include 'layout.phtml';
```

Afficher toutes les catégories sous forme de grille.

```php
<section class="grille grilleCategory">
    <?php foreach($categories as $categorie) : ?>

            <div>
                <p><img src="image/<?= $categorie['imagec']; ?>" alt="image de catégorie"></p>
                <a href="produits.php?categorie=<?= $categorie['nom']; ?>">voir plus</a>
                <h3><?= $categorie['nom']; ?></h3>
                <p><?= $categorie['descc']; ?></p>
            </div>

    <?php endforeach;?>
</section>
```

6. Affichage des produits

```php
<?php
session_start();
require('application/database.php');
$nom=$_GET['categorie'];

if(isset($_SESSION['id_client'])){
    $userInfo=getIdClient($_SESSION['id_client']);
}

//liste des articles
$produits=getListProduitById($nom);
//liste des catégories
$categories=listCategories();

// Sélection et affichage du template PHTML.
$template = 'produits';
include 'layout.phtml';
```

Affichage des produits sous forme de grille liés à une catégorie.

```php
<section class="categoryProductGrille">
    <section  class="grille grilleCategory">
        <?php foreach($produits as $produit) : ?>

            <div>
                <p><img src="image/<?= $produit['imagep']; ?>" alt="image du produit"></p>
                <h3><?= $produit['nomp']; ?></h3>
                <p><?= $produit['prixUnite']; ?> €</p>
                <a href="detailsProduits.php?id_produit=<?= $produit['id_produit']; ?>">
                    voir plus</a>
            </div>

        <?php endforeach;?>
    </section>
</section>
```

7. Affichage du détail d'un article à louer

```php
function getDetailProduitById($id){
    $pdo=createconnection();
    $req=$pdo->prepare('SELECT * FROM produit WHERE id_produit=?');

    $req->execute(array($id));
    $details=$req->fetchAll(PDO::FETCH_ASSOC);
    return $details;
}
```

```php
<?php foreach($produit as $unProduit) : ?>
    <div>
        <div class="detailproduit">
            <img src="image/<?= $unProduit['imagep']; ?>" alt="image du produit">
        </div><!--
        --><div class="detailproduit">
                <h2><?= $unProduit['nomp']; ?></h2>
                <h3><?= $unProduit['prixUnite']; ?> €</h3>
                <p><?= $unProduit['descpUn']; ?></p>
<?php endforeach;?>
```

8.Gestion de stock:

```sql
/*trigger qui permet la gestion de stock des articles à louer*/
drop trigger if exists gestionStock;
delimiter //
create trigger gestionStock
after insert on detail_com
for each row
begin
update produit
set quantestock=quantestock-new.quantite_com
where id_produit=new.id_produit;
end //
delimiter ;
```

```sql
/*trigger qui permet la gestion de stock des articles à louer*/
drop trigger if exists updateGestionStock;
delimiter //
create trigger updateGestionStock
after update on detail_com
for each row
begin
update produit
set quantestock=quantestock-new.quantite_com
where id_produit=new.id_produit;
end //
delimiter ;
```

```sql
/*trigger qui permet la gestion de stock des articles à louer*/
drop trigger if exists deleteGestionStock;
delimiter //
create trigger deleteGestionStock
after delete on detail_com
for each row
begin
update produit
set quantestock=quantestock+old.quantite_com
where id_produit=old.id_produit;
end //
delimiter ;
```

9.Gestion du montant de la commande:

```
/*trigger qui permet la gestion de commande*/
drop trigger if exists gestionMontantComm;
delimiter //
create trigger gestionMontantComm
after insert on detail_com
for each row
begin
declare MTH decimal(10,2);
    select sum(prixUnite * new.quantite_com) into MTH
        from detail_com d,produit p
        where  d.id_produit = p.id_produit and id_com=new.id_com;
    update commande
        set mtht=mtht+MTH,
        tva=mtht*0.2,
        mttc=mtht+tva
        where id_com=new.id_com;
end //
delimiter ;

create table indexCom(
    id_ic int auto_increment primary key
);
```

```
/*trigger qui permet la gestion de commande*/
drop trigger if exists updateUestionMontantComm;
delimiter //
create trigger updateGestionMontantComm
before update  on detail_com
for each row
begin
declare qte int ;
declare mth decimal(10,2) default 0;
if new.quantite_com < old.quantite_com
then
    set qte=old.quantite_com - (select new.quantite_com from detail_com
                                where id_com=old.id_com
                                and id_produit=old.id_produit);
    select sum(prixUnite * qte) into mth
    from produit p
    where old.id_produit=p.id_produit ;
    update commande
    set mtht=mtht- mth,
    tva= mtht * 0.2 ,
    mttc=mtht + tva
    where id_com=old.id_com;
else
    set qte=(select new.quantite_com from detail_com
            where id_com=old.id_com
            and id_produit=old.id_produit) -old.quantite_com ;
            select sum(prixUnite * qte) into mth
                from produit p
                where old.id_produit=p.id_produit ;
            update commande
            set mtht=mtht + mth,
            tva= mtht * 0.2 ,
            mttc=mtht + tva
            where id_com=old.id_com;
end if ;
end //
delimiter ;
```

