Project Definition:

1. Admin Panel 1 :
   1. Use a ready template for it.
   2. Write all codes needed.
2. Site design:
   1. All page: responsive.
      1. Pages: Home page, features, portfolios, About us, contact us
3. Membership system: (Admin Panel 2): prepared with: bootstrap.
4. Home page:
   1. Content: Text, Background
   2. Text, background: can be edited by Admin Panel
5. About us:
   1. Contents: About us, Our vision, Our mission
   2. All these: with Admin Panel 1: Add, Edit and Delete able.
6. Features:
   1. Contents: customers, comments and portfolio
   2. Design by yourself
7. Portfolio
   1. This should be totally complete.
   2. All with JQuery, Ajax
   3. A page of all products: image, title, link, description
   4. On mouse over: Detail should be display.
   5. All Link and description should be editable by Admin Panel 1
   6. The target page no needed. Just open a new window.
8. Contact us
   1. Should be controlled by admin panel
9. Notes:
   1. Memberships have to be editable,
   2. pages have to be editable, added and deleted,
   3. The forms have to be readable from the admin panel.

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |

\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Administration System

|  |  |
| --- | --- |
| There are some roles:   * System Admin (Role = SystemAdmin) * Membership Admin * Homepage Admin * Aboutus Admin * Portfolio Admin * Features Admin * Contactus Admin * Comments Admin * User | User Access :   * Register & Login * Change password * View All pages (else of Administration Page) * Write comments for features * Insert record in contact us   Other Roles:   * User Access + Their Access |

Membership Administration:

* All access

Homepage Administration:

* View homepage image and change it.
* View text and change it

About us Administration:

* View 3 sections of About us
* Insert/Delete/Edit 3 section

Portfolio Administration:

* View list of portfolios
* Add/Edit/remove a portfolio (Image/Title/Description/Link)

Features Administration:

* View all features
* Add/Edit/Deactivate a feature (Image, Title, description, Category)

Contact us Administration:

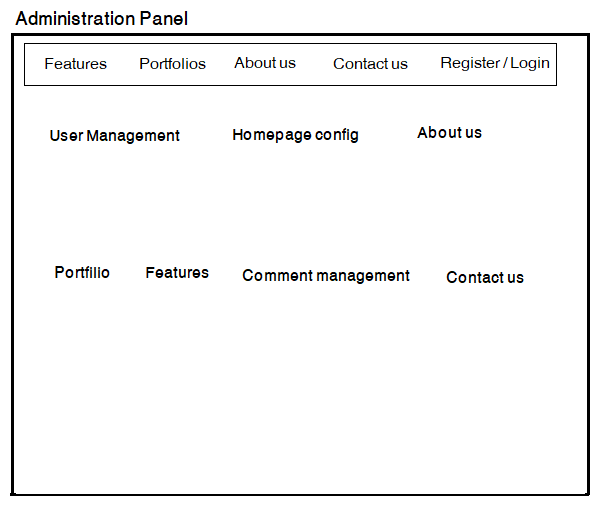
* View all user’s messages (new messages/all messages/replayed or not replayed/a message and its replays)
* Replay a message

User comments administration:

* View all comments (every comment has the username and product name) / filter on confirmed or unconfirmed messages
* Confirm or reject a comment

System administration:

* All Administration Access

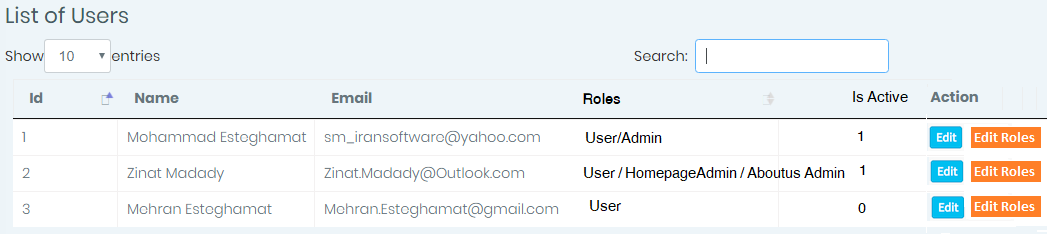


This is a sample :

<https://wrappixel.com/demos/admin-templates/admin-wrap/landingpage/index.html>

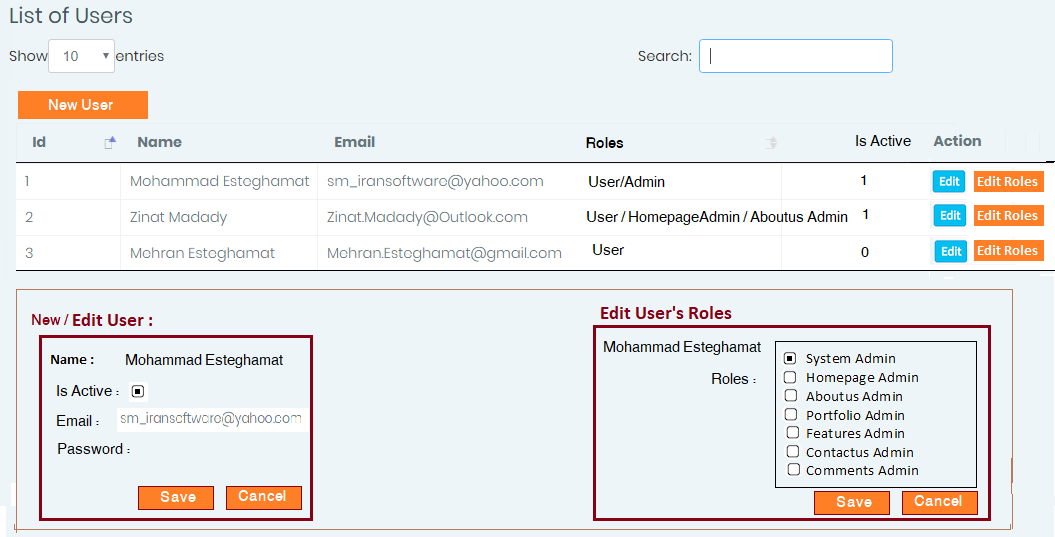
User Management:

1. In Default View list of all users and their roles.



Roles are string.

1. Edit



1. .

Relationships : Good site :

<https://appdividend.com/2018/05/17/laravel-many-to-many-relationship-example/>

تا اینجا جدول های Roles , UserRoles را اضافه کردیم و ارتباط Many to Many را هم برقرار کردیم.

اکنون با استفاده از DataTables می خواهیم Crud مربوط به مدیریت کاربران را بنویسیم.

برای ادامه مجبوریم که Package مربوط به Yajra را نصب کنیم.

DataTable ها امکان پشتیبانی از Pivot Table ها و ویرایش آنها (مثلا ویرایش Role ها یک کاربر) را ندارند.

Breadcrump : خورده های نان – مسیری که هانسل و گرتل انتخاب کردند تا بتوانند بعدا به خانه بازگردند.



با این خطا روبرو هستم :

# **Uncaught TypeError: Cannot read property 'querySelector' of null**

سوال اول

Query Selector چی هست ؟

اول باید ببینیم DOM و یا همان HTML DOM چیست ؟

DOM در واقع مدل ساختار یک سند HTML و یا XML است. و JavaScript با استفاده از مدل DOM، می تواند به تمام عناصر یک صفحه HTML دسترسی داشته باشد.

مثلا :

|  |  |
| --- | --- |
| نمایشی از DOM یا مدل این ساختار | یک ساختار HTML |
|  | <table>  <tbody>  <tr>  <td>Shady Grove</td>  <td>Aeolian</td>  </tr>  <tr>  <td>Over the River, Charlie</td>  <td>Dorian</td>  </tr>  </tbody>  </table> |

مدل DOM یک ساختار درختی از اشیاء موجود در صفحه HTML می سازد:

|  |  |
| --- | --- |
|  | JavaScript با استفاده از مدل شیء گرای DOM، یک صفحه HTML داینامیک یا پویا ایجاد می کند:  JavaScript می تواند، تمام عناصر یک صفحه ی HTML را تغییر دهد.  JavaScript می تواند، تمام خصوصیات عناصر یک صفحه HTML را تغییر دهد.  JavaScript می تواند، تمام سبک های CSS یک صفحه HTML را تغییر دهد.  JavaScript می تواند، به تمام رویدادهای یک صفحه HTML واکنش نشان دهد. |

مثلا زمانی که یک عنصر را از طریق id پیدا می کنیم ، این کار را از طریق DOM انجام داده ایم :

var x=document.getElementById("intro");

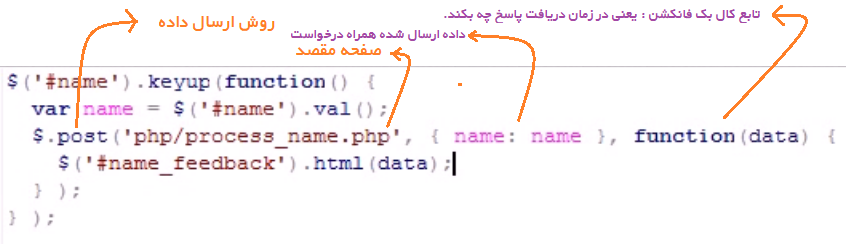
حالا QuerySelector چیست ؟

The **querySelector**() method returns the first element that matches a specified CSS selector(s) in the document.

