# Laravel

**laravel□□□□**

- □□□□

# □□□□

# □□laravel

## php & composer

```
sudo apt install php composer php-cli php-common php-mbstring php-xml php-zip php-mysql php-json php-bcmath php-curl php-gd php-tokenizer php8.1-xml php-fpm mariadb-server nginx
```

□□□□□□□□□□apache or nginx□nginx□□□□php-fpm

```
php -v
```

```
sudo apt install curl php-cli php-mbstring git unzip
curl -sS https://getcomposer.org/installer | sudo php -- --install-dir=/usr/local/bin --filename=composer
```

```
composer --version
```

## install laravel

```
composer create-project --prefer-dist laravel/laravel your-project-name
cd your-project-name
```

# install apache2 or nginx

## apache

```
sudo vi /etc/apache2/sites-available/your-project-name.conf
```

```
<VirtualHost *:80>
    ServerName your-domain-or-ip
    DocumentRoot /var/www/html/your-project-name/public
    <Directory /var/www/html/your-project-name>
        AllowOverride All
    </Directory>
</VirtualHost>
```

```
sudo a2enmod rewrite
```

Enable the virtual host:

```
sudo a2ensite your-project-name.conf
```

Restart Apache for the changes to take effect:

```
sudo systemctl restart apache2
```

## nginx

```
sudo vi /etc/nginx/sites-available/your-project-name
```

```
server {
    listen 80 default_server;
    server_name your-domain-or-ip;
    return 301 https://$server_name$request_uri;
    }

server {
    listen 443 ssl default_server;
    server_name your-domain-or-ip;

    include snippets/ssl;
    include snippets/ssl;

    root /var/www/html/your-project-name/public;
    index index.php;

    location / {
 #       try_files $uri $uri/ /index.php?$query_string;
    try_files $uri $uri/ /index.php?$args;
    }

    error_page 404 /404.html;
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
    root /var/www/html;
    }
```

```
        location ~ \.php$ {
                include snippets/fastcgi-php.conf;
        #       With php-fpm (or other unix sockets):
                fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
                fastcgi_index index.php;
                fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
                include fastcgi_params;
        #       # With php-cgi (or other tcp sockets):
        #       fastcgi_pass 127.0.0.1:9000;
        }
        # deny access to .htaccess files, if Apache's document root
        # concurs with nginx's one
        #

        add_header X-XSS-Protection "1; mode=block";
#       add_header 'Access-Control-Allow-Origin' *;
#       add_header 'Access-Control-Allow-Credentials' true;
#       add_header Strict-Transport-Security "max-age=31536000; includeSubdomains; preload";
#       add_header Content-Security-Policy "default-src 'self' http: https: data: blob: 'unsafe-inline'" always;

        location ~*  \.(jpg|jpeg|gif|png|svg)$ {
                expires 365d;
        }

        location ~*  \.(pdf|css|html|js|swf)$ {
                expires 2d;
        }

        location ~ /\.ht {
                deny all;
        }

        location ~ ^/\.user\.ini {
                deny all;
        }

       location ~ /.well-known/acme-challenge {
        root /var/www/html;
              allow all;
```

```
    }

    location ~ /\.ht {
        deny all;
    }
}
```

Replace your-domain-or-ip with your actual domain name or server IP address.
Enable the Nginx server block:

```
sudo ln -s /etc/nginx/sites-available/your-project-name /etc/nginx/sites-enabled/
```

Test the Nginx configuration for any syntax errors:

```
sudo nginx -t
```

Restart Nginx for the changes to take effect:

```
sudo systemctl restart nginx
```

# MariaDB □□□

## □□□□ MariaDB □□□□□□□□□□□□□□□□□□□□□□□□

MariaDB/MySQL □□□□□□□□

```
sudo mysql_secure_installation
```

□□ root □□ MariaDB/MySQL □□□

```
sudo mysql -u root -p
```

## □□ laravel □□□

```
CREATE DATABASE laravel;
CREATE USER `user`@`localhost` IDENTIFIED BY 'yourpassword';
GRANT ALL ON laravel.* TO `user`@`localhost`;
FLUSH PRIVILEGES;
```

□□ Laravel □□□□□□ .env □□□□□□□□□□□□□□□□□□

```
DB_DATABASE=laravel
DB_USERNAME=user
DB_PASSWORD=yourpassword
```

除了 .env 的設定檔之外，也可以直接編輯 config/database.php 中的資料庫設定。

```
php artisan serve 伺服器   #開啟服務，預設8000
php artisan migrate        #建立使用資料庫
php artisan about          #快速瀏覽環境設定
```

### Configure Laravel

```
cp .env.example .env
```

### Generate a new application key

```
php artisan key:generate
```

### Set the appropriate permissions on Laravel directories

```
sudo chown -R www-data:www-data /var/www/html/your-project-name/storage
sudo chmod -R 775 /var/www/html/your-project-name/storage
```

