```php
<?php
/*******************************************************************************
AUTEUR      : Constantin Herrmann
LIEU        : CFPT Informatique Genève
DATE        : Avril 2020
TITRE PROJET: RESA
DESCRIPTION : Ce script contient toutes les fonctions pour effectuer des READ sur la table 'user'
VERSION     : 1.0
*******************************************************************************/

// On inclu le connecteur de la base de données
include '../../pdo.php';

// On inclu le fichier que génère le mot de passe pour l'utilisateur
include '../password/index.php';

include '../../vars.php';

// Get all users permet de récupérer tous les users de la base de données
function GetAllUsers(){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT `user`.`id`, `user`.`first_name`, `user`.`last_name`, `user`.`phone`, `user`.`email`, `user`.`username` FROM `user`';
    $query = database()->prepare($req);
  }

  try {
    $query->execute();
    $res = $query->fetchAll(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  return $res;
}

/*
* Récupère les users d'après leur permission
* Params :
*   - $idPermission : L'id de la permission recherchée
*/
function GetUsersByPermission($idPermission){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT `user`.`first_name`, `user`.`last_name`, `user`.`phone`, `user`.`email`, `user`.`username` FROM `user` WHERE `user`.`id` IN (SELECT `is_in_as`.`idUser` FROM `is_in_as` WHERE `is_in_as`.`idPermission` = '.$idPermission.')';
    $query = database()->prepare($req);
  }

  try {
    $query->execute();
    $res = $query->fetchAll(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  return $res;
}

/*
* Récupère les informations sur les permissions d'un user d'après son id
* Params :
*   - $id : l'id de l'utilisateur recherché
*/
function GetUserPermissionById($id){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT IFNULL(e.name, "-") etablishment_name, IFNULL(p.name, "-") as permission_name FROM user as u LEFT JOIN is_in_as as iia ON iia.idUser = u.id LEFT JOIN establishment as e ON e.id = iia.idEtablishement LEFT JOIN permission as p ON p.id = iia.idPermission WHERE u.id ='.$id;
    $query = database()->prepare($req);
  }
```

```php

  try {
    $query->execute();
    $res = $query->fetchAll(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  return $res;
}

/*
* Récupère les données d'un utilisateur d'après son id
* Params :
*   - $id : l'id de l'utilisateur recherché
*/
function GetUser($id){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT `id`, `first_name`, `last_name`, `phone`, `email`, `username` FROM `user` WHERE `id` = '.$id;
    $query = database()->prepare($req);
  }

  try {
    $query->execute();
    $res = $query->fetchAll(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  return $res;
}

/*
* Récupère les données d'un utilisateur si celui-ci travail bien dans l'établissement en question
* Params :
*   - $username : le nom d'utilisateur
*   - $password : le mot de passe déjà hashé en sha256
*   - $idEtablishment : l'id de son établissement
*/
function LoginEtablishment($username, $password, $idEtablishment){
  global $key;
  $pass = hash('sha256', hash('sha256', $key).$password);

  static $query = null;

  if ($query == null) {
    $req = 'SELECT u.id as idUser, u.first_name as firstnameUser, u.last_name as lastnameUser, u.phone as phoneUser, u.email as emailUser, p.name as namePermission, p.level as levelPermission, IFNULL(e.id, "-") as idEtablishment, IFNULL(e.name, "-") as nameEtablishment FROM `is_in_as` as iia INNER JOIN `permission` as p ON p.id = iia.idPermission LEFT JOIN `establishment` as e ON e.id = iia.idEtablishement INNER JOIN `user` as u ON u.id = iia.idUser WHERE iia.idUser IN (SELECT `id` FROM `user` WHERE `username` = :u AND `password` = :p) AND (iia.idEtablishement = :e OR iia.idPermission = 1)';
    $query = database()->prepare($req);
  }

  try {
    $query->bindParam(":u", $username, PDO::PARAM_STR);
    $query->bindParam(":p", $pass, PDO::PARAM_STR);
    $query->bindParam(":e", $idEtablishment, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  if($res == null || $res == false){
    return false;
  }else{
    return $res;
  }
}

```

```php
/*
* Récupère les données d'un utilisateur d'après son id
* Params :
*   - $email : l'email du client
*   - $password : le mot de passe déjà hashé en sha256
*/
function Login($email, $password){
  global $key;
  $pass = hash('sha256', hash('sha256', $key).$password);