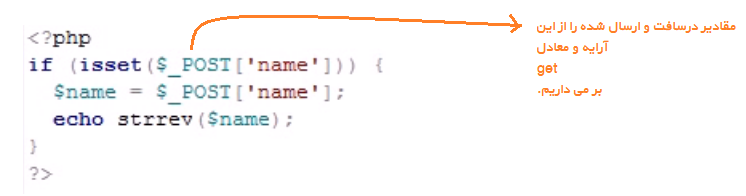
ارسال درخواست از طریق JQuery:

نکته :

* اینکه ارسال درخواست به سمت سرور ، حال با هر روشی که باشد چند پارامتر نیاز دارد :
  + صفرم : اینکه به کدام روش ارسال کند : Post و یا Get .
  + اول : صفحه مقصد یعنی اینکه به کدام صفحه درخواست ارسال شود.
  + دوم : اینکه اگر مقداری را می خواهد ارسال کند همراه درخواست ، باید برای هر متغیر یک اسم و یک مقدار ارسال شود.
  + سوم : اینکه یک تابع Callback نیاز دارد که هر زمان که پاسخ رسید ، با آن پاسخ چه کند و داده دریافت شده را چه کند ؟



در سمت مقصد :



و نکته جالب اینکه :

1. ارسال درخواست از طریق post در واقع همان ajax است.
2. در ارسال درخواست باید حساب csrf را هم کرد. وگرنه خطای 419 unknown status می گیریم.

$.post("{{ route('userListByButton') }}",

                        {listName:$listName, \_token: '{{csrf\_token()}}' },

                        function(data){

                            alert('data');

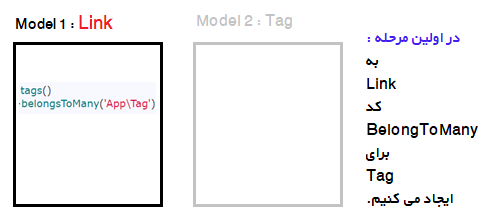
                        });

1. در دستور Ajax می توانیم با پارامتر Data به سرور داده هم بفرستیم. اما اگر داده ای بفرستیم باید مشخص کنیم که نحوه ارسال آین داده ها چطور باشد؟ با Post و یا با Get . و اگر مشخص نکنیم همان Get را در نظر می گیرد.
2. زمانی در web.php برای تعیین Route از Resource استفاده می کنیم که همه متدها را در کنترولر مطابق با اسامی پیش فرض و استاندارد تعریف کرده باشیم. و resource خود متشکل از 7 یا 8 روتینگ است.

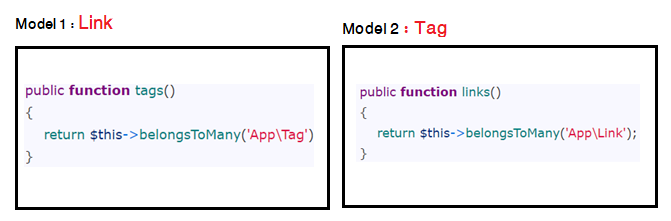
مثالی برای BelongsToMany :

1. [#Add The **belongsToMany** To Your first Model.](https://vegibit.com/laravel-belongstomany-example/#belongsToMany)
   * یک جدول (یا همان Model ) داریم با نام Link :
   * در این مدل یک BelongsToMany برای Tags اضافه می کنیم.

به این معنی که هر لینک می تواند چندین Tag داشته باشد.



1. [#Create The **Associated** Eloquent Model.](https://vegibit.com/laravel-belongstomany-example/#associated-model)
   * اکنون Associated Eloquent یا همان مدل Tag را ایجاد می کنیم.
2. [#Create A **Migration** For The Tags Table and the Pivot Table.](https://vegibit.com/laravel-belongstomany-example/#create-migration)
   * اکنون باید مدل ها را Migrate کنیم . که در ضمن جدول Pivot که همان رابط بین آن دو جدول مورد نظر است، ایجاد می شود.
   * یعنی به زبان ساده ساخت همین مدل ها در دیتابیس. تا اینجای کار را خودمان هم ایجاد کردیم. یعنی دو جدول اصلی و جدول رابط بین این دو جدول.
   * اگر از طریق رسمی Migration رفته باشیم جدول رابط یا همان Pivot Table ما Link\_Tag نام دارد.
3. [#**Migrate** The Database.](https://vegibit.com/laravel-belongstomany-example/#run-migration)
   * php artisan migrate
4. [#Configure The **Inverse belongsToMany** On The Tag Model.](https://vegibit.com/laravel-belongstomany-example/#inverse-manytomany)
   * حالا باید بریم روی مدل Tag و ارتباط معکوس را ایجاد کنیم.
   * یعنی به همان شکل برای مدل Tag .



1. [#Test Our Relationship With **Tinker**.](https://vegibit.com/laravel-belongstomany-example/#tinker)
   * می توان از طریق ترمینال این ارتباط را چک کرد.
2. [#Update The **create() method** Method For Tag Support with lists().](https://vegibit.com/laravel-belongstomany-example/#create-method)
   * <https://vegibit.com/laravel-belongstomany-example/>
3. [#Update The **Create Form View** For Tag Support.](https://vegibit.com/laravel-belongstomany-example/#create-form-view)
4. [#Update The **store() method** for Tag Support with attach().](https://vegibit.com/laravel-belongstomany-example/#store-method)
5. [#Update The **edit() method** for Tag Support with lists().](https://vegibit.com/laravel-belongstomany-example/#edit-method)
6. [#Update The **Edit Form View** for Tag Support.](https://vegibit.com/laravel-belongstomany-example/#edit-form-view)
7. [#Update The **update() method** for Tag Support with sync().](https://vegibit.com/laravel-belongstomany-example/#update-method)
8. [#Update The **Show View** For Tag Support.](https://vegibit.com/laravel-belongstomany-example/#show-view)
9. [#Make The Tag Select Box **Pretty**.](https://vegibit.com/laravel-belongstomany-example/#pretty-tags)

DataTables جداول Pivot را ساپورت نمی کند.

اکنون پیاده سازی :

1. می خواهیم با کلیک روی هر Button لاراول یک کار انجام دهد.
   1. مساله این است که همه این کارهای ماهیت تغییر داده دارند و منجر به یک درخواست Ajax می شوند.
   2. اکنون سوال این است که آیا در یک صفحه می توانیم چند در خواست Ajax داشته باشیم؟ اگر می توانیم چگونه و همه انها باید در روی $(document).ready مستقر شوند ؟
      1. مشکلی نیست. می توانیم چندین درخواست Ajax در صفحه داشته باشیم.
2. اما نکته : مدیریت Event ها . (یا همان Event Handler)
   1. مثلا می خواهیم روی Click روی کلیدی ، کاری را انجام دهیم.
   2. نکته مهم این است که این کلید **Static** ایجاد شده و یا **دینامیک** ؟ که تعریف رخداد برای اینها متفاوت است ؟
      1. برای موارد **استاتیک** :

$(document).ready(function($){

Example 1 :

$('.editmodalopen').click(function(){…});

Example 2 :

$('#addNewCourseBox').click(function(){…});

Example 3 :

$('#modalForm').on('submit', function(event) {…});

Example 4 :

$("#button1").on("click" , function(){...});

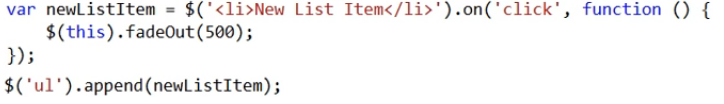
* + 1. برای موارد **دینامیک** :

مثلا باتنی که خودش بصورت دینامیک تهیه شده ، باید به طریق دیگری رخدادها را رویش فعال کرد.

چرا ؟ چون رخدادهای نوع استاتیک زمانی ایجاد می شود که صفحه لود شده باشد. و زمانی که صفحه لود شد ، رخدادها اجرا می شوند و بعد از آنکه کلیدهای دینامیک ایجاد شدند ، چون زمان لود صفحه گذشته دیگر هیچ زمان آن رخدادها اجرا نمی شوند. پس چطور باید آنها را اجرا کنیم.

Method 1 :

یا اینکه در زمان ایجاد آن المان دینامیک ، در همان زمان رخداد را هم به آن از طریق on اضافه کنیم.



اما این روش برای نرم افزار های بزرگ مناسب نیست چونکه این تعداد Event handler توان سیستم را کاهش می دهد. در واقع در این روش به ازای هر آیتم یک Event Handler ایجاد می کنیم.

Method 2 :

در این روش پارامتر سومی را اضافه می کنیم که Parent آن المان را اعلام می کنیم. و یک بار یک Eventhandler برای آن ایجاد می شود و این برای همه نمونه های آن نوع کار خواهد کرد. یعنی در دو مرحله ، یک Event handler برای همه المان ها تعریف می کنیم البته با ایجاد پارامتر سوم. و سپس آن المان های دینامیک را ایجاد می کنیم.

|  |  |
| --- | --- |
| 2 | 1 |
|  | Ul همان المان پرنت است. و در داخل on هم به نوع المانی که قرار است این را بگیرد اشاره می کنیم : li |

یه مثال دیگر :

$("body").on("click" , ".button-container button" , function(){...});

یعنی به نحوی چون خود آنها دینامیک هستند با استفاده از Parent آنها که استاتیک هستند روی این المنت ها کار می کنیم.

