**PHỤ LỤC**

[I. Cấu trúc, mô hình hoạt động của Vportal 3](#_Toc370830056)

[1. Mô hình tổng quan Vportal 3](#_Toc370830057)

[2. Mô hình hoạt động của vPortalCore 5](#_Toc370830058)

[3. Ưu nhược điểm của vPortal 6](#_Toc370830059)

[II. Hàm trong VportalCore 6](#_Toc370830060)

[1. vPortal\_Authenticate 6](#_Toc370830061)

[1.1. isInternalUser 6](#_Toc370830062)

[2. vPortal\_Cache 6](#_Toc370830063)

[2.1. getInstance 6](#_Toc370830064)

[2.2. save 7](#_Toc370830065)

[2.3. load 7](#_Toc370830066)

[2.4. remove 7](#_Toc370830067)

[3. vPortal\_Database 7](#_Toc370830068)

[3.1. getInstance 7](#_Toc370830069)

[3.2. closeConnection 8](#_Toc370830070)

[4. vPortal\_CoreData 8](#_Toc370830071)

[4.1. \_getSectionByPath 8](#_Toc370830072)

[4.2. \_checkSectionExist 8](#_Toc370830073)

[4.3. \_validateFeatureByFeatureCode 8](#_Toc370830074)

[4.4. \_getValidPosition 8](#_Toc370830075)

[4.5. \_getBlockInstanceBySection 9](#_Toc370830076)

[4.6. \_getBlockInstanceConfig 9](#_Toc370830077)

[4.7. \_getTemplate 9](#_Toc370830078)

[5. vPortal\_Hook 10](#_Toc370830079)

[5.1. fetchHook 10](#_Toc370830080)

[6. vPortal\_Language 10](#_Toc370830081)

[6.1. setLang 10](#_Toc370830082)

[6.2. getLang 10](#_Toc370830083)

[6.3. translate 10](#_Toc370830084)

[7. vPortal\_Session 10](#_Toc370830085)

[7.1. set 10](#_Toc370830086)

[7.2. get 11](#_Toc370830087)

[7.3. remove 11](#_Toc370830088)

[7.4. destroy 11](#_Toc370830089)

[7.5. getCookie 11](#_Toc370830090)

[7.6. setCookie 11](#_Toc370830091)

[7.7. removeCookie 12](#_Toc370830092)

[8. vPortal\_Utility 12](#_Toc370830093)

[8.1. convertPath 12](#_Toc370830094)

[8.2. downloadFile 12](#_Toc370830095)

[8.3. dump 12](#_Toc370830096)

[8.4. getClientIp 12](#_Toc370830097)

[8.5. getConfig 13](#_Toc370830098)

[8.6. setConfig 13](#_Toc370830099)

[8.7. getCurrDate 13](#_Toc370830100)

[8.8. getCurrTime 13](#_Toc370830101)

[8.9. loadModel 13](#_Toc370830102)

[8.10. loadHelper 14](#_Toc370830103)

[8.11. loadView 14](#_Toc370830104)

[8.12. \_compile 14](#_Toc370830105)

[8.13. pagingInfo 15](#_Toc370830106)

[8.14. subString 15](#_Toc370830107)

[8.15. writeFile 15](#_Toc370830108)

[9. vPortal\_Display 15](#_Toc370830109)

[9.1. redirect 15](#_Toc370830110)

[9.2. doOutput 16](#_Toc370830111)

[III. Phát triển tính năng mới trong vPortalCore. 16](#_Toc370830112)

[1. Thêm thư viện mới 16](#_Toc370830113)

[2. Thêm biến và Replace tag trong template 16](#_Toc370830114)

[3. Thêm cấu trúc biên dịch template 16](#_Toc370830115)

[IV. Import thư viện VportalCore vào website. 17](#_Toc370830116)

[V. Database sử dụng trong vPortalCore 17](#_Toc370830117)

**vPortalCore**

1. Cấu trúc, mô hình hoạt động của Vportal
   1. Mô hình tổng quan Vportal

**Site 1**S

**Site 2**S

**…**

vPortal Aplication 1

vPortal Aplication 2

…

**vPortal**

**Library**

**Zend**

**Database**

**Memcache**

**Memcache Queue**

**Cpanel**

**Worker**

Access Data

Write Log

Access Data & Write Cache

Modify memcache data

Input data

Get info from Queue

Write Info (Clear cache)

