```php
<?php

/*
 * SADEK Serena
 * Juin 2017
 * TripTracker
 * DataInsertModif.php
 *
 * Toute les fonctions de cette page sont dédiées à l'insertion des données dans
 * le serveur et dans la base de donnée. Cette page fait également office de protail
 * pour les call AJAX commandant l'insertion et la modification d'éléments.
 */

session_start();

//Connexion PDO requise
require_once '../connection.php';

/**
 * Protocole visant à supprimer un voyage, ses étapes et les média qui leur sont
 * attribués de la base de donnée. Le protocole supprime également le fichier du
 * tracé et les images du serveur.
 */
if (isset($_POST["deleteTrip"])) {
    //Nécessité de récupérer des données en lecture
    require_once './navigationData.php';

    $idTrip = $_POST["idToDelete"];

    try {
        $wps = getWps($idTrip);

        if (count($wps > 0)) {
            $connection = getConnection();
            $connection->beginTransaction();

            for ($index7 = 0; $index7 < count($wps); $index7++) {
                //Suppression des media
                removeMediaFile($wps[$index7]["idWaypoint"], $connection);
                deleteMediaOfWp($wps[$index7]["idWaypoint"], $connection);
                //Suppression de l'étape
                deleteWaypoint($wps[$index7]["idWaypoint"], $connection);
            }

            //Suppression du voyage
            removePathOfTrip($idTrip, $connection);
            deleteTrip($idTrip, $connection);

            $connection->commit();
        }
    } catch (Exception $ex) {
        $connection->rollBack();
        echo json_encode($ex->getTraceAsString());
    }
}


/**
 * Procédure déclenchée par la présence de la variable insert dans le post
 * Récupère le path et les informations pour les insérer dans la base de donnée
 * l'aide d'une transaction.
 */
if (isset($_POST["insert"])) {
    try {
        $ids = array();

        $connection = getConnection();
        $connection->beginTransaction();

```

```php
            $title = filter_input(INPUT_POST, "title", FILTER_SANITIZE_STRING);

            $tripId = InsertTrip($title, $connection);
            $fileId = CreatePathTextFile($_POST["path"]);
            setPath($tripId, $fileId, $connection);

            $path = filter_input(INPUT_POST, "path", FILTER_SANITIZE_STRING);

            $content = $_POST["content"];
            $content = json_decode($content);

            for ($index = 0; $index < count($content); $index++) {

                $title = $content[$index]->title;
                $comment = $content[$index]->comment;
                $date = $content[$index]->date;
                $date = implode("-", array_reverse(explode("/", $date)));

                $lat = $content[$index]->lat;
                $lng = $content[$index]->lng;
                $address = $content[$index]->address;

                $wpId = InsertWaypoint($tripId, $title, $comment, $date, $lat, $lng,
                $address, $connection);
                //Cookie destiné au call AJAX d'insertion des media
                setcookie("WP" . $content[$index]->ref, $wpId, time() + 10);

                array_push($ids, $wpId);
            }

            $connection->commit();

            echo json_encode($ids);
        } catch (Exception $ex) {
            $connection->rollBack();
            if (isset($fileId)) {
                unlink("../usersRessources/path/" . $fileId);
            }

            echo json_encode($ex->getTraceAsString());
        }
    }

    /**
     * Protocole visant à éditer un voyage existant. On compare les informations
     * soumises aux informations existantes, et selon le cas on les met à jour,
     * on les  insère ou on les supprime.