این ویدئو فوق العاده است :

<https://www.youtube.com/watch?v=Wxnd21_f_pc>

<https://www.pragimtech.com/>

<https://www.pragimtech.com/courses/bootstrap-tutorial-for-beginners/>

<https://www.youtube.com/watch?v=a59kOE2Ma1Q&list=PL6n9fhu94yhVDV697uvHpavA3K_eWGQap&index=1>

<https://www.youtube.com/watch?v=rI4kdGLaUiQ&index=1&list=PL6n9fhu94yhUbctIoxoVTrklN3LMwTCmd>

نکته جالب :

* می توانیم $(this) را به success هم پاس بدهیم. چگونه ؟
  + ابتدا و قبل از $ajax آنرا به یک متغیر می دهیم.
  + سپس به راحتی آنرا در success استفاده می کنیم.

نمونه خوب برای کار با جداول دینامیک :

<https://www.codemag.com/Article/1511031/CRUD-in-HTML-JavaScript-and-jQuery>

دسته بندی کردن یا Chunk کردن خروجی Eloquent :

* Chunking و یا بخش بندی کردی خروجی
  + در زمانهایی که خروجی تعداد زیادی رکورد است برای پروسس کردن آنها می توان آنها را Chunk و یا بخش بندی کرد و بخش بخش آنها را پروسس کنیم. یعنی استفاده از متد Chunk .

Flight::chunk(200, function ($flights) {

foreach ($flights as $flight) {

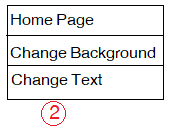
//

}

});

* + یکی از مزیت های Chunk این است که در هر بار اجرا به تعدادی که مشخص کردیم به سراغ دیتابیس می رود و بیشتر از آن را واکشی نمی کند.

**صفحه مدیریت تغییرات Home Page :**



* **ابتدا ایجاد مدل** 
  + این مدل را به همراه Migration آن ایجاد می کنیم.

php artisan make:model Homepage --migration

* + ایجاد ساختار جدول در مدل ایجاد شده.

public function up()

{

Schema::create('homepages', function (Blueprint $table) {

$table->increments('id');

$table->timestamps();

$table->string('homepage\_imagename');

$table->text('homepage\_text');

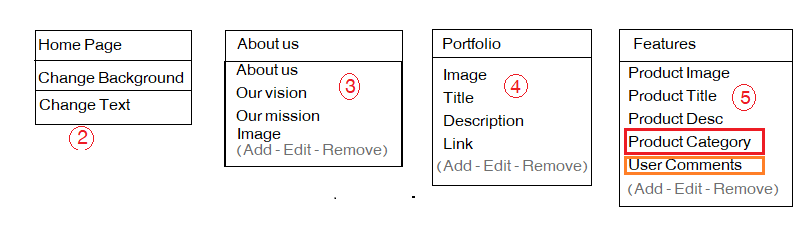
});

}

* + مدل ایجاد شده را migrate می کنیم.

php artisan migrate --path=database/migrations/2019\_10\_08\_142235\_create\_homepages\_table.php

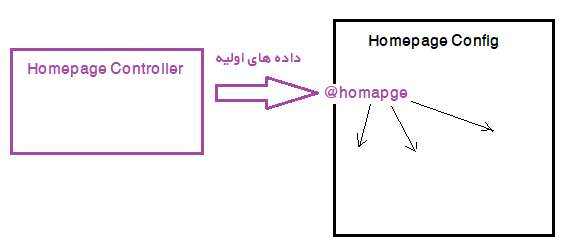
* + این جدول فقط باید یک رکورد داشته باشد. برای این جدول یک رکورد بصورت دستی ایجاد می کنیم و در فرم مدیریت آن هم امکان ایجاد قرار نمی دهیم.
* **طراحی صفحه** 
  + ابتدا باید در نظر بگیریم که 4 صفحه با همین الگو داریم.
  + یعنی 4 صفحه که مدیریت آنها شامل ایجاد – ویرایش و حذف رکوردهایی شامل تصویر و متن و عنوان می باشند.



* + فقط صفحه Features دو مورد متفاوت دارد. یکی دسته بندی و یکی Comment هاست.
  + همه صفحه ها : Image و متن و عنوان دارند.
  + طراحی کلی صفحه به این شکل در نظر می گیریم :

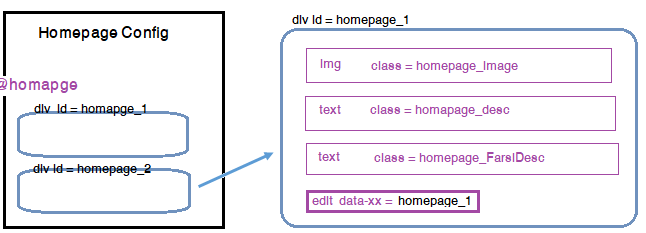
|  |  |
| --- | --- |
| **Title** – String  Font Name – Font Color – Font weight | Image 1 1 |
| **Description** – String  Font Name – Font Color – Font weight |
| Text 1  Font Name – Font Color – Font weight |
| Text 2  Font Name – Font Color – Font weight |
| Text 3  Font Name – Font Color – Font weight |
| Link |
| Category |

* الگوی کار
  + دریافت داده ها برای لیست اولیه : این داده های را از کنترولر می گیریم.



* + حال در یک حلقه For هر ردیف اطلاعاتی را در یک div با یک id یکتا چاپ می کنیم. این div نقش Parent آن رکورد اطلاعاتی را ایفا می کند. و کلید Edit و Delete آن رکورد هم دقیقا همین ساختار id را در data خود دارند.

و هر فیلد اطلاعاتی در یک input با یک کلاس مشخص و متمایز قرار می گیرد.



* + اکنون با کلیک روی دکمه Edit از هر رکورد ، آن رکورد را تشخیص می دهیم حال با کلاس هر input به داده های فیلدها هم دسترسی داریم.
* var homepageText = $('#'+ homepageDivId + ' .homepageText').text();
* حال برای ویرایش یک رکورد :
  + قبلا در مدال (اگر بخواهیم از فرم استفاده کنیم) چند input مخفی برای درج id رکورد و یا اطلاعاتی مثل ویرایش کننده رکورد و ... درج می کنیم.
  + پس از اینکه اطلاعات همه فیلدهای یک رکورد از لیست را گرفتیم ، آنها را به فیلدهای Modal منتقل می کنیم.
    - قبل از هر مقدار دهی حتما محتوای فیلد مورد نظر در مدال را پاک می کنیم.

خطا های مرتبط با Ajax :

خطای مرحله 1 در زمان فراخوانی متد Ajax :

شرایط خطا :

* من از فرم استفاده کردم .
  + و در داخل فرم هم از csrf استفاده کردم.
* اما در زمان Ajax عملکرد فرم را متوقف می کنم با : event.preventDefault();
  + به نظرم این دستور حتی csrf فرم را هم متوقف می کند.
* و با اجرای دستور ajax من با خطای زیر رو برو شدم.

VM3192:4 POST http://localhost:8000/admin/homepageConfig/update 419 (unknown status)

* راهکار:
  + Csrf را برای ajax جدای از فرم و بصورت جداگانه تعریف کردم. یکی در meta در بالای page و یکی بصورت setup و قبل از متد Ajax .
  + مشکل رفع شد.

خطای مرحله 2 در زمان استفاده از data ارسال شده به کنترولر :

* با فعال شدن کنترولر جهت استفاده از داده های ارسال شده :
* خطای زیر تولید شد :

POST http://localhost:8000/admin/homepageConfig/update 404 (Not Found)

و با دابل کلیک روی آن:

The GET method is not supported for this route. Supported methods: POST.

* بررسی
  + داده های ارسال شده به controller را که بررسی می کنیم ، همه خالی است و یک آرایه خالی ارسال شده است.
  + اطلاعات را به این شکل ارسال می کردم.

var data = $('#modalForm').serialize();

$.ajax({ …. Data : data ….});

* راهکار :
  + مرحله 1 : ارسال داده ها را به این شکل تغییر دادم و به نظر داده های فرم به سمت کنترولر ارسال می شود. .

data : new FormData(this)

* + - اما باز هم خطا تولید شد. همان 404 Not found .
  + مرحله 2 : نکته : زمانیکه اطلاعات را با فرم ارسال می کنیم در سمت کنترولر فیلدها را به نام name آن input از فرم می شناسد و نه به id آن input. .
    - اینبار خطای دیگری تولید شد.

POST http://localhost:8000/admin/homepageConfig/update 500 (Internal Server Error)

* + مرحله 3 : از Validator استفاده کرده بودم اما کلاس اون رو use نکرده بودم.

