

How to insatall CakePHP (version 3.5.*) with Composer & PostgreSQL



CakePHP

CakePHP 3.x feature::

Better performance:

Version 3 incorporates performance improvements to the bootstrap process, the routing process, and several parts of process for generating helper templates.

Enhanced components and helpers:

Version 3 provides enhanced support for “flash messages” with its new FlashHelper and FlashComponent. In addition, the CookieComponent has been enhanced, making it easier to separate the configuration of cookie namespaces and the handling of cookie data.

Improved session management:

Session management has always been a static class in CakePHP which has proven to be problematic in a number of ways. With version 3, you can now access the session from the request object `$this->request->session()`. This change also makes the session easier to test, and enables CakePHP to use PHPUnit 4.x.

Improved consistency of conventions:

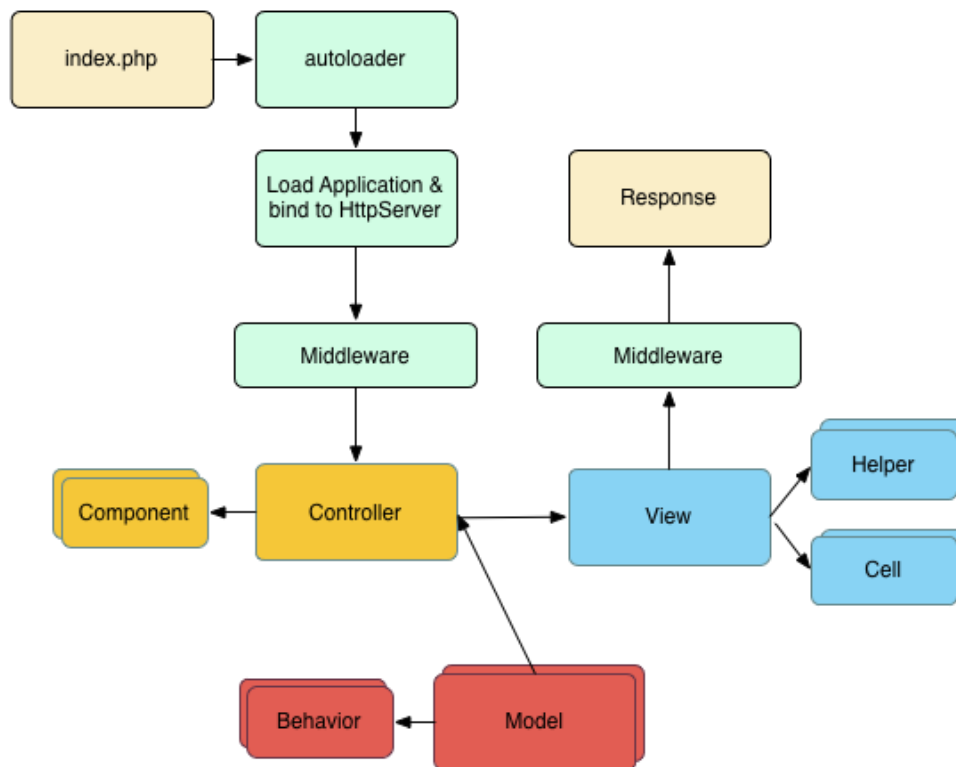
The application skeleton and plugin skeletons have been updated to use the same directory structure in order to be more consistent with one another.

Themes and plugins merged:

A key goal of CakePHP 3 was to make themes more powerful and robust. Working toward that goal, it became apparent that what was really needed was for themes to provide the same capabilities as plugins. Accordingly, any plugin may now be used as a theme, which also simplifies packaging and redistribution.

ORM Improvements:

Several API changes have been made to the ORM (Object-relational mapping). Most notably, it's now simpler to specify deep associations for saving operations, and a couple of conventions have been changed to reduce the learning curve and confusion among new adopters.

Cake Life Cycle:

The typical CakePHP request cycle starts with a user requesting a page or resource in your application. At a high level each request goes through the following steps:

1. The webserver rewrite rules direct the request to `webroot/index.php`.
2. Your Application is loaded and bound to an `HttpServer`.
3. Your application's middleware is initialized.
4. A request and response is dispatched through the PSR-7 Middleware that your application uses. Typically this includes error trapping and routing.
5. If no response is returned from the middleware and the request contains routing information, a controller & action are selected.
6. The controller's action is called and the controller interacts with the required Models and Components.
7. The controller delegates response creation to the View to generate the output resulting from the model data.
8. The view uses Helpers and Cells to generate the response body and headers.
9. The response is sent back out through the Middleware.
10. The `HttpServer` emits the response to the webserver.

CakePHP Folder Structure:

After you've downloaded the CakePHP application skeleton, there are a few top level folders you should see:

- The bin folder holds the Cake console executables.
- The config folder holds the (few) Configuration files CakePHP uses. Database connection details, bootstrapping, core configuration files and more should be stored here.
- The plugins folder is where the Plugins your application uses are stored.
- The logs folder normally contains your log files, depending on your log configuration.
- The src folder will be where your application's files will be placed.
- The tests folder will be where you put the test cases for your application.
- The tmp folder is where CakePHP stores temporary data. The actual data it stores depends on how you have CakePHP configured, but this folder is usually used to store translation messages, model descriptions and sometimes session information.
- The vendor folder is where CakePHP and other application dependencies will be installed by composer. Editing these files is not advised, as composer will overwrite your changes next time you update.
- The webroot directory is the public document root of your application. It contains all the files you want to be publically reachable.
- Make sure that the tmp and logs folders exist and are writable, otherwise the performance of your application will be severely impacted. In debug mode, CakePHP will warn you, if these directories are not writable.

The src Folder

CakePHP's src folder is where you will do most of your application development. Let's look a little closer at the folders inside src.

Controller

Contains your application's controllers and their components.

Locale

Stores string files for internationalization.

Model

Contains your application's tables, entities and behaviors.

Shell

Contains the console commands and console tasks for your application. For more information see Shells, Tasks & Console Tools.

View

Presentational classes are placed here: views, cells, helpers.

Template

Presentational files are placed here: elements, error pages, layouts, and view template files.

CakePHP Conventions:

We are big fans of convention over configuration. While it takes a bit of time to learn CakePHP's conventions, you save time in the long run. By following conventions, you get free functionality, and you liberate yourself from the maintenance nightmare of tracking config files. Conventions also make for a very uniform development experience, allowing other developers to jump in and help.

Controller Conventions

Controller class names are plural, CamelCased, and end in Controller. UsersController and ArticleCategoriesController are both examples of conventional controller names.

Public methods on Controllers are often exposed as 'actions' accessible through a web browser. For example the /users/view maps to the view() method of the UsersController out of the box. Protected or private methods cannot be accessed with routing.

URL Considerations for Controller Names

As you've just seen, single word controllers map to a simple lower case URL path. For example, UsersController (which would be defined in the file name UsersController.php) is accessed from http://example.com/users.

While you can route multiple word controllers in any way you like, the convention is that your URLs are lowercase and dashed using the DashedRoute class, therefore /article-categories/view-all is the correct form to access the ArticleCategoriesController::viewAll() action.

File and Class Name Conventions

In general, filenames match the class names, and follow the PSR-0 or PSR-4 standards for autoloading. The following are some examples of class names and their filenames:

The Controller class LatestArticlesController would be found in a file named LatestArticlesController.php

The Component class MyHandyComponent would be found in a file named MyHandyComponent.php

The Table class OptionValuesTable would be found in a file named OptionValuesTable.php.

The Entity class OptionValue would be found in a file named OptionValue.php.

The Behavior class EspeciallyFunkableBehavior would be found in a file named EspeciallyFunkableBehavior.php

The View class SuperSimpleView would be found in a file named SuperSimpleView.php

The Helper class BestEverHelper would be found in a file named BestEverHelper.php

Model and Database Conventions:

Table class names are plural, CamelCased and end in Table. UsersTable, ArticleCategoriesTable, and UserFavoritePagesTable are all examples of conventional model names.

Table names corresponding to CakePHP models are plural and underscored. The underlying tables for the above mentioned models would be users, article_categories, and user_favorite_pages, respectively.

The convention is to use English words for table and column names. If you use words in another language, CakePHP might not be able to process the right inflections (from singular to plural and vice-versa). If you need to add your own language rules for some words, you can use the utility class Cake\Utility\Inflector. Besides defining those custom inflection rules, this class also allows you to check that CakePHP understands your custom syntax for plurals and singulars words. See the documentation about Inflector for more information.

Field names with two or more words are underscored: `first_name`.