     */
    if (isset($_POST["edit"])) {
        require './navigationData.php';

        //Réucpération du contenu initial du trip

        $data = array();
        $data["trip"] = getTrip($_POST["tripId"]);

        if (count($data["trip"]) > 0) {
            $data["waypoints"] = getWps($_POST["tripId"]);

            for ($index1 = 0; $index1 < count($data["waypoints"]); $index1++) {
                $media = getMediaOfWp($data["waypoints"][$index1]["idWaypoint"]);
                $mediaTitle = array();
                for ($index2 = 0; $index2 < count($media); $index2++) {
                    $filename = "usersRessources/image/" . $media[$index2]["mediaName"];
                    array_push($mediaTitle, $filename);
                }
                $data["waypoints"][$index1]["wpDate"] = implode("/",
                array_reverse(explode("-", $data["waypoints"][$index1]["wpDate"])));

```

```php
            $data["waypoints"][$index1]["media"] = $mediaTitle;
        }

        //Modification contextuelle du contenu

        try {
            $connection = getConnection();
            $connection->beginTransaction();

            $idTrip = $_POST["tripId"];
            $title = $_POST["title"];
            $path = $_POST["path"];

            //Mise à jour du voyage
            updateTrip($idTrip, $title, $connection);
            updatePathTextFile(getPathName($idTrip, $connection), $path);

            $content = json_decode($_POST["content"]);
            $contentCopy = json_decode($_POST["content"]);

            for ($index3 = 0; $index3 < count($content); $index3++) {

                $title = $content[$index3]->title;
                $date = $content[$index3]->date;
                $date = implode("-", array_reverse(explode("/", $date)));
                $comment = $content[$index3]->comment;
                $lat = $content[$index3]->lat;
                $lng = $content[$index3]->lng;
                $address = $content[$index3]->address;

                //Les étapes possédant un id (présentes dans la base) sont mises à jour
                if (isset($content[$index3]->id)) {

                    $idWp = $content[$index3]->id;
                    updateWaypoint($idWp, $idTrip, $title, $comment, $date, $lat, $lng, $address, $connection);
                    //Cookie destiné au call AJAX d'insertion des media
                    setcookie("WP" . $content[$index3]->ref, $idWp, time() + 10);

                    for ($index4 = 0; $index4 < count($data["waypoints"]); $index4++) {
                        if (isset($data["waypoints"][$index4]["idWaypoint"])) {
                            if ($data["waypoints"][$index4]["idWaypoint"] == $idWp) {
                                $data["waypoints"][$index4] = null;
                            }
                        }
                    }
                } else {
                    //Les étapes ne possédant pas d'id sont nouvelles et donc insérées dans la base
                    InsertWaypoint($idTrip, $title, $comment, $date, $lat, $lng, $address, $connection);
                }
            }

            //Les étapes de la base absentes des étapes mises à jour sont supprimées
            for ($index5 = 0; $index5 < count($data["waypoints"]); $index5++) {
                if ($data["waypoints"][$index5] !== null) {
                    $wpid = ($data["waypoints"][$index5]["idWaypoint"]);
                    removeMediaFile($wpid, $connection);
                    deleteMediaOfWp($wpid, $connection);
                    deleteWaypoint($wpid, $connection);
                }
            }

            //Les images dans la fille d'attente de suppression sont supprimées.