نحوه فراخوانی صفحه داخلی در یک Panel بدون Refresh شدن کل صفحه

**روش اول :**

در این روش دو قسمت لینک ها و فراخوانی صفحات به این شکل هست :

* قسمت لینک ها : از <a href … > استفاده شده است.
* قسمت فراخوانی صفحات :
  + مجبوریم برای جلوگیری از لود همه صفحه به خاطر a href از event.preventDefault(); استفاده کنیم.
  + در قسمت بعد هم از متد Load بهره می گرفتیم. $('.content').load($(this).attr('href'));
* مشکلات : این روش باعث این نوع خطا ها می شود : (هم به دلیل preventDefault و هم load )

[Deprecation] Synchronous XMLHttpRequest on the main thread is deprecated because of its detrimental effects to the end user's experience. For more help, check https://xhr.spec.whatwg.org/.

jquery.min.js:3 [Violation] Added non-passive event listener to a scroll-blocking 'touchstart' event. Consider marking event handler as 'passive' to make the page more responsive. See https://www.chromestatus.com/feature/5745543795965952

**روش دوم :**

* در قسمت لینک ها از Button استفاده کردیم و لینک را در Value قرار دادیم. این باعث شد که لازم نباشد از preventDefault استفاده کنیم.
* برای قسمت دوم از Ajax متد Ajax بهره گرفتیم.

$(document).ready(function($)

{

$('.sidebarsMenu').click(function (event)

{

$.ajax({

url:$(this).val(),

dataType: 'html'

})

.done(function(data) {

// Assuming the request returns HTML, replace content

$('.content').html(data);

});

});

});

صفحه About Us :

یک رکورد برای About Us ایجاد می کنیم.

اطلاعاتی را بصورت Default می دهیم .

بعد به ازای هر فیلد و تصویر یک مقدار نمایش یا عدم نمایش قرار می دهیم.

* ایجاد یک مدل به همراه migration آن

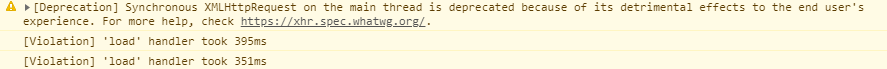
php artisan migrate --path=database/migrations/2019\_10\_11\_193321\_create\_aboutuses\_table.php

* نکته : همانطور که ساختار را در Model می دهیم همان ساختار را برای ایجاد باید در migration هم قرا ردهیم.
  + در واقع migration برای ایجاد و تغییرات Schema است. DDL
    - Migrations are like version control for your database, allowing your team to easily modify and share the application's database schema.
  + اما مدل در واقع یک wrapper روی دیتابیس است و شامل تمام عملیات تغییرات داده ها و انجام واکشی دادهاست. DML
* نمونه های About us

<https://coschedule.com/blog/write-about-us-page-template-examples/>

PHP and Python languages,  
Restfull API development; develop collaborative projects with multiple APIs,

* خطا در زمان کار کردن با About us
  + یک button برای Add new Record گذاشتیم که کار نمی کند و صفحه Modal را باز نمی کند.
  + هیچ خطایی هم نداریم. مگر خطاهای browser :



* + در بررسی انجام شده یک نکته باید اعتراف شود : اینکه صفحات داخلی بصورت دینامیکی لود می شوند.

('.content').html(data);

درست است که در داخل PHP همین Button را بصورت استاتیک ایجاد کردم ، اما بصورت دینامیک لود می شود.

* + - یک راهکار استفاده از

$(document).on ("click", "#addNewAboutus", function () {

به جای

$('#addNewAboutus').click(function(){

که کار نکرد.

* + - بررسی کردم و با قرار دادن Alert متوجه شدم که اصلا Document هم لود نمی شود و بالاتر حتی Script هم لود نمی شود. چرا ؟
    - همه محتوای صفحه پاک شد. اکنون اسکریپت اجرا می شود. احتمالا خطایی در متن وجودداشته است.
    - همه بخش ها را به ترتیب اضافه کردم و فقط در زمان اضافه کردن Modal متوجه شدم که کا رنمی کند.
    - مشکل وجود Text Area در Modal بود. چرا ؟
      * اینکه Tag را نبسته بودم. این را اضافه کردم و همه چیز درست شد. </textarea>

اکنون بعد از اضافه کردن رکورد جدید باید همان بخش از صفحه Update شود :

ویدئوهای مرتبط :

<https://www.youtube.com/watch?v=Ga3pOn7wXBA>

<https://www.youtube.com/watch?v=ytKc0QsVRY4>

اما من می خواهم یک رکورد جدید را که اضافه کردم به صفحه نمایش دهم و هدف نمایش یک صفحه کامل نیست.

ابتدا این روش را امتحان کردم که نتیجه داد .

$('#myTable tr:last').after('<tr>...</tr><tr>...</tr>');

و یا

$('#myTable tbody').append('<tr class="child"><td>blahblah</td></tr>');

یک نکته :

* برای انتخاب دکمه های Edit و Delete آنها که ابتدای لود صفحه ساخته شده اند مشکلی نیست. اما آنها که در همان حال بصورت دینامیک ساخته شده اند دیگر بصورت **//$('.deletemodalopen').click(function(){** قابل استفاده نیستند. و باید به ترتیبی که به عناصر دینامیک اشاره می کنیم انتخاب شوند. یعنی به این شکل :

$('table tbody').on('click','tr :button.deletemodalopen',function() {

طراحی و پیاده سازی Portfolios

E:\0-Job in Turky\0-Online Marketing\00-Vahid\newxampp\htdocs\UNCS\_test\_project\Z-MyDocs\Templates\templatemo\_539\_simple\_house

تعریف : **پرتفولیو چیست؟** به طور خلاصه می شه گفت پرتفولیو آلبوم  کار های شماست. پروژه هایی که در طول ترم ارائه دادید. اسکیس- راندو- ماکت سازی- همه ی شیت بندی ها- کارهای سه بعدی و..... رو در یک دفترچه تدوین می کنید و به عنوان پرتفولیو به دانشگاه یا محل کار ارائه می دید.

فیلدها :

* عنوان
* توضیح
* تصویر
* لینک

مراحل ایجاد کار :

* ابتدا ایجاد مدل
* به همراه ایجاد Migration
  + فعلا با مدل کار نداریم، ادامه کار با Migration
* Migrate کردن و ایجاد جدول
* ایجاد صفحه برای Portfolios
* آدرس ساخت کارت ها :

<https://www.w3schools.com/howto/howto_css_column_cards.asp>

<https://modernweb.com/5-things-you-should-stop-doing-with-jquery/>

طراحی و پیاده سازی Products یا Features

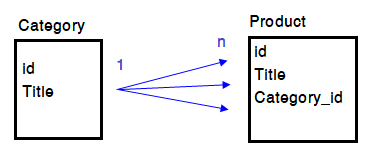
ایجاد هدر فیکس در بالای صفحه :   
<https://www.w3schools.com/howto/howto_js_sticky_header.asp>

ایجاد کارت :

کدهای لازم قرار داده شد.

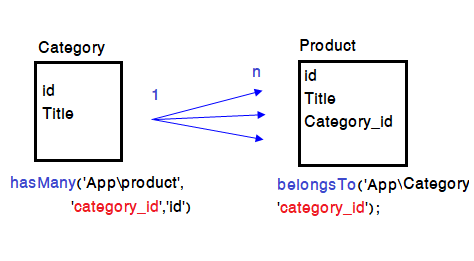
ایجاد مدل ها . اینجا یک رابطه یک به چند داریم. یعنی یک Category در چند محصول قرار می گیرد.

به این شکل :



برای این نوع ارتباط در سمت مدل 1 باید رابطه hasMany را برقرار کرد و در سمت n رابطه belongTo .

به این شکل :



مشکل : با ایجاد checkbox و redio button مشکل نمایش داشتم و قسمتی نمایش داده نمی شد.

بهترین کار ، مشاهده این المنت ها روی inspect element هست.

در ضمن Checkbox کلاس form-control نمی خواهد.

یک نکته مهم در مورد CheckBox ها :

When a checkbox is ticked, it's value is present in the posted data. When it is unticked, it's value is not present. This means when you do $request->all() it will contain only the checkboxes that were ticked.

یعنی حتی else ان هم که مقداری را چک می کند خطا تولید می کند.

        if(is\_string($request['modalproductIsEnabled'])) {

            $product->product\_isenabled = 1;

        }

        else

        {

            $product->product\_isenabled = 0;

        }

باید Else را برداشت و به نحو دیگری مقدار را تنظیم کرد.

و راهکار آن نیز قرار دادن مقدار پیش فرض برای Checkbox است که در هر صورت مقداری بازگشت داده شود. یعنی اینقدر گیر داد که مجیور شدم یه فیلد مخفی تعریف کنم که وضعیت چک باکس رو توی اون نگهداری کنم.

خوب چه کنم که سرعت بیشتری داشته باشم :

چیزهایی که سرعت منو می گیره :

1. خطاهایی که نمی شناسم .
   * مثلا همون Checkbox که اگر تیک نخورده باشه ، هیچ دیتایی به طرف دیگه ارسال نمی شه .
   * یا اینکه از Default Prevent بزنیم حتی csrf ی که توی فرم زدیم هم کنسل می شه و باید مجددا ارسال بشه.
   * یا اینکه در سمت کنترولر فقط با اسامی المان ها کار می کنه نه آی دی ها .
2. خطاهایی که از روی بی دقتی و یا کپی کاری ایجاد شده .
   * مثلا نام های اشتباه و ...
3. چیزهای جدیدی که می خوام یاد بگیرم.
   * خوب درآوردن مطالب و خوندن کمی زمانبره
4. کار روی ظاهر صفحات
   * مثال من هنوز یک ایده و الگوی خوب برای کارت ها ایجاد نکردم.
5. یکی توقف های فکری الکی و کار نکردن و حواس پرتی و ایجاد وقفه توی کار
6. یکی به نظرم الگوهایی که کامل نمی شن و هی تمام کد کپی می شه .

