

TrusCRUD

Introduction



As a developer, I once bought and used a tool such as a **CRUD Builder**. but when I first used it I felt it took quite a long time to learn. I want to have a tool that can be **learned quickly** and **immediately used** and supported by several databases. that's why I made this tool.

TrusCRUD is a tool to assist developers in building web applications with the **Laravel Framework**, with TrusCRUD developers no longer need to think about :

- Generate CRUD (**CRUD Builder** with CLI)
- User Management
- login with Social Accounts (Multiple provider per user)
- Role and Permissions
- Menu Management

TrusCRUD is very easy to develop or if you want to customize the results of the Generated **CRUD** .

TrusCRUD **Support several Databases** supported by Laravel because TrusCRUD utilizing migration tools from Laravel, [Read More](#)



We will continue to develop this tool as long as laravel continues to be developed, please support by buying our product

Installation

Server Requirements

Because this is made with the **Laravel**, Server Requirements the same as that. [Please Read](#)

Installing CRUD Generator

After you get the **Source Code**, extract it and open it in the terminal.

```
$ cd crud-project
```


Then install all package dependencies with the **composer**

```
$ composer install
```

Make sure all dependencies are installed, copy the `.env.example` to `.env` and after that edit the database configuration in the `.env` file, for example :

```
.env

1  #DB Section
2  DB_CONNECTION=pgsql #mysql,sqlsrv,sqlite
3  DB_HOST=127.0.0.1
4  DB_PORT=5432
5  DB_DATABASE=truscud
6  DB_USERNAME=root
7  DB_PASSWORD=root
```


 SQLite is not stable, maybe some page get errors

Run database migrate with seeder

```
$ php artisan migrate --seed
```

Run development server

```
$ php artisan serve --port=3000
```

 Login with default account

email : admin@mail.com / password: password

TrusCRUD successfully installed.

Guide

Users Management

You can manage users that have been created or registered, for manage user Open the user menu in the super admin section.

Menu Management

You can add or remove menu and change or reorder menu, in **Permission > Menu Accesses**.

CRUD Generator

Initial Generator

Before generate CRUD, you must **Create Initial Generator** code first. same like [make:migration](#) in Laravel.

Run this command for create generator code :

```
$ php artisan make:crud ClassName
```

And if you want make more than 1 CRUD you can do this command :

```
$ php artisan make:crud ClassNameOne,ClassNameTwo,ClassNameN
```

and then, you have new file in `{project_path}/app/TrusCRUD/Generator/CRUD`

Screen Capture

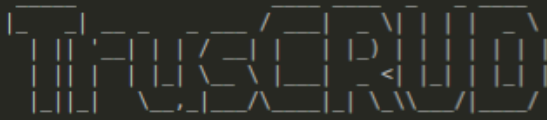
```
mydan3@linux:~/trus-crud-v1$ ./artisan make:crud Customer

  TRUSCRUD  | CLI - Crud Builder V1.0
             | Laravel 6.12.0
             | www.dani.work

CRUD Generator created: Customer.php
=> /home/mydan3/trus-crud-v1/app/TrusCRUD/Generator/CRUD/Customer.php
```

Make Single Crud

```
mydan3@linux:~/trus-crud-v1$ ./artisan make:crud Customer,Service,Invoice,Transaction
```



```
CLI - Crud Builder V1.0  
Laravel 6.12.0  
www.dani.work
```

```
CRUD Generator created: Customer.php
```

```
=> /home/mydan3/trus-crud-v1/app/TrusCRUD/Generator/CRUD/Customer.php
```

```
-----  
CRUD Generator created: Service.php
```

```
=> /home/mydan3/trus-crud-v1/app/TrusCRUD/Generator/CRUD/Service.php
```

```
-----  
CRUD Generator created: Invoice.php
```

```
=> /home/mydan3/trus-crud-v1/app/TrusCRUD/Generator/CRUD/Invoice.php
```

```
-----  
CRUD Generator created: Transaction.php
```

```
=> /home/mydan3/trus-crud-v1/app/TrusCRUD/Generator/CRUD/Transaction.php
```

Make Mutiple Crud

Setup CRUD Generator

To make the generator code run properly, you need to prepare some.