## Laravel Breeze

```
composer require laravel/breeze --dev      # 安裝Breeze
```

安裝npm跟nodejs，請注意版本問題

```
sudo apt install npm
sudo apt install nodejs
```

**ubuntu** 系統函式庫的**nodejs**版本通常會太舊，可以用下方方式處理，請問注意版本，或可以從**https://nodejs.org/en**下載

```
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs
php artisan breeze:install
php artisan migrate
npm install
```

直接運作

```
npm run dev
```

# php artisan□□

```
php artisan env:encrypt      #□□□□□□
```

**□□□□□□□□.env□□□□.env.encrypted□□□□□□□□□□□□□□□□□□□□□□□□□□**

```
php artisan env:encrypt --key=□□
php artisan env:decrypt      #□□
```

**□□□--key □□□□**

```
php artisan env:decrypt --force
```

php artisan serve　　#□□□webagent□□□□□□□8000port

```
php artisan serve      #□□□webagent□□□□□□□8000port
```

## □□mode□database□□□□

□□□□□□□□□□□□□□□□

make:model->migrate->make:factory->make:seeder->db:seed->make:controller->create view->set route

```
php artisan make:model Movie -m
```

```
php artisan migrate     #□□□□□
php artisan make:migration create_□table□□_table     #□□□table
php artisan make:migration create_aboutme_photo — table=aboutme
php artisan migrate:status     #□□□□□□□
php artisan migrate□rollback  #□□□□□
php artisan migrate:refresh     #□□□□□□□□□□□□
```

## □□Factory□□□

```
php artisan make:factory MovieFactory --model=Movie
```

## □□seeder

```
php artisan make:seeder MovieSeeder
```

## □□control

```
php artisan make:controller MovieController
```

## □□□□□□

```
php artisan down
```

□□□
□□□□□□--refresh=(□□) □□□□□□□□□□□□□□□□
--render="errors::503" □□□□503□□□view

## □□□□□□

```
php artisan up
```

□□□□□□□□□□

```
php artisan down --secret="1630542a-246b-4b66-afa1-dd72a4c43515"
```

□□□□□https://example.com/1630542a-246b-4b66-afa1-dd72a4c43515

## □app□□job□□

```
php artisan make:job
```

**□□□php artisan make list □□□□□□□□□□**

## route□□

```
php artisan route:list
```

**□□□**

1. **--except-vendor□□□□□□□□□□route**
2. **--only-vendor □□□□□□□□**
3. **-v □□□Route Middleware**
4. **--path=api cjo4vu04g4eo32u/4**
5. **□□□□URI□□□Route**

```
php artisan vendor:publish --tag=laravel-pagination
```

**□□□□tailwindcss □□□□ bootstrap**
**□□□□□□ Laravel □□□□□□□□ resources/views/vendor/pagination**

## □□cache

```
php artisan cache:clear
php artisan route:clear
php artisan view:clear
```

# Route□□

```
Route::get($uri, $callback);
Route::post($uri, $callback);
Route::put($uri, $callback);
Route::patch($uri, $callback);
Route::delete($uri, $callback);
Route::options($uri, $callback);
```

```
Route::match(['get', 'post'], '/', function () {
 // ...
});
Route::any('/', function () {
 // ...
});
```

## □□□□

```
Route::redirect('/here', '/there');
Route::redirect('/here', '/there', 301);
Route::permanentRedirect('/here', '/there');
```

□□□□□□ users/index □□□

```
redirect()->to("users/index");
```

□□□□□□□□□□□□□□□

```
redirect("users/index");
```

□□□□□□, □□□□□□□□

```
Redirect::to("users/index");
```

□□□□□□

```
Redirect("users/index");
```

```
Route::view('/welcome', 'welcome');
Route::view('/welcome', 'welcome', ['name' => 'Taylor']);
```

## to()

```
function to($to = null, $status = 302, $header = [], $secure = null)
```

$to : □□□□□□□
$status □ HTTP □□
$header □□□□□□□□ HTTP □□
$secure : http vs. https □□□□□

## route()

```
function route($to = null, $parameters =[], $status = 302, $headers = [])
```

$to : □□□□□□
$parameters : □□□□□□□

## withInput()

□□□□□□□□□□□□□□□

```
redirect("users.edit")
    ->withInput()
    ->with(["success" => "update success", "id" => 2]);
```

## env□□□□

□□□□□□□□

```
'debug' => env('APP_DEBUG', false),
```

## □□□□

□.env

```
APP_TIMEZONE='America/New_York'
```

□app.php(□□□□□□□1)

```
'timezone' => 'America/New_York',
'timezone' => env('APP_TIMEZONE', 'UTC'),
```