چیزهایی که سرعت منو زیاد می کنه :

1. علاقه به برنامه نویسی و انگیزه تکمیل سایت
2. دید خوب روی کد نویسی داشتن.

راهکارها برای افزایش سرعت :

1. خطاهایی که نمی شناسم .
   * **راهکار ها :** 
     1. مهمترین راهکار تغییر رویکرد در مواجهه با خطاست :
        1. خطا یک بازدانده نیست . خطا یک آموزگار است و همراه.
     2. اینکه درگیر آزمون و خطا کمتر بشوم و ابتدا سریع ریشه یابی مشکل را داشته باشم و سپس یادگیری برای حل مشکل.
     3. یادگیری بیشتر google Inspector و Error Tracking که خطاها راحتر تر رد گیری شوند.
2. خطاهایی که از روی بی دقتی و یا کپی کاری ایجاد شده .
   * **راهکارها:** 
     1. خواندن کد ایجاد شده و مرور آن .
3. چیزهای جدیدی که می خوام یاد بگیرم.
   * **راهکارها:** 
     1. یادگیری بهتر مفاهیم لاراول : مثلا Blade Template Engine
     2. مرور Github و اصولا مرور کارهای آماده و گرفتن ایده و ...
4. کار روی ظاهر صفحات
   * **راهکار :** 
     1. استفاده کامل از فریم ورک ها مثلا Bootstrap و درگیر نشدن با ایجاد المان ها و ...
5. یکی توقف های فکری الکی و کار نکردن و حواس پرتی و ایجاد وقفه توی کار
6. یکی به نظرم الگوهایی که کامل نمی شن و هی تمام کد کپی می شه .

طراحی و پیاده سازی Comment Management

طرح اولیه :

لیست کامنت های جدید (تعیین وضعیت نشده ) 0 = بررسی نشده و 1 = تایید شده و 2 = رد شده

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| View | Status | Comment | Grade | P. Category | Product | Comment Date | User |
| View | Non | This ….. | Average | Tools | Ampul | 2019/10/20 | m.estegh |
| View | Non | I bought … | good | Homeware | Glasses | 2019/10/21 | Ali |
| … |  | … | … | … | … | … | … |

After View Click:

|  |  |  |
| --- | --- | --- |
| User : m.esteghamat | | Date : 2019/10/20 |
| Product : | Ampul …. | |
| Product category : | Tools | |
| Grade : | Average | |
| Comment Text | This product has a good view but when you … . | |
| Confirm  Reject | | |

کارهایی که باید انجام شود :

1. در صفحه لیست کامنتها :
2. در زمان لود :
   1. لیست همه کامنت هایی که بررسی و تایید نشده اند می آید.
      1. یعنی فیلد comment\_confirmdate آنها Null است. و یا confirmationstatus = 0 است.
      2. در ابتدا نتایج بر اساس تاریخ درج کامنت مرتب شده است.
3. کلید اصلی بالای صفحه در حالت کامنت های بررسی نشده است.
   1. حالت ها : بررسی نشده ، بررسی شده و همه کامنت هاست.
4. امکان Paging نیاز است چون تعداد کامنت ها زیاد است.
5. امکان مرتب سازی بر اساس هر یک از ستون های بالا وجود دارد. ( به غیر از متن کامنت )
6. امکان جستجو بر اساس همه ستون ها وجودداشته باشد . بصورت پیش فرض روی ستونی که مرتب شده و نیز امکان جستجو در متن نیز فراهم است.
7. با انتخاب کلید View جزییات کامنت در قسمت پایین صفحه لود می شود.
8. امکان ویرایش فقط برای متن کامنت فراهم است و بقیه بصورت Readonly هستند. و نیز امکان تایید و یا رد آنها وجوددارد.
9. اگر صفحه را در حالتی نمایش همه کامنت ها و یا کامنت های تایید شده هم ببریم باز همه رکوردها قابل ویرایش هستند.

لیست کامنت ها

بهترین راه برای لود لیست و نمایش چیست ؟

* با یک لیست چند جدوله روبرو هستیم . یعنی یک جدول که با یک یا چند جدول دیگر ارتباط دارد.
* لود اولیه را از یک صفحه کنترولر بگیرد و بقیه Paging را بصورت Ajax کار کند ؟
* بعد از لود اطلاعات در ردیف ها ، به نحوی همه اطلاعات در المان های صفحه پنهان باشد که در زمان نمایش جزییات نیاز به Ajax نداشته باشیم ؟
* نحوه مرتب سازی بر اساس ستون ها به چه نحو است ؟ آیا با کلیلک روی هر ستون باید مجددا با Ajax اطلاعات را بخوانیم و مرتب کنیم ؟
* با جستجو مجددا نتایج از جدول بصورت Ajax خوانده می شوند و در صفحه لیست می شوند.

نتایج جستجوی اینترنتی

1. امکان Paging در لاراول وجود دارد. در Eloquent .

<https://appdividend.com/2018/02/23/laravel-pagination-example-tutorial/>

$users = User::paginate(15); And … .

<https://laravel.com/docs/5.8/pagination>

1. امکان جستجو :

<https://freek.dev/1182-searching-models-using-a-where-like-query-in-laravel>

<https://dev.to/mohammadfouladgar/making-the-advanced-search-query-with-eloquent-builder-in-laravel-30g3>

<https://stackoverflow.com/questions/27641911/laravel-routing-and-controller-for-search>

<https://www.cloudways.com/blog/live-search-laravel-ajax/>

<https://www.youtube.com/watch?v=2U1t9f-64_4>

1. واقعا Laravel Blade چیست ؟

<https://colorlib.com/polygon/cooladmin/table.html>

<https://laravel.com/docs/5.8/blade>

<https://www.youtube.com/watch?v=k1GDORPzHFI>

1. مرتب سازی

<https://investmentnovel.com/laravel-column-sorting-with-pagination-tutorial-with-example/>

<https://www.youtube.com/watch?v=NGGZEdS8jhU>

1. K .

<https://www.youtube.com/watch?v=_6BUIktFzLs>

# **Pagination in Laravel using Ajax**

<https://www.youtube.com/watch?v=j4_ICnyFTy8>

# **Sorting with pagination in Laravel using Ajax**

<https://www.youtube.com/watch?v=2U1t9f-64_4>

# **Laravel Live Data Search with Sorting & Pagination using Ajax**

-\_---\_\_\_-\_\_\_--\_\_-\_\_--\_\_\_-\_\_\_-\_\_--\_\_-\_\_\_-\_\_\_-\_\_\_-\_\_\_--\_\_-\_\_--\_\_\_-\_\_\_-\_\_--\_\_-\_-\_\_\_-\_\_\_--\_\_-\_\_--\_\_\_-\_\_\_

<https://www.youtube.com/watch?v=_6BUIktFzLs>

# **Pagination in Laravel using Ajax**

برای صفحه بندی :

1. کنترولر – متد index و نمایش Page

Public function index()

{

$data = Post::all()->**paginate(5);**

Return view(‘pagination’ , compact(‘data’));

}

پس Page اصلی که می خواهیم در آن صفحه بندی را نمایش دهیم pagination است.

1. ایجاد یک View دیگر که داخل آن رکوردها را نمایش می دهیم و لینک صفحه بندی را قرار می دهیم.

View name : Pagination\_data

(نمایش لینک مربوط به صفحه بندی )

* + قرار دادن این کد بعد از نمایش لیست در صفحه

@foreach ($date -> $row)

….

@endforeach

**{{ !! $data -> links() !! }}**

1. در صفحه اصلی ، این ویو نمایش داده ها و صفحه بندی را include می کنیم.

<div id=”table\_date”>

@include(‘pagination\_data’);

</div>

1. در Route آدرس صفحه اصلی را می دهیم :

Route::get(‘/pagination’ , ‘paginationcontroller@index’);

**اکنون فاز اول Pagination آماده است.** یعنی با کلیک روی هر عدد در پایین صفحه Page مورد نظر لود می شود. اما با هر کلیک یکبار صفحه لود می شود. و فاز بعدی استفاده از Ajax است.

1. دریافت Click Event و دریافت آن Page که کلیک شده است. (دریافت href آن و شماره آن صفحه )

در همان صفحه اصلی pagination اسکریپت نویسی را انجام می دهیم.

این دریافتی را در متغیر page قرار می دهیم.

$(document).ready(function(){

$(document).on(‘click’ , ‘.pagination a’ , function(event){

Event.preventdefault();

Var page = $(this).attr(‘href’).split(‘page=’)[1];

این کد می گوید دریافتی href را بر اساس عبارت ‘page=’ تقسیم کن و اولین قسمت را برگردان.

و به این شکل شماره صفحه درخواستی را به دست می آوریم.

اکنون یک تابعی را برای لود آن صفحه که شماره اش را داریم ، فراخوانی می کنیم.

**Fetch\_page(page);**

});

});

1. نوشتن متد Ajax

Function fetch\_data(page){

$.ajax({

url = “/pagination/fetch\_date?page=”+page,

این فقط یک آدرس است که در آنجا باید اطلاعات را بخوانیم و بفرستیم اینجا.

