```php
<?php

/*
  Fichier: function.php
  Auteur: Kevin Zaffino
  Date: 15/06/2016
  Version:1.10
  Description: Contient les différentes fonctions de traitement
  Copyright (TPI 2016 - Kevin Zaffino © 2016)
 */

require_once './phpScript/mysql.inc.php';

/**
 * Créé un pointeur sur la base
 * @staticvar type $dbc
 * @return PDO
 */
function &myDatabase() {
    static $dbc = null;

    if ($dbc == null) {
        try {
            $dbc = new PDO('mysql:host=' . DB_HOST . ';dbname=' . DB_NAME, DB_USER,
            DB_PASSWORD, array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8", PDO
            ::ATTR_PERSISTENT => true));
        } catch (Exception $e) {
            echo 'Erreur : ' . $e->getMessage() . '<br />';
            echo 'N : ' . $e->getCode();
            die('Could not connect to MySQL');
        }
    }
    return $dbc;
}


//Gestion des Utilisateurs

/**
 * Ajoute une adresse dans la base
 * @staticvar type $ps_adress
 * @staticvar type $ps_id
 * @param string $country
 * @param string $city
 * @param string $street
 * @return int  => le dernier id ajouté dans la base
 */
function addAddress($country, $city, $street) {

    //Prepared statements
    static $ps_adress = null;
    static $ps_id = null;

    //query
    $sql_adress = 'insert into adress (adress.city,adress.street,adress.country_iso)
    values (:city,:street,:iso);';

```

```php
    if ($ps_adress == null) {
        $ps_adress = myDatabase()->prepare($sql_adress);
    }
    if ($ps_id == null) {
        $ps_id = myDatabase()->prepare('SELECT LAST_INSERT_ID()');
    }

    try {
        $ps_adress->bindParam(':city', $city, PDO::PARAM_STR);
        $ps_adress->bindParam(':street', $street, PDO::PARAM_STR);
        $ps_adress->bindParam(':iso', $country, PDO::PARAM_STR);
        $ps_adress->execute();

        $ps_id->execute();
        $last_id = $ps_id->fetchAll(PDO::FETCH_NUM);

        return $last_id[0][0];
    } catch (PDOException $e) {
        $isok = null;
    }
    return $isok;
}

/**
 * Insere un nouvel utilisateur dans la base de données
 * @param string $lastName    Le nom de famille
 * @param string $firstName   Le prénom
 * @param string $gender      Le genre
 * @param string $mail        Le email
 * @param string $pwd         Le mot de passe pas encore crypté
 * @param string $phone       Le numero de telephone
 * @param string $country     Le code ISO du pays
 * @param string $city        Le nom de la ville
 * @param string $street      Le nom de la rue
 * @param int $image          L'id de l'image
 * @return boolean  True si correctement ajouté, autrement False.