After you Create Initial Generator, open that file in

```
{project_path}/app/TrusCRUD/Generator/CRUD
```

 for you edit some code, for example :

Customer.php

```
1  <?php  
2  namespace App\TrusCRUD\Generator\Crud;  
3  
4  use App\TrusCRUD\Generator\CrudGenerator;  
5  
6  class Customer extends CrudGenerator {  
7  
8      public function initial() {  
9          //$this->setName('Customer');  
10         //$this->setSearchable('name');  
11         //$this->addColumn('name', 'type');  
12  
13         $this->setName('Customer');  
14         $this->setSearchable('fullname');  
15         $this->addColumn('fullname', 'string');
```

```
16         $this->addColumn('address', 'text');
17         $this->addColumn('email', 'string');
18         $this->addColumn('join_at', 'date');
19     }
20
21 }
```

The commented code is default, you can add more column with copy in new line and provide the name and type of the column.

Available Types for Column

- bigIncrements
- bigInteger
- binary
- boolean
- char
- date
- dateTime
- dateTimeTz
- decimal
- double
- enum
- float
- geometry
- geometryCollection
- increments
- integer
- ipAddress
- json
- jsonb
- lineString
- longText
- etc.

For more information about types available you can [Read More](#) in **Available Column Types** section.

Relationship

Also you can related the column with another model class, however **the class must already exist** or you must created before

for example :

```
Invoice.php

1  <?php
2  namespace App\TrusCRUD\Generator\Crud;
3
4  use App\TrusCRUD\Generator\CrudGenerator;
5
6  class Invoice extends CrudGenerator {
7
8      public function initial() {
9          $this->setName('Invoice');
10         $this->setSearchable('code');
11         $this->addColumn('trasaction_date', 'date');
12         $this->addColumn('code', 'string');
13         $this->addColumn('email', 'string');
14
15         $this->addColumn('customer_id', 'bigInteger')
16             ->relation('belongsTo', 'Customer');
17     }
18
19 }
```

The `relation($type , $className)` is method for add relation to column, in example column `customer_id` related to `Customer` .

Params

name	descriptions
\$type	Relation type (belongsTo, hasMany).
\$className	The class name you have.

Custom Form Field

You can change default form type with `formAs` for example you can make upload file in images field :

```
$this->addColumn('image', 'string')->formAs('upload');
```

Available :

- upload
 - ... coming soon.
-

Generate and Build CRUD

After configure and add what is needed in the Initial Generator, then You register the class in the `TrusCRUD/Generator/CURD/CrudBuilder.php` file,

for example :

CrudBuilder.php

```
1  <?php
2  namespace App\TrusCRUD\Generator\CRUD;
3
4  use App\TrusCRUD\Generator\CrudGenerator;
5
6  class CrudBuilder extends CrudGenerator {
7
8
9      public function run($cmd) {
10         $this->cmd = $cmd;
11
12
13         //Add All Class CRUD Generator
14         $this->call(Customer::class);
15         $this->call(Employee::class);
16
```

```
17      //.....
18      //Add More
19  }
20
21 }
```

Generate CRUD

Magic time..!

Run this command for generate the CRUD based on your **Setup**

```
$ php artisan crud:generate --migrate
```

`--migrate` use this option if you want **migrate directly**.

Remove CRUD

If doing mistake when you making CRUD, you can delete all file (controller, model, view, migration), menu, and table which create on CRUD generator.

Run this command for remove

```
$ php artisan crud:remove ClassName,ClassName --table
```

`--table` If you want to also delete the table that has been migrated. **be careful if the table has record.**

Changelog

v1.0.0 - 2020-01-03

Initial Release

- The awesome tool has been release

Added

- CRUD Generator (CLI)
- User Management
- Menu Management
- Role & Permission
- Socialite (Login with Social Account) multi provider per user