Success function(data)

{

$(“table\_data”).html(data);

}

});

}

در واقع اینجا آدرس برگشتی را داخل همان div در تابع اصلی دادیم. و گفتیم که یک Html را به آنجا خواهیم فرستاد.   
 یعنی اینکه ماباید همه Html را در کنترولر ایجاد کنیم ؟؟؟؟ جواب خیر است .

ما در کنترولر داده ها را می گیریم و به سمت آن View میانی که داده ها را داخل جدول می چیند می فرستیم. و این View هم که بصورت اتوماتیک در آن View اصلی include شده است.

1. ایجاد متد مناسب در کنترولر برای دریافت داده های درخواستی و ارسال به Page میانی

Public function fetch\_page(Request $request)

{

$data = Post::all()->**paginate(5);**

Return view(‘pagination\_data’ , compact(‘data’))->render();

}

اینجا متوجه نشدم که پارامتر ارسالی را چطور در fetch داده ها دخالت داد ؟ یعنی صفحه چندم را گرفت ؟

1. و تنظیم route برای این مورد

Route::get(“pagination/fetch\_date” , “paginationcontroller@fetch\_data”);

**اکنون فاز دوم Pagination هم آماده شد.**

<https://www.youtube.com/watch?v=j4_ICnyFTy8>

# **Sorting with pagination in Laravel using Ajax**

1. اضافه کردن مرتب سازی در اولین لود صفحه

Public function index()

{

$data = Post::all()->orderBy(“id” , asc) ->**paginate(5);**

Return view(‘pagination’ , compact(‘data’));

}

1. اضافه کردن 4 مورد جدید به قسمت Header جدول نمایش داده ها در View اصلی :
   1. اضافه کردن کلاس **Sorting**  به هدر ستون مورد نظر
   2. اضافه کردن attr با نام **data-sorting\_type = “asc”**  به هدر ستون های مورد نظر
   3. اضافه کردن attr با نام **data-column\_name = “…”**  به هدر ستون های مورد نظر
   4. اضافه کردن span با id مرتبط برای هر ستون . مثلا : <span id=”post\_title\_icon”></span>
2. View میانی یا همان pagination\_data تغییری نسبت به قبل ندارد.
3. Include کردن pagination\_data در ویو اصلی.
4. تعریف Route

**فاز اول یعنی همان pagination برقرار شد.**

اکنون می خواهیم که همین را با Ajax انجام بدهیم و نسبت به روش قبلی با کمی تغییر.

1. **سه input** در ویو اصلی، بعد از جدول نمایش داده ها قرار می دهیم.

در واقع در Value این input ها :

* + اولی : **شماره page**ی که کلیک شده است را نگهداری می کنیم.
* <input type=’hidden’ name=’hidden\_page’ id=’hidden\_page’ **value=’1’** />
  + دومی : **نام ستونی** که کلیک شده است را نگهداری می کنیم.
* <input type=’hidden’ name=’hidden\_column\_name’ id=’hidden\_column\_name’ **value=’id’** />
  + سومی : **نوع مرتب سازی ستونی** که کلیک شده است را نگهداری می کنیم.
* <input type=’hidden’ name=’hidden\_sort\_type’ id=’hidden\_sort\_type’ **value=’asc’** />

1. اکنون در قسمت اسکریپت همان ویو اصلی :
   1. به ازای کلیک روی یک page از لینک های Pagination
      1. مقادیر جدید به المان ها داده می شود.
      2. مقادیر ستون و page جمع آوری می شود.
      3. متد fetch\_data با سه پارامتر : page , columnname , sorttype فراخوانی می شود.

$(document).on(‘click’ , ‘.pagination a’ , function(event){

Event.preventdefault();

Var page = $(this).attr(‘href’).split(‘page=’)[1];

$(“hidden\_page”).val(page);

Var column\_name = $(“hidden\_column\_name”).val();

Var sort\_type = $(“hidden\_sort\_type”).val();

**Fetch\_page(page, sort\_type , Column\_name);**

});

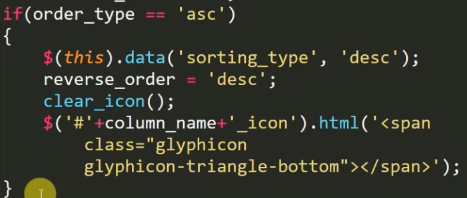
* 1. به ازای کلیک روی یک هدر ستون داده ها برای مرتب سازی آن ستون
     1. ابتدا مقادیر ستون و page جمع آوری می شود.
     2. مقادیر جدید به المان ها داده می شود.
     3. متد fetch\_data با سه پارامتر : page , columnname , sorttype فراخوانی می شود.

$(document).on(‘click’ , ‘.sorting’, function(){

Var column\_name = $(“hidden\_column\_name”).val();

Var sort\_type = $(“hidden\_sort\_type”).val();

Var reverse\_order = ‘’;





$(“#hidden\_column\_name”).val(column\_name);

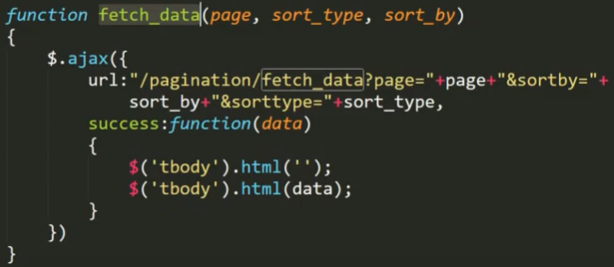
$(“#hidden\_sort\_type”).val(reverse\_order);

Var page = $(‘#hidden\_page’).val();

**Fetch\_page(page, sort\_type , Column\_name);**

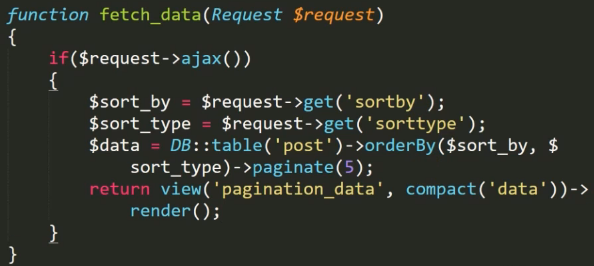
});

1. اضافه کردن متد Ajax :



1. اکنون سمت کنترولر این تابع یا متد fetch\_page را پیاده سازی می کنیم.

داده ها به این متد بصورت Ajax می رسند. (؟ چگونه )



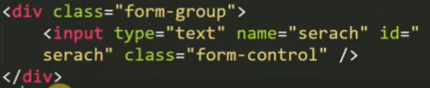
1. تنظیم Route

**اتمام فاز دوم و راه اندازی صفحه بندی و مرتب سازی با هم .**

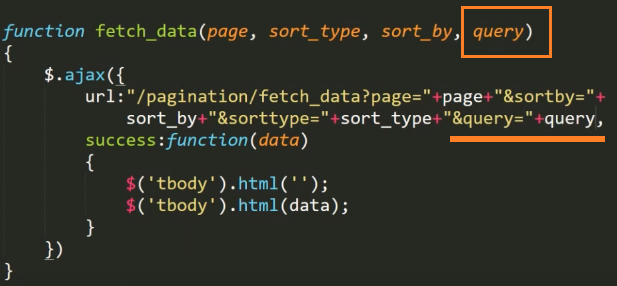
<https://www.youtube.com/watch?v=2U1t9f-64_4>

# **Laravel Live Data Search with Sorting & Pagination using Ajax**

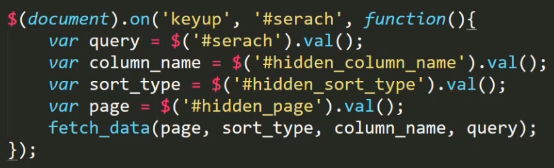
1. اضافه کردن یک input text در قسمت بالای جدول



1. تکمیل تابع fetch\_data برای دریافت و ارسال مقدار جستجو – اضافه کردن پارامتر جدید



1. اضافه کردن متدی برای خواندن کلیدها در باکس جستجو



1. اکنون پارامتر جستجو و یا همان Query را در هر دو تابع مرتبط با رخداد کلیک روی paging و یا کلیک روی Column Header اضافه می کنیم.



1. تکمیل خود متد fetch\_data برای دریافت پارامتر جستجو و اعمال آن



))))))))))

ادامه پروژه :

* در تولید داده های Fake باید دقت کنیم که رکوردهایی که comment\_confirmationstatus = 0 است یعنی هنوز بررسی نشده اند پس این رکوردها باید confirmdate , confirmuserid = Null باشد .

|  |  |
| --- | --- |
| تا اینجا Pagination با refresh صفحه انجام شد. یعنی بدون Ajax .  در ضمن این refresh شدن باعث شد که از قالب صفحه Admin Panel بیرون بیاید.  نکته در مورد پهنای ستون ها :  اگر بخواهیم پهنای تعیین شده برای ستون ها موثر باشد باید ابتدا برای Table این پروپرتی ها را مشخص کنیم.  style="table-layout: fixed; width: 100%;"  بعد برای ستون ها می توانیم width بدهیم .  <th style="width:5%"> |  |

با Ajax هم انجام شد.