 */
function insertUser($lastName, $firstName, $gender, $mail, $pwd, $phone, $country,
$city, $street, $image = DEFAULT_IMAGE_ID) {

    //Prepared statements
    static $ps_user = null;

    //Query
    $sql_user = 'insert into users
    (users.firstname,users.lastname,users.gender,users.mail,users.phone,users.banned,u
    sers.password,users.id_Images,users.id_Adress, users.privilege) values
    (:firstname,:lastname,:gender,:mail,:phone,0,:password,:idimage,:idadress,
    :privilege);';

    if ($ps_user == null) {
        $ps_user = myDatabase()->prepare($sql_user);
    }

    //ajout et récuperation de l'identifiant de l'adresse
    $id_adress = intval(addAddress($country, $city, $street));

```

```php
    //cryptage du mot de passe
    $pwd_sha1 = sha1($pwd);

    $priv = PRIV_USER;

    try {

        $ps_user->bindParam(':firstname', $firstName, PDO::PARAM_STR);
        $ps_user->bindParam(':lastname', $lastName, PDO::PARAM_STR);
        $ps_user->bindParam(':gender', $gender, PDO::PARAM_INT);
        $ps_user->bindParam(':mail', $mail, PDO::PARAM_STR);
        $ps_user->bindParam(':phone', $phone, PDO::PARAM_STR);
        $ps_user->bindParam(':password', $pwd_sha1, PDO::PARAM_STR);
        $ps_user->bindParam(':idimage', $image, PDO::PARAM_INT);
        $ps_user->bindParam(':idadress', $id_adress, PDO::PARAM_INT);
        $ps_user->bindParam(':privilege', $priv, PDO::PARAM_INT);

        $isok = $ps_user->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Verification des identifiants de l'utilisateur
 * @staticvar type $ps
 * @param string $mail      L'adresse email
 * @param string $pwd       Le mot de passe pas encore crypté
 * @return boolean
 */
function login($mail, $pwd) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'select users.id, users.password from users where users.mail = :mail';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    //cryptage du mot de passe
    $pwd_sha1 = sha1($pwd);

    try {
        $ps->bindParam(':mail', $mail, PDO::PARAM_STR);
        $isok = $ps->execute();
        $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
        if (isset($isok[0])) {
            if ($isok[0]['password'] == $pwd_sha1) {
                $isok = intval($isok[0]['id']);
            } else {
                $isok = FALSE;
            }
        } else {
```

```php
            $isok = FALSE;
        }
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Deconnecte l'utilisateur
 */
function logOut() {
    destroySession();
    header('Location: index.php');
}

/**
 * Retourne un tableau avec les infos de l'utilisateur
 * @staticvar type $ps
 * @param int $id   Id de l'utilisateur
 * @return Array    Tableau associatif contenant les données de l'utilisateur
 */
function getUserInfo($id) {
    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'select * from user_info where user_info.id = :id';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $id, PDO::PARAM_INT);

        $isok = $ps->execute();
        $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
        $isok = $isok[0];
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Inverse l'etat de bannissement de l'utilisateur
 * @staticvar type $ps
 * @param int $uid      Identifiant de l'utilisateur
 * @return boolean
 */
function banUnbanUser($uid) {
    $banstate = getUserInfo($uid)['banned'];
    $finalestate = ($banstate == "0") ? 1 : 0;

    //Prepared statement
    static $ps = null;
```

```php

    //Query
    $sql = 'UPDATE `users` SET `banned` = :ban WHERE `id` = :id';
    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }
    try {
        $ps->bindParam(':ban', $finalestate, PDO::PARAM_INT);
        $ps->bindParam(':id', $uid, PDO::PARAM_INT);
        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Modifie les informations de l'utilisateur
 * @staticvar type $ps
 * @param string $lastName      Nom de l'utilisateur
 * @param string $firstName     Prenom de l'utilisateur
 * @param Bool $gender          Genre de l'utilisateur (0 pour Femme et 1 pour Homme)
 * @param string $mail          Email de l'utilisateur
 * @param string $phone         Numero de telephone de l'utilisateur
 * @param int $id               Identifiant d el'utilisateur
 * @return boolean
 */
function updateUserInfo($lastName, $firstName, $gender, $mail, $phone, $id) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = "update users set users.firstname = :firstname ,users.lastname =
