CakePHP

[[Thao tác với model] 2](#_Toc534361612)

[Truy vấn SQL: 2](#_Toc534361613)

[+ Validate data 6](#_Toc534361614)

[Gom nhóm validate 1 trường 9](#_Toc534361615)

[[Thao tác vơi controller] 10](#_Toc534361616)

[[Thao tác với view] 10](#_Toc534361617)

[[Helper - View]: 11](#_Toc534361618)

[+ Breadcrumbs 11](#_Toc534361619)

[Add templates 12](#_Toc534361620)

[+ Url: 12](#_Toc534361621)

[Có icon trong thẻ a: 12](#_Toc534361622)

[+ Flash: 12](#_Toc534361623)

[[Thao tác vơi Router]: 13](#_Toc534361624)

[[Xử lý tìm kiếm] 13](#_Toc534361625)

[[Xử lý phân trang] 15](#_Toc534361626)

[[Xử lý time ] 16](#_Toc534361627)

[[Xử lý session] 17](#_Toc534361628)

[[Xử lý mã hóa] 18](#_Toc534361629)

[[Xử lý sort] 18](#_Toc534361630)

[[Xử lý popup] 18](#_Toc534361631)

[[Migration]: 18](#_Toc534361632)

[[CMD:] 19](#_Toc534361633)

[[Share FB Twitter Xing Google Plus Linkdln] 19](#_Toc534361634)

[[Xử lý radio] 19](#_Toc534361635)

[[Login with facebook] 20](#_Toc534361636)

[[Xử lý file CSV] 20](#_Toc534361637)

[[Xử lý select list] 21](#_Toc534361638)

[[CSRF Token] 21](#_Toc534361639)

[[Xử lý real time với websocket.io] 22](#_Toc534361640)

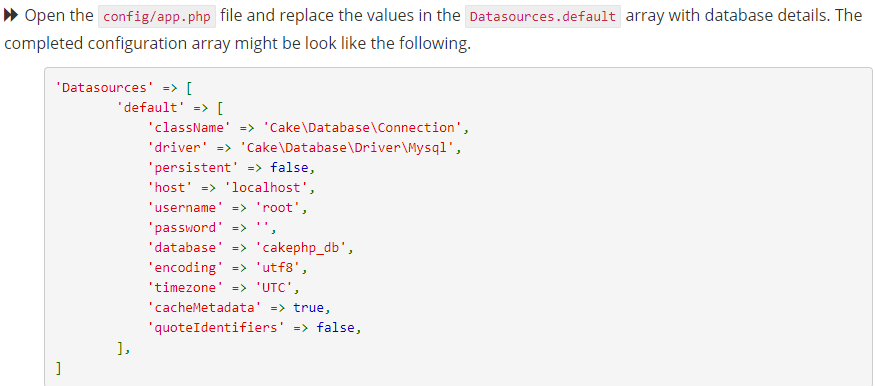
[[Mã hóa password - DefaultPasswordHasher] 22](#_Toc534361641)

[[Chạy batch - SHELL] 22](#_Toc534361642)

**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**

[Thao tác với model]

## Truy vấn SQL:

***Xác định là edit hay create:***

***+Nếu là edit thì gọi đối tượng:***

$bank = $this->WithdrawalInfo->find()->where([ 'userId' => $id, 'flagShowData' => WITHDRAWAL\_INFO\_FLAW\_SHOW\_DATA\_OFF, ])->first();

***+Khi create thì***

if (!$bank) { $bank = $this->WithdrawalInfo->newEntity(); }

***+Thực thi:***

$updateData = $this->WithdrawalInfo->patchEntity($bank, $data); // edit and create

Với $data là dự liệu insert: (tối ưu dòng code)

$data[‘abc’] = value1 //abc là tên column

$data[‘xyz’] = value2

<?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();

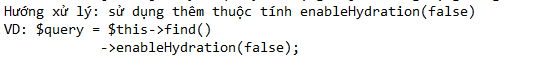
foreach ($data as $key => $value) {

echo $value->name;

}

}

## Chuyển query từ dạng object sang dạng mảng



## + Validate data

**+** 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

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.',

]

]);

Truy vấn trong Cake:

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

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

[Thao tác vơi 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;

[Thao tác với 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’

<?= $this->Html->meta ?> dẫn tởi thư mục webroot

Đưa tất cả thư mục template cần thiết vào webroot.

