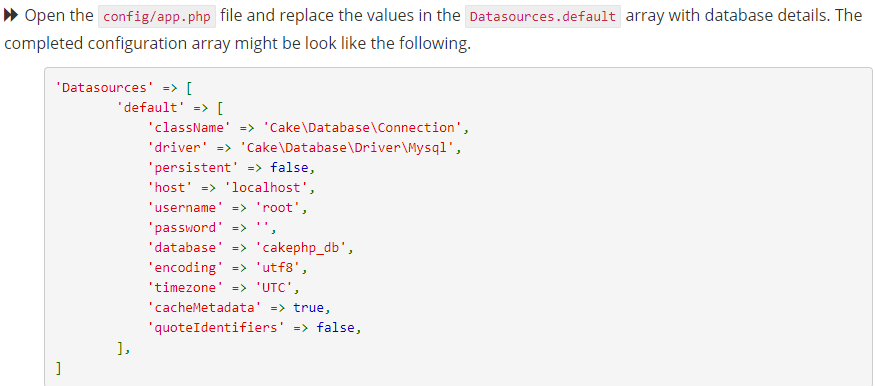
**Cake PHP**

[**Link tài liệu**](https://book.cakephp.org/3.0/en/quickstart.html)

[**Link connect db**](https://www.codexworld.com/cakephp-3-x-tutorial-for-beginners/)

****

**+ Src\Model\name\_model.php**

Đối với model phải tạo model.

<?php

namespace App\Model\Table;

use Cake\ORM\Table;

class FruitsTable extends Table {

}

use Cake\Datasource\ConnectionManager;

use Cake\ORM\TableRegistry;

$this->connection = ConnectionManager::get('default');

private $connection;

private $table;

+ Thêm:

function insertData() {

$this->connection->insert('users',[

'email' => 'admin@gmail.com',

'password' => 'admin',

'phone' => '456',

]);

+ Thêm dưới dạng table:

function tableData() {

$this->autoRender = false;

$tableIns = $this->table->newEntity();

// $tableIns['email'] = 'demo@gmail.com';//C1

// $tableIns['password'] = 'demo';

// $tableIns['phone'] = '987';

$tableIns->email = 'demo1@gmail.com';//C2

$tableIns->password = 'demo1';

$tableIns->phone = '9871';

$this->table->save($tableIns);

}

+ Thêm theo model:

function insertModel() {

$this->autoRender = false;

//cách 1

$fruitObj = $this->Fruits->newEntity();

$fruitObj->name = 'apple';

$fruitObj->description = 'this is best for health';

$this->Fruits->save($fruitObj);

//cách 2:

$fruitObj = $this->Fruits->query();

$fruitObj->insert(['name', 'description'])->values([

'name' => 'Mango',

'description' => 'King of fruits'

])->execute();

}

+ Xóa:

function deleteData() {

$this->connection->delete('users', [

'id' => 4

]);

}

+ Xóa dự liệu table:

function deleteTable() {

$this->autoRender = false;

$data = $this->table->get(9);

$this->table->delete($data);

}

+ Xóa dữ liệu theo model:

function deleteDataModel() {

$this->autoRender = false;

//c1

$data = $this->Fruits->get(3);

if ($this->Fruits->delete($data)) {

echo 'deleted';

}

//c2

$data = $this->Fruits->query();

$data->delete()->where(['id'=>2])->execute();

+ Sửa:

function updateData() {

$this->connection->update('users',[

'email' => 'minhthanh@gmail.com',

'password' => '123456',

'phone' => '987654321',

], [

'id' => 2

]);

}

+ Sửa dữ liệu table:

function updateTable() {

$this->autoRender = false;

$data = $this->table->get(5);

$data->email = 'update@gmail.com';

$data->password = 'updatePass';

if ($this->table->save($data)) {

echo 'Success';

}

}

+ Sửa dữ liệu theo model:

function updateDataModel() {

$this->autoRender = false;

//c1:

$data = $this->Fruits->get(1);