Input data

* + - * Library: Hệ thống các lớp, các hàm được sử dụng để xử lý các chức năng trên Vportal và hệ thống thư viện này dựa trên nền tàng của Zend Framework
      * Memcache Queue: Hệ thống lưu trữ thời gian tạo memcache, thời gian clear memcache.
      * Memcache: Nhằm giảm tải lượng connect trực tiếp vào DB hệ thống vPortal sử dụng Memcache để lưu các kết quả query ở các bước trước đó hoặc kết quả trả về của một url.
      * Database: Mỗi website được xây dựng bằng Vportal sẽ sử dụng 2 DB chính là portal và data
      * DB portal: lưu trữ cấu hình, thông tin các module, block, … trên một website
      * DB data: lưu trữ nội dung ( tin tức, sự kiện, bài viết…) trên website
      * Cpanel: Hệ thống quản trị nội dung website, khi dữ liệu được cập nhật trên Cpanel thì đồng thời sẽ cập nhật thời gian xóa cache vào Memcache Queue
      * Worker: Hệ thống cronjob chạy tự đồng để update lại memcache khi có sự thay đổi dữ liệu từ Cpanel. Worker sẽ dựa vào thông tin từ memcache queue để tìm các cache sẽ được update lại dữ liệu.
  1. Mô hình hoạt động của vPortalCore

**Initialize.php**



**- Parse URL.**

**- Check downsite**

**- Load section & validate.**

**- Load section template master**

**- Load module.**

**- Load block instances.**

**Hiển thị content lên web browser**

**- Close all db connections.**

**- Close all MQ connections.**

**- Close all Memcache connections.**

**bootstrap.php**

* 1. Ưu nhược điểm của vPortal
     1. Ưu điểm
        + Nhanh chóng, dễ dàng triển khai một website
        + Hệ thống được tất cả các website đang được vận hành tại cùng một nơi
        + Chịu tải được lượng request lớn.
     2. Nhược điểm
        + Hệ thống source code khá phức tạp, rất khó cho việc tiếp cận ban đầu
        + Nhiều hệ thống db khác nhau, rất khó quản lý và tốn kém cơ sở hạ tầng.
        + Cơ chế lưu trữ cache phức tạp và rắc rối
        + Chưa có cơ chế tự động zip code css, js

1. Hàm trong VportalCore
   1. vPortal\_Authenticate
   2. isInternalUser

* Cú pháp: <boolean> vPortal\_Authenticate::isInternalUser(<string>$ip=’’)
* Chức năng: check xem IP user có thuộc dãy IP cho phép xem debug hay không
* Trả về: true nếu hợp lệ, false: không hợp lệ.
* Tham số:
  + $ip dùng để check ip của user với $ip xem có hợp lệ hay không?
  1. vPortal\_Cache

1. 1. getInstance

* Cú pháp: <mixed> vPortal\_Cache::getInstance(<string>$key,<array> $config, <boolean>$isStore=true).
* Chức năng: Khởi tạo cache instance.
* Trả về: cache object nếu khởi tạo thành công, false nếu khởi tạo không thành công.
* Tham số:
  + $key: giá trị xác định instance cần khởi tạo là duy nhất.
  + $config: thông tin config cache. Các thông tin base trên Zend Cache. Config cache có thể đặt bất kỳ chỗ nào chỉ cần load thông tin lên và truyền vào method.
  + $isStore: cho phép method có lưu trong Vportal\_Cache hay không. Nếu muốn khởi tạo không cần lưu lại thì set =false, mặc định là sẽ lưu.
  1. save
* Cú pháp: vPortal\_Cache::save($value, $key, $tags = array(), $specificLifetime = false)
* Chức năng: Lưu giá trị của cache
* Trả về: false nếu lưu không thành công, ngược lại thì trả về object cache
* Tham số:
  + $value: giá trị của cache
  + $key: key của cache
  + $tag: mãng tag key
  + $specificLifetime: thời gian tồn tại
  1. load
* Cú pháp: vPortal\_Cache::load($key)
* Chức năng: Loa giá trị của cache
* Trả về: false nếu load không thành công, ngược lại thì trả về object cache
* Tham số:
  + $key: key của cache
  1. remove
* Cú pháp: vPortal\_Cache:: remove ($key)
* Chức năng: xóa giá trị của cache
* Trả về: false nếu xóa không thành công, ngược lại thì trả về true
* Tham số:
  + $key: key của cache
  1. vPortal\_Database

Các phương thức xử lý DB trong vPortalCorel sử dụng thư viện của Zend Framework. Các hàm bên dưới chỉ để khởi tạo đối tượng DB trong Zend.