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

[Helper - View]:

# + 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>',

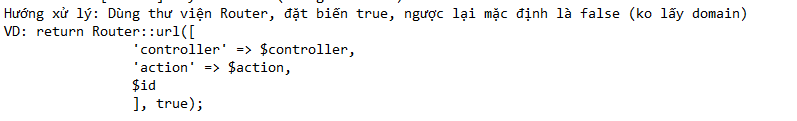
'item' => '<li><a href="{{url}}">{{title}}</a></li>'

]);

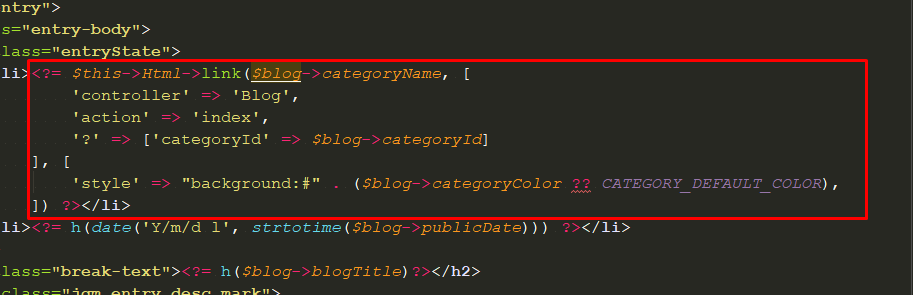
echo $this->Breadcrumbs->render(['separator' => '<i class="fa fa-angle-right"></i>']);

# + Url: (link)

# Lấy full url (bao gồm domain)

****

# Chỉnh style cho link

****

# Get data từ URL

****

# Có icon trong thẻ a:

<?= $this->Html->link('<i class="icon icon-nav-home"></i>ホーム',

                    [

                        'controller' => 'web',

                        'action' => 'index'

                    ],

                    [

                        'class' => 'global-nav-link',

                        'escape' => false

                    ]) ?>

//new

<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.

]);

[Thao tác vơi Router]:

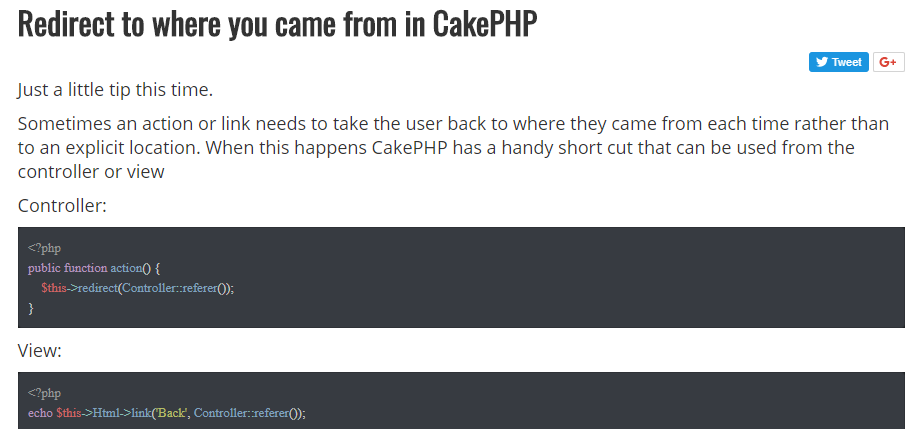
Tạo đường dẫn:

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

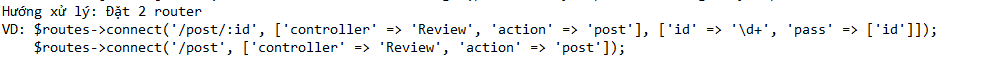
Đường dẫn chứa param:

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

+ Trở lại trang trước đó:



+ Khi đặt tên cho url thì có 2 trường hợp là có truyền param và không truyền param



[Xử lý tìm kiếm]

+ 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>

[Xử lý phân trang]

+ Có template:

$this->Paginator->setTemplates([

'prevActive' => '<li class="pagenation-item">

<a href={{url}}><div class="pagenation-btn pagenation-prev">{{text}}</div></a></li>',

'prevDisabled' => '<li class="pagenation-item prev disable">

<a><div class="pagenation-btn pagenation-prev btn-disable" href="" onclick="return false;">{{text}}</div></a></li>',

'nextActive' => '<li class="pagenation-item">

<a href={{url}}><div class="pagenation-btn pagenation-next">{{text}}</div></a></li>',

'nextDisabled' => '<li class="pagenation-item next disable">

<a class="pagenation-btn pagenation-prev btn-disable" href="" onclick="return false;">{{text}}</div></a></li>',

'number' => '<li class="pagenation-item">

<a href={{url}} ><div class="pagenation-btn">{{text}}</div></a></li>',

'current' => '<li class="pagenation-item">

<a href={{url}}><div class="pagenation-btn is-active">{{text}}</div></a></li>',

]);

?>

<ul class='pagenation'>

<?php

$pageCount = $this->paginator->params()['pageCount'];

$currentPage = $this->paginator->params()['page'];

if ($pageCount > 1) {

echo $currentPage === 1 ? '' : $this->Paginator->prev('Prev');

echo $this->Paginator->numbers();

echo $currentPage === $pageCount ? '' : $this->Paginator->next('Next');

}

?>

</ul>

+ Controller:

use Cake\Controller\Component\PaginatorComponent;

public $ = [

'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>";

?>

[Xử lý time ]

+ Format:

public function time() {

$time = Time::now();

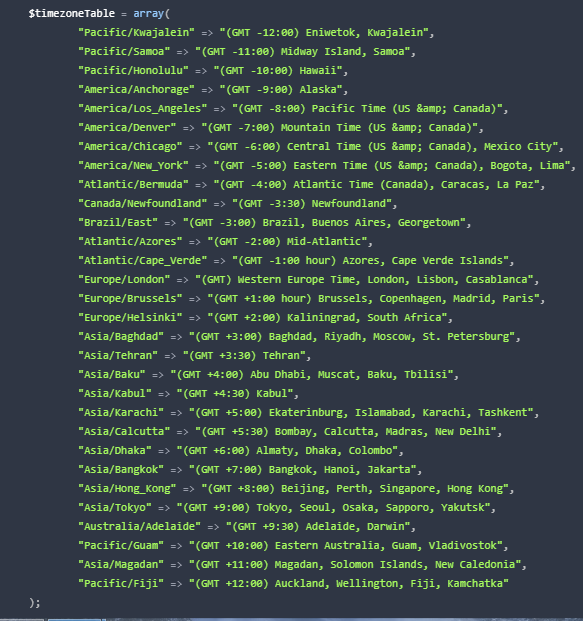
echo ($time->i18nFormat('dd-MM-yyyy HH:mm:ss'));

$this->autoRender = false;

}

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

vào app để chỉnh lại



[Xử lý session]

$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

}

?>

[Xử lý mã hóa]

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

[Xử lý sort]

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

[Xử lý 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'

]

);

?>

[Seeds]:

+ Tạo file seeds

bin/cake bake seed name\_table

+ Chạy file seeds

bin/cake migrations seed

[Shell]: (Batch)



