

## TD PHP MVC

### Étape 16 : Formulaires de modification d'un product

L'objectif de cette étape est d'ajouter une route permettant la modification d'un produit (par l'administrateur du site)

La route vers cette vue sera : `/product/update/id`. Il faut donc compléter notre controller `ProductController` avec une méthode `update` et créer le template associé : `template/product/update.view.php`. Cette étape va beaucoup ressembler au TD 15 (et 15 bis) création d'un product.

Nous allons d'abord modifier notre vue `template/product/read.view.php` pour y ajouter un bouton « Modifier » pointant vers la route `/product/update/id`. Nous en profitons pour tester l'affichage de l'image du produit en ajoutant une balise `img`.

```
template > product > read.view.php
8  Description : <?= $product->description ?>
9  </h4>
10 <h4>
11 Prix : <?= $product->price ?>
12 </h4>
13 <div>
14     <img alt="<?= $product->name ?>" src="<?= $product->image_path ?? ' ' ?>" />
15 </div>
16 <br>
17 <a href="/product/update/<?= $product->id?>" class="btn btn-sm btn-success mb-3">
18     Modifier
19 </a>
```

En testant, on peut se rendre compte que l'image ne s'affiche pas ... une petite correction dans la méthode `create` du `ProductController` est nécessaire pour corriger ce bug.

```
controller > product.controller.php > ProductController > create
48
49 if(isset($files['image_path']) && $files['image_path']['error'] == 0){
50     $file = $files['image_path'];
51     $posted['image_path'] =
52         str_replace($_SERVER["DOCUMENT_ROOT"], "", str_replace("/temp", "", $file['tmp_name']));
53 }
54 if(isset($posted['image_path'])){
55     $temp_image_path = explode('/', $posted['image_path']);
56     $indexToInsert = count($temp_image_path) - 1;
57     array_splice($temp_image_path, $indexToInsert, 0, "temp" );
58     $temp_image_path = implode('/', $temp_image_path);
59     $temp_image_path = $_SERVER["DOCUMENT_ROOT"] . $temp_image_path ;
60 }
61
62 $repo = new MainRepository("product");
63 $errors = $repo->validate($posted);
64 if(count($errors) == 0){
65     $result = $repo->insertOne($posted);
66     if($result != false){
67         if (isset($posted['image_path'])) {
68             $test = rename($temp_image_path, $_SERVER["DOCUMENT_ROOT"] . $posted['image_path']);
69         }
70         header("Location: /product/read/$result->id");
71         die;
72     }
73 }
74
75 }
```

Testons à nouveau le chemin `/product/create` qui redirigera vers `product/read/id` une fois l'insertion en base de données faite. L'image s'affiche correctement et nous pouvons voir le bouton « Modifier » également.

← → ↺ 🏠 ⚠ Non sécurisé | td-php-mvc.loc:81/product/read/38

HEADER

# Détail du produit : Gâteau Fruits Rouges

## Catégorie du produit : Les Desserts

Description : Délicieux gâteau aux fruits rouges

Prix : 5.9




Modifier

FOOTER

Cela étant fait nous pouvons créer la méthode `update` de notre contrôleur sur le même principe que la méthode `create` (pour l'instant sans le traitement du formulaire)

```
controller > 🐘 product.controller.php > 🛠 ProductController > 📦 update
86
87     function update(){
88
89         $errors = [];
90         $posted = [];
91
92         // TODO Submit
93
94         $repository = new MainRepository("category");
95         $categories = $repository->getAll();
96
97         $this->entities = [ 'categories' => $categories,
98                           'errors' => $errors,
99                           'posted' => $posted ];
100
101         $this->render();
102     }
```

Nous créons le template correspondant en reprenant le même code que pour le create et en y ajoutant l'image (code en surbrillance).

```
template > product >  update.view.php
1 <h2>Modification Produit</h2>
2 <form action="" method="post" enctype="multipart/form-data">
3 <label for="name">Nom</label><br>
4 <input type="text" name="name" id="name" value="<?=$posted['name'] ?? ''>">
5 <br><br>
6 <label for="description">Description</label><br>
7 <textarea name="description" id="description" cols="30" rows="10"><?=$posted['description'] ?? ''></textarea><br><br>
8 <label for="price">Prix</label><br>
9 <input type="text" name="price" id="price" value="<?=$posted['price'] ?? ''>">
10 <label class="text-danger"><?=$errors['price'] ? 'Champ invalide, saisir un prix SVP.' : ''></label><br><br>
11 <label for="image_path">Image</label><br>
12 <div>
13 .....<img alt="<?=$posted['name'] ?? ''>" src="<?=$posted['image_path'] ?? '/assets/img/default_product_image.jpg'>" /><br>
14 .....</div>
15 <input type="file" name="image_path" id="image_path" value="Change"><br><br>
16 <label for="category_id">Categorie</label><br>
17 <select id="category_id" name="category_id">
18 <option value="null">Choisir ...</option>
19 <?php foreach($categories as $category){ ?>
20 <option value="<?=$category->id"><?=$category->id == ($posted['category_id'] ?? '') ? 'selected' : '' ?> >
21 | <?=$category->name?>
22 | </option>
23 <?php } ?>
24 </select><br><br>
25 <div class="d-flex">
26 <input type="submit" name="create" id="create" value="Valider">
27 <input type="submit" name="cancel" id="cancel" value="Annuler" class="ml-2">
28 </div><br>
29 <label class="text-danger"><?=$errors ? 'Erreur de saisie dans le formulaire.' : ''></label>
30 </form>
```