<http://framework.zend.com/manual/1.12/en/zend.db.html>

1. 1. getInstance

* Cú pháp: <Zend\_Db Object> vPortal\_Database::getInstance(<string>$name)
* Chức năng: Khởi tạo db instance.
* Trả về: zend db object nếu khởi tạo thành công, ghi log và redirect sang trang thông báo lỗi khi không thành công.
* Tham số:
  + $name: tên key config. Key này là key của array config db trong application/config/database.php.
  1. closeConnection
* Cú pháp: vPortal\_Database:: closeConnection();
* Chức năng: Đóng các kết nối Database
* Trả về: void()
  1. vPortal\_CoreData

Class này lấy thông tin cấu hình của toàn bộ site từ database vProtal.

* 1. \_getSectionByPath
* Cú pháp: <array>vPortal\_CoreData::\_ getSectionByPath(<string>$sectionPath)
* Chức năng: Lấy thông tin tin của một session từ database
* Trả về: Array thông tin của section nếu section tồn tại, false nếu không tồn tại
* Tham số:
  + $sectionPath: tên của section
  1. \_checkSectionExist
* Cú pháp: <boolean>vPortal\_CoreData::\_ checkSectionExist (<string>$sectionPath)
* Chức năng: Kiểm tra section có tồn tại.
* Trả về: true nếu tồn tại, false nếu không
* Tham số:
  + $sectionPath: tên của section
  1. \_validateFeatureByFeatureCode
* Cú pháp: <array>vPortal\_CoreData::\_ validateFeatureByFeatureCode (<string>$ featureCodes, <string>$version)
* Chức năng: Lấy thông tin của module
* Trả về: Array thông tin module nếu có, ngược lại trả về false
* Tham số:
  + $featureCodes: Mã của feature php
  + $version: Phiên bản vPortal
  1. \_getValidPosition
* Cú pháp: <array>vPortal\_CoreData::\_ getValidPosition (<int>$ templateMasterId, <string>$ tplMasterName)
* Chức năng: Lấy thông tin các vị trí trên template master của
* Trả về: Array
* Tham số:
  + $ templateMasterId: Id của template master
  + $ tplMasterName: tên của template master
  1. \_getBlockInstanceBySection
* Cú pháp: <array>vPortal\_CoreData::\_ getBlockInstanceBySection(<int >$sectionId, <array>$positions, <string>$sectionPath, <string>$tplMasterName = '')
* Chức năng: Lấy thông tin của các block trên một section
* Trả về: Array
* Tham số:
  + $ sectionId: Id của section
  + $positions: mảng các possitions trên một section
  + $sectionPath: Tên của section cần lấy block
  + $ tplMasterName: tên của template master
  1. \_getBlockInstanceConfig
* Cú pháp: <array>vPortal\_CoreData::\_ getBlockInstanceBySection(<int>$blockInstanceId, <int>$sectionId='', <string>$blockName='', <string>$positionName='')
* Chức năng: Lấy thông tin cấu hình của block trên một section
* Trả về: Array nếu có cấu hình, false nếu k có
* Tham số:
  + $blockInstanceId: Id của block tại một possition
  + $sectionId: Mã của một section
  + $blockName: Tên của một block
  + $positionName: Tên của vị trí block
  1. \_getTemplate
* Cú pháp: <array>vPortal\_CoreData::\_getTemplate(<string>$featureCode, <string>$templateName)
* Chức năng: Lấy thông tin của template
* Trả về: Array nếu lấy được nội dung, false nếu k có
* Tham số:
  + $featureCode: Mã của feature
  + $templateName: Tên của template
  1. vPortal\_Hook