            for ($index6 = 0; $index6 < count($_SESSION["picOnDelete"]); $index6++) {
                removeMedia($_SESSION["picOnDelete"][$index6], $connection);
```

```
                deleteMedia($_SESSION["picOnDelete"][$index6], $connection);
            }

            $_SESSION["picOnDelete"] = array();

            $connection->commit();

            echo 'OK';
        } catch (Exception $exc) {
            $connection->rollBack();

            echo $exc->getMessage();
        }
    } else {
        echo 'No Result';
    }
}

/**
 * Ajoute un voyage dans la base de donnée avec les informations données par le
 * biais de la connexion donnée
 * @param type $title
 * @param type $co
 * @return type
 */
function InsertTrip($title, $co) {
    $req = $co->prepare("INSERT INTO trip(idUser, tpTitle) values (:idUser, :title)");
    $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
    $req->bindParam(":title", $title, PDO::PARAM_STR);
    $req->execute();
    return $co->lastInsertId();
}

/**
 * Attribue le nom du fichier contenant son tracé au voyage donné par le biais
 * de la connexion donnée
 * @param type $idTrip
 * @param type $pathLocation
 * @param type $co
 */
function setPath($idTrip, $pathLocation, $co) {
    $req = $co->prepare("UPDATE trip set pathObject = :path WHERE idTrip = :idTrip
    AND idUser = :idUser");
    $req->bindParam(":path", $pathLocation, PDO::PARAM_STR);
    $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
    $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
    $req->execute();
}

/**
 * Ajoute une étape par le biais de la connexion donnée
 * @param int $idTrip
 * @param string $title
 * @param string $comment
 * @param string $date
 * @param double $lat
 * @param double $lng
 * @param string $address
 * @param PDO $co
 * @return int $idWp : id du Waypoint ajouté
 */
function InsertWaypoint($idTrip, $title, $comment, $date, $lat, $lng, $address, $co) {
    $req = $co->prepare("INSERT INTO waypoint(idTrip, wpTitle, wpComment, wpDate,
    lat, lng, address) values (:idTrip, :title, :comment, :date, :lat, :lng,
    :address)");
    $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
    $req->bindParam(":title", $title, PDO::PARAM_STR);
    $req->bindParam(":comment", $comment, PDO::PARAM_STR);
    $req->bindParam(":date", $date, PDO::PARAM_STR);
```

```php
        $req->bindParam(":lat", $lat, PDO::PARAM_STR);
        $req->bindParam(":lng", $lng, PDO::PARAM_STR);
        $req->bindParam(":address", $address, PDO::PARAM_STR);
        $req->execute();
        return $co->lastInsertId();
    }

    /**
     * Genève un fichier au nom aléatoire pour stocker la variable passée en paramètre
     * @param string $content : contenu à insérer dans un fichier texte
     * @return string $id : shortName du fichier texte créé
     */
    function CreatePathTextFile($content) {
        $id = uniqid("path", true);
        $id .= ".txt";

        $stream = fopen("../usersRessources/path/" . $id, 'a+');
        fwrite($stream, $content);
        fclose($stream);

        return $id;
    }

    /**
     * Ré-écrit le chemin du voyage dans le fichier donné
     * @param type $name
     * @param type $content
     */
    function updatePathTextFile($name, $content) {
        $stream = fopen("../usersRessources/path/" . $name, 'w');
        fwrite($stream, $content);
        fclose($stream);
    }

    /**
     * Efface le voyage donné se sa table par le biais de la connexion donnée
     * @param type $tripId
     */
    function deleteTrip($tripId, $co) {
        $req = $co->prepare("DELETE FROM trip where idTrip = :id and idUser = :idUser");
        $req->bindParam(":id", $tripId, PDO::PARAM_INT);
        $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
        $req->execute();
    }

    /**
     * Efface l'étape de sa table par le biais de la connexion donnée
     * @param type $wpId
     * @param type $co
     */
    function deleteWaypoint($wpId, $co) {
        $req = $co->prepare("DELETE FROM waypoint where idWaypoint = :id");
        $req->bindParam(":id", $wpId, PDO::PARAM_INT);
        $req->execute();
    }

    /**
     * Efface les réféférences des Média du voyage donné dans la table par le biais
     * de la connexion donnée
     * @param type $wpId
     * @param type $co
     */
    function deleteMediaOfWp($wpId, $co) {
        $req = $co->prepare("DELETE FROM media where idWaypoint = :id");