**□□□□ □□ php artisan config:clear □□□□**

## □□□□

**□□□□□□□□□□□**

```
<!DOCTYPE html>
<html>
<head>
    <title>Laravel 10 Eloquent Model Search Example - ItSolutionStuff.com</title>
    <meta name="csrf-token" content="{{ csrf_token() }}">
    <link href="https://cdnjs.cloudflare.com/ajax/libs/twitter-bootstrap/5.0.1/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body>
<div class="container">
    <div class="card">
      <div class="card-header">
        <h2>Laravel 10 Eloquent Model Search Example - ItSolutionStuff.com</h2>
      </div>
      <div class="card-body">
          <form class="row g-3" method="GET" action="{{ route('users.index') }}">
            <div class="col-auto">
              <label for="search" class="visually-hidden">Search</label>
              <input type="text" class="form-control" id="search" placeholder="Search" name="search"value="{{
request()->search }}">
            </div>
            <div class="col-auto">
              <button type="submit" class="btn btn-primary mb-3">Search</button>
            </div>
          </form>
          <table class="table table-striped">
            <tr>
                <th>ID</th>
```

```
                <th>Name</th>
                <th>Email</th>
            </tr>
            @foreach ($users as $user)
                <tr>
                    <td>{{ $user->id }}</td>
                    <td>{{ $user->name }}</td>
                    <td>{{ $user->email }}</td>
                </tr>
            @endforeach
        </table>
        {{$users->links()}}
    </div>
  </div>
</div>
</body>
</html>
```

**□ {{$users->links()}}**
**□□ {{$users->links('pagination::bootstrap-5')}}**

□□□□□□□□□□

```
<!-- a Tag for previous page -->
<a href="{{$employees->previousPageUrl()}}">
    <!-- You can insert logo or text here -->
</a>
@for($i=0;$i<=$employees->lastPage();$i++)
    <!-- a Tag for another page -->
    <a href="{{$employees->url($i)}}">{{$i}}</a>
@endfor
<!-- a Tag for next page -->
<a href="{{$employees->nextPageUrl()}}">
    <!-- You can insert logo or text here -->
</a>
```

□□□□□□

```
$row = User::paginate(15);
```

□□□□□□url □query□□ withQueryString()

```
$row = User::where('name', 'LIKE', "%{$search}%")
      ->paginate(15)
      ->withQueryString();  □□□□□
```

□□□□□□□□□□□□□□□□□□□□□□□□

```
xxx.com/user/keyword=xxx&page=2
```

## fragment() □□□□

```
$row = User::where('name', 'LIKE', "%{$search}%")
    ->paginate(15)
    ->fragment('user_data')    #□□□□
    ->withQueryString();
```

□□□□□□□□□□□□□□□□□□□□□□□□

```
xxx.com/user/keyword=xxx&page=2#user_data
```

## □□□□□ Tailwindcss, □□□□□□ Bootstrap

```
//app/Providers/AppServiceProvider.php #□□
use Illuminate\Pagination\Paginator;    #□□
    public function boot(){
        Paginator::useBootstrap();      #□□
    }
/**
 * Get the validation rules that apply to the request.
 *
 * @return array
 */
public function rules()
{
    return [
        'title' => 'required|unique:posts|max:255',
        'body' => 'required',
    ];
}
```

required(□□) unique: table□□□□□□□max:□□□255 min:□□□□
email(email□□) date(□□) nullable(□□□□)

mimes:png,jpg,jpeg,webp 檔案格式 decimal:0,2 數字 size:11 數字數量
備註: 如果要使用到檔案需要在form新增enctype="multipart/form-data"

驗證規則

```
request()->validate([
    'password' => 'required|confirmed'
])
```

confirmed (密碼驗證
備註: 另一個值需要是password_confirmation, 如下

```
<div class="form-group row">
    <label for="password-confirm" class="col-md-4 col-form-label text-md-right">Confirm Password</label>
    <div class="col-md-6">
        <input id="password-confirm" type="password" name="password_confirmation" required="required"
class="form-control">
    </div>
</div>
```

# Selects

從資料表中取得所有的資料列

```
$users = DB::table('users')->get();
foreach ($users as $user)
{
    var_dump($user->name);
}
```

從資料表中取得單一資料列

```
$user = DB::table('users')->where('name', 'John')->first();
var_dump($user->name);
```

取得單一欄位值的列表

```
$roles = DB::table('roles')->lists('title');
```

這個方法將會回傳資料表 role 的 title 欄位值的陣列。你也可以透過下面的方法，為回傳的陣列指定自訂鍵值。

```
$roles = DB::table('roles')->lists('title', 'name');
```

□□□□□ (Select Clause)

```
$users = DB::table('users')->select('name', 'email')->get();
$users = DB::table('users')->distinct()->get();
$users = DB::table('users')->select('name as user_name')->get();
```

□□□□□□□□□□□□

```
$query = DB::table('users')->select('name');
$users = $query->addSelect('age')->get();
```

□□ where □□□□

```
$users = DB::table('users')->where('votes', '>', 100)->get();
```

## □or□□□

```
$users = DB::table('users')
                ->where('votes', '>', 100)
                ->orWhere('name', 'John')
                ->get();
```