$data->name = "Apple";

if($this->Fruits->save($data)) {

echo 'success';

}

//c2

$data = $this->Fruits->query();

$data->update()->set([

'name'=>'Grapes',

'description'=>'This is updated fruit'

])->where(['id'=>2])->execute();

}

+ Select:

function selectData() {

$this->autoRender = false;

$data = $this->connection->execute('select \* from users limit 1')->fetchAll('assoc');//assoc hiện key, limit 1: chỉ xuất 1 dòng

print\_r($data);

//where có 2 cách dung

…where id = 1 Or …where id = :id”, [“id”=>2]

//fetch and fetchAll: xuất 1 hoặc xuất tất cả

//Cách 2:

$data = $this->connection->newQuery()->select('\*')

->from('users')->order(['id'=>'desc'])->execute()->fetchAll('assoc');

}

+ Select dữ liệu dạng table:

function selectTable() {

$this->autoRender = false;

$data = $this->table->find('all')->select(['email', 'password'])->order(['id'=>'asc'])->toList();

foreach ($data as $key => $value) {

echo $value['email'] . '<br>';

}

$data = $this->table->get(5); //lấy theo id

}

+ Select dữ liệu theo model:

function getdatamodel() {

$this->autoRender = false;

$data = $this->Fruits->find('all',[

'order'=>['id'=>'desc']

])->toArray();

foreach ($data as $key => $value) {

echo $value->name;

}

}

[**Link hướng dẫn**](https://www.youtube.com/watch?v=z71XwXX3nVo&list=PLT9miexWCpPXtHIZmS_Yn1IGpgcEoVqZr&index=7)

**Controller**

* Truyền param từ controller sang view:

C1: $this->set(‘name’, $value);

C2: $this->set(compact(‘name’));

Lưu ý phần name không chứa ký tự $.

* Router truyền param cho function:

Function(param1, param2) {

Echo param1;

}

Router tự truyền tham số vào tên function được gọi

* Function tự nhận param:

$this->request->getParam(‘ten\_bien\_param’)[vi\_tri\_nếu\_cần];

* Lấy thông tin url:

Khai báo package: use Cake\Routing\Router;

$this->url = Router::url(‘/’, true);

Echo $this->url;

**View**

[**Form help trong view**](http://nongdanit.info/php-mysql/cakephp-bai-18-formhelper-trong-view-cua-cakephp.html)

* Tắt yêu cầu view

$this->autoRender = false;

* Gọi tới element con:

Tạo element con ở thư mục: Template->element.

<?= $this->element(‘header’) ?>

Nội dung

<?= $this->element(‘footer’) ?>

public function initialize() {

parent::initialize();

$this->viewBuilder()->setLayout('layout');

}

* Trỏ tới file css, js, img:

<?= $this->Html->css(‘đường dẫn trong webroot’) ?>

<?= $this->Html->script(‘đường dẫn trong webroot’) ?>

<?= base\_url ?> . ‘đường dẫn ảnh trong webroot’

Đưa tất cả thư mục template cần thiết vào webroot.

Đối với ảnh có thể truyền base\_url(phía trên) để dẫn ảnh.

**Model:**

**+** Thư viện

use Cake\ORM\Table;

use Cake\Validation\Validator; // khai báo rang buộc

use Cake\ORM\Rule\IsUnique;

use Cake\ORM\RulesChecker; // tự tạo rule

**+** Code:

function validationDefault(Validator $validator) {

->requirePresence('password', 'create', 'Chưa có password')

->notEmpty('password', 'Chưa nhập password')

->ascii('password', 'Không bỏ dấu')// không cho bỏ dấu

->add('password', [

'length' => [

'rule' => ['lengthbetween', 5, 20], //between

'message' => 'Độ dài mật khẩu từ 5 đến 20 ký tự',

'on' => 'create'

]

])

->requirePresence('email', 'update', 'Chưa có email') khai báo

->notEmpty('email', 'Chưa nhập email')