        $req->bindParam(":id", $wpId, PDO::PARAM_INT);
        $req->execute();
    }

    /**
```

```php
     * Efface les réféférences du média donné par le biais de la connexion donnée
     * @param type $wpId
     * @param type $co
     */
    function deleteMedia($idMedia, $co) {
        $req = $co->prepare("DELETE FROM media where idMedia = :id");
        $req->bindParam(":id", $idMedia, PDO::PARAM_INT);
        $req->execute();
    }

    /**
     * Efface le média du serveur
     * @param type $wpId
     * @param type $co
     */
    function removeMedia($idMedia, $co) {
        $name = getMediaName($idMedia, $co);
        if (file_exists("../usersRessources/image/" . $name)) {
            unlink("../usersRessources/image/" . $name);
        }
    }

    /**
     * Effece la tracé du serveur
     * @param type $idTrip
     * @param type $co
     */
    function removePathOfTrip($idTrip, $co) {
        $pathName = getPathName($idTrip, $co);
        unlink("../usersRessources/path/" . $pathName);
    }

    /**
     * Récupère le nom du média à partir de son id par le biais de la connexion donnée
     * @param type $wpId
     * @param type $co
     */
    function getMediaName($mediaId, $co) {
        $req = $co->prepare("SELECT mediaName FROM media where idMedia = :id");
        $req->bindParam(":id", $mediaId, PDO::PARAM_INT);
        $req->execute();
        $result = $req->fetch(PDO::FETCH_ASSOC);
        return $result["mediaName"];
    }

    /**
     * Efface les images de l'étape donnée par le biais de la connexion donnée
     * @param type $wpId
     */
    function removeMediaFile($wpId, $co) {
        $media = getMediaOfWp($wpId);
        for ($index = 0; $index < count($media); $index++) {
            if (file_exists("../usersRessources/image/" . $media[$index]["mediaName"])) {
                unlink("../usersRessources/image/" . $media[$index]["mediaName"]);
            }
        }
    }

    /**
     * Met à jour le voyage donnée par le biais de la connexion donnée
     * @param type $idTrip
     * @param type $title
     * @param type $co
     */
    function updateTrip($idTrip, $title, $co) {
        try {

            $req = $co->prepare("UPDATE trip set tpTitle = :title where idTrip = :id");
            $req->bindParam(":id", $idTrip, PDO::PARAM_INT);
```

```
405             $req->bindParam(":title", $title, PDO::PARAM_STR);
406             $req->execute();
407         } catch (Exception $ex) {
408             echo $ex->getMessage();
409         }
410     }
411
412     /**
413      * Met à jour l'étape donnée par le biais de la connexion donnée
414      * @param type $wpId
415      * @param type $idTrip
416      * @param type $title
417      * @param type $comment
418      * @param type $date
419      * @param type $lat
420      * @param type $lng
421      * @param type $address
422      * @param type $co
423      */
424     function updateWaypoint($wpId, $idTrip, $title, $comment, $date, $lat, $lng,
       $address, $co) {
425         try {
426             $req = $co->prepare("UPDATE waypoint set wpTitle = :title, wpComment =
                :comment, wpDate = :date, lat = :lat, lng = :lng, address = :address where
                idWaypoint = :wpId");
427             $req->bindParam(":wpId", $wpId, PDO::PARAM_INT);
428             $req->bindParam(":title", $title, PDO::PARAM_STR);
429             $req->bindParam(":comment", $comment, PDO::PARAM_STR);
430             $req->bindParam(":date", $date, PDO::PARAM_STR);
431             $req->bindParam(":lat", $lat, PDO::PARAM_STR);
432             $req->bindParam(":lng", $lng, PDO::PARAM_STR);
433             $req->bindParam(":address", $address, PDO::PARAM_STR);
434             $req->execute();
435         } catch (Exception $ex) {
436             echo $ex->getMessage();
437         }
438     }
439
440     /**
441      * Retounre le nom du fichier contenant le tracé do voyage donné
442      * @param type $idTrip
443      * @param type $co
444      * @return type
445      */
446     function getPathName($idTrip, $co) {
447         $req = $co->prepare("SELECT pathObject FROM trip WHERE idTrip = :idTrip");
448         $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
449         $req->execute();
450         $result = $req->fetch(PDO::FETCH_ASSOC);
451         return $result["pathObject"];
452     }
453
```