اکنون برای قسمت ابتدایی کار ، یعنی تکمیل لیست ، چند کار باقی مانده است :

1. جایگذاری مقادیر title از جداول مرتبط به جای id
2. اضافه کردن مقدار جستجو روی همه موارد
3. اضافه کردن کلید ویرایش

برای قسمت اول ، ما این نمودار را داریم که باید با Eloquent آنرا پیاده سازی کنیم.

|  |  |
| --- | --- |
| * باید دو بار با جدول users ارتباط برقرار کنیم. * یکبار با جدول product * یک بار هم که قبلا جدول product با categories ارتباط برقرار کرده و ما اینجا باید عنوان Category را هم داشته باشیم.   یک راهکار دیگر :   * می توانیم یک view هم داشته باشیم. |  |

**In Comment Model :**

class Comment extends Model

{

    public function user()

    {

        return $this-> belongsTo('App\User' , "comment\_userid");

    }

    public function confirmerUser()

    {

        return $this-> belongsTo('App\User' , "comment\_confirmeruserid");

    }

    public function product()

    {

        return $this->belongsTo("App\Product" , "comment\_productid");

    }

}

**In Views: (For Example)**

<td>{{ $comment->product->product\_title }}</td>

اما نکته مهم اینکه کلیدهای خارجی همه باید درست باشند.

<https://stackoverflow.com/questions/7045653/setting-max-height-for-table-cell-contents>

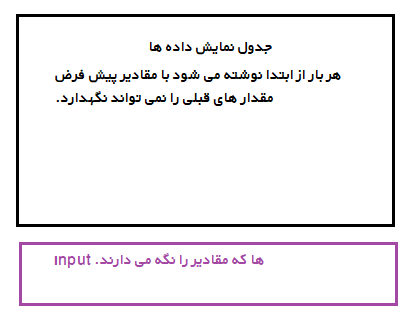
\*\*\*\*\*\*\*\*\* حتما قبل از استفاده از متغیر هایی که به View پاس می دهیم چک کنیم که آیا مقدار دارند و یا خیر :

isset

@php

    $p\_pagenumber = isset($p\_pagenumber) ? $p\_pagenumber : 1;

@endphp

برای مرتب سازی ستون ها یک نکته مهم است :   


<https://stackoverflow.com/questions/52370510/laravel-migration-specified-key-was-too-long-max-key-length-is-767-bytes>

جدید

1. فایل های js رو در مسیر resource نمی توان گذاشت.
2. آنها رادر مسیر Public قرار دادم.
   1. مشکل : اما مشکل این است که چطور این اسکریپت ها را داخل Document ready بیاورم ؟
   2. با ابن ترتیبی که اضافه کردم امکان شناسایی متغیرهای عمومی را هم ندارد که در اسکریپت اصلی معرفی شده اند.
      1. ظاهرا یکی از راهکارها استفاده از $.getScript("js/adminpanel/markalar.js"); هست که راهنمای کامل آن اینجاست. <https://api.jquery.com/jQuery.getScript/>
      2. یه نکته اینه که این تابع کار عجیب غریبی نمی کنه و در واقع داره با ajax کد شما رو فراخوانی می کنه :

This is a shorthand Ajax function, which is equivalent to:

**$.ajax({**

**url: url,**

**dataType: "script",**

**success: success**

**});**

* + 1. اما باید دقت کرد که $.getScript is an asynchronous method این تابع یک asynchronous است. یعنی ممکنه که خروجی ها به ترتیبی که مورد انتظار شماست نباشد و یا به ترتیبی که نوشته اید نباشد. مثلا :

$.getScript('a.js', function () {

console.log('Loaded a');

});

$.getScript('b.js', function () {

console.log('Loaded b');

});

$.getScript('c.js', function () {

console.log('Loaded c');

});

// Output could be:

// Loaded c

// Loaded a

// Loaded b

این هم توضیح خیلی خوبی داده :

<https://stackoverflow.com/questions/13762243/having-difficulty-working-with-global-variables-and-getscript>

آموزش ساخت Admin Panel

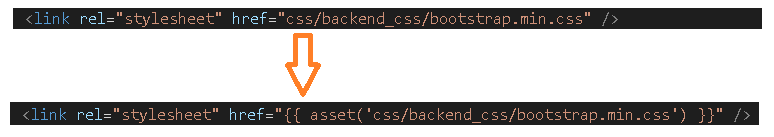
<https://www.youtube.com/watch?v=-t7oMHzIpBQ&list=PLLUtELdNs2ZY5drPxIWzpq5crhantlzp7>

1. انتخاب یک template رایگان برای admin panel
   1. انتخاب Matrix از سایت Bootstrap free admin panel template با آدرس wrappixel.com

<https://wrappixel.com/demos/free-admin-templates/matrix-admin/index.html>

* 1. دانلود این Template

1. انتقال این template به داخل لاراول
   1. دو فولدر Backend\_css , Frontend\_css را ایجاد می کنیم. در فولدر css و تمام css های این جا را به Backend\_css کپی می کنیم.
   2. به همین ترتیب دو فولدر برای js می سازیم و فایل های js را به آن کپی می کنیم. در Backend\_js
   3. به همین ترتیب در مورد image ها
   4. و فونت ها
2. ایجاد کنترلر admin
   1. Php artisan make:controller AdminController
   2. در این کنترولر متد لاگین را می نویسیم. متدی که فقط view با نام admin\_login را بر می گرداند.
3. ساخت ویو لاگین
   1. ابتدا فولدر admin را در فولدر resources می سازیم.
   2. سپس فایل admin\_login.blade.php را ایجاد می کنیم.
   3. حالا همه محتوای فایل login.html را از template برداشته و به داخل این فایل کپی می کنیم.
   4. اصلاح مسیر ها برای css , js , images , fonts
      1. از asset هم برای آدرس دهی استفاده می کنیم.
      2. Asset در اقع فایل های css , js , image , font های ما در مسیر public است. و لارارول با استفاده از تابع asset() یک URL برای این مسیر ها می سازد.



1. ایجاد route برای لاگین
   1. طبق این route با درج آدرس admin اولین جایی که لود می شود فرم لاگین است.

Route::get(‘/admin’ , ‘AdminController@login’)

1. ایجا فرآیند لاگین :
   1. Migrate کردن .

رفع اشکال migrate در لاراول : به فایل appserviceprovider این را اضافه می کنیم.

use Illuminate\Support\Facades\Schema;

public function boot()

{

Schema::defaultStringLength(191);

}

1. ایجاد authentication
   1. با استفاده از دستور php artisam make:auth ساختار امنیتی را ایجاد می کنیم.
   2. می تونیم با این آدرس هم این موضوع را چک کنیم.

<http://127.0.0.1:8000/register>

* 1. سپس فیلد admin از نوع Boolean را به جدول users اضافه می کنیم.
  2. اکنون به صفحه login خودمان می رویم و action آنرا اصلاح می کنیم.

method="POST" action="{{ url('/admin') }}"

* 1. و نیز {{ csrf\_field() }} را فراموش نمی کنیم.
  2. به فیلدهای فرم name را هم اضافه می کنیم.
  3. باتن لاگین را هم از حالت href به حالت input و submit تغییر می دهیم.
  4. اصلاح route به نحوی هم که get و هم post را قبول کند.

Route::match(['get' , 'post'] , '/admin', 'AdminController@login');

* 1. اکنون کنترولر لاگین را کامل می کنیم.
     1. ابتدا نوع متد دریافتی را چک می کنیم که مطمئن شویم که submit رخ داد.
     2. بعد با استفاده از Auth::attempt چک می کنیم نام کاربری و رمز عبور کاربر را.

    public function login(Request $request)

    {

        if ($request->isMethod('post'))

        {

            $data = $request->input();

            $credentials = $request->only('email', 'password' , 'admin');

            if (Auth::attempt($credentials))

            {

                echo "success!!";

                die;

            }

            else

            {

                echo "failed!!";

                die;

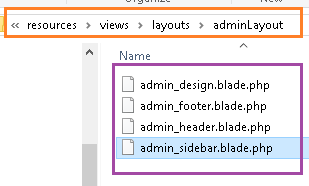
            }

        }

        return view('admin.admin\_login');

    }

1. ساخت داشبور – به همراه Layout ها
   1. ابتدا دو Layout درست می کنیم . هم برای admin و هم برای front .
   2. این دو لای اوت را در فولدرهای جداگانه هم قرار می دهیم.
   3. اول برای admin که شامل header , Footer , … برای Admin است.
      1. فولدری با نام adminLayout می سازیم. که شامل این فایل ها خواهد شد .
      2. Admin\_design.blade.php که شامل طراحی کامل صفحه admin است .
         1. یعنی فایل های بعدی را هم در خودش دارد.
      3. Admin\_header.blade.php
      4. Admin\_footer.blade.php
      5. Admin\_sidebar.blade.php



* + 1. حالا کلیه محتوای داشبورد (فایل index از template ) را به این فایل یعنی admin\_design.blade.php منتقل می کنیم .
    2. و قسمت های مربوط به header , footer , sidebar را به فایل هایی که گفتیم منتقل می کنیم.
    3. و در فایل اصلی یعنی design فایل های Header , footer , sidebar را هم include می کنیم.
    4. سپس فایلی برای محتوای اصلی تهیه می کنیم. Dashboard.blade.php و آنرا در فولدر admin قرار می دهیم.
       1. اما مجددا این فایل را include نمی کنیم بلکه بصورت @yield قرار می دهیم.
       2. سپس در فایل داشبورد این فایل را extend می کنیم.
    5. در ادامه اصلاح مسیرهای css و ... و ... در این فایل ها.
    6. ایجاد route مناسب برای داشبورد.
       1. یعنی از لاگین به داشبورد برسیم.
       2. در داخل کنترلر به ازای موفقیت باز می فرستیم سراغ مسیریاب admin/dashboard

1. ساخت صفحه نمایش خطای لاگین ناموفق و نیز فرآیند Logout
   1. مثلا برای نام کاربری و یا رمز عبور اشتباه
   2. ابتدا در AdminController به ازای faile باید پیغام خطا را هم ارسال کنیم.

    public function login(Request $request)

    {

        if ($request->isMethod('post'))

        {

            $data = $request->input();

            $credentials = $request->only('email', 'password' , 'admin');

            if (Auth::attempt($credentials))

            {

                // echo "success!!";

                // die;

            return redirect('/admin/dashboard');

            }

            else

            {

                // echo "failed!!";

                // die;

                return redirect('/admin')->with( 'flash\_message\_error' , 'Invalid Username or Password!!');

            }

        }

        return view('admin.admin\_login');

    }

* 1. حال باید پیغام خطای ارسال شده را در admin\_login.blade.php نمایش دهیم.