Foreign keys in `hasMany`, `belongsTo`/`hasOne` relationships are recognized by default as the (singular) name of the related table followed by `_id`. So if `Users` `hasMany` `Articles`, the `articles` table will refer to the `users` table via a `user_id` foreign key. For a table like `article_categories` whose name contains multiple words, the foreign key would be `article_category_id`.

Join tables, used in `BelongsToMany` relationships between models, should be named after the model tables they will join, arranged in alphabetical order (`articles_tags` rather than `tags_articles`).

In addition to use an auto-increment key as the primary key, you may also use UUID columns. CakePHP will create a unique 36 character UUID (`Cake\Utility\Text::uuid()`) whenever you save a new record using the `Table::save()` method.

View Conventions

View template files are named after the controller functions they display, in an underscored form. The `viewAll()` function of the `ArticlesController` class will look for a view template in `src/Template/Articles/view_all.ctp`.

The basic pattern is `src/Template/Controller/underscored_function_name.ctp`.

By naming the pieces of your application using CakePHP conventions, you gain functionality without the hassle and maintenance tethers of configuration. Here's a final example that ties the conventions together:

Database table: "articles"

Table class: `ArticlesTable`, found at `src/Model/Table/ArticlesTable.php`

Entity class: `Article`, found at `src/Model/Entity/Article.php`

Controller class: `ArticlesController`, found at `src/Controller/ArticlesController.php`

View template, found at `src/Template/Articles/index.ctp`

Using these conventions, CakePHP knows that a request to `http://example.com/articles/` maps to a call on the `index()` function of the `ArticlesController`, where the `Articles` model is automatically available (and automatically tied to the 'articles' table in the database), and renders to a file. None of these relationships have been configured by any means other than by creating classes and files that you'd need to create anyway.

:: Work Shop ::

System Required

How to setup for our server

WAMP Sever (LAMP stack server bundle of Windows, Apache, MySQL, PHP) recommended version 3.1.0 with x64:

<http://www.wampserver.com/en/>

Download Link: <https://sourceforge.net/projects/wampserver/files/>



Install path:

D:\wamp

Apache and PHP Configure:

Enabled Extension PHP extension

Open php.ini and follow for each uncommented

- extension=php_intl.dll
- extension=php_pdo_pgsql.dll
- extension=php_pgsql.dll
- extension=php_mbstring.dll

Setting PHP default time zone in php.ini

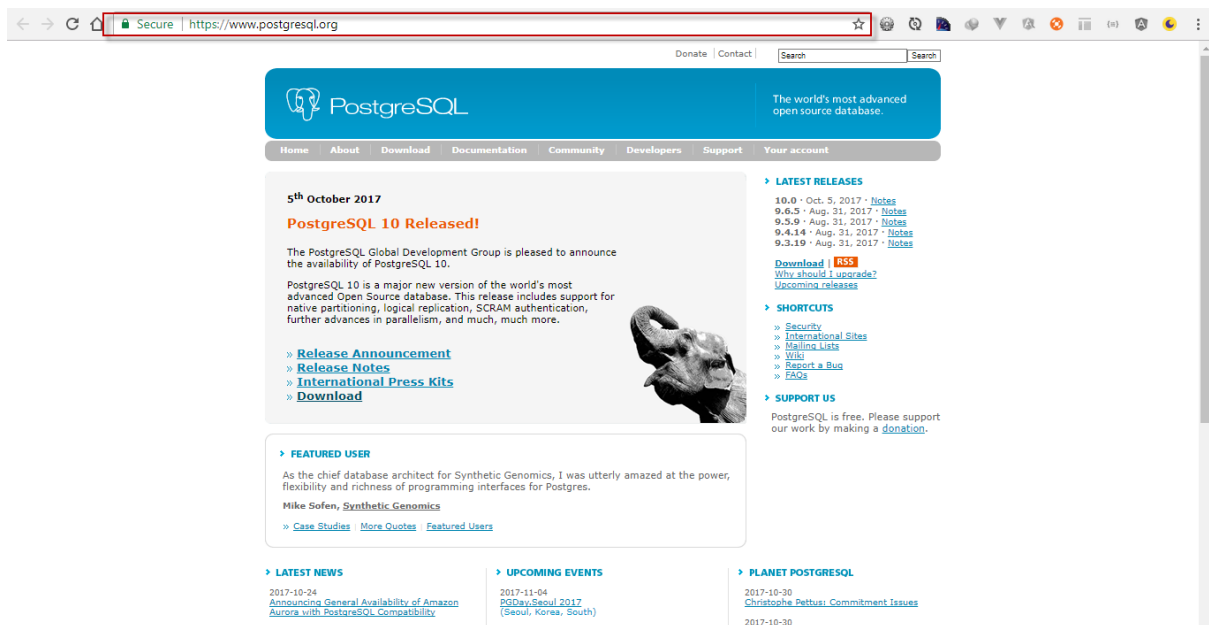
date.timezone = Asia/Bangkok

Enable for Apache mod rewrite rule in file httpd.conf

LoadModule rewrite_module modules/mod_rewrite.so

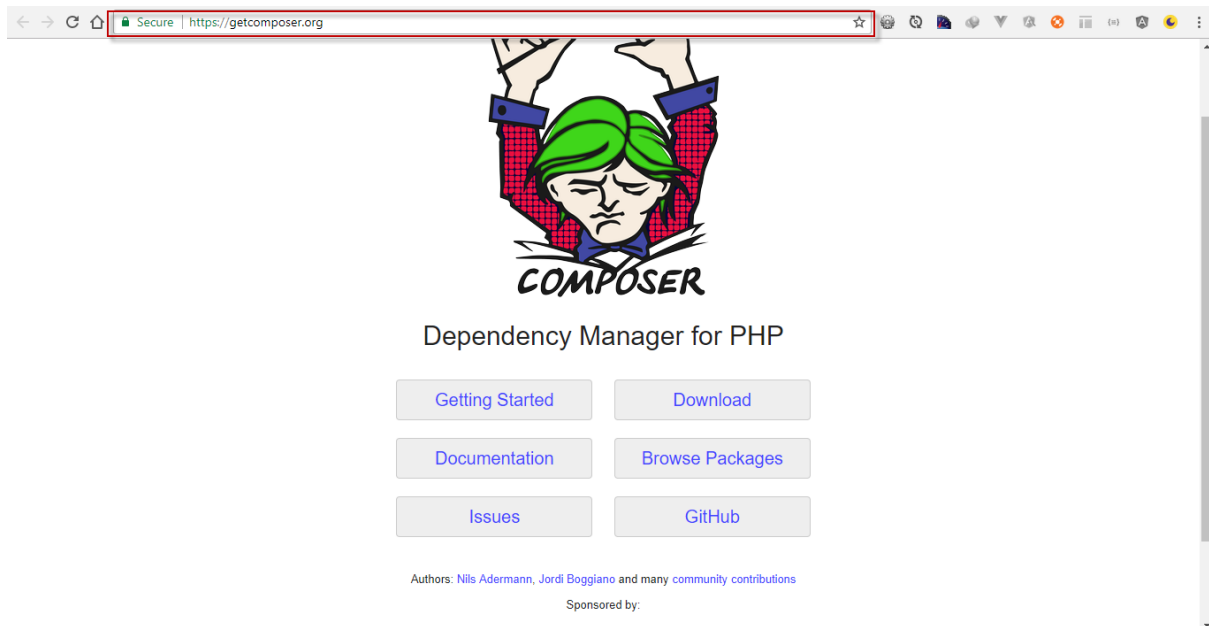
PostgreSQL (SQL Server) recommended 9.4.x:

<https://www.postgresql.org/>



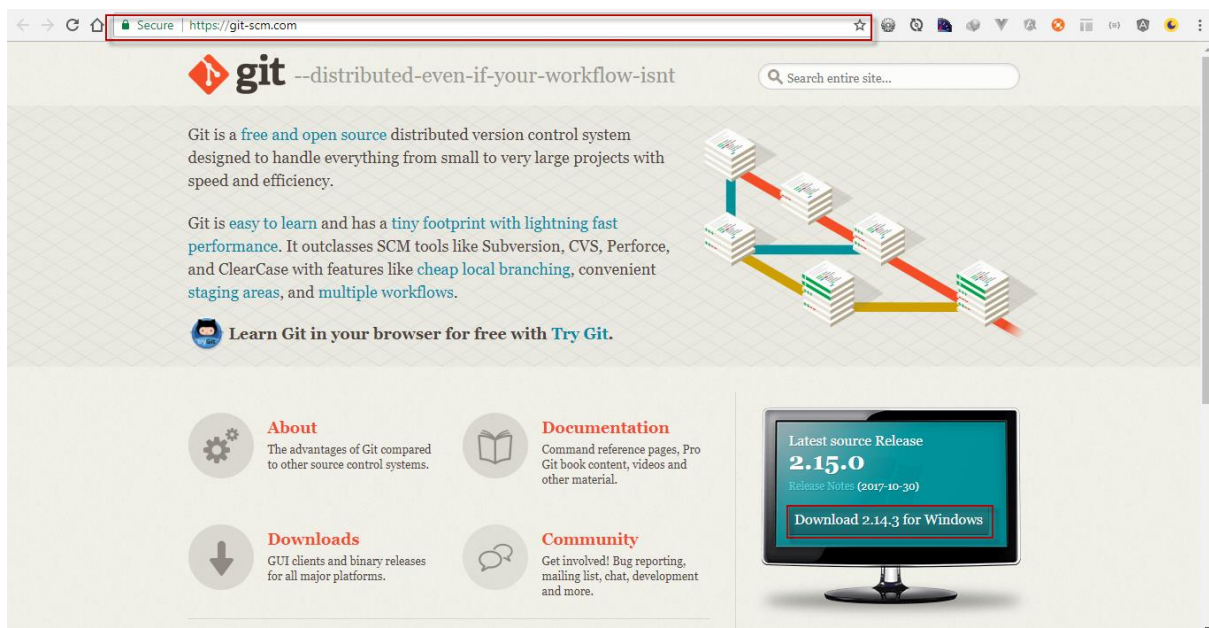
Composer (Dependency of PHP):

<https://getcomposer.org/Composer-Setup.exe>



GIT Repository:

<https://git-scm.com/>

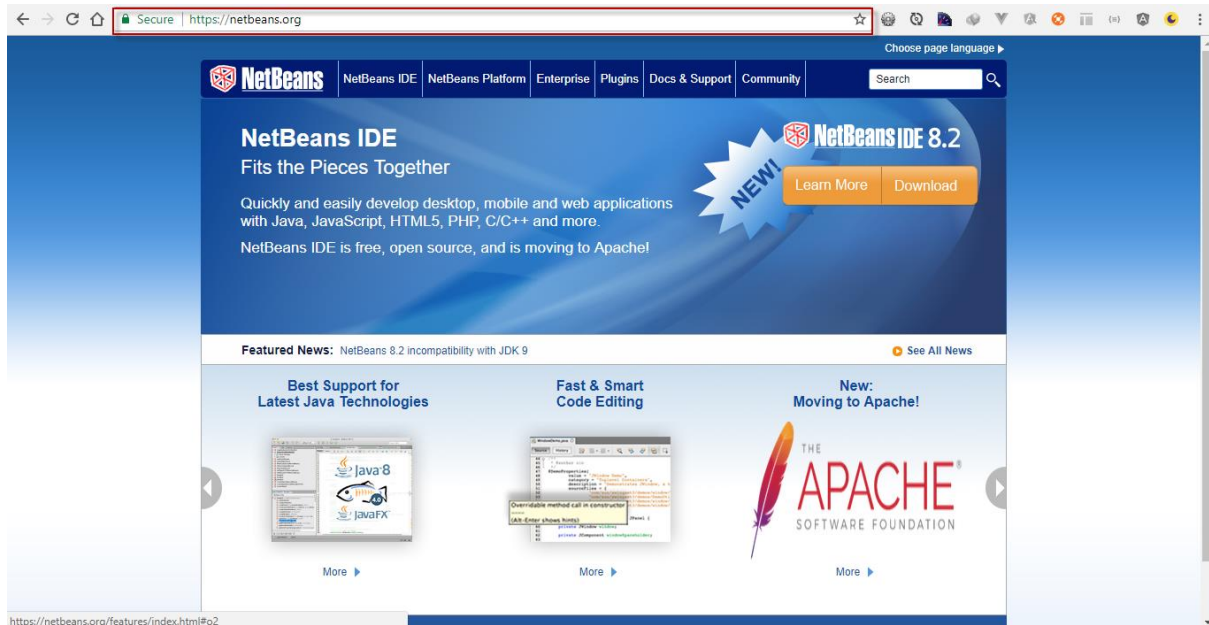


Java JDK Version 1.8 x86-64:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

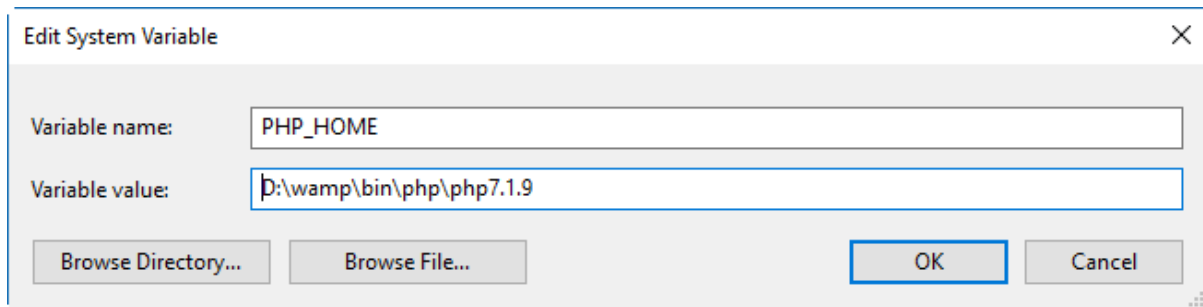
NetBean Editor Version 8.1:

<https://netbeans.org/downloads/> (Your Java JDK needed to install first)



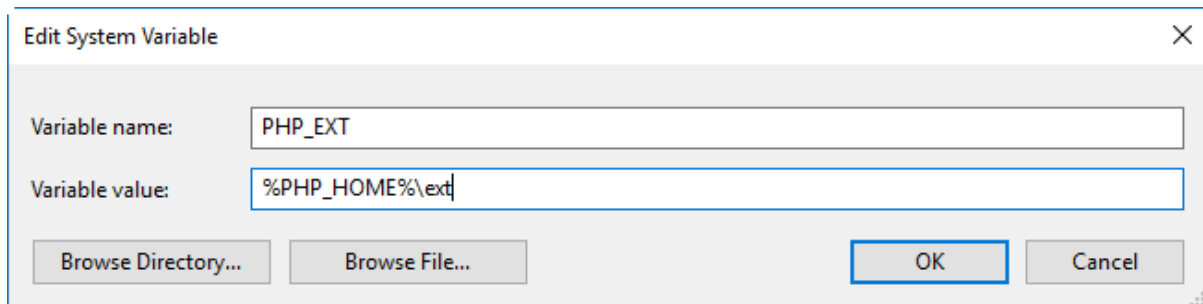
Setting for Environment path:

PHP_HOME = D:\wamp\bin\php\php7.1.9



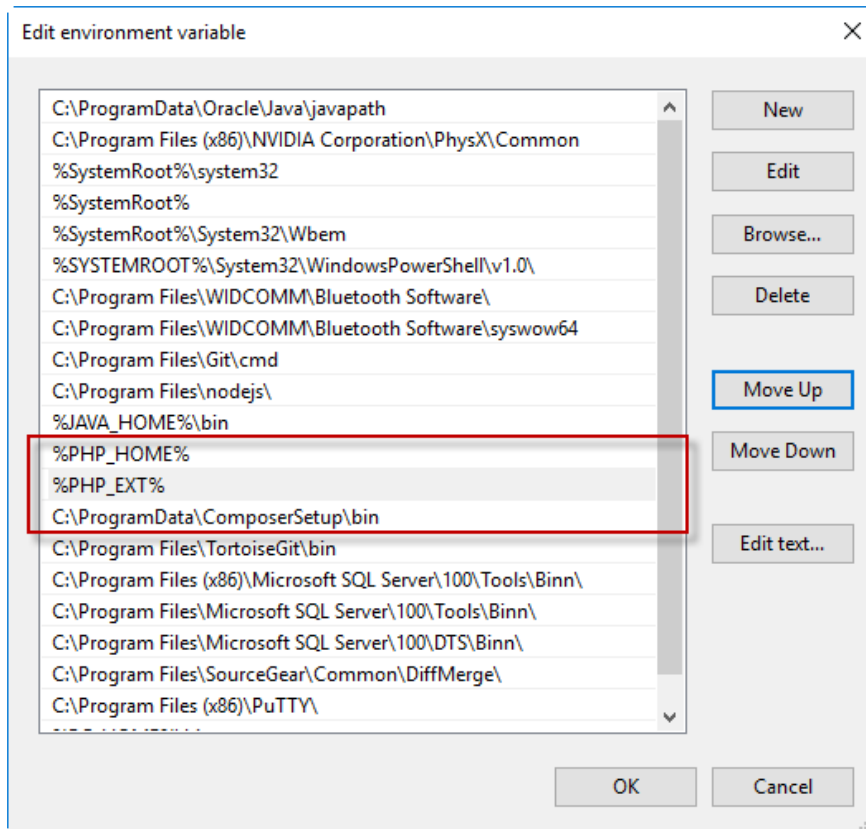
The screenshot shows a Windows 'Edit System Variable' dialog box. The title bar reads 'Edit System Variable' with a close button (X) on the right. Inside the dialog, there are two text input fields. The first field is labeled 'Variable name:' and contains the text 'PHP_HOME'. The second field is labeled 'Variable value:' and contains the text 'D:\wamp\bin\php\php7.1.9'. Below these fields, there are four buttons: 'Browse Directory...', 'Browse File...', 'OK', and 'Cancel'. The 'OK' button is highlighted with a blue border. The dialog box has a standard Windows XP-style appearance with a light gray background and a blue border.