1. 1. fetchHook

* Cú pháp: <void>vPortal\_Hook::fetchHook(<string>$feature,<string>$hookName).
* Chức năng: Thực hiện load các plugin dùng tại hookName ở feature nào đó.
* Trả về: void.
* Tham số:
  + $feature: Feature code của 1 feature nào đó.
  + $hookName: tên hook cần fetch.
  1. vPortal\_Language

1. 1. setLang

* Cú pháp: <void>vPortal\_Language::setLang(<string>$lang).
* Chức năng: Thực hiện set language cho site.
* Trả về: void.
* Tham số:
  + $lang: language code cần set.
  1. getLang
* Cú pháp: <string>vPortal\_Language::getLang().
* Chức năng: lấy giá trị language code hiện tại.
* Trả về: string.
  1. translate
* Cú pháp: <string>vPortal\_Language::translate(<string>$featureCode, <string>$text, <string>$lang).
* Chức năng: dịch đoạn text sang ngôn ngữ truyền vào param $lang.
* Trả về: string. Trả về chuỗi với ngôn ngữ tương ứng, nếu không có thì trả về chuỗi gốc.
* Tham số:
  + $featureCode: feature code cần translate.
  + $text: giá trị text cần dịch.
  + $lang: language code cần dịch.
  1. vPortal\_Session

1. 1. set

* Cú pháp: <void>vPortal\_Session::set(<string>$key,<mixed>$value).
* Chức năng: set giá trị cho session.
* Trả về: void.
* Tham số:
  + $key: key của session.
  + $value: giá trị của session ứng với key.
  1. get
* Cú pháp: <mixed>vPortal\_Session::get(<string>$key).
* Chức năng: get giá trị cho session key nào đó.
* Trả về: mixed. Nếu key của session không tồn tại thì trả về null.
* Tham số:
  + $key: key của session.
  1. remove
* Cú pháp: <void>vPortal\_Session::remove(<string>$key).
* Chức năng: remove session với key nào đó.
* Trả về: void.
* Tham số:
  + $key: key của session.
  1. destroy
* Cú pháp: <void>vPortal\_Session::get().
* Chức năng: destroy tất cả session.
* Trả về: void.
  1. getCookie
* Cú pháp: <mixed>vPortal\_Session::getCookie(<string>$key).
* Chức năng: get giá trị cookie với key nào đó.
* Trả về: mixed. Nếu key của cookie không tồn tại thì trả về null.
* Tham số:
  + $key: key của cookie.
  1. setCookie
* Cú pháp: <void>vPortal\_Session::setCookie(<string>$key, <mixed>$value).
* Chức năng: set giá trị cho session key nào đó.
* Trả về: mixed. Nếu key của session không tồn tại thì trả về null.
* Tham số:
  + $key: key của cookie.
  1. removeCookie
* Cú pháp: <void>vPortal\_Session::removeCookie(<string>$key).
* Chức năng: remove cookie với key nào đó.
* Trả về: coid.
* Tham số:
  + $key: key của cookie.
  1. vPortal\_Utility