## □□ Where Between

```
$users = DB::table('users')
                ->whereBetween('votes', array(1, 100))->get();
```

## □□ Where Not Between

```
$users = DB::table('users')
                ->whereNotBetween('votes', array(1, 100))->get();
```

## □□ Where In □□□

```
$users = DB::table('users')
                ->whereIn('id', array(1, 2, 3))->get();

$users = DB::table('users')
                ->whereNotIn('id', array(1, 2, 3))->get();
```

## □□(Order By)□□□(Group By) □ Having

```
$users = DB::table('users')
                ->orderBy('name', 'desc')
                ->groupBy('count')
                ->having('count', '>', 100)
                ->get();
```

## □□(Offset) □ □□(Limit)

```
$users = DB::table('users')->skip(10)->take(5)->get();
```

## Joins

□□□□□□□□□□ join □□□□□□□□□□□□
□□□ Join □□

```
DB::table('users')
            ->join('contacts', 'users.id', '=', 'contacts.user_id')
            ->join('orders', 'users.id', '=', 'orders.user_id')
            ->select('users.id', 'contacts.phone', 'orders.price')
            ->get();
```

Left Join □□

```
DB::table('users')
        ->leftJoin('posts', 'users.id', '=', 'posts.user_id')
        ->get();
```

□□□□□□□□□□ join □□

```
DB::table('users')
        ->join('contacts', function($join)
        {
            $join->on('users.id', '=', 'contacts.user_id')->orOn(...);
        })
        ->get();
```

□□□□□□□ join □□□ where □□□□□□□□□□ join □□□□□ where □ orWhere □□□□□□□□□□□□□ contacts □□□□□ user_id □□□□□

```
DB::table('users')
        ->join('contacts', function($join)
        {
```

```
        $join->on('users.id', '=', 'contacts.user_id')
             ->where('contacts.user_id', '>', 5);
    })
    ->get();
```

## 進階 Where

群組化參數

有些時候你需要更進階的 where 子句，像是「where exists」或巢狀的群組化參數。Laravel 的查詢產生器也可以處理這樣的情況；

```
DB::table('users')
        ->where('name', '=', 'John')
        ->orWhere(function($query)
        {
            $query->where('votes', '>', 100)
                  ->where('title', '<>', 'Admin');
        })
        ->get();
```

上面的查詢語法會產生下方的 SQL:

```
select * from users
where exists (
    select 1 from orders where orders.user_id = users.id
)
```

### 聚合

查詢產生器也提供各式各樣的聚合方法，像是 count、max、min、avg 及 sum。

使用聚合方法

```
$users = DB::table('users')->count();
$price = DB::table('orders')->max('price');
$price = DB::table('orders')->min('price');
$price = DB::table('orders')->avg('price');
$total = DB::table('users')->sum('votes');
```

## Raw Expressions

有些時候你需要使用 raw expression 在查詢語句裡，這樣的表達式會成為字串插入至查詢，因此要小心勿建立任何 SQL 隱碼攻擊點。要建立 raw expression，你可以使用 DB::

□□ Raw Expression

```
$users = DB::table('users')
                    ->select(DB::raw('count(*) as user_count, status'))
                    ->where('status', '<>', 1)
                    ->groupBy('status')
                    ->get();
```

□□□□□□□□□□

```
DB::table('users')->increment('votes');
DB::table('users')->increment('votes', 5);
DB::table('users')->decrement('votes');
DB::table('users')->decrement('votes', 5);
```

□□□□□□□□□□□□

```
DB::table('users')->increment('votes', 1, array('name' => 'John'));
```

□□
□□□□□□□□□□

```
DB::table('users')->insert(
    array('email' => 'john@example.com', 'votes' => 0)
);
```

□□□□□□ (Auto-Incrementing) ID □□□□□□□

□□□□□□□□□□□□ID□□□□□ insertGetId □□□□□□□□ ID□

```
$id = DB::table('users')->insertGetId(
    array('email' => 'john@example.com', 'votes' => 0)
);
```

□□□□□□□□□□

```
DB::table('users')->insert(array(
    array('email' => 'taylor@example.com', 'votes' => 0),
    array('email' => 'dayle@example.com', 'votes' => 0),
));
```

□□
□□□□□□□□□

```
DB::table('users')
            ->where('id', 1)
            ->update(array('votes' => 1));

Invoices::where('id', $id)->update($request->all());

Invoices::where('id', '>', 0)->update($request->all());
```

□□□

```
$data = DB::table('store_list')->where('ID',2)->where('area','□□')->get();
```

□□
□□□□□□□□□

```
DB::table('users')->where('votes', '<', 100)->delete();
```

□□□□□□□□□□

```
DB::table('users')->delete();
```

□□□□□

```
DB::table('users')->truncate();
```

## Unions

□□□□□□□□□□□□□□□□□□□□ (union)□□□□□□□□□

```
$first = DB::table('users')->whereNull('first_name');

$users = DB::table('users')->whereNull('last_name')->union($first)->get();
```

unionAll □□□□□□□□□ union □□□□□□□□□

## □□□□ (Pessimistic Locking)

□□□□□□□□□□□□□□□ SELECT □□□□□□□□□□□

□□□□ SELECT □□□□□□Shard lock□□□□□□□□□ sharedLock□

```
DB::table('users')->where('votes', '>', 100)->sharedLock()->get();
```

