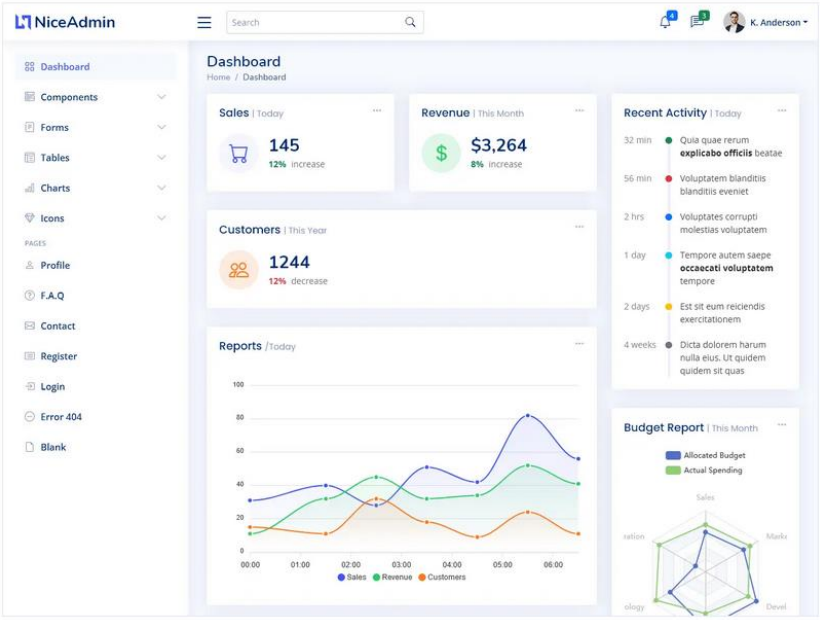
Create your WEB Application with MVC paradigm and OPP – PAP support

Tutorial 10

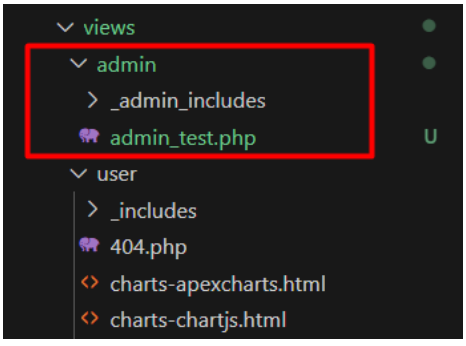
Up to this moment, you have a full authentication system with registration, login, and logout. Next, you will introduce a section that will manage your entire system, the administrator section.

In this section, you will grant certain privileges to one or a couple of users to administer your system. You will add a section to add, update, and delete contents.

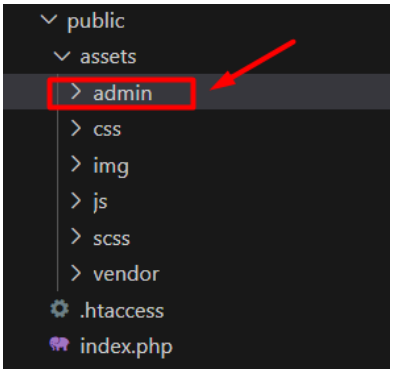
As I saw before, you can use the same template from the previous implementation or add a new one. In these cases, we use a template called CRM to administer, like the example above.



To use a different template please choose a new one and add a new folder inside the views like the example:



And inside the folder assets, you need also to add a new folder with the libraries of the admin template:



Next when you want to call the new template, you need to specify the correct path when you call the view, like the example:

Don't forget to change the path to the CSS files on the headers of the new template, you can also do a new _includes folder to put the header and the footer.

Now let's construct the admin controller to see if the template work well. Create a new file controller with the name admin.php inside the controllers folder:

Next you need to insert this code inside the new file:

```php
class Admin extends Controller
{
    //Public default metodo index, mesmo que o utilizador coloque ou não qualquer URL,
    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']);
        }

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

        /*
        require ABSPATH . '/views/user/_includes/admin_header.php';

        require ABSPATH . '/views/user/index.php';

        require ABSPATH . '/views/user/_includes/admin_footer.php';
        */
    }
}
```

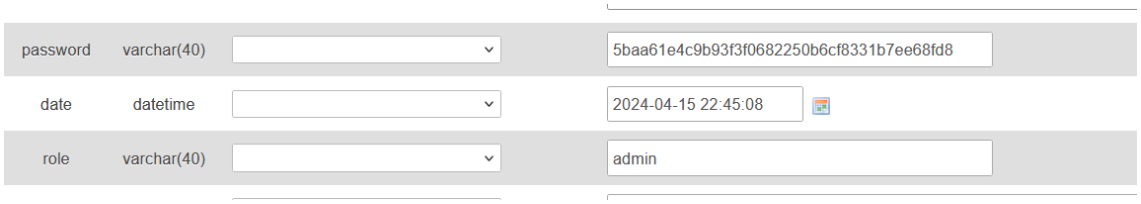
Attention to the path view, mine is different because of my file structure, so you'll need to adapt yours accordingly..