->add('email',[

'required' => array(

'on' => 'create',

'rule' => 'notBlank',

'message' => 'Enter your email address',

'required' => true,

'last' => true

),

'email' => [

'rule' => 'email',

'message' => 'Sai định dạng email',

'last' => true

],

'emailUnique' => [

'rule' => [$this, 'checkUniqueEmail'], gọi rule tự tạo

'message' => 'Trùng email'

]

])

->requirePresence('phone', 'create', 'Chưa có điện thoại')

->notEmpty('phone', 'Chưa nhập số điện thoại')

->numeric('phone', 'Chỉ được là số', 'create')

->add('phone', 'length', [

'rule' => ['lengthbetween', 8, 10],

'message' => 'Số điện thoại từ 8 đến 10 chữ số'

])

->requirePresence('avatar', 'create', 'Chưa có hình ảnh')

->allowEmpty('avatar', 'Cho phép ko chọn ảnh')

->add('avatar',[

'uploadError' => array(

'rule' => 'uploadError',

'message' => 'The logo upload failed.',

'last' => true,

'allowEmpty' => true

),

'mimeType' => array(

'rule' => array('mimeType', array('image/gif', 'image/png', 'image/jpg', 'image/jpeg')),

'message' => 'Chỉ upload được ảnh',

'allowEmpty' => true

),

'fileSize' =>array (

'rule' => array('fileSize', '<=', '1MB'),

'message' => 'Logo image must be less than 1MB.',

'allowEmpty' => true

),

])

->requirePresence('address', 'create', 'Chưa có địa chỉ')

->notEmpty('address', 'Chưa nhập địa chỉ')

->add('address', [

'length'=>[

'rule' => ['lengthbetween', 10, 150],

'message' => 'Địa chỉ từ 10 đến 150 ký tự'

]

]);

return $validator;

}

+ Rule tự tạo chạy khi gọi ở hàm trên: (gồm controller và table)

public function checkUniqueEmail($data) {

$isUnique = $this->find('all',[

//'fields' => ['id'],//lấy thẻ id

'conditions' => [

'email' => $data,

]

]);

if($isUnique->first() != NULL){

if(isset($this->id)){

if ($this->id == $isUnique->first()['id']) {

return true;

} else {

return false;

}

}else{

return false;

}

}else{

return true;

}

}

Table

$data = json\_decode($user, true); phải chuyển mới có thể check

$this->loadModel('Users'); gọi model để sử dụng

$validation = $this->Users->newEntity($data); bắt đầu check

$validationErr = $validation->getErrors(); lấy các lỗi ra

if (!empty($validationErr) ) {

$this->set(compact('validationErr')); đưa lỗi sang view

}

Lưu ý: Khi check thì check dạng ($data), lưu thì lưu trực tiếp $user

Controller

<?php if (isset($validationErr['name'])) {

foreach ($validationErr['name'] as $key => $value) {

echo "<span class='messageError'>$value</span>";

}

}

?>

View

+ Rule tự tạo chạy khi save or delete:

Lưu ý khi đặt vị trí save or delete để bắt được lỗi

public function buildRules(RulesChecker $rules)

{

$rules->add($rules->isUnique(['email'], 'Email không được trùng'));

return $rules;

}

Table

$err = $user->getErrors();

$this->set('err',$err);

Controller

if (isset($err)) {

foreach ($err['email'] as $key => $value) {

echo "<span class='messageError'>$value</span>";

}

}

View

**Helper:**

# + Breadcrumbs

Hiển thị từng đoạn , sử dụng trong file .ctp

$this->Breadcrumbs->add(

'Product', // Tên hiển thị

['controller'=>'test','action'=>'index'] // đường dẫn

);

echo $this->Breadcrumbs->render(

['class' => 'breadcrumbs-trail'] //show List ở trên ra.