PHP_EXT = D:\wamp\bin\php\php7.1.9 \ext



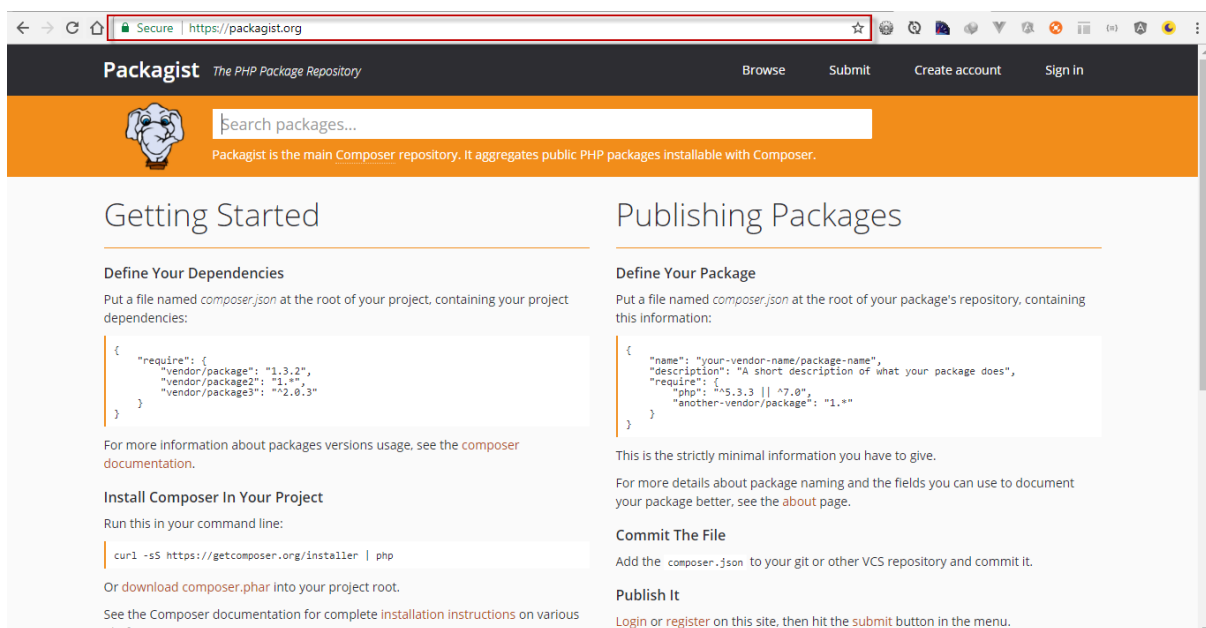
The screenshot shows a Windows 'Edit System Variable' dialog box. The title bar reads 'Edit System Variable' with a close button (X) on the right. Inside the dialog, there are two text input fields. The first field is labeled 'Variable name:' and contains the text 'PHP_EXT'. The second field is labeled 'Variable value:' and contains the text '%PHP_HOME%\ext'. Below these fields, there are four buttons: 'Browse Directory...', 'Browse File...', 'OK', and 'Cancel'. The 'OK' button is highlighted with a blue border. The dialog box has a standard Windows XP-style appearance with a light gray background and a blue border.

And then adding above variable to “path” variable



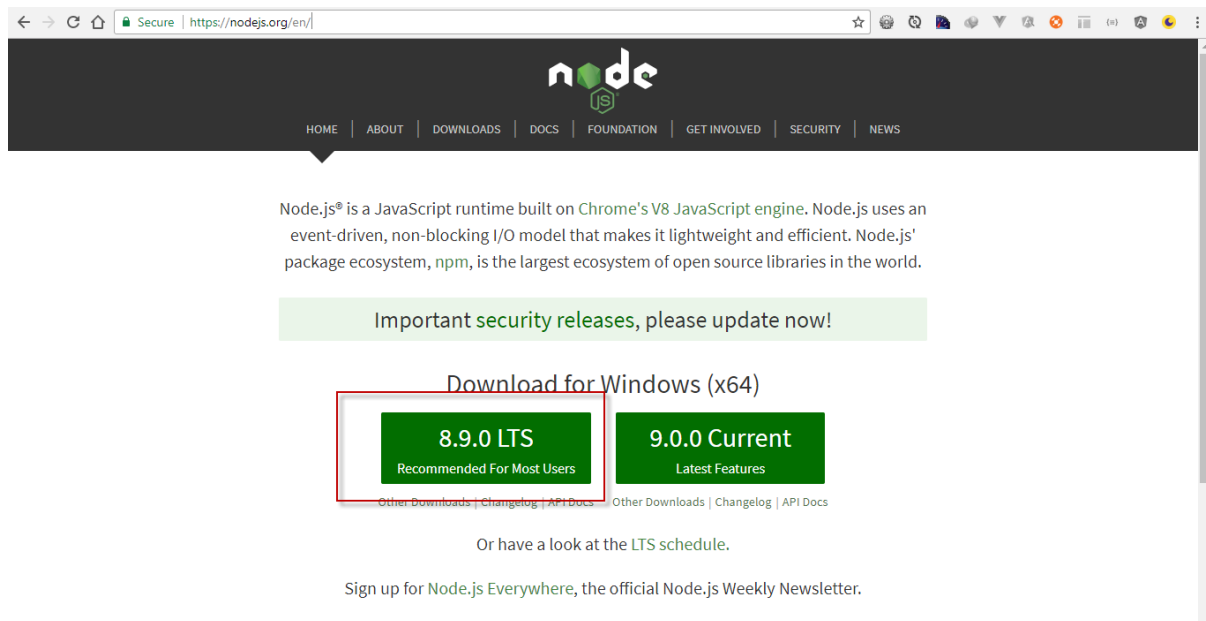
Composer and packagist: (PHP Dependency Repository)

<https://packagist.org/>



NodeJS: (JavaScript Compiler)

<https://nodejs.org/en/>



Checking for NodeJS completed installed

\$ node -v

\$ npm -v

```
MINGW64:/d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
dk_antikorn@DKANTIKORN MINGW64 /d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
$ node -v
v8.9.0
dk_antikorn@DKANTIKORN MINGW64 /d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
$ npm -v
5.4.2
dk_antikorn@DKANTIKORN MINGW64 /d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
$ |
```

Install CakePHP 3.x with Composer:

\$ cd d:

```
$ cd wamp/www/
```

```
$ composer self-update && composer create-project --prefer-dist cakephp/app: "3.5.*" DEMO_CAKEPHP3_CRUD/www
```

```
$ cd DEMO_CAKEPHP_CRUD/www
```

[illegible]

Setup for visual hosting:

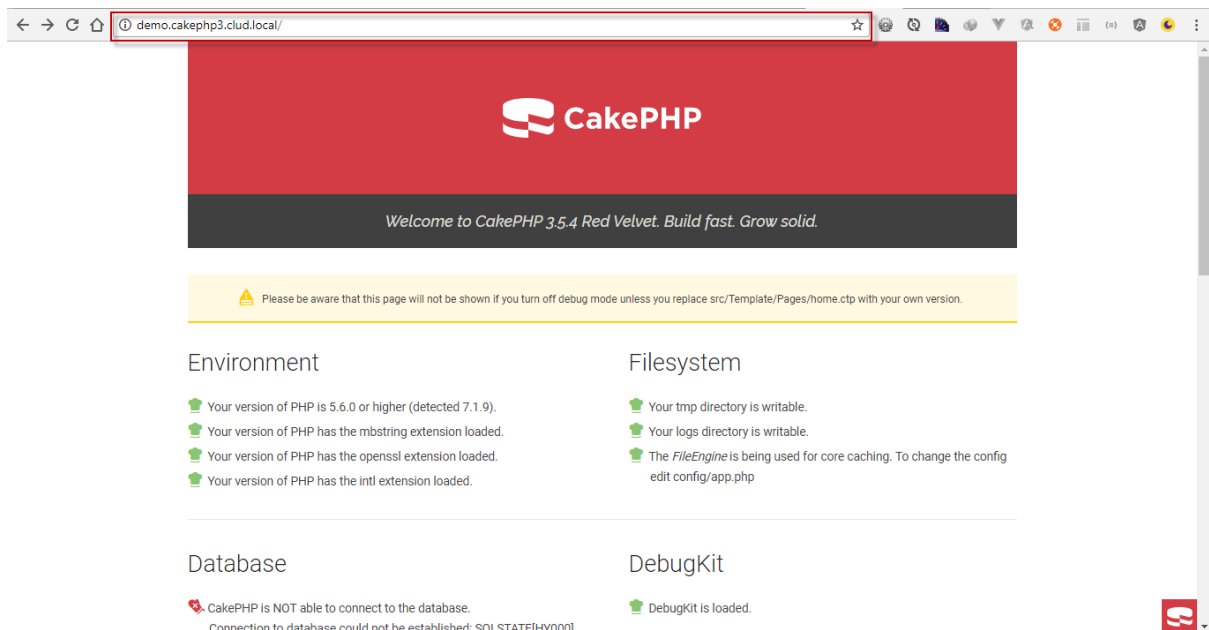
Add 2 line below to => C:\Windows\System32\drivers\etc\hosts

```
127.0.0.1    demo.cakephp3.crud.local
::1         demo.cakephp3.crud.local
```

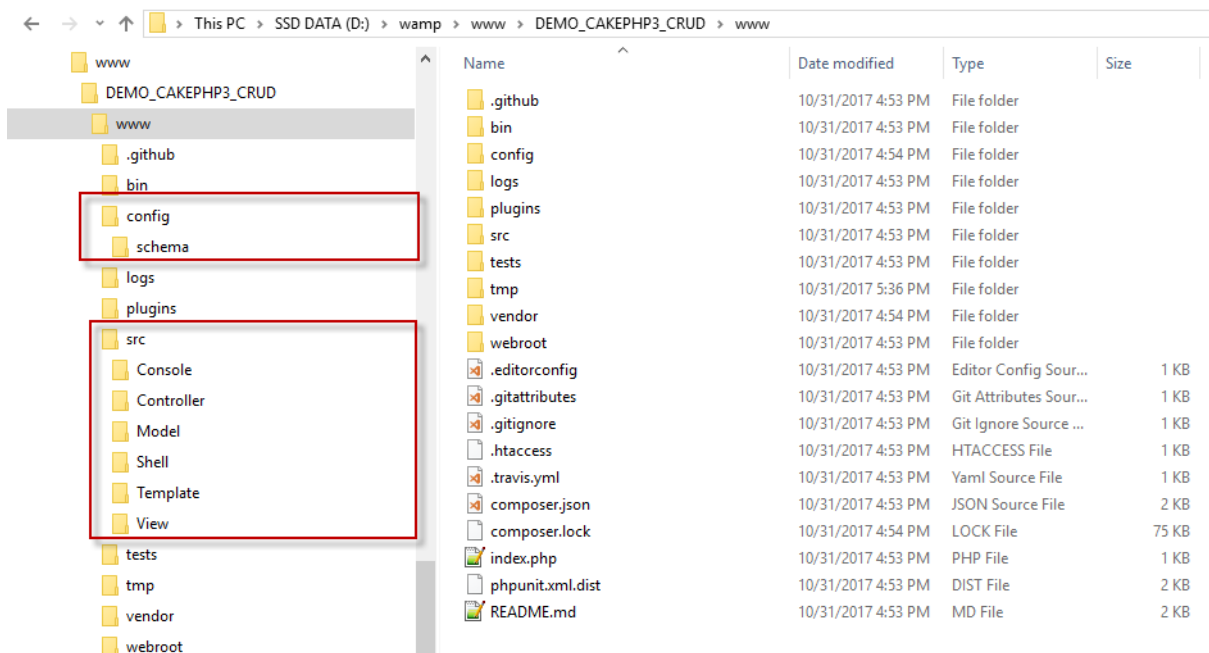
Add content below to => D:\wamp\bin\apache\apache2.4.27\conf\extra\httpd-vhosts.conf

```
<VirtualHost *:80>
    ServerName demo.cakephp3.crud.local
    ServerAlias demo.cakephp3.crud.local
    DocumentRoot "${INSTALL_DIR}/www/DEMO_CAKEPHP3_CRUD/www/"
    ErrorLog "logs/demo.cakephp3.crud.local-error.log"
    CustomLog "logs/demo.cakephp3.crud.local-access.log" common
    <Directory "${INSTALL_DIR}/www/DEMO_CAKEPHP3_CRUD/www/">
        Options +Indexes +Includes +FollowSymLinks +MultiViews
        AllowOverride All
        Require local
    </Directory>
</VirtualHost>
```


And then open your browser:



Folder Structure:



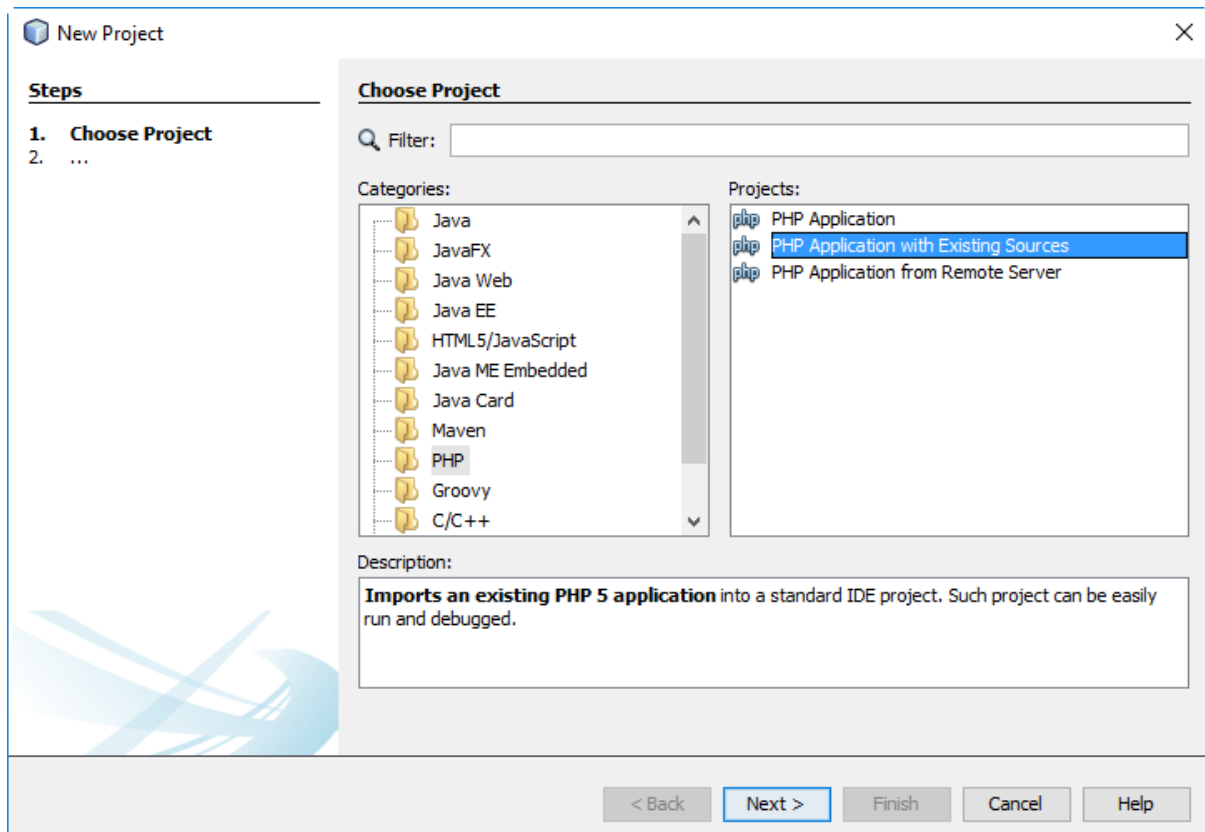
Tree view CakePHP3.x folder structure:

```

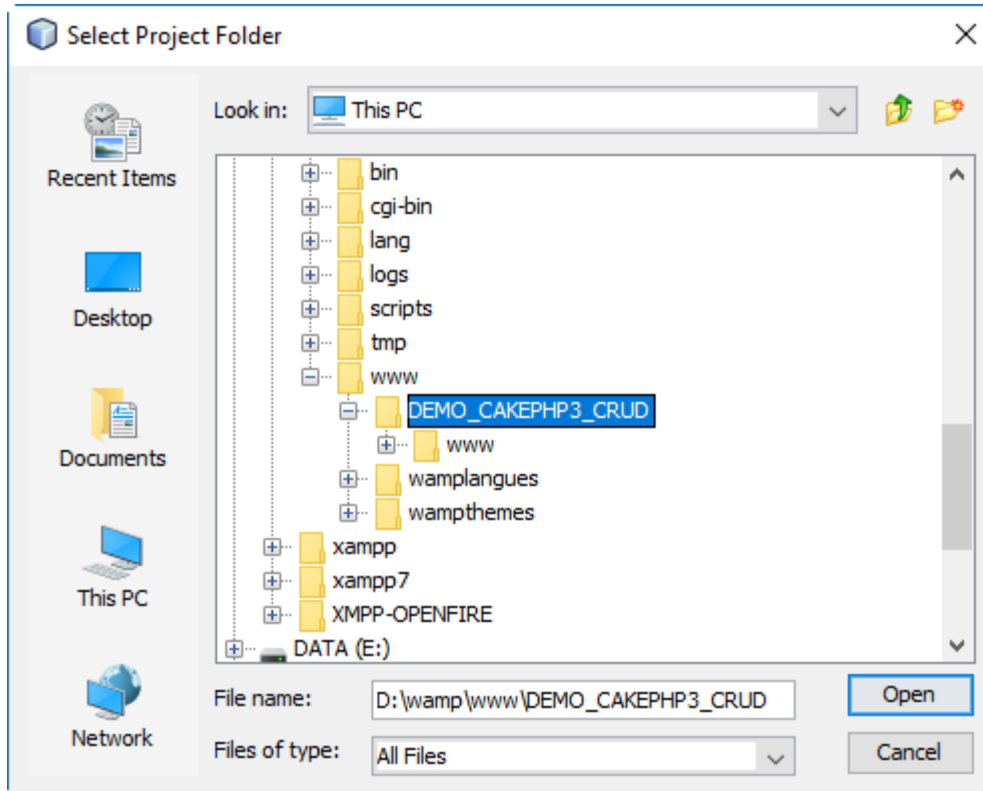
Administrator: Command Prompt
D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src>tree
Folder PATH listing for volume SSD DATA
Volume serial number is 000000B6 EE7D:90E1
D:..
-- Console
-- Controller
-- Component
-- Model
--   -- Behavior
--   -- Entity
--   -- Table
-- Shell
-- Template
--   -- Element
--   --   -- Flash
--   -- Email
--   --   -- html
--   --   -- text
--   -- Error
--   -- Layout
--   --   -- Email
--   --   --   -- html
--   --   --   -- text
--   --   -- rss
--   -- Pages
-- View
--   -- Helper
D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src>

```

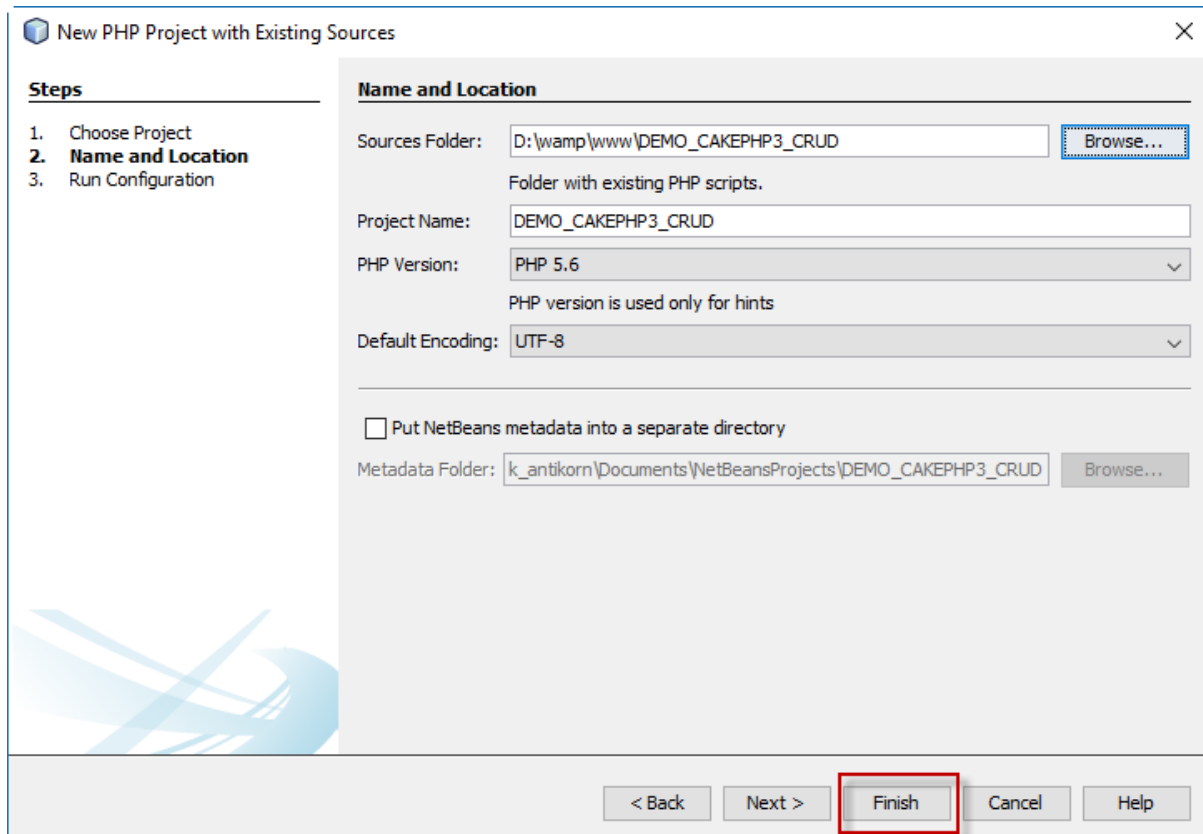
Open Netbean and file => New project => create new PHP project with existing source



Browse:



And Click Open



New PHP Project with Existing Sources

Steps

1. Choose Project
- 2. Name and Location**
3. Run Configuration

Name and Location

Sources Folder: D:\wamp\www\DEMO_CAKEPHP3_CRUD Browse...

Folder with existing PHP scripts.