In simple display :

        @if(Session::has('flash\_message\_error'))

            {!! session('flash\_message\_error') !!}

        @endif

With bootstrap view :

…

* 1. Logout
     1. ابتدا آدرس Logout را در Header پیدا می کنیم و به جای آن route صحیح Logout را قرار می دهیم.

<li class=""><a title="" href="{{ url('/logout') }}"><i class="icon icon-share-alt"></i> <span class="text">Logout</span></a></li>

* + 1. در web.php یک Route جدید ایجاد می کنیم. برای Logout

Route::get('/logout', 'AdminController@logout');

* + 1. اکنون یک متد جدید را هم در کنترلر ایجاد می کنیم.

    public function logout()

    {

        //Session::flash();

        return redirect('/admin')->with('flash\_message\_success' , 'Logout Successfuly!!');

    }

1. محافظت از داشبرد
   1. الان اگر در حالت Logout هم آدرس داشبورد را بزنیم وارد می شود. که باید محافطت شود.
   2. از دو راه یم توانیم محافظت کنیم :
      1. با session
      2. با middleware
   3. از راه session :
      1. ابتدا در زمان لاگین ما session را مقدار می دهیم و سپس برای هر عملیات داخل admin انرا چک می کنیم.
      2. این ساده ترین راه است. ابتدا این روش را امتحان می کنیم.
         1. ابتدا در کنترلر در زمان موفقیت لاگین یک session با نام adminsession ایجاد می کنیم.

Session::put('adminSession',$data['email']);

* + - 1. حالا در متدد داشبورد این رو چک می کنیم.

        if(Session::has('adminSession'))

        {

            return view('admin.dashboard');

        }

        else

        {

            return redirect('/admin')->with('flash\_message\_error' , 'Please login to access admin panel');

        }

* 1. از راه Middleware :
     1. ابتدا باید در Router یک مسیر Group برای همه مسیر های admin ایجاد کنیم و سپس آنرا چک کنیم.
     2. فعلا موارد session رو غیر فعال می کنیم و این متد جدید را شروع می کنیم.
     3. **ابتدا** به سراغ Route می رویم :

Route::group(['middleware'=>['auth']], function(){

    Route::get('/admin/dashboard', 'AdminController@dashboard');

});

* + 1. **اکنون** در قسمت middleware ها ما یک فایل آماده ایجاد شده داریم با نام RedirectIfAuthenticated و باید داخل آن تغییراتی بدهیم.
    2. این else را به کد موجود اضافه میکنیم :

        else

        {

            return redirect()->action('AdminController@login')->with('flash\_message\_error','please login to access');

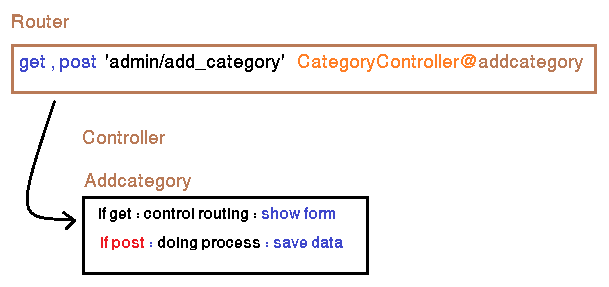
        }

تمام

تا اینجا نگاه کردم.

<https://www.youtube.com/watch?v=TILCwtTtmWA>

1. Page Change password
   1. ایجاد change password form
   2. ابتدا تهیه فرمی در قالب داشبورد
   3. کپی از فرم های template
   4. Create route
   5. Create controller method
   6. New we should copy all javascript code to validate
   7. Remove extra froms
   8. Add new fileds
   9. حالا باید برویم و در فایل اسکریپت مربوط به form Validation ها در template
   10. نام این فایل matrixform\_validation.js است.
       1. Github.com/stachdevelopers
   11. اکنون باید با استفاده از Ajax چک کنیم که آیا پسوردها درست است یا خیر ؟
   12. ابتدا بعد از ورود پسورد فعلی و در زمان ورود به پسورد جدید چک می کنیم که ایا این پسورد درست است یا خیر. تغییرات را هم در همان فایل js می دهیم.
   13. بعد سراغ route می رویم .
   14. Controller – create method
   15. <https://www.youtube.com/watch?v=Nn9qkHCb8GA>
   16. در adminController این دو باید use شوند : app , hash
   17. اکنون update کردن پسورد که باید در یک متد دیگر انجام شود.
   18. سپس بدون استفاده از ajax آنرا Update می کنیم. می توانستیم با آزکس هم آپدیت کنیم.
   19. اضافه کردن قسمت نمایش پیغام ها در فرم settings .
2. First insert Markalar
   1. Create table by migration
   2. At first start with a bigger model – Like Category – with more fileds
   3. Run migrate
   4. Create controller
   5. Create Model By artisan : php artisan make:model Category
   6. Create Roate for add category



* 1. Create method addcategory in controller - for first step (will change)

    public function addcategory()

    {

        return view('add\_category');

    }

* 1. Now we create add\_category.blade.php
  2. برای این کار ابتدا داخل فولدر admin یک فولدر categories درست می کنیم و داخل آن فایل را می سازیم.
  3. حال در زمان شروع به ساخت add\_category.blade.php ابتدا design layout آنرا ایجاد می کنیم. .

@extends('layouts.adminLayout.admin\_design')

@section('content')

@endsection

* 1. حالا می توانیم هم فرم خودمان را بسازیم و هم می توانیم از الگوهای form validation استفاده کنیم. مثلا همان password validation .
  2. فعلا صفحه را با این آدرس چک می کنیم. <http://127.0.0.1:8000/admin/add_category>
  3. مجددا از اسکریپت موجود برای کنترل استفاده می کنیم.
  4. فرم را تمیز می کنیم و فیلدهای لازم را اضافه می کنیم.
  5. Base\_validation را در فایل اسکریپت کپی کرده و تکرار می کنیم و کنترل فیلدها را با آن انجام میدهیم.

    // Category Validation

    $("#add\_category").validate({

        rules: {

            category\_name: {

                required: true

            },

            description: {

                required: true,

            },

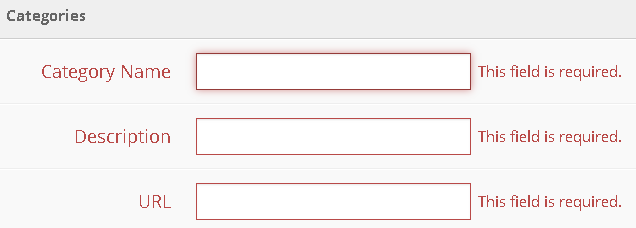
            url: {

                required: true,

            }

        },

* 1. تا همین مرحله چک می کنیم و جواب می دهد :



* 1. تکمیل کنترولر – ابتدا تست آن

    public function addcategory(Request $request)

    {

        if($request->isMethod('post'))

        {

            $data = $request->all();

            echo '<pre>';

            print\_r($data);

            die;

        }

        return view('admin.categories.add\_category');

    }

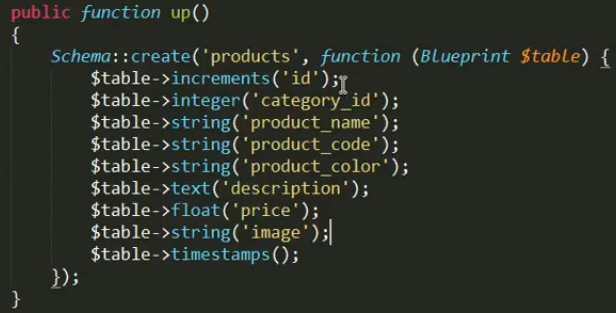
* 1. New complete method

1. نمایش Category ها با استفاده از Datatable Jquery .

<https://www.youtube.com/watch?v=-YBGa17v2MQ>

Create Product

1. Create table



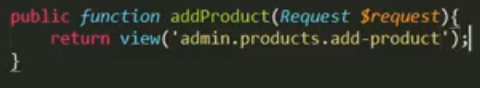
1. Create Controller

Php artisan make:controller **ProductsController**

1. Create Model Product
2. Create Route for add product



1. Create addProduct method in controller



1. Create add\_product.blade.php file in product folder