);

$this->Breadcrumbs->insertAt(

3,// vị trí breadcrumbs trong list

'Productss',

['controller' => 'products', 'action' => 'index']

);

$this->Breadcrumbs->insertBefore(

'Productt', //<= thêm vào trước đối tượng tên này

'Productss',

['controller' => 'products', 'action' => 'index']

);

$this->Breadcrumbs->insertAfter(

'Productt', // <= thêm vào sau đối tượng này

'Productss',

['controller' => 'products', 'action' => 'index']

);

Gom nhóm lại các breadcrumbs

$this->Breadcrumbs->add([

['title'=>'Product', 'url'=> ['controller'=>'test', 'action'=>'index']],

['title'=>'Product', 'url'=> ['controller'=>'test', 'action'=>'index']]

]);

Add templates :

$this->Breadcrumbs->setTemplates([

'wrapper' => '<ol class="breadcrumbs">{{content}}</ol>',

'item' => '<li><a href="{{url}}">{{title}}</a></li>'

]);

echo $this->Breadcrumbs->render(['separator' => '<i class="fa fa-angle-right"></i>']);

# + Url:

# // hoạt động như router

<a href="<?= $this->Url->build(['controller'=>'test','action'=>'index']); ?>">Test</a>

//C2: truyền kèm parameter

<a href="<?= $this->Url->build([

"controller" => "Test",

"action" => "index",

"?" => ["foo" => "bar"],

"#" => "first",

]); ?>">Test</a>

# + Flash:

$this->loadComponent('Flash');//gọi trong initialize mới dung dc

$this->Flash->set(‘Nội dung message’, [

'element' => 'success'//show message theo css sẵn có.

]);

**Router:**

Tạo đường dẫn:

Router:connect(‘/ten\_duong\_dan/…/’, [‘controler’ =>’ten\_controller’, ‘action’ => ‘ten\_function’]);

Đường dẫn chứa param:

Router:connect(‘/ten\_duong\_dan/:param1/:param2’, [‘controler’ =>’ten\_controller’, ‘action’ => ‘ten\_function’], [‘ten\_bien\_param’ => [‘param1’, ‘param2’]]);

**Validate model:**

**+** Controller**:**

use Cake\Validation\Validator;

function testDataValidate() {

$this->autoRender = false;

$validator = new Validator;

$req\_data = array(

'name' => 'Hiaaaa',

'email' => 'aaa@gmail.com',

'age' => '8',

'url' => 'google.com',

);

//rules

$validator

->requirePresence('name', 'create', 'name field should be needed')

->add('name', 'length',

[

'rule' => ['minlength', 6],

'message' => ['Nam should min length > 5']

]

)

->requirePresence('email', 'create', 'Email should be needed')

->add('email', 'email' ,

[

'rule' => 'email',

'message' => 'Invalid email'

]

)

->requirePresence('age', 'create', 'Need Age')

->numeric('age', 'Age only number', 'create')

->requirePresence('url', 'create', 'Need Url')

->notEmpty('url', 'Url not empty', 'create')

->url('url', 'Invalid url', 'create');

//->allowEmpty('url');

//->notEmpty('url', 'Url shold be required', 'create');

$validation = $validator->errors($req\_data);

if (!empty($validation)) {

var\_dump( $validation);

} else {

echo 'Passed Successfully';

}

}

+ Model:

B1: Tạo mode (trên db cũng phải có) và validate bên model:

Gom nhóm validate 1 trường:

// Define multiple fields for mixed modes

$validator->requirePresence([

'author\_id' => [

'mode' => 'create',

'message' => 'An author is required.',

],

'published' => [

'mode' => 'update',

'message' => 'The published state is required.',

]

]);

Vd:

<?php

namespace App\Model\Table;

use Cake\ORM\Table;

use Cake\Validation\Validator;

class TestsTable extends Table {

function validationDefault(Validator $validator) {

$validator->requirePresence('email', 'create', 'Email should be needed')

->add('email',[

'unique'=> [

'rule'=>'validateUnique',

'provider' => 'table',

'message' => 'Email should be unique'

]

])

->requirePresence('password', 'create', 'password should be needed')

->requirePresence('comfirm\_password', 'create', 'comfirm\_password should be needed')