    :lastname ,users.gender = :gender ,users.mail = :mail ,users.phone = :phone
    where users.id = :id";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':firstname', $firstName, PDO::PARAM_STR);
        $ps->bindParam(':lastname', $lastName, PDO::PARAM_STR);
        $ps->bindParam(':gender', $gender, PDO::PARAM_INT);
        $ps->bindParam(':mail', $mail, PDO::PARAM_STR);
        $ps->bindParam(':phone', $phone, PDO::PARAM_STR);

        $ps->bindParam(':id', $id, PDO::PARAM_INT);

        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

```

```php
/**
 * Modifie le chemin de l'avatar de l'utilisateur
 * @staticvar type $ps
 * @param int $id
 * @param type $path
 * @return boolean
 */
function updateUserImage($id, $path) {
    static $ps = null;

    $sql = "update images set images.path = :path where images.id = :id";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps_user->bindParam(':path', $city, PDO::PARAM_STR);
        $ps_user->bindParam(':id', $id, PDO::PARAM_INT);

        $isok = $ps_user->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Modifie l'adresse d'un utilisateur
 * @staticvar type $ps
 * @param int $id                   Identifiant de l'utilisateur
 * @param string $countryiso        Code ISO du pays
 * @param string $city              Nom de la ville
 * @param string $street            Nom de la rue
 * @return boolean
 */
function updateUserAdress($id, $countryiso, $city, $street) {

    //Prepared stateent
    static $ps = null;

    //Query
    $sql = "update adress set adress.city = :city ,adress.street = :street
    ,adress.country_iso = :iso where adress.id = :id";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {

        $ps->bindParam(':city', $city, PDO::PARAM_STR);
        $ps->bindParam(':street', $street, PDO::PARAM_STR);
        $ps->bindParam(':iso', $countryiso, PDO::PARAM_STR);
        $ps->bindParam(':id', $id, PDO::PARAM_INT);

        $isok = $ps->execute();
```

```php
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Retourne un tableau avec tous les utilisateurs
 * @staticvar type $ps
 * @return array        Tableau associatif contenant tous les utilisateurs
 */
function getAllUser() {
    static $ps = null;

    $sql = "SELECT id, firstname as Prenom,lastname as Nom,mail, banned as Bannis
    FROM user_info WHERE privilege=1;";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Modifie l'id de l'image d'un utilisateur avec un id donné
 * @staticvar type $ps
 * @param int $user_id      Identifiant d el'utilisateur
 * @param int $image_id     Identifiant d el'image a attribuer
 * @return boolean
 */
function changeUserImage($user_id, $image_id) {
    static $ps = null;

    $sql = "update users set users.id_Images = :iid where users.id = :uid";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':iid', $image_id, PDO::PARAM_INT);
        $ps->bindParam(':uid', $user_id, PDO::PARAM_INT);

        $isok = $ps->execute();
    } catch (PDOException $ex) {
        $isok = false;
    }

    return $isok;
```

```php
}

/**
 * Récupere tous les pays
 * @staticvar type $ps
 * @return boolean
 */
function getAllCountry() {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'select * from country order by name';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

//Gestion des annonces

/**
 * Insert une nouvelle annonce dans la base de donnée
 * @staticvar type $ps
 * @param string $name          libelle de l'annonce
 * @param string $description   Description de l'annonce
 * @param string $price         Prix de l'annonce
 * @param string $date          Date de creation de l'annonce
 * @param string $uid           Identifiant du créateur de l'annonce
 * @param bool $mailvisible     Email visible ou non
 * @param bool $phonevisible    Numero de telephone visible ou non
 * @param bool $adressvisible   Adresse visible ou non
 * @return boolean
 */
function insertArticle($name, $description, $price, $date, $uid, $mailvisible,
$phonevisible, $adressvisible) {

    //Prepared statements
    static $ps = null;
    static $ps_id = null;

    //Query
    $sql = 'insert into articles
    (name,description,price,state,creationdate,banned,id_Users,mailvisible,phonevisibl
    e,adressvisible) values
    (:name,:description,:price,1,:date,0,:uid,:mvis,:pvis,:avis)';


```

```
441     if ($ps == null) {
442         $ps = myDatabase()->prepare($sql);
443     }
444 
445     if ($ps_id == null) {
446         $ps_id = myDatabase()->prepare('SELECT LAST_INSERT_ID()');