Now let's introduce a validation role to check if the logged in user is available to visit the admin area:

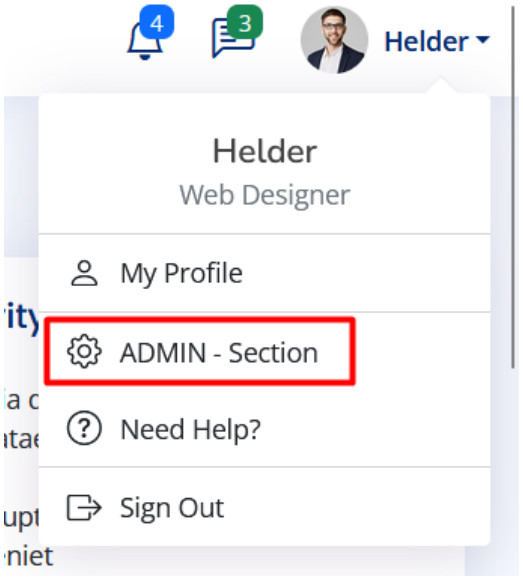
```
</li> -->
<!-- Admin choice -->
<?php if(isset($data['user_data']) && $data['user_data']->role == 'admin') : ?>
<li>
  <a class="dropdown-item d-flex align-items-center" href="<?=ROOT?>admin">
    <i class="bi bi-gear"></i>
    <span>ADMIN - Section</span>
  </a>
</li>
<?php endif; ?>
```

In my example I use this section inside a dropdown menu, but you can use it where you want it.

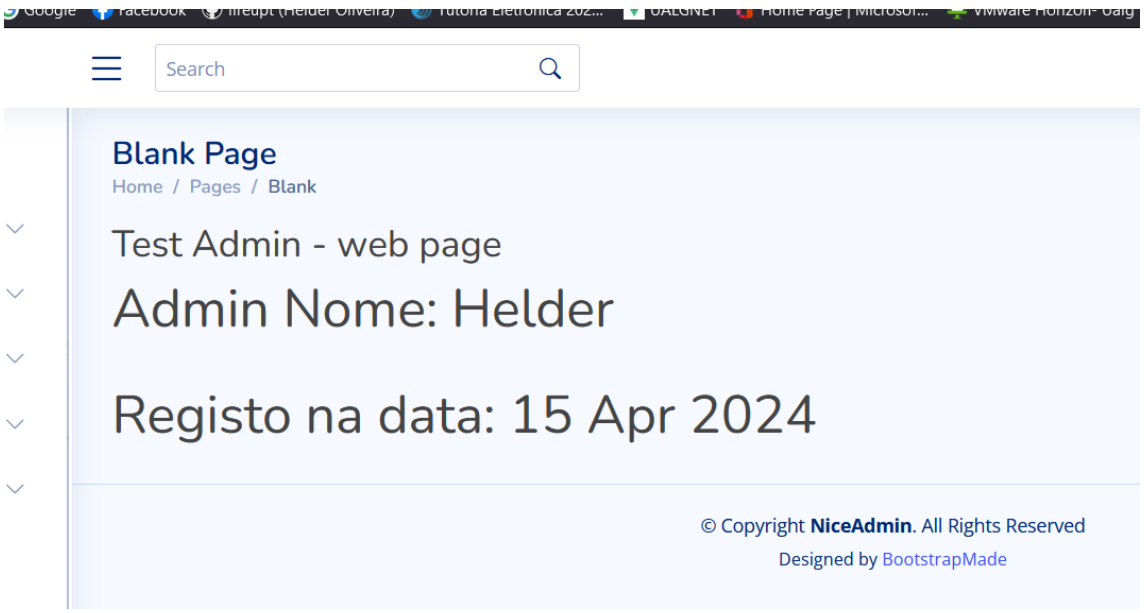
Now go to the database and change one user role do admin:



Now, with the changes made and the `href` defined, if you check, your admin span should appear. If you click on them, it's supposed to go to the admin view that you defined earlier.



My example admin view:



If you want to use the same template, please add a new controller and a new view.

END Tutorial 10

Reference: **PHP Ecommerce website development | Define Admin section | MVC OOP - Quick programming**

Template: NICEADMIN, design by BOOTSTRAP