Chạy file shell:

Bin/cake ten\_file\_shell (Vd: bin/cake demo)

[Migration]:

Lưu ý: ***Khi migration đã chạy qua file migrate đó rồi thì sẽ không chạy lại file đó nữa, muốn chạy file đó phải rollback lại. Các lệnh rollback sau:***

+ Rollback lại hết tất cả

bin/cake migrations rollback

+ Rollback theo 1 file nhất định

bin/cake migrations rollback -t 20150103081132

Thêm column trong table:

-Bin/cake bake migration ChatOnlineV5:

$this->table('chatOnline')

->addColumn('bikeId', 'biginteger', ['after' => 'receivedUserId'])

->update();

Change column:

-Bin/cake bake migration TenFileMigrate:

$this->table('mailHistory')

->changeColumn('userId', 'text', ['null' => true, 'default' => NULL])

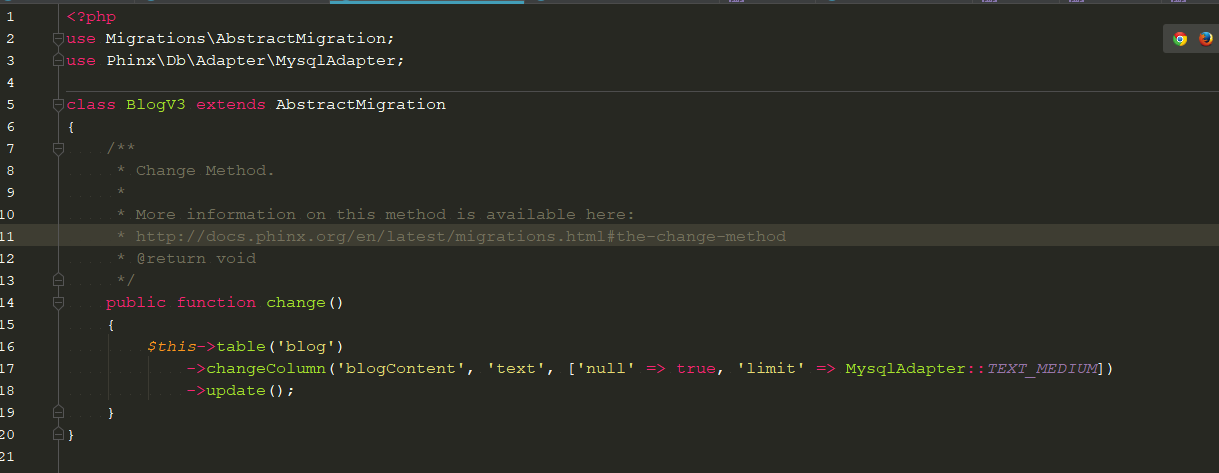
->update();

-Chạy file migration:

+ [All] bin/cake migrations migrate

+ [Phiên bản] bin/cake migrations migrate –t mã\_phiên\_bản

## + Medium Text



[CMD:]

(dung trong thư mục bin)

+ [DB] bin/cake bake model --no-fixture --no-test --force TjCouponSegment]

+ [Controller] cake bake controller –prefix admin user --no-actions --no-test

Với admin là folder con, user là tên controller

+ [view] cake bake template --prefix admin users

[Share FB Twitter Xing Google Plus Linkdln]

[Link code tham khảo](https://codepen.io/patrickkahl/pen/DxmfG)

[Link hướng dẫn](https://css-tricks.com/simple-social-sharing-links/)

[Xử lý radio]

+ Có template:

<?php

$templates = [

'inputContainer' => '{{content}}', // hủy div do form input tạo ra

'nestingLabel' => ' // cấu trúc label

<li class="form-radio-item">

<label class="form-radio">

{{input}} // gọi tới input

<span class="form-radio-label">{{text}}</span>

</label>

</li>',

'radio' => '<input type="radio" name="availableType" value="{{value}}" checked = "true">' // cấu trúc radio