447     }
448 
449 
450     try {
451         $ps->bindParam(':name', $name, PDO::PARAM_STR);
452         $ps->bindParam(':description', $description, PDO::PARAM_STR);
453         $ps->bindParam(':price', $price, PDO::PARAM_STR);
454         $ps->bindParam(':date', $date, PDO::PARAM_STR);
455         $ps->bindParam(':uid', $uid, PDO::PARAM_INT);
456         $ps->bindParam(':mvis', $mailvisible, PDO::PARAM_INT);
457         $ps->bindParam(':pvis', $phonevisible, PDO::PARAM_INT);
458         $ps->bindParam(':avis', $adressvisible, PDO::PARAM_INT);
459 
460 
461         $isok = $ps->execute();
462 
463         $ps_id->execute();
464         $last_id = $ps_id->fetchAll(PDO::FETCH_NUM);
465 
466         return intval($last_id[0][0]);
467     } catch (PDOException $e) {
468         $isok = false;
469     }
470     return $isok;
471 }
472 
473 /**
474  * Edition des informations d'une annonce
475  * @staticvar type $ps
476  * @param int $idarticle            Identifiant de l'annonce a editer
477  * @param string $name              Libelle del'annonce
478  * @param string $description       Description del'annonce
479  * @param string $price             Prix de l'annonce
480  * @param bool $mailvisible         Email visible ou non
481  * @param bool $phonevisible        Numero de telephone visible ou non
482  * @param bool $adressvisible       Adresse visible ou non
483  * @return boolean
484  */
485 function editArticleInfo($idarticle, $name, $description, $price, $mailvisible,
    $phonevisible, $adressvisible) {
486 
487     //Prepared statement
488     static $ps = null;
489 
490     //Query
491     $sql = 'update articles set name=:name ,description=:description ,price=:price
        ,mailvisible=:mvis, phonevisible=:pvis ,adressvisible =:avis where id=:id';
492 
493     if ($ps == null) {
494         $ps = myDatabase()->prepare($sql);
495     }
```

```
496
497     try {
498
499         $ps->bindParam(':name', $name, PDO::PARAM_STR);
500         $ps->bindParam(':description', $description, PDO::PARAM_STR);
501         $ps->bindParam(':price', $price, PDO::PARAM_STR);
502         $ps->bindParam(':mvis', $mailvisible, PDO::PARAM_INT);
503         $ps->bindParam(':pvis', $phonevisible, PDO::PARAM_INT);
504         $ps->bindParam(':avis', $adressvisible, PDO::PARAM_INT);
505         $ps->bindParam(':id', $idarticle, PDO::PARAM_INT);
506
507
508         $isok = $ps->execute();
509
510         return $isok;
511     } catch (PDOException $e) {
512         $isok = false;
513     }
514     return $isok;
515 }
516
517 /**
518  * Insert une image de l'article dans la base de données
519  * @staticvar type $ps
520  * @param int $aid      Identifiant de l'article
521  * @param int $iid      Identifiant de l'image
522  * @return boolean
523  */
524 function insertArticleImage($aid, $iid) {
525     //Prepared statement
526     static $ps = null;
527
528     //Query
529     $sql = "update images set images.id_articles = :articleid where images.id =
        :imageid";
530
531     if ($ps == null) {
532         $ps = myDatabase()->prepare($sql);
533     }
534
535     try {
536         $ps->bindParam(':articleid', $aid, PDO::PARAM_INT);
537         $ps->bindParam(':imageid', $iid, PDO::PARAM_INT);
538         $isok = $ps->execute();
539     } catch (PDOException $e) {
540         $isok = false;
541     }
542
543     return $isok;
544 }
545
546 /**
547  * Retourne les images des articles en fonction de l'id de l'article
548  * @staticvar type $ps
549  * @param int $idarticle    Identifiant de l'article
550  * @return boolean
551  */
```

```php
function articleImages($idarticle) {
    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'SELECT id, path FROM images where images.id_articles = :id';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $idarticle, PDO::PARAM_INT);
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Ouvre/Ferme un article
 * @staticvar type $ps
 * @param int $aid      Identifiant del'article
 * @return boolean
 */
function openCloseArticle($aid) {
    //Recuperation des infos de l'article
    $banstate = articleInfo($aid)['state'];
    $finalestate = ($banstate == "0") ? 1 : 0;

    //prepared statement
    static $ps = null;

    //Query
    $sql = 'UPDATE `articles` SET `state` = :state WHERE `id` = :id';
    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }
    try {
        $ps->bindParam(':state', $finalestate, PDO::PARAM_INT);
