Create your WEB Application with MVC paradigm and OPP – PAP support

Tutorial 17

In this tutorial, you will show your collected data in a table and apply the edit and delete content from the table icons.

If you cannot see your collected data inside your table, please, don't advance, because it is useless. Review your last tutorial to achieve this goal.

Okay, now, there is a several ways to show data in a table.

This is the easy away, but not the best, because the data will be available to everyone to access on the front end.

Ok, first you need to make a query to your data base, to get the data. So, in you model category, please introduce this new function code:

```php
public function get_categories()
{
    $DB = Database::newInstance();

    $query    = "select * from categories";
    $result = $DB->read($query);


    return json_decode(json_encode($result), true);

}
```

Ok, now you need to access to your return data from your controller. In your controller, update your index code with this lines:

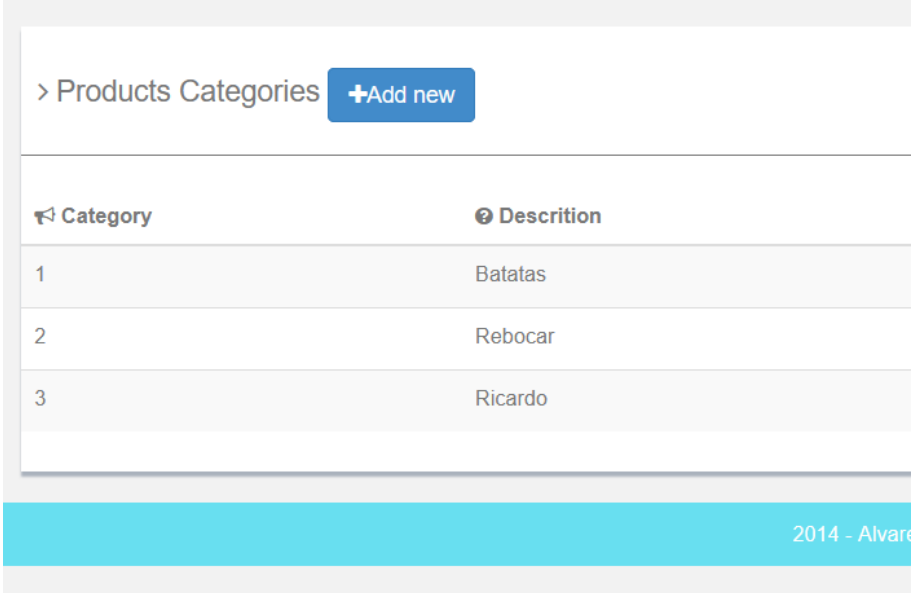
```php
public function index()
{

//Load model User, to access database
$User = $this->load_model('User');

//Validate if is login and if is an admin
$data['user_data'] = $User->check_login(true, ["admin"]);

//validate if the user is really log in
if(is_array($data['user_data']))
{
    $data['user_data'] = $user_data;
    //show($data['user_data']);
}

/*//get category list
*/
   $categoryModel = $this->load_model('Category');
   $data['categories'] = $categoryModel->get_categories();

  //$this->title = 'Admin - Dashboard';
  $data['page_title'] = "Admin - Categories";
  //Rota onde esta a view que vai carregar
  $this->view("../admin/categories", $data);
}
```

Let's debug: In the first function you have query database to get the data, and you return a encode JSON data, to send it to the controller. Next in your controller you receive the data and added to the existing array data variable, with a tag, categories.

Next in your front-end template, you will access to your data, until the data array as you can see in the picture below.