->add('comfirm\_password', [

'password\_mismatch'=>[

'rule'=>['compareWith', 'password'],

'message'=>'Password khong trung'

]

]);

return $validator;

}

}

B2: Bên controller

function validateDataByModel() {

$this->autoRender = false;

$req\_data = [

'email'=>'assa',

'password'=>'sa',

'comfirm\_password'=>'sa'

];

$this->loadModel('Tests');//trên database cũng phải có trường này

$validation = $this->Tests->newEntity($req\_data);

$validationError = $validation->errors();

if (!empty($validationError)) {

print\_r($validationError);

} else {

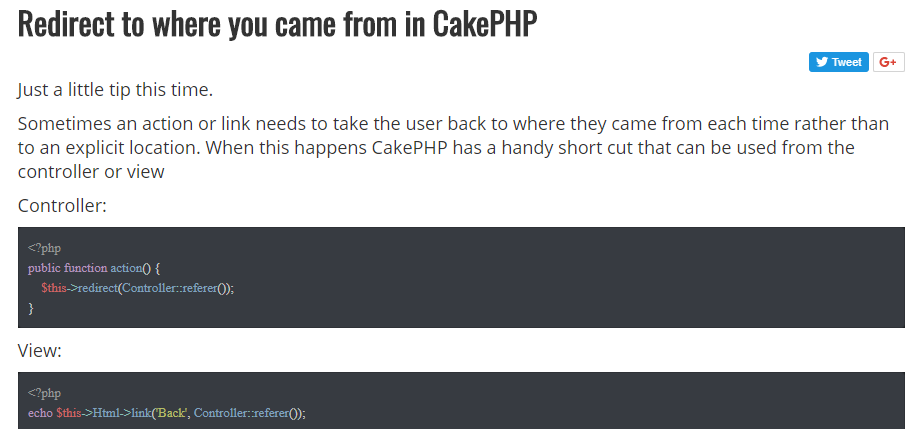
echo 'Success';

}

}

Bảng validate:

|  |  |
| --- | --- |
| lengthBetween('username', [4, 8]); | Độ dài giữa min and max |
|  |  |
|  |  |
|  |  |



Search:

+ controller

Vd1:

$keywork = $this->request->getQuery('Name');

if (!empty($keywork)) {

$this->paginate = [

'conditions' => [

'or' => [

'email LIKE' =>'%' . $keywork . '%',

'name LIKE' =>'%' . $keywork . '%',

'phone LIKE' => '%' . $keywork . '%'

]

]

];

}

VD2:

$email = $this->request->getQuery('Email');

$phone = $this->request->getQuery('Phone');

$address = $this->request->getQuery('Address');

$id = ($this->request->getQuery('Id'));

$sex = $this->request->getQuery('Sex');

$active = $this->request->getQuery('Active');

if (!empty($name) || !empty($email) || !empty($phone) || !empty($address) ||

is\_numeric($active) || is\_numeric($sex) ) {

$this->paginate = [

'conditions' => [

'name LIKE' =>'%' . $name . '%',

'email LIKE' =>'%' . $email . '%',

'phone LIKE' => '%' . $phone . '%',

'address LIKE' => '%' . $address . '%',

]

];

if (is\_numeric($sex)) {

$this->paginate = [

'conditions' => [

'name LIKE' =>'%' . $name . '%',

'email LIKE' =>'%' . $email . '%',

'phone LIKE' => '%' . $phone . '%',

'address LIKE' => '%' . $address . '%',

'sex ' => $sex

]

];

}

}

+ view

<?= $this->Form->create('', ['type' => 'get', 'style' => 'display:flex']); ?>

<?= $this->Form->control('Name',['default' => $this->request->getQuery('Name')])?>

<?= $this->Form->control('Email',['default' => $this->request->getQuery('Email')])?>

<?= $this->Form->control('Phone',['default' => $this->request->getQuery('Phone')])?>

<?= $this->Form->control('Address',['default' => $this->request->getQuery('Address')])?>