        $ps->bindParam(':id', $aid, PDO::PARAM_INT);
        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Ban/Unban un article
 * @staticvar type $ps
 * @param int $aid          Identifiant de l'article
 * @return boolean
 */
function banunbanArticle($aid) {
```

```php
    //Recuperation de l'etat actuel
    $banstate = articleInfo($aid)['banned'];

    //inversement de l'etat
    $finalestate = ($banstate == "0") ? 1 : 0;

    //Prepared statement
    static $ps = null;

    //Query
    $sql = "UPDATE `articles` SET `banned` = :ban WHERE `id` = :id";

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }
    try {
        $ps->bindParam(':ban', $finalestate, PDO::PARAM_INT);
        $ps->bindParam(':id', $aid, PDO::PARAM_INT);
        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Recupere toutes les annonces de la base
 * @staticvar type $ps
 * @return boolean
 */
function getAllArticles() {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'select id, name as Libelle,description,price as Prix,state as
    Etat,creationdate as "Date de creation",banned as Bannis from articles';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Recupere tous les articles qui sont actif et non bannis
 * @staticvar type $ps
 * @return boolean
```

```php
 */
function listArticles() {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'SELECT * FROM articles where state = 1 and banned = 0 order by creationdate DESC';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Recupere toutes les annonces d'un utilisateur en particulier
 * @staticvar type $ps
 * @param int $uid         Identifiant dle'utilisateur
 * @return boolean
 */
function getUserArticles($uid) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'SELECT * FROM articles where banned = 0 and articles.id_Users = :id order by creationdate';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $uid, PDO::PARAM_INT);
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Retourne les informations d'un article
 * @staticvar type $ps
```

```php
 * @param int $id       $identifiant de l'article
 * @return boolean
 */
function articleInfo($id) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'select * from articles where id = :id';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $id, PDO::PARAM_INT);

        $isok = $ps->execute();
        $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
        $isok = $isok[0];
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Supprime toutes les images d'un article dans la base de donnée
 * @staticvar type $ps
 * @param int $id_article       Identifiant del'article
 */
function deleteArticleImages($id_article) {
    $articleimage = articleImages($id_article);

    foreach ($articleimage as $value) {
        $id = intval($value['id']);
        deleteFile(deleteImageEntry($id));
    }
}

//Gestion des commentaires

/**
 * Ajoute un commentaire dans la base
 * @staticvar type $ps
 * @param int $uid          Identifiant de l'utilisateur
 * @param int $aid          Identifiant de l'article
 * @param string $date      Date du commentaire
 * @param string $com       Texte du commentaire
 * @return boolean
 */
function addComment($uid, $aid, $date, $com) {

    //Prepared statement
    static $ps = null;
```

```php

    //Query
    $sql = "insert into comments
    (comments.id_Users,comments.id_articles,comments.date_com,comments.comm,comments.s
    tate,comments.banned) values (:uid,:aid,:date,:com,1,0)";
    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':uid', $uid, PDO::PARAM_STR);
        $ps->bindParam(':aid', $aid, PDO::PARAM_STR);
        $ps->bindParam(':date', $date, PDO::PARAM_STR);
        $ps->bindParam(':com', $com, PDO::PARAM_STR);

        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }
    return $isok;
}

/**
 * Recupere les commentaires d'une annonce
 * @staticvar type $ps
 * @param int $aid          Identifiant de l'annonce
 * @return boolean
 */
function getArticleComments($aid) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = "select * from comments where id_articles = :id and comments.banned = 0";
    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $aid, PDO::PARAM_INT);
        $isok = $ps->execute();
        $isok = $isok = $ps->fetchAll(PDO::FETCH_ASSOC);
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * met a jour l'etat du commentaire (en attente de moderation ou non)
 * @staticvar type $ps
 * @param int $idcomment        Identifiant du commentaire