Project Name: DEMO_CAKEPHP3_CRUD

PHP Version: PHP 5.6 ▼

PHP version is used only for hints

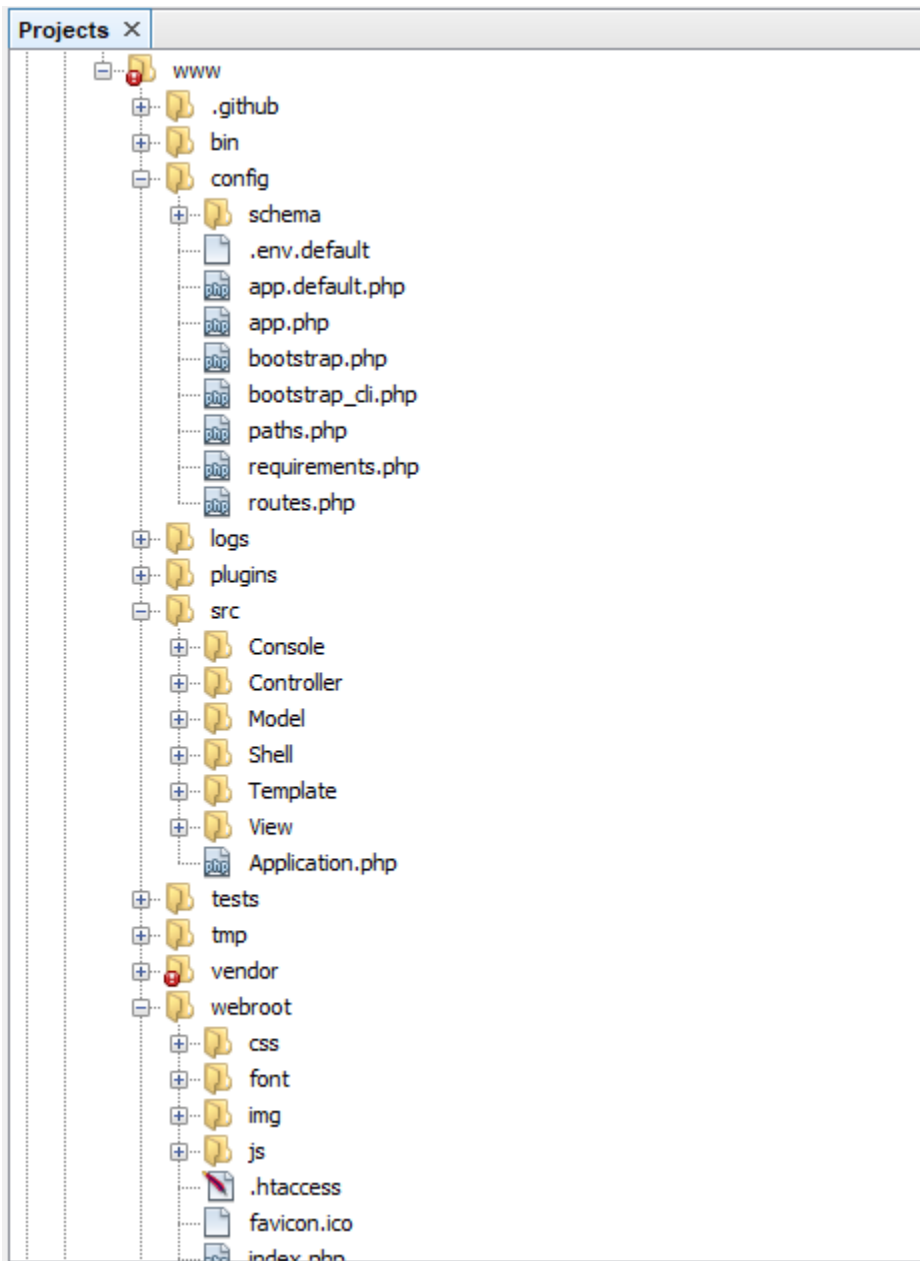
Default Encoding: UTF-8 ▼

☐ Put NetBeans metadata into a separate directory

Metadata Folder: k_antikorn\Documents\NetBeansProjects\DEMO_CAKEPHP3_CRUD Browse...

< Back Next > **Finish** Cancel Help

And then click finish



Create database role for the demo app:

role name: demo_cakephp3_crud

password: password

New Login Role...

Properties Definition Role privileges Role membership Variables Se

Role name demo_cakephp3_crud

OID

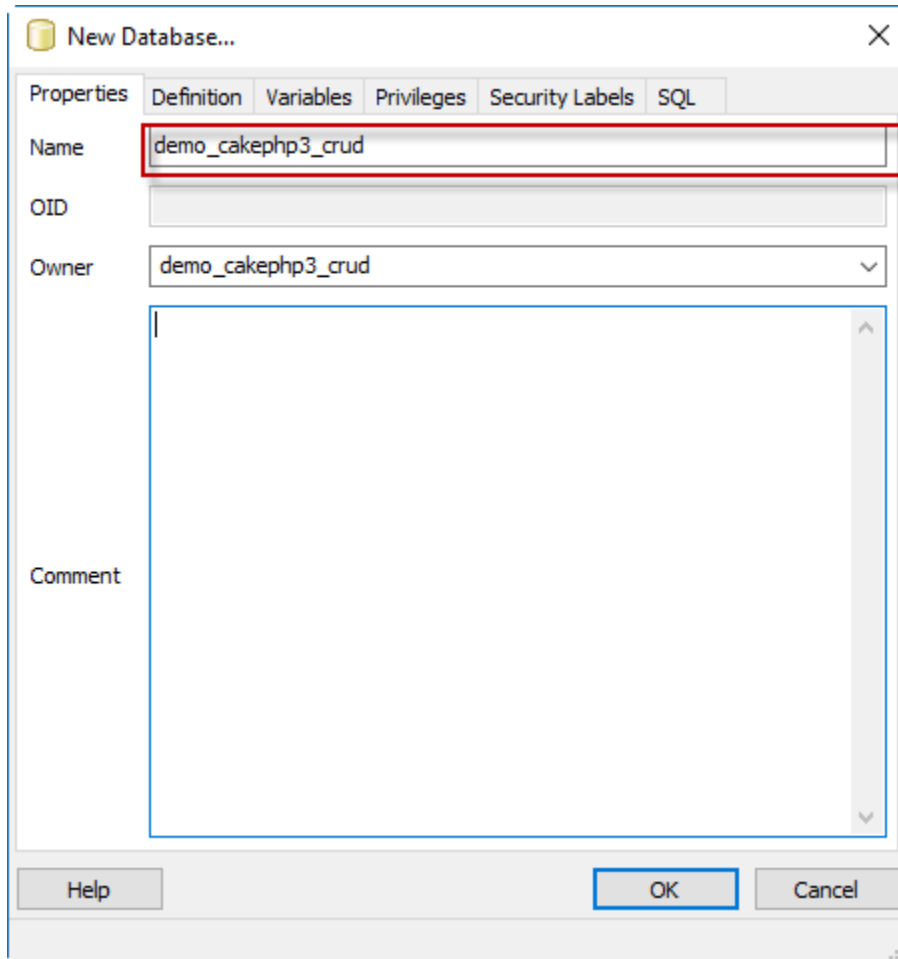
Comment demo_cakephp3_crud fro database demo_cakephp3_crud

Use Slony

Help OK Cancel

The passwords entered do not match!

Create database `demo_cakephp3_crud` owner with `demo_cakephp3_crud`:



The image shows a 'New Database...' dialog box with the following fields and controls:

- Name:** A text input field containing 'demo_cakephp3_crud', highlighted with a red border.
- OID:** An empty text input field.
- Owner:** A dropdown menu showing 'demo_cakephp3_crud'.
- Comment:** A large empty text area.
- Buttons:** 'Help', 'OK' (highlighted with a blue border), and 'Cancel'.

Configure for CakePHP3.x Database Connection:

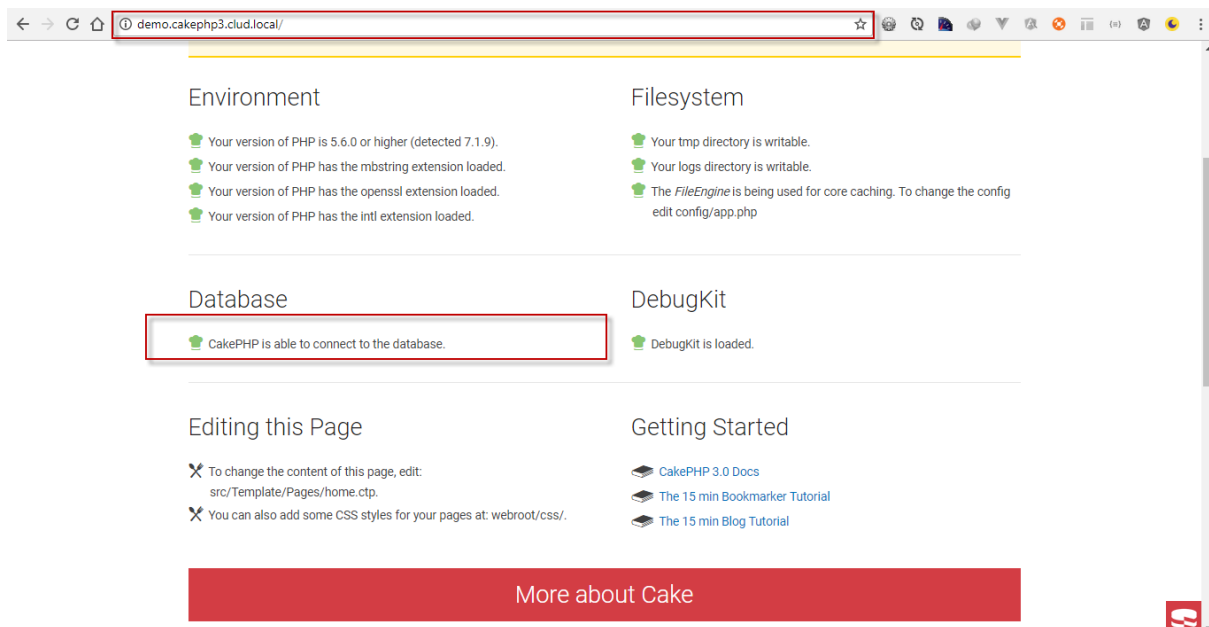
Open config/app.php and then change below line

```
'Datasources' => [
    'default' => [
        'className' => 'Cake\Database\Connection',
        'driver' => 'Cake\Database\Driver\Mysql',
        .....
        'username' => 'my_app',
        'password' => 'secret',
        'database' => 'my_app',
        'encoding' => 'utf8',
        .....
    ],
```