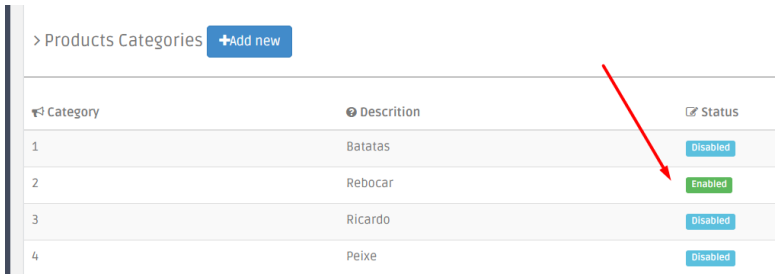
```
<tbody>
    <?php if (!empty($data['categories'])): ?>
        <?php foreach ($data['categories'] as $index => $category): ?>
            <tr>
                <td><?php echo $index + 1; ?></td>
                <td><?php echo htmlspecialchars($category['category']); ?></td> <!-- Access as array -->
                <td>12000.00$</td>
                <td><span class="label label-info label-mini">Enable</span></td>
                <td>
                    <button class="btn btn-success btn-xs"><i class="fa fa-check"></i></button>
                    <button class="btn btn-primary btn-xs"><i class="fa fa-pencil"></i></button>
                    <button class="btn btn-danger btn-xs"><i class="fa fa-trash-o"></i></button>
                </td>
            </tr>
        <?php endforeach; ?>
    <?php else: ?>
        <tr>
            <td colspan="5">No categories found.</td>
        </tr>
    <?php endif; ?>
</tbody>
```

In this code, you make a interaction in the data categories with a foreach, and echo the data inside a row of the table, and the result should be:



Ok, this is the easy away, but you can only create a table inside your model, turn it in a function, and echo the table, inside the front-end code. This permit security, and no data manipulation code, because the code, will be not accessed in the model or the controller. For this propose, I will use this away, but be free to explore another ways.

ASSIGMENT: add your enable/disabled option to your table and put it on working, so if the result is 0 is disabled, otherwise is 1, then is enable.



Next step is to adapt your code to create a modal to delete and edit process. First, I will start with the delete, because it's more easier to do.

Okay, to delete a category, you need the 'ID'. To get the reference you can use the data that you already receive in your foreach statement. Follow the example below:

```
<!-- Botão de Exclusão -->
<button class="btn btn-danger btn-xs" onclick="openDeleteModal(<?php echo htmlspecialchars($category['id']); ?>)">
    <i class="fa fa-trash-o"></i>
</button>
```

To use a modal, you need to create a handler function. When you do the click, the function is called, and the modal show up. This modal is necessary, because you need to ask the user if he really want it to delete the category.

If you repair, you need also to send the ID in the scope of the function, otherwise, you cannot access to the ID.

Please add this handler function into your script's area inside your view categories file:

```
//Function to handler the delete modal
function openDeleteModal(id) {
    // Define o valor do campo oculto no modal de exclusão
    document.getElementById('deleteGroupId').value = id;
    //console.log(id);
    // Abre o modal de exclusão
    $('#deleteGroupModal').modal('show');
}
```

Now, add your delete modal, next the add_category modal:

```
    <!-- Delete Modal HTML -->
<div id="deleteGroupModal" class="modal fade" tabindex="-1"
        aria-labelledby="deleteGroupModal-Label" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <form id="deleteGroup">
                <div class="modal-header">
                    <h4 class="modal-title">Apagar grupo</h4>
                    <button type="button" class="btn-close" data-bs-dismiss="modal"
                        aria-label="Close"></button>
                </div>
                <div class="modal-body">
                    <p>Tem a certeza que quer apagar este grupo?</p>
                    <p class="text-warning"><small>A ação não pode ser defeita.</small></p>
                    <input id="deleteGroupId" name="deleteGroupId" type="hidden" class="form-control" value="">
                </div>
                <div class="modal-footer">
                    <button type="button" class="btn btn-secondary" data-dismiss="modal">Cancelar</button>
                    <input  class="btn btn-danger" value="Apagar"
                    onclick="delete_row(document.getElementById('deleteGroupId').value)">
                </div>
            </form>
        </div>
    </div>
</div>
```

The important thing is to add onclick the reference scope to the ID row like the example:

```
<div class="modal-footer">
    <button type="button" class="btn btn-secondary" data-dismiss="modal">Cancelar</button>
    <input  class="btn btn-danger" value="Apagar"
        onclick="delete_row(document.getElementById('deleteGroupId').value)">
</div>
form>
```

This is my modal, but feel free to add your ony example. Beware with input submit option, because you need to add another onclick function, in this case will be the delete row function.