要「鎖住更新(lock for update)」在 select 語法時，你可以使用「lockForUpdate」方法:

要在 SELECT 語句中「鎖住更新」，你僅需在查詢語句中使用 lockForUpdate 方法即可:

```
DB::table('users')->where('votes', '>', 100)->lockForUpdate()->get();
```

### 快取查詢結果

使用 remember 方法，你可以輕鬆的快取查詢結果:

```
$users = DB::table('users')->remember(10)->get();
```

這個範例中，查詢結果將會快取 10 分鐘。當結果被快取時，查詢語句將不會被執行，而會從應用程式指定的快取驅動器中載入快取的結果。

如果你正在使用的是 支援的快取驅動器，你也可以為快取增加標籤:

```
$users = DB::table('users')->cacheTags(array('people', 'authors'))->remember(10)->get();
```

## 如何使用auth

```
use Illuminate\Support\Facades\Auth;
auth()->user()
```

### 按照那個項目做排序

```
$results = Project::all()->orderBy("name");

$users = User::query()
    ->orderBy('name', 'desc')
    ->get();

$users = User::query()
    ->orderBy('name', 'desc') // [tl! --]
    ->orderByDesc('name') // [tl! ++]
    ->get();

$users = User::query()
    ->orderBy('name', 'desc')
    ->orderBy('email', 'asc')
    ->get();
```

```
$ordered = User::query()->orderBy('name');

$reorderedByEmail = $query->reorder('email', 'desc')->get();
```

## if□□□□

```
if($request->digital_signature == 'APPROVED'){
 // your code
}
elseif($request->digital_signature == 'NOT-APPROVED'){
// your code
}
```

## □□□□||□□□□□□□□

```
public function new_approvel_update(Request $request, $id)
{
    if($request->digital_signature == 'WITH DIGITAL SIGNATURE')
    {
          $input= Student::Where('delete_status','NOT DELETED')->find($id);
          $input['center_approved'] = strtoupper ('APPROVED');
          $input['date_of_join'] = $request->date_of_join;
    }
elseif($request->digital_signature == 'WITHOUT DIGITAL SIGNATURE') {
          $input= Student::Where('delete_status','NOT DELETED')->find($id);
          $input['center_approved'] = strtoupper ('NOT-APPROVED');
          $input['date_of_join'] = $request->date_of_join;
    }

  $certificate->save();

  return redirect('new_application')->with('success',' APPLICATION APPROVED SUCCESSFULLY .');
}
```

## □□□□

```
Carbon::now()->format('d-m-Y')      #□□□□□□□□□□
```

## □□□□□□□

□□□□file□storage □□□□

```
use Illuminate\Support\Facades\File;

$file_path = public_path('path/to/your/file.txt');
if (File::exists($file_path)) {
    File::delete($file_path);
    echo 'File deleted successfully.';
} else {
    echo 'File does not exist.';
}
```

## □□□□□□

```
$files_to_delete = ['path/to/file1.txt', 'path/to/file2.txt', 'path/to/file3.txt'];

foreach ($files_to_delete as $file_path) {
    if (File::exists($file_path)) {
        File::delete($file_path);
        echo 'File deleted successfully: ' . $file_path . PHP_EOL;
    } else {
        echo 'File does not exist: ' . $file_path . PHP_EOL;
    }
}
```

## Download□□

```
use Illuminate\Support\Facades\Storage;

public function downloadFile($id){
  $path = Student::where("id", $id)->value("file_path");
  return Storage::download($path);}

public function downloadFile()
{
    $file_path = public_path('path/to/file.pdf');
    $file_name = 'custom_file_name.pdf';

    return response()->download($file_path, $file_name);
```

```
}
```

## □□□□

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\File;
class FileUpload extends Controller
{
  public function createForm(){
    return view('file-upload');
  }
  public function fileUpload(Request $req){
        $req->validate([
        'file' => 'required|mimes:csv,txt,xlx,xls,pdf|max:2048'
        ]);
        $fileModel = new File;
        if($req->file()) {
            $fileName = time().'_'.$req->file->getClientOriginalName();
            $filePath = $req->file('file')->storeAs('uploads', $fileName, 'public');
            $fileModel->name = time().'_'.$req->file->getClientOriginalName();
            $fileModel->file_path = '/storage/' . $filePath;
            $fileModel->save();
            return back()
            ->with('success','File has been uploaded.')
            ->with('file', $fileName);
        }
   }
}
```

## □□□□□hash□

```
$user->forceFill([
□'password' => Hash::make($request->password),
□'remember_token' => Str::random(60),
])->save();
```

## □□hash□□