 * @param bool $state           Valeur a assigner
 * @return boolean
 */
```

```php
function changeComState($idcomment, $state) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'update comments set comments.state = :state where comments.id = :id';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $idcomment, PDO::PARAM_INT);
        $ps->bindParam(':state', $state, PDO::PARAM_INT);
        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

/**
 * Bannis un commentaire
 * @staticvar type $ps
 * @param int $idcomment        Identifiant du commentaire
 * @return boolean
 */
function banComment($idcomment) {

    //Prepared statement
    static $ps = null;

    //Query
    $sql = 'update comments set comments.banned = 1 where comments.id = :id';

    if ($ps == null) {
        $ps = myDatabase()->prepare($sql);
    }

    try {
        $ps->bindParam(':id', $idcomment, PDO::PARAM_INT);
        $isok = $ps->execute();
    } catch (PDOException $e) {
        $isok = false;
    }

    return $isok;
}

//Gestion des images

/**
 * Insert une image dans la base de données et retourne son id
 * @staticvar type $ps_image
 * @staticvar type $ps_id
```

```
889     * @param string $path           Chemin de l'image
890     * @param int $idarticle        (a specifier si on souhaite assigner l'image a un
        article)
891     * @return int                   Identifiant de l'image
892     */
893    function insertImage($path, $idarticle = FALSE) {
894
895        //Prepared statements
896        static $ps_image = null;
897        static $ps_id = null;
898
899
900        //Query
901        if ($idarticle) {
902            $sql_image = 'insert into images (path,images.id_articles) values
                (:path,:id)';
903        } else {
904            $sql_image = 'insert into images (path) values (:path)';
905        }
906
907        $sql_id = 'select id from images where path = :path';
908
909        if ($ps_image == NULL) {
910            $ps_image = myDatabase()->prepare($sql_image);
911        }
912
913        if ($ps_id == NULL) {
914            $ps_id = myDatabase()->prepare($sql_id);
915        }
916
917        try {
918            $ps_image->bindParam(':path', $path, PDO::PARAM_STR);
919            if ($idarticle) {
920                $ps_image->bindParam(':id', $idarticle, PDO::PARAM_INT);
921            }
922
923            $ps_image->execute();
924
925            $ps_id->bindParam(':path', $path, PDO::PARAM_STR);
926
927            $isok = $ps_id->execute();
928            $isok = $ps_id->fetchAll(PDO::FETCH_NUM);
929        } catch (PDOException $ex) {
930            $isok = false;
931        }
932
933        return $isok[0][0];
934    }
935
936    /**
937     * Upload une image
938     * @return type
939     */
940    function imageUpload() {
941
942        //Extensions autorisées
943        $extArray = array('jpg', 'gif', 'png', 'jpeg');
```

```
944
945         $ext = '';
946         $error = '';
947         $imageName = '';
948
949     //Si le dossier cible n'existe pas, alors on essaye de le créer
950         if (!is_dir(TARGET)) {
951             if (!mkdir(TARGET, 0755)) {
952                 exit('Erreur : le répertoire cible ne peut-être créé ! Vérifiez que vous
                    diposiez des droits suffisants pour le faire ou créez le manuellement !');
953             }
954         }
955
956     //Verification si le champ n'est pas vide
957         if (!empty($_FILES[INPUT]['name'])) {
958
959     //recupère l'extension du fichier
960             $ext = pathinfo($_FILES[INPUT]['name'], PATHINFO_EXTENSION);
961
962     //verification de l'extension
963             if (in_array(strtolower($ext), $extArray)) {
964                 $infosImg = getimagesize($_FILES[INPUT]['tmp_name']);
965
966
967     //Verification du type de l'image
968                 if ($infosImg[2] >= 1 && $infosImg[2] <= 14) {
969
970
971     //Verification des dimensions et de la taille
972                     if (($infosImg[0] <= WIDTH_MAX) && ($infosImg[1] <= HEIGHT_MAX) && (
                        filesize($_FILES[INPUT]['tmp_name']) <= MAX_SIZE)) {
973
974                         if (isset($_FILES[INPUT]['error']) && UPLOAD_ERR_OK === $_FILES[
                            INPUT]['error']) {
975
976                             $imageName = uniqid() . '.' . $ext;
977
978
979                             if (move_uploaded_file($_FILES[INPUT]['tmp_name'], TARGET .