Ok, this function add also reference to you ID, with the get element attribute.

Next you need to add a new function to get the data from the submit button to send it to the controller, like the create category. Put this function in the view next to the add_category function:

```
// Function to delete row
function delete_row(id)
{
    send_data(data = {
        data_type:"delete_row",
        id:id
    })

}
```

Ok, now go to your controller AJAX, and add this code to the existing add category function:

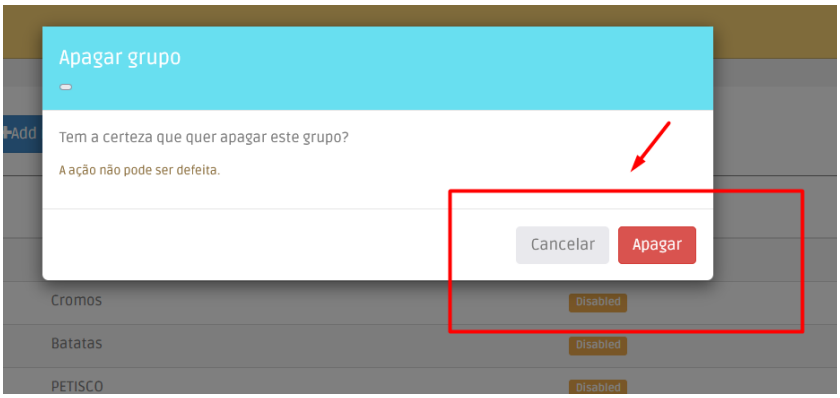
```
if (is_object($data) && isset($data->data_type) && $data->data_type == 'add_category') {
    // Carrega o modelo de categoria
    $category = $this->load_model('Category');
    $check = $category->create($data);

    if (!empty($_SESSION['error'])) {
        $arr['message'] = $_SESSION['error'];
        $_SESSION['error'] = "";
        $arr['message_type'] = "error";
    } else {
        $arr['message'] = "Categoria adicionada com sucesso!";
        $arr['message_type'] = "info";
    }

    $arr['data'] = "";

    // Handle deleting a category
    elseif ($data->data_type == 'delete_row') {

        $check = $category->delete($data->id);
        if ($check) {
            $arr['message'] = "A sua categoria foi removida com sucesso!";
            $arr['message_type'] = "info";
        } else {
            $arr['message'] = "Erro ao remover a categoria!";
            $arr['message_type'] = "error";
        }
        $_SESSION['error'] = "";
        $arr['data'] = "";
        $arr['data_type'] = "delete_row";echo json_encode($arr);
}
```

Like the code before, in this update, you will call the model and send the message result.

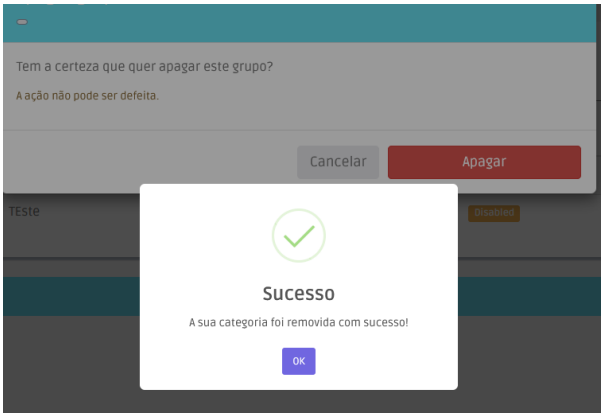
In the model category, update the delete function:

```php
public function delete($id)
{
    $DB = Database::newInstance();
    $id = (int)$id;
    $query = "delete from categories where id = '$id' limit 1";
    $DB->write($query);
}
```

Ok, now test your code, and if everting is ok, the result should be:



When you delete, the message result:



Remember, always comment your code, to facilitate in the future to read them.

END Tutorial 17

Reference: **PHP Ecommerce website development | Send data to database | MVC OOP - Quick programming**

Template: NICEADMIN, design by BOOTSTRAP

Modal: https://getbootstrap.com/docs/4.0/components/modal/