];

$options = [

['value' => '1', 'text' => \_\_('利用可能')],

['value' => '0', 'text' => \_\_('利用できない')]

];

?>

<?=

$this->Form->input('type', [

'type' => 'radio',

'options' => $options,

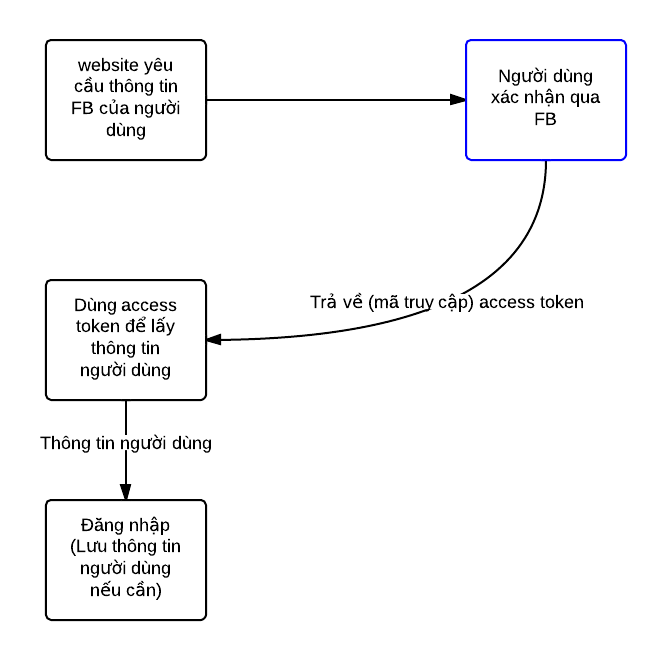
'templates' => $templates,

'label' => false

]);

[Login with facebook]

composer require facebook/graph-sdk



[Xử lý file CSV]

header('Content-Type: text/csv');

header('Content-Disposition: attachment;filename=' . $fileName);//tên file

$fp = fopen('php://output', 'w');

fputcsv($fp, $header); // nội dung đầu file

foreach ($dataArr as $key => $data) {// xử lý data (nếu cần)

if (isset($data['createdDate'])) {

$data['createdDate'] = $data['createdDate']->i18nFormat('yyyy-MM-dd HH:mm:ss');

}

if (isset($data['borrowDate'])) {

$data['borrowDate'] = $data['borrowDate']->i18nFormat('yyyy-MM-dd HH:mm:ss');

}

if (isset($data['returnDate'])) {

$data['returnDate'] = $data['returnDate']->i18nFormat('yyyy-MM-dd HH:mm:ss');

}

fputcsv($fp, $data);// lưu data vào csv

}

fclose($fp);

exit;

[Xử lý select list]

$this->Form->select('title',

[

'key' => \_\_('選択してください'),

\_\_('登録について'),

\_\_('料金について'),

\_\_('シェア取引について'),

\_\_('事故・トラブルについて'),

\_\_('補償・ロードサービスについて'),

\_\_('退会・お客様情報について'),

\_\_('売上金について'),

\_\_('その他のお問い合わせ'),

\_\_('サービスへのご意見・ご要望')

],

[

'default' => isset($\_POST['back']) ? $data['title'] : ['key'],

'disabled' => ['key'],

'class' => 'form-select'

]

[CSRF Token]

$this->loadComponent('Csrf'); // khai báo component

$token = $this->request->getParam('\_csrfToken'); // lấy giá trị của token

$this->getEventManager()->off($this->Csrf); // tắt token khi cần

{

data: JSON.stringify(data),

\_csrfToken: $('#csrfToken').val() // khi sử dụng ajax post, truyền kèm theo

}

[Xử lý real time với websocket.io]

composer require cboden/ratchet //cài đặt ratchet

composer require react/zmq

[Mã hóa password - DefaultPasswordHasher]