En l'état actuel nous affichons un formulaire vide qui ne contient pas les informations du produit que nous souhaitons modifier, nous avons donc besoin de récupérer le produit en base de donnée pour pouvoir l'afficher dans le formulaire.

```
controller > product.controller.php > ProductController > update
88
89     $errors = [];
90     $posted = [];
91
92     $repository = new MainRepository("product");
93     $product = $repository->getOne($this->id);
94     foreach($product as $k => $v){
95         $posted[$k] = $v;
96     }
97
98     //TODO Submit
99
```

Il nous reste à traiter le submit du formulaire, mais avant nous allons devoir créer un méthode updateOne dans le MainRepository qui servira à mettre à jour le produit en base de donnée.

```
repository > main.repository.php > MainRepository > updateOne
128
129     function updateOne($fields){ //TODO updateWhere
130         $set = "";
131         $valuesToBind = array();
132         $id = $fields['id'];
133         unset($fields['id']);
134         foreach($fields as $k=>$v){
135             $set .= $k."=?, ";
136             array_push($valuesToBind,$v);
137         }
138         $set = trim($set,",");
139         $where = "id = ?";
140         array_push($valuesToBind,$id);
141         $sql = "UPDATE $this->table SET $set WHERE $where";
142         $statement = $this->connect()->prepare($sql);
143         $result = $statement->execute($valuesToBind);
144         $test = $statement->rowCount() == 1;
145         if($result && $test){
146             $entityClass = $this->entity;
147             $fields['id'] = $id;
148             $entity = new $entityClass($fields);
149             return $entity;
150         }
151         return false;
152     }
153
```

## Traitement du submit dans le controller :

```
controller > product.controller.php > ProductController > update
94     foreach($product as $k => $v){
95         $posted[$k] = $v;
96     }
97
98     if (isset($_SESSION['post'])) {
99         foreach($_SESSION['post'] as $k => $v){
100             $posted[$k] = $v;
101         }
102         unset($_SESSION['post']);
103         $files = $_SESSION['files'];
104         unset($_SESSION['files']);
105
106         if (isset($posted['cancel'])) {
107             //TODO Delete temp image file if exists
108             header("Location: /product/read/$this->id");
109             die;
110         }
111
112         if(isset($files['image_path']) && $files['image_path']['error'] == 0){
113             $file = $files['image_path'];
114             $posted['image_path'] =
115                 str_replace($_SERVER["DOCUMENT_ROOT"], "", str_replace("/temp", "", $file['tmp_name']));
116         }
117         if(isset($posted['image_path'])){
118             $temp_image_path = explode('/', $posted['image_path']);
119             $indexToInsert = count($temp_image_path) - 1;
120             array_splice( $temp_image_path, $indexToInsert, 0, "temp" );
121             $temp_image_path = implode('/', $temp_image_path);
122             $temp_image_path = $_SERVER["DOCUMENT_ROOT"] . $temp_image_path ;
123         }
124     }
```

```
controller > product.controller.php > ProductController > update
124
125     $needToBeUpdated = false;
126     foreach($posted as $k => $v){
127         if(isset($product->{$k}) && $product->{$k} != $v){
128             $needToBeUpdated = true;
129             break;
130         }
131     }
132
133     if($needToBeUpdated == true){
134         $posted['id'] = $this->id;
135         $errors = $repository->validate($posted);
136         if(count($errors) == 0){
137             $result = $repository->updateOne($posted);
138             if($result != false){
139                 if (isset($posted['image_path'])) {
140                     $test = rename($temp_image_path, $_SERVER["DOCUMENT_ROOT"] . $posted['image_path']);
141                 }
142                 header("Location: /product/read/$this->id");
143                 die;
144             }
145         }
146     }
147     else{
148         header("Location: /product/read/$this->id");
149         die;
150     }
151 }
152
153 $repository = new MainRepository("category");
154 $categories = $repository->getAll();
```

Testez en debug pas à pas pour comprendre.