```
if (Hash::check('plain-text', $hashedPassword)) {
}
```

## Left()

```
select left(dffgrrerew,3)     #□□□□□3□□□
```

## right()

```
select right(dffgrrerew,3)    #□□□□□3□□□
```

## substring()

```
select substring(dsafwegre,3,4)    #□□□□□□□□□4□□□□
select substring(dsafwegre,3)      #□□□□□□□□
select substring(dsafwegre,-3)     #□□□□□□□□□
select substring(dsafwegre,-4,2)   #□□□□□□□□2□
select substring(dsafwegre,w,2)    #□w□□□2□□□
select substring(dsafwegre,w,-2)    #□w□□□2□□□
```

□□□□□□□□□□□□□□□□□□

# Excel □□import/export

```
composer require maatwebsite/excel
```

□□import□export□app□□□

```
php artisan make:import UsersImport --model=User
```

app/Imports/UsersImport.php

```
<?php

namespace App\Imports;

use App\Models\User;
use Maatwebsite\Excel\Concerns\ToModel;
use Maatwebsite\Excel\Concerns\WithHeadingRow;
use Hash;
```

```
  class UsersImport implements ToModel, WithHeadingRow
{
    /**
    * @param array $row
    *
    * @return \Illuminate\Database\Eloquent\Model|null
    */
    public function model(array $row)
    {
        return new User([
            'name'     => $row['name'],
            'email'    => $row['email'],
             'password' => Hash::make($row['password']),
        ]);
    }
}
```

```
php artisan make:export UsersExport --model=User
```

app/Exports/UsersExport.php

```
<?php

namespace App\Exports;

use App\Models\User;
use Maatwebsite\Excel\Concerns\FromCollection;
use Maatwebsite\Excel\Concerns\WithHeadings;

  class UsersExport implements FromCollection, WithHeadings
{
    /**
    * @return \Illuminate\Support\Collection
    */
    public function collection()
    {
        return User::select("id", "name", "email")->get();
    }
      /**
```

```
     * Write code on Method
     *
     * @return response()
     */
    public function headings(): array
    {
        return ["ID", "Name", "Email"];
    }
}
```

```
php artisan make:controller UserController
```

app/Http/Controllers/UserController.php

```
<?php
  namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Exports\UsersExport;
use App\Imports\UsersImport;
use Maatwebsite\Excel\Facades\Excel;
use App\Models\User;

  class UserController extends Controller
{
    /**
    * @return \Illuminate\Support\Collection
    */
    public function index()
    {
        $users = User::get();
          return view('users', compact('users'));
    }

    /**
    * @return \Illuminate\Support\Collection
    */
    public function export()
     {
        return Excel::download(new UsersExport, 'users.xlsx');
```

```
    }

    /**
    * @return \Illuminate\Support\Collection
    */
    public function import()
    {
        Excel::import(new UsersImport,request()->file('file'));
        return back();
    }
}
```

routes/web.php

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\UserController;

/*
|--------------------------------------------------------------------------|
 Web Routes
|--------------------------------------------------------------------------|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider within a group which
| contains the "web" middleware group. Now create something great!
|
*/

Route::controller(UserController::class)->group(function(){
    Route::get('users', 'index');
    Route::get('users-export', 'export')->name('users.export');
    Route::post('users-import', 'import')->name('users.import');

});
```

resources/views/users.blade.php

```
<!DOCTYPE html>
<html>
```

```
<head>
   <title>Laravel 10 Import Export Excel to Database Example - ItSolutionStuff.com</title>
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
    <div class="container">
   <div class="card bg-light mt-3">
      <div class="card-header">
         Laravel 10 Import Export Excel to Database Example - ItSolutionStuff.com
      </div>
      <div class="card-body">
         <form action="{{ route('users.import') }}" method="POST" enctype="multipart/form-data">
            @csrf
            <input type="file" name="file" class="form-control">
            <br>
            <button class="btn btn-success">Import User Data</button>
         </form>
          <table class="table table-bordered mt-3">
            <tr>
               <th colspan="3">
                  List Of Users
                  <a class="btn btn-warning float-end" href="{{ route('users.export') }}">Export User Data</a>
               </th>
            </tr>
            <tr>
               <th>ID</th>
               <th>Name</th>
               <th>Email</th>
            </tr>
            @foreach($users as $user)
            <tr>
               <td>{{ $user->id }}</td>
               <td>{{ $user->name }}</td>
               <td>{{ $user->email }}</td>
            </tr>
            @endforeach
         </table>
        </div>
   </div>
</div>
```

```
    </body>
</html>
```