+ check(string $password , string $hashedPassword)

Tạo hàm băm cho password và so sánh với password đã được mã hóa (login)

+ hash(string $password)

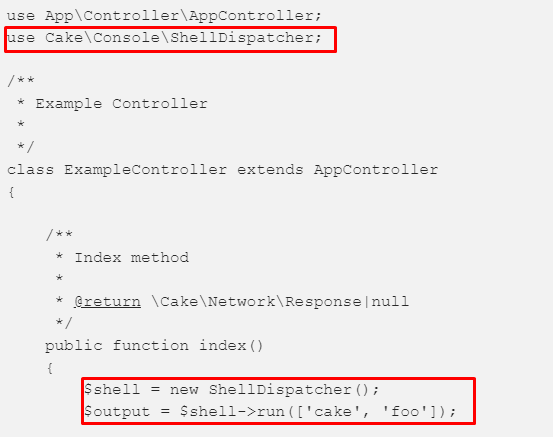
băm password

+ needsRehash().

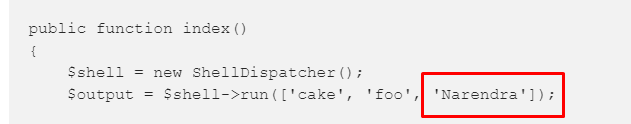
[Chạy batch - SHELL]

[Link hướng dẫn sử dụng shell](https://medium.com/@narendravaghela/cakephp-execute-shell-command-from-controller-90c8323558a)

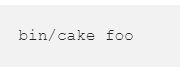
+ Chạy shell từ controller



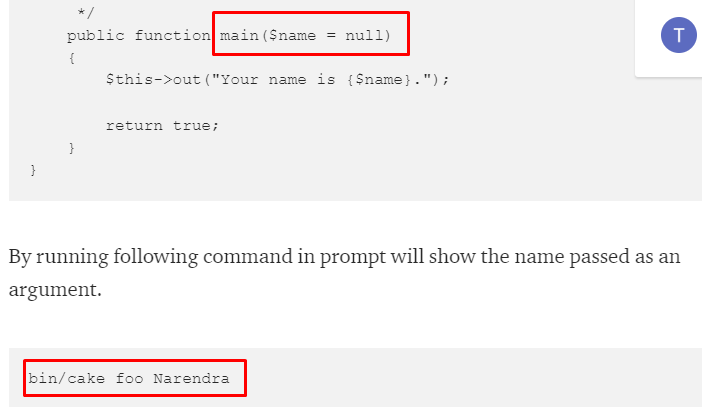
+ Chạy shell từ controller có kèm param



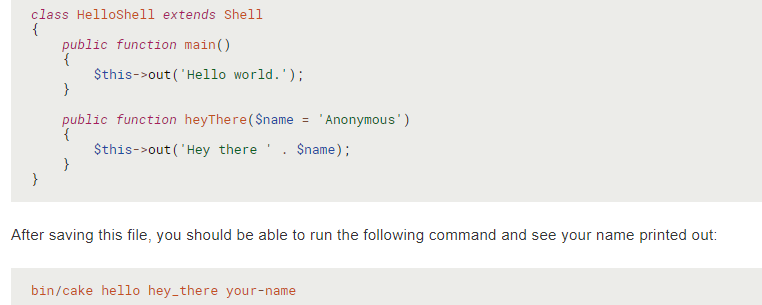
+ Chạy shell từ command line



+ Chạy shell từ command line kèm theo param



Chạy shell theo tên hàm mình đặt và truyền param



[Auth]

+ Sử dụng trên view, check login

C1: $this->request->session()->read('Auth.User.username');

C2: khai báo biến trên app controller, truyền sang view

$this->isLogin = ($this->Auth->user) ?? null;

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

+ Sử dụng trên controller

$this->Auth->user() //check login