```
/*trigger qui permet la gestion de commande*/
drop trigger if exists deleteUestionMontantComm;
delimiter //
create trigger deleteGestionMontantComm
before delete  on detail_com
for each row
 begin
update commande
    set mtht=mtht - (select sum(prixUnite * old.quantite_com)
                     from produit p
                     where old.id_produit=p.id_produit
                     group by old.id_com ) ,
                tva= mtht * 0.2 ,
                mttc=mtht + tva
              where id_com=old.id_com;
end //
delimiter ;
```

10.Réservation d'un matériel:

```
DELIMITER //
DROP FUNCTION IF EXISTS ressD //
CREATE FUNCTION ressD (idp int(11),datD date,datF date)
 RETURNS INT
BEGIN
declare x int;
select count(*) into x from detail_com where datD between dateD and dateF;
RETURN x;
END//
DELIMITER ;
```

```
DELIMITER //
DROP FUNCTION IF EXISTS ressF //
CREATE FUNCTION ressF (idp int(11),datD date,datF date)
 RETURNS INT
BEGIN
declare y int;
select count(*) into y from detail_com where datF between dateD and dateF;
RETURN y;
END//
DELIMITER ;
```

```
DELIMITER //
DROP TRIGGER IF EXISTS verif_reservation;
CREATE TRIGGER verif_reservation
BEFORE INSERT ON detail_com
FOR EACH ROW
BEGIN
IF ressD (new.id_produit,new.dateD,new.dateF) || ressF (new.id_produit,new.dateD,new.dateF)
THEN

    INSERT INTO Erreur (erreur) VALUES
    ("l'engein est déja réservé pour ses dates !");


END IF;
END //
DELIMITER ;
```

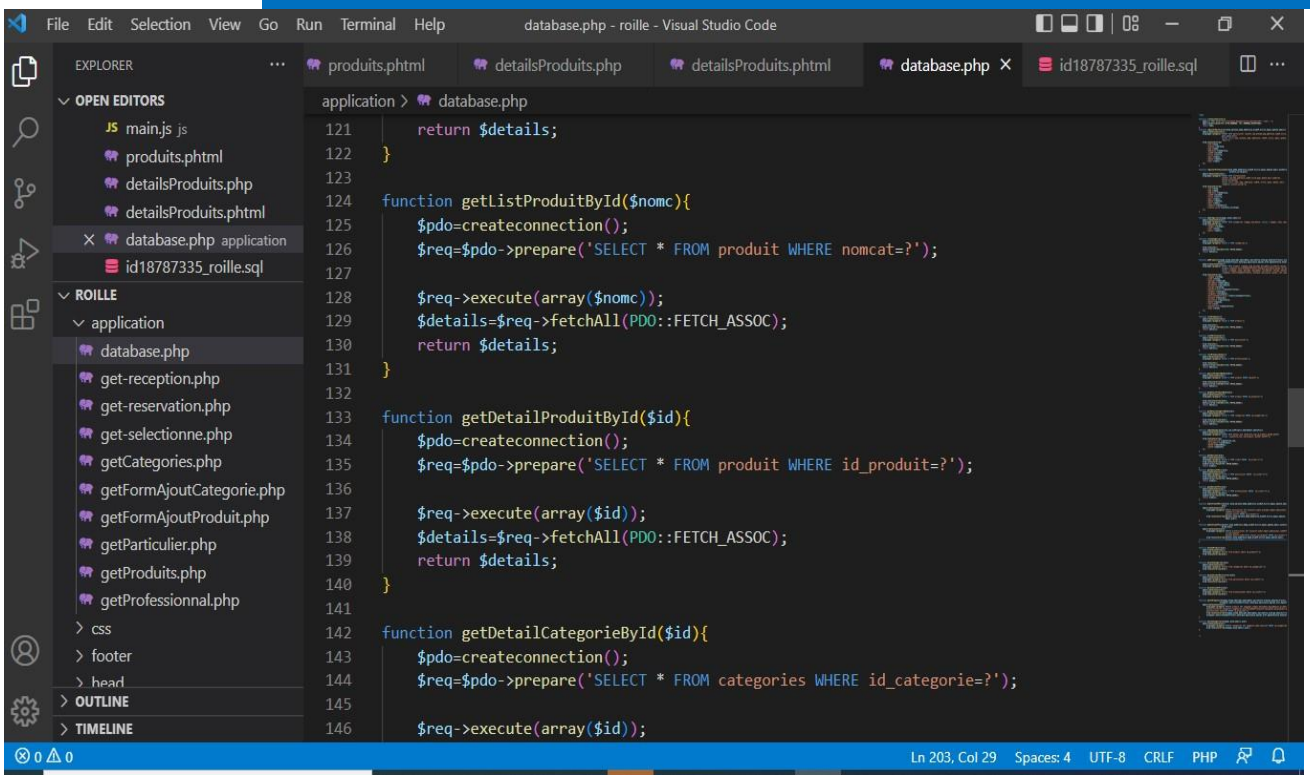
III. Annexe

1. Code Visual Studio

Pour faciliter et accélérer la création du site, nous avons choisi d'utiliser l'édituer de texte Code Visual Studio

Lien pour la source officielle:
https://code.visualstudio.com/

Aspect général :



2. PDO (PHP Data Objects)

Pour que l'application Java se connecte à la BDD, nous devons utiliser le driver JDBC pour MYSQL.

Source :
https://www.php.net/manual/fr/book.pdo.php