# autoload□□□(□□□□□□)

```
composer install --optimize-autoloader --no-dev

php artisan config:cache

php artisan route:cache

php artisan view:cache

APP_DEBUG = false         #□debug□□

{{ Auth::user()->name }}   #□□□□□□□□□

use App\Models\User;       #□□□□□□□□

$users = User::all();

<input type = "text" value="{{ old('title') }}" name = "title" class = "w-2/5 border-gray-300 p-2 mx-2">

<textarea name = "content" rows = "20" class = "w-4/5 m-auto border-gray-300 p-2 " >{{ old('content')
}}</textarea>

<textarea name = "summary" rows = "5" class = "w-4/5 m-auto border-gray-300 p-2" >{{ old('summary')
}}</textarea>

<input type = "date" value="{{ old('start_time') }}" name = "start_time" class = "border-gray-300 p-2 mx-2">

<input type = "date" value="{{ old('end_time') }}" name = "end_time" class = "border-gray-300 p-2 mx-2">
```

**old□□□□□□□□**

```
<form action="/register" method="POST" id="registration-form">
  @csrf
```

```
  <div class="form-group">
    <label for="username-register" class="text-muted mb-1"><small>Username</small></label>
    <input value="{{old('username')}}" name="username" id="username-register" class="form-control"
type="text" placeholder="Pick a username" autocomplete="off" />
      @error('username')
          <p class="m-0 small alert alert-danger shadow-sm">{{$message}}</p>
      @enderror
  </div>

  <div class="form-group">
    <label for="email-register" class="text-muted mb-1"><small>Email</small></label>
    <input value="{{old('email')}}" name="email" id="email-register" class="form-control" type="text"
placeholder="you@example.com" autocomplete="off" />
      @error('email')
          <p class="m-0 small alert alert-danger shadow-sm">{{$message}}</p>
      @enderror
  </div>

  <div class="form-group">
    <label for="password-register" class="text-muted mb-1"><small>Password</small></label>
    <input name="password" id="password-register" class="form-control" type="password"
placeholder="Create a password" />
      @error('password')
          <p class="m-0 small alert alert-danger shadow-sm">{{$message}}</p>
      @enderror
  </div>

  <div class="form-group">
    <label for="password-register-confirm" class="text-muted mb-1"><small>Confirm
Password</small></label>
    <input name="password_confirmation" id="password-register-confirm" class="form-control"
type="password" placeholder="Confirm password" />
      @error('password_confirmation')
          <p class="m-0 small alert alert-danger shadow-sm">{{$message}}</p>
       @enderror
  </div>

  <button type="submit" class="py-3 mt-4 btn btn-lg btn-success btn-block">Sign up for OurApp</button>
</form>
```

**@error**

## Laravel

```
Storage::disk('local')->put('example.txt', 'Contents');
```

**storage/app**

```
./vendor/bin/sail artisan storage:link
```

**public/storage** **storage/app/public** **storage/app/public** **artisan**

AWS S3 Laravel

```
./vendor/bin/sail composer require league/flysystem-aws-s3-v3 "^3.0"
```

S3 config/filesystem.php .env

```
's3' => [
'driver' => 's3',
'key' => env('AWS_ACCESS_KEY_ID'),
'secret' => env('AWS_SECRET_ACCESS_KEY'),
'region' => env('AWS_DEFAULT_REGION'),
'bucket' => env('AWS_BUCKET'),
'url' => env('AWS_URL'),
'endpoint' => env('AWS_ENDPOINT'),
'use_path_style_endpoint' => env('AWS_USE_PATH_STYLE_ENDPOINT', false),
'throw' => false,
],
```

.env

```
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
AWS_USE_PATH_STYLE_ENDPOINT=false
```

```
$contents = Storage::get('file.jpg');
```

如果希望取得一些基礎的檔案資訊，也非常簡單比方說，size() 可以取得檔案的大小

```
Storage::size('file.jpg');
```

lastModified() 可以取得檔案最後修改的時間

```
$time = Storage::lastModified('file.jpg');
```

## Laravel 存取其他伺服器: Http Client

Laravel 裡面已經安裝好了 guzzlehttp 套件，並且做了一層包裝。

我們只要使用 Illuminate\Support\Facades\Http 就能取得其他伺服器的回應

```
use Illuminate\Support\Facades\Http;


$response = Http::get('https://www.google.com');
```

取得回應之後，我們就能根據該回應的內容，判斷後續對應的行為。

比方說，可以取得回應的 Http Status

```
$response->status()
```

或者取得回應的 Header key-value pair 的陣列

```
$response->headers()
```

又或者取得回應的 body

```
$response->body()
```

當然，有時候資料不能這麼簡單的透過 HTTP GET 取得

可能必須透過 POST 取得

```
$response = Http::post('http://example.com/users', [
    'name' => 'Steve',
    'role' => 'Network Administrator',
]);
```

有些網站接收的不是 API 格式的內容，而是網頁的 Form Submit

```
$response = Http::asForm()->post('http://example.com/users', [
    'name' => 'Sara',
    'role' => 'Privacy Consultant',
]);
```

□□□□□□□□□□□□□□□□□□□□□ body □□□□□

```
$response = Http::withBody(
    base64_encode($photo), 'image/jpeg'
)->post('http://example.com/photo');
```