                            $imageName)) {
980                                 $path = TARGET . $imageName;
981
982                                 //insertion dans la base de donnée et retour de l'id
983                                 return insertImage($path);
984                             } else {
985                                 return 'Problème lors de l\'upload !';
986                             }
987                         } else {
988                             return 'Une erreur interne a empêché l\'upload de l\'image';
989                         }
990                     } else {
991                         return 'Erreur dans les dimensions de l\'image !';
992                     }
993                 } else {
994                     return 'Le fichier à uploader n\'est pas une image !';
995                 }
996             } else {
```

```
997             return 'L\'extension du fichier est incorrecte !';
998         }
999     } else {
1000        return 'Image non renseignée!';
1001    }
1002 }
1003
1004 /**
1005  * Parcours le $_FILE pour envoyer toutes les images dans un article
1006  * @param int $id_article       Identifiant de l'article
1007  * @return boolean
1008  */
1009 function multiUpload($id_article) {
1010
1011     //Nombre d'image
1012     $nb = count($_FILES[INPUT]['name']);
1013     for ($i = 0; $i < $nb; $i++) {
1014         $name = uniqid();
1015         $extension_upload = strtolower(substr(strrchr($_FILES[INPUT]['name'][$i], '.'
         ), 1));
1016         $destination = TARGET . $name . "." . $extension_upload;
1017         //deplacement des fichiers
1018         move_uploaded_file($_FILES[INPUT]['tmp_name'][$i], $destination);
1019         //insertion del'image dans la base
1020         insertImage($destination, $id_article);
1021     }
1022     return TRUE;
1023 }
1024
1025 /**
1026  * Modifie le chemin d'une image dans la base d edonnée
1027  * @staticvar type $ps
1028  * @param type $id_image    Identifiant d el'image a modifier
1029  * @param type $newpath     Nouveau chemin
1030  * @return boolean
1031  */
1032 function editImagePath($id_image, $newpath) {
1033     static $ps = null;
1034
1035     $sql = "update images set images.path = :path where images.id = :id";
1036
1037     if ($ps == null) {
1038         $ps = myDatabase()->prepare($sql);
1039     }
1040
1041     try {
1042         $ps->bindParam(':path', $newpath, PDO::PARAM_STR);
1043         $ps->bindParam(':id', $id_image, PDO::PARAM_INT);
1044
1045         $isok = $ps->execute();
1046     } catch (PDOException $ex) {
1047         $isok = false;
1048     }
1049
1050     return $isok;
1051 }
1052
```

```php
/**
 * Supprime une image de la base de données
 * @staticvar type $ps_path
 * @staticvar type $ps_delete
 * @param int $image_id     Identifiant de l'image
 * @return boolean
 */
function deleteImageEntry($image_id) {

    static $ps_path = null;
    static $ps_delete = null;

    $sql_path = "SELECT path FROM images where id=:id;";
    $sql_delete = "DELETE from images WHERE id=:id";


    if ($ps_path == null) {
        $ps_path = myDatabase()->prepare($sql_path);
    }

    if ($ps_delete == null) {
        $ps_delete = myDatabase()->prepare($sql_delete);
    }

    try {
        $ps_path->bindParam(':id', $image_id, PDO::PARAM_INT);
        $path = $ps_path->execute();
        $path = $ps_path->fetchAll(PDO::FETCH_ASSOC);


        $ps_delete->bindParam(':id', $image_id, PDO::PARAM_INT);
        $delete = $ps_delete->execute();
    } catch (PDOException $ex) {
        $path = false;
    }

    return $path[0]['path'];
}

/**
 * Supprime un fichier
 * @param string $path   Emplacement du fichier a supprimer
 */
function deleteFile($path) {
    if (file_exists($path)) {
        unlink($path);
    }
}
```