  static $query = null;

  if ($query == null) {
    $req = 'SELECT `id`, `first_name`, `last_name`, `phone`, `email` FROM `user` WHERE `email` = :e AND `password` = :p';
    $query = database()->prepare($req);
  }

  try {
    $query->bindParam(":e", $email, PDO::PARAM_STR);
    $query->bindParam(":p", $pass, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  if($res == null || $res == false){
    return false;
  }else{
    return $res;
  }
}


function LoginEtablishementManager($n, $e, $p){
  global $key;
  $pass = hash('sha256', hash('sha256', $key).$p);

  static $query = null;

  if ($query == null) {
    $req = 'SELECT e.id FROM establishment as e INNER JOIN is_in_as as iia ON iia.idEtablishement = e.id WHERE e.name = :n AND iia.idUser IN (SELECT u.id FROM user AS u WHERE email = :e AND password = :p) AND iia.idPermission = 2';
    $query = database()->prepare($req);
  }

  try {
    $query->bindParam(":e", $e, PDO::PARAM_STR);
    $query->bindParam(":p", $pass, PDO::PARAM_STR);
    $query->bindParam(":n", $n, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  if($res == null || $res == false){
    return false;
  }else{
    return $res;
  }
}

/*
* Vérifie si il y à déjà un compte avec le mail
* Params :
*   - $email : l'email à vérifier
*/
function CheckMailExists($email){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT `id` FROM `user` WHERE `email` = :email';
    $query = database()->prepare($req);
  }

  try {
```

```php
    $query->bindParam(":email", $email, PDO::PARAM_STR);
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  if($res == null || $res == false){
    return false;
  }else{
    return true;
  }
}


function IsAllowedToBeHere($idUser, $idEtab){
  static $query = null;

  if ($query == null) {
    $req = 'SELECT p.id as permission FROM `is_in_as` as iia INNER JOIN permission AS p ON
p.id = iia.idPermission WHERE (iia.idEtablishement ='.$idEtab.' AND iia.idUser = '.$idUser.')
OR (iia.idPermission = 1 AND iia.idUser = '.$idUser.')';
    $query = database()->prepare($req);
  }

  try {
    $query->execute();
    $res = $query->fetch(PDO::FETCH_ASSOC);
  }
  catch (Exception $e) {
    error_log($e->getMessage());
    $res = false;
  }

  if($res == null || $res == false){
    return false;
  }else{
    return true;
  }
}



if(isset($_GET['byPermission'])){
  $idPermission = $_GET['byPermission'];
  // Vérification qu'il s'agit bien d'un int
  if(is_numeric($idPermission)){
    echo json_encode(GetUsersByPermission($idPermission));
  }else{
    echo json_encode("Valeur non integer");
  }
}
else if(isset($_GET['permissions']) && isset($_GET['id'])){
  $id = $_GET['id'];
  // Vérification qu'il s'agit bien d'un int
  if(is_numeric($id)){
    echo json_encode(GetUserPermissionById($id));
  }else{
    echo json_encode("Valeur non integer");
  }
}
else if(isset($_GET['user']) && isset($_GET['id'])){
  $id = $_GET['id'];
  // Vérification qu'il s'agit bien d'un int
  if(is_numeric($id)){
    echo json_encode(GetUser($id));
  }else{
    echo json_encode("Valeur non integer");
  }
}
else if(isset($_GET['login_e']) && isset($_GET['username']) && isset($_GET['password'])){
  $username = $_GET['username'];
  //$password = hash('sha256', $_GET['password']);
  $password = $_GET['password'];

  $user = LoginEtablishment($username, $password, 1);
  if($user != false){
    echo json_encode($user);
  }else{
    echo json_encode("Utilisateur ou mot de passe incorrect");
  }
```

```php
}
else if(isset($_GET['login']) && isset($_GET['email']) && isset($_GET['password'])){
  $email = $_GET['email'];
  //$password = hash('sha256', $_GET['password']);
  $password = $_GET['password'];

  $user = Login($email, $password);
  echo json_encode($user);
}
else if(isset($_GET['n']) && isset($_GET['e']) && isset($_GET['p'])){
  echo json_encode(LoginEtablishementManager($_GET['n'], $_GET['e'], $_GET['p']));
}
else if(isset($_GET['id'])){
  $id = $_GET['id'];
  // Vérification qu'il s'agit bien d'un int
  if(is_numeric($id)){
    $user = GetUser($id);
    $permissions = GetUserPermissionById($id);

    $user['permissions']= $permissions;
    echo json_encode($user);
  }else{
    echo json_encode("Valeur non integer");
  }
}
else if(isset($_GET['checkmail']) && isset($_GET['email'])){
  echo json_encode(CheckMailExists($_GET['email']));
}
else if(isset($_GET['u']) && isset($_GET['e'])){
  echo json_encode(IsAllowedToBeHere($_GET['u'], $_GET['e']));
}
else{
  echo json_encode(GetAllUsers());
}
```