□□□□□□□□□□□□□□□□□□□ Header □□□□□

```
$response = Http::withHeaders([
    'X-First' => 'foo',
    'X-Second' => 'bar'
])->post('http://example.com/users', [
    'name' => 'Taylor',
]);
```

## □□

```
@switch($role)
    @case('admin')
        <p>You are an admin.</p>
        @break
    @case('user')
        <p>You are a user.</p>
        @break
    @default
        <p>Your role is not defined.</p>
@endswitch
```

```
@isset($products)
    ...
@endisset
```

```
@empty($records)
    ...
@endempty
```

```
@for ($i = 0; $i < 10; $i++)
    The current value is {{ $i }}
@endfor
```

```
@foreach($products as $product)
    <li>{{ $product->name }}</li>
@endforeach
```

```
@forelse($products as $product)
    <li>{{ $product->name }}</li>
@empty
    <p>No products found.</p>
@endforelse
```

```
@while (true)
    <p>I'm looping forever.</p>
@endwhile
```

```
@auth
    <p>You're logged in</p>
@endauth
```

```
@guest
    <p>Please login</p>
@endguest
```

{{-- Add raw php code --}}

```
@php
    $counter = 1;
@endphp
```

{{-- Show JSON data --}}

```
<pre>@json($products, JSON_PRETTY_PRINT)</pre>
```

{{-- Create a stack (in a component)--}}

```
@stack('script')
```

{{-- Push to a stack (when using the component) --}}

```
@push('script')
    <script>
        console.log('Script is working!')
    </script>
@endpush
```

# CRUD

## Create

```
$newUser = User::create(['name' => 'John Doe', 'email' => 'john@example.com']);
```

## Read

```
$users = User::all();
$user = User::find(1);
$activeUsers = User::where('status', 'active')->get();
$tags = Product::where('name', 'Some Product')->first()->tags; // Because we define relationships it's easy to get related dat
```

## Update

```
$user = User::find(1);
$user->update(['name' => 'Updated Name']);
User::where('status', 'inactive')->update(['status' => 'active']);
```

## Delete

```
$user = User::find(1);
$user->delete();
User::where('status', 'inactive')->delete();
```

# Query Builder

## Selecting Data

```
$users = DB::table('users')
    ->select('name', 'email')
    ->where('name', 'like', "%vince%")
    ->orWhere('name', 'like', "%patrick%")
    ->where('active', true)
    ->orderBy('created_at', 'desc')
    ->get();
```

## Joining Tables and select specific Columns with conditions

```
$users = DB::table('users')
    ->join('orders', 'users.id', '=', 'orders.user_id')
    ->select('users.id as user_id', 'users.name as user_name', 'orders.id as order_id', 'orders.amount',
'orders.created_at')
    ->where('orders.status', '=', 'completed')
    ->where('users.active', '=', true)
    ->orderBy('orders.created_at', 'desc')
    ->get();
```

## Aggregates

```
$totalOrders = DB::table('orders')->count();
$averagePrice = DB::table('products')->avg('price');
```

## Insert, Update, Delete

```
DB::table('users')->insert(['name' => 'John Doe', 'email' => 'john@example.com']);
DB::table('users')->where('id', 1)->update(['name' => 'Updated Name']);
DB::table('users')->where('id', 1)->delete();
```

# Storage(□□□)

## Store a file in the default disk (usually 'public')

```
Storage::put('file.txt', 'Hello, Laravel!');
```

## Get the contents of a file

```
$contents = Storage::get('file.txt');
```

### Delete a file

```
Storage::delete('file.txt');
```

### Check if a file exists

```
if (Storage::exists('file.txt')) {
    // File exists
}
```

### Create a directory

```
Storage::makeDirectory('images');
```

### Delete a directory

```
Storage::deleteDirectory('images');
```

# Authentication

### Get the currently authenticated user

```
$user = auth()->user();             // or Auth::user();
```

### Get one attribute off the currently authenticated user (e.g. name, email, id, ...)

```
$name = auth()->user()->name;       // or Auth::user()->name;
```

# Middleware

### You can register your custom middleware for easier usage in app/Http/Kernel.php.

```
protected $middlewareAliases = [
    'auth' => \App\Http\Middleware\Authenticate::class,
    ...
    'active' => \App\Http\Middleware\ActiveUser::class,
```

```
];
```

Use your middleware on routes (for example in routes/web.php) which will cause the logic to be executed everytime a request is made to that route. Beware of the order you use when assigning middleware, this is also the order of execution.

// When registered you can use middleware like this.

```
Route::get('/profile', function () {
    // ...
})->middleware(['auth', 'active']);
```

// Alternative way is to use it like this (also possible when it's not registered)

```
Route::get('/profile', function () {
    // ...
})->middleware(Authenticate::class, Active::class);
```

□□□□□□□□

□□□□□□□□□□□?

□□□□□□□□□□□□□□□□□□□?□□□□□□□□□□□□□

```
$thisIsInt = 10;     → □□ (int)


$thisIsStr = "10";  → □□ (string)
```