1. 1. convertPath

* Cú pháp: <string>vPortal\_Utility::convertPath(<string>$path).
* Chức năng: chuyển đường dẫn về đúng dấu phân cách của HĐH.
* Trả về: string. Trả về đường dẫn với path seperator của HĐH.
* Tham số:
  + $path: đường dẫn cần thực hiện.
  1. downloadFile
* Cú pháp: <integer>vPortal\_Utility::downloadFile(<string>$filePath).
* Chức năng: thực hiện download file với file path truyền vào.
* Trả về: integer. 0: thành công, 1: File not found, 2: Extension of file is invalided.
* Tham số:
  + $filePath:đường dẫn đến file cần download.
  1. dump
* Cú pháp: <void>vPortal\_Utility::dump(<mixed>$variable, <string>$ip=’’, <boolean>$exit=false).
* Chức năng: thực hiện dump giá trị của 1 biến nào đó với những IP nào được config trong application/config/debug.php hoặc là ip truyền trực tiếp vào.
* Trả về: void.
* Tham số:
  + $variable: Biến cần dump.
  + $ip: ip xác định việc dump variable theo ip đó.
  + $exit: xác định việc dump có exit hay không.
  1. getClientIp
* Cú pháp: <string>vPortal\_Utility::getclientIp().
* Chức năng: lấy IP của máy client.
* Trả về: string.
  1. getConfig
* Cú pháp: <mixed>vPortal\_Utility::getConfig(<string>$name, <string>$key=’’).
* Chức năng: lấy thông tin config trong application/config/\*.
* Trả về: mixed. Nếu tồn tại file config với tên là $name thì sẽ load thông tin của file đó lên. Nếu có key thì sẽ trả về thông tin config với $key truyền vào. Nếu không load đuợc config sẽ return false.
* Tham số:
  + $name: tên của file config (bỏ đi .php).
  + $key: key trong config file config cần lấy.
  1. setConfig
* Cú pháp: <boolean>vPortal\_Utility::setConfig(<string>$name, <string>$key, <mixed>$value).
* Chức năng: set giá trị cho config.
* Trả về: boolean. Trả về true nếu set thành công, ngược lại trả về false.
* Tham số:
  + $name: tên của file config (bỏ đi .php).
  + $key: key trong config file config cần lấy.
  + $value: giá trị của config.
  1. getCurrDate
* Cú pháp: <string>vPortal\_Utility::getCurrDate(<string>$format).
* Chức năng: trả về ngày hiện tại theo format giống như hàm date.
* Trả về: mixed. Nếu key của session không tồn tại thì trả về null.
* Tham số:
  + $format: định dạng cần trả ra theo dạng nào.
  1. getCurrTime
* Cú pháp: <integer>vPortal\_Utility::getCurTime().
* Chức năng: trả về thời gian hiện tại.
* Trả về: integer.
  1. loadModel
* Cú pháp: <mixed>vPortal\_Utility::loadModel(<string>$featureCode, <string>$className).
* Chức năng: trả về instance của 1 model, được lưu theo dạng singleton.
* Trả về: mixed. Trả về false nếu không load model được, trả về instance nếu load model được.
* Tham số:
  + $featureCode: Feature cần load.
  + $className: tên class cần load.
  1. loadHelper
* Cú pháp: <mixed>vPortal\_Utility::loadHelper(<string>$featureCode, <string>$className).
* Chức năng: Load helper cho một module
* Trả về: mixed - Return instance of a class if load class successfull else return false
* Tham số:
  + $featureCode: Feature cần load.
  + $className: tên class cần load.
* Ví dụ:

*$news = vPortal\_Utility::loadHelper('news','Application\_News\_Helper\_News'),* file helper sẽ được lưu: application/feature/module/helper/news.php

* 1. loadView
* Cú pháp: <string>vPortal\_Utility::loadView(<string>$featureCode, <string>$view, <array>$data, <array>$config).
* Chức năng: trả về nội dung của view.
* Trả về: string. Trả về nội dung của view nếu load ok, trả về rỗng nếu không load được view, trả về mã lỗi nếu có lỗi xảy ra trong quá trình load view.
* Tham số:
  + $featureCode: Feature cần load.
  + $view: Tên view của feature cần load.
  + $data: dữ liệu cần out ra view.
  + $config: các config cần out ra view.
  1. \_compile
* Cú pháp: <string> vPortal\_Utility::\_compile(<string>$featureCode, <string>$view, <string>$template)
* Chức năng: biên dịch template thành nội dung PHP và hiển thị lên view
* Trả về: string. Trả về nội dung PHP của view nếu load thành công, ngược lại thì trả về false
* Tham số:
  + $featureCode: Feature cần load
  + $view: tên view của feature cần load
  + $template: Tên template của view cần load
  1. pagingInfo
* Cú pháp: <array>vPortal\_Utility::pagingInfo(<integer>$rowTotal, <integer>$currentPage, <integer>$itemPerPage, <integer>$pageNumLimit).
* Chức năng: trả về các giá trị cần thiết cho việc phân trang.
* Trả về: array.
* Tham số:
  + $rowTotal: tổng số record.
  + $currentPage: trang hiện tại.
  + $itemPerpage: số record trên 1 trang.
  + $pageNumLimit: số trang trên 1 lần hiển thị.
  1. subString
* Cú pháp: <string>vPortal\_Utility::subString(<string>$content, <integer>$num).
* Chức năng: cắt chuỗi content với số ký tự nào đó.
* Trả về: string.
* Tham số:
  + $content: nội dung chuỗi cần cắt.
  + $num: số lượng ký tự cần cắt.
  1. writeFile
* Cú pháp: <boolean>vPortal\_Utility::writeFile(<string>$content, <string>$filePath, <string>$type=’w’).
* Chức năng: ghi vào file với nội dung nào đó.
* Trả về: boolean. Trả về true nếu write file thành công, false nếu write file không thành công.
* Tham số:
  + $content: nội dung cần write file.
  + $filePath: đường dẫn cần write.
  + $type: ghi theo loại nào. Mặc định là w.
  1. vPortal\_Display
  2. redirect
     + - Cú pháp: vPortal\_Display:: redirect(<string>$url,<bool> $direct301 = false)
       - Chức năng: di chuyển tới một url khác
       - Trả về: void
       - Tham số:

$url: Link sẽ redirect tới

$direct301: có phải redirect 301, dành cho SEO

* 1. doOutput
     + - Cú pháp: vPortal\_Display:: doOutput()
       - Chức năng: Biên dịch content cho vPortal

- Get all parse content

- Replace UI constant.

- Insert seo content (title, keyword, description)Trả về: string

* + - * Tham số:

1. Phát triển tính năng mới trong vPortalCore.
   1. Thêm thư viện mới

vPortalCore là tập hợp các class, function xử lý một chức năng nào đó và được sử dụng chung cho nhiều site. Để phát triển thêm một chức năng trong vPortalCore chỉ cần tạo file class trong thư mục: corereal/library/vPortal

Ví dụ: cần tạo một class xử lý về String

* File: corereal/library/vPortal/String.php
* Class name: class vPortal\_String
  1. Thêm biến và Replace tag trong template

Để thêm các biến cố định sử dụng cho toàn bộ template ta có thể thêm trong file application/config/ui.php của site

Ví du: return array(‘SITE\_URL’ => ‘application|siteUrl’) biến SITE\_URL sẽ được lấy giá trị từ biến siteUrl của config/application.php

Để replace các tag mới trong template cần điều chỉnh lại function vPortal\_Display::replaceUiConstant($content) trong file Display.php. Hiện tại hệ thống đã có 2 tag CSS\_URL và TOKEN. Nếu các biến được lấy lên từ DB cần thêm hoặc chỉnh sửa các function xử lấy data trong **CoreData.php**

* 1. Thêm cấu trúc biên dịch template

Để thêm các cấu trúc biên dịch template chúng ta sẽ điều chỉnh function vPortal\_Utility::\_ compile ($featureCode, $view, $template) trong file Utility.php. Function này sẽ tìm cấu trúc được định dạng trên template chuyển thành code PHP để xử lý tiếp.

1. Import thư viện VportalCore vào website.

Để sử dụng portal thư viện vPortalCore vào website, trong file index.php ta chèn vào dòng code sau:

include $corePath . '/initialize.php';

$corePath: là đường dẫn ổ đĩa tới thư viện của vPortalCore.

Để sử dụng các function trong thư viện vPortalCore ta sử dụng cấu trúc sau:

Class::function(param1, param1,..)

Chi tiết các function đã được giới thiệu trong phần **II**, xem thêm mô hình hoạt động của vPortalCore tại phần **I.2**

1. Database sử dụng trong vPortalCore

vProtalCore làm việc chính với database vProtal, ví dụ: vng\_vportal, hungba\_vportal.

**Các table chính được sử dụng:**

* vp\_section: Lưu trữ thông tin của các section
* vp\_feature: Thông tin các module của website
* vp\_block: Thông tin của các block
* vp\_block\_instance: Các block hiển thị trên một section
* vp\_template\_master: Lưu trữ các giao diện chính của website
* vp\_template\_master\_position: Thông tin vị trí trên các template master
* vp\_css: Thông tin đường dẫn css của các template
* vp\_skin: Danh sách các skin của web
* vp\_position: Danh sách các vị trí trên website.
* vp\_template: Giao diện của block, module của các feature