<div>

<label>Giới tính</label>

<?= $this->Form->select(

'Sex',

['0' => 'Nữ','1' => 'Nam'],

['empty' => 'Giới tính',

'default' => $this->request->getQuery('Sex')

]

)?>

</div>

<button>Search</button>

<?= $this->Form->end() ?></i>

Phân trang:

+ Controller:

use Cake\Controller\Component\PaginatorComponent;

public $paginate = [

'limit' => '5'

];

public function initialize()

{

parent::initialize();

$this->viewBuilder()->setLayout('adminLayout');

$this->url = Router::url('/', true);

$base\_url = $this->url;

$this->set(compact('base\_url'));

$this->loadComponent('Paginator'); //chạy paginator

}

public function index()

{

$data = $this->Users->find('all');

$this->set('data', $this->paginate($data));

}

+ View

<?php

echo "<div class='center'><ul class='pagination'>";

echo $this->Paginator->prev( ('previous'), array('tag' => 'li', 'currentTag' => 'a', 'currentClass' => 'disabled'), null, array('class' => 'prev disabled'));

echo $this->Paginator->numbers(array('separator' => '','tag' => 'li', 'currentTag' => 'a', 'currentClass' => 'active'));

echo $this->Paginator->next(\_\_('next'), array('tag' => 'li', 'currentTag' => 'a', 'currentClass' => 'disabled'), null, array('class' => 'next disabled'));

echo "</ul></div>";

?>

Thời gian trong Cakephp:

+ Format:

public function time() {

$time = Time::now();

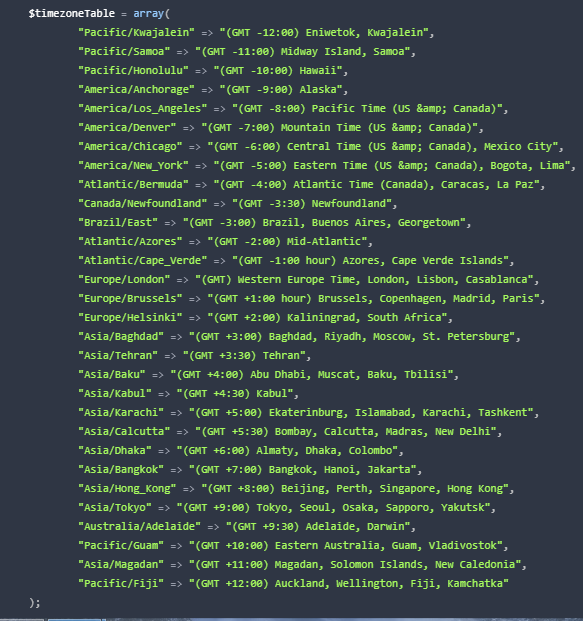
echo ($time->i18nFormat('dd-MM-yyyy HH:mm:ss'));

$this->autoRender = false;

}

+ Time zone: [Link](https://alvarotrigo.com/blog/timezone-user-configuration-at-cakephp/)

vào app để chỉnh lại



Truy vấn trong Cake:

[Link](https://alvinalexander.com/php/cakephp-find-conditions-cakephp-find-all-list-count)

Session trong cakephp:

$session = $this->request->getSession(); //gọi session

$session->write('user', ['name' => $user['name'], //viết session

'avatar' => $user['avatar'], 'id' => $user['id']]);

$session->delete('user'); // xóa session

+Controller

<?php

$session = $this->request->getSession();

if ($session->check('user')) {

echo $session->read('user')['name'];//doc session

}

?>

+View

Mã hóa:

$hashPassword = Security::hash($data['password'], 'md5');

Sort trên view:

<?= $this->Paginator->sort('id', null, ['direction' => 'desc']);?>

Xóa hiển thị popup

<?php echo $this->Html->link(

'Delete',

['controller' => 'Users', 'action' => 'deleteUser', $value['id']],

[

'confirm' => "Bạn có chắc muốn xóa id $value[id]",

'class' =>'btn btn-danger'

]

);

?>