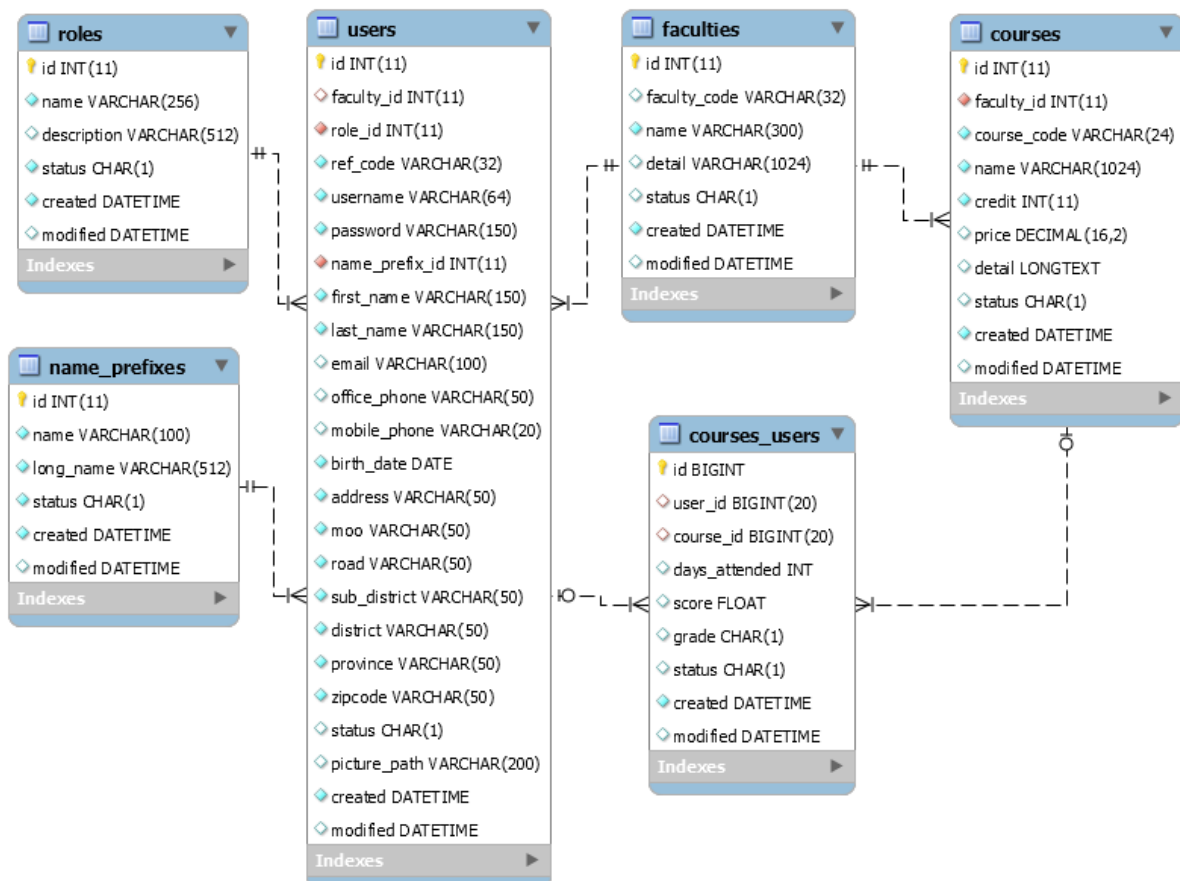
To::

```
'Datasources' => [
    'default' => [
        'className' => 'Cake\Database\Connection',
        'driver' => 'Cake\Database\Driver\Postgres',
        'persistent' => false,
        'host' => 'localhost',
        'username' => 'demo_cakephp3_crud',
        'password' => 'password',
        'database' => 'demo_cakephp3_crud',
        .....
    ],
```


And then open in your browser:



Creating the Database for our CakePHP3.x demo:



Generating Code With Backing Scaffolding:

Required PHP CLI 5.6.2 or Higher Checking typing for `$ php -v` or `$ bin/cake bake`

```
$ bin/cake bake

Welcome to CakePHP v3.4.6 Console
-----
App : src
Path: /var/www/cakephp.dev/src/
PHP : 5.6.20
-----
The following commands can be used to generate skeleton code for your application.

Available bake commands:

- all
- behavior
- cell
- component
- controller
- fixture
- form
- helper
- mailer
- migration
- migration_diff
- migration_snapshot
- model
- plugin
- seed
- shell
- shell_helper
- task
- template
- test

By using `cake bake [name]` you can invoke a specific bake task.
```

Generating our scaffold code

On Windows you'll need to use `bin\cake` instead:

`$ bin\cake bake all users`

On Linux typing the following command:

`$./cake bake all users`

`$./cake bake all roles`

`$./cake bake all faculties`

`$./cake bake all courses`

`$./cake bake all name_prefixes`

`$./cake bake all courses_users`

Or one command for bake all

\$./cake bake all everything

```

MINGW64/d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/bin
Creating file D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Template\Roles\edit.ctp
Wrote 'D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Template\Roles\edit.ctp'
Bake All complete.
d:\wamp\www\DEMO_CAKEPHP3_CRUD\www\bin
$ ./cake bake all faculties
Bake All
One moment while associations are detected.
Baking table class for Faculties...
Creating file D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Model\Table\FacultiesTable.php
Wrote 'D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Model\Table\FacultiesTable.php'
Baking entity class for Faculty...
Creating file D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Model\Entity\Faculty.php
Wrote 'D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\src\Model\Entity\Faculty.php'
Baking test fixture for Faculties...
Creating file D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\tests\Fixture\FacultiesFixture.php
Wrote 'D:\wamp\www\DEMO_CAKEPHP3_CRUD\www\tests\Fixture\FacultiesFixture.php'
Bake is detecting possible fixtures...

```

Open up your browser:

demo.cakephp3.clud.local/users

Users

Documentation API

ACTIONS

- New User
- List Faculties
- New Faculty
- List Roles
- New Role
- List Name Prefixes
- New Name Prefix
- List Courses
- New Course

Users

Id	Faculty	Role	Ref Code	User Name	First Name	Last Name	Email	Office Phone	Mobile Phone	Birth Date	Address	City	Road	Sub District	District	Province	Zip Code	Status	Picture Path	Create	Modify	Actions
< previous next >																						
Page 1 of 1, showing 0 record(s) out of 0 total																						

CakePHP3.x Helper:

Install for CakePHP Bootstrap Helper

Documentation for bootstrap helper inject bootstrap class to CakePHP Form:

<https://holt59.github.io/cakephp3-bootstrap-helpers/>

<https://github.com/Holt59/cakephp3-bootstrap-helpers>

Run the command on your terminal

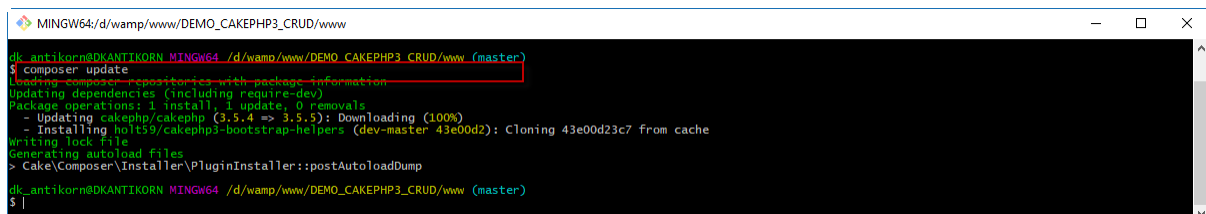
\$ composer require holt59/cakephp3-bootstrap-helpers:dev-master

Or add this line on your composer.json

```
"require": {
    "holt59/cakephp3-bootstrap-helpers": "dev-master"
}
```

And then run in your terminal:

\$ composer update



Add line to your config/bootstrap.php

```
Plugin::load('Bootstrap');
```

Add below to your AppController.php

```
public $helpers = [  
    'Form' => ['className' => 'Bootstrap.Form'],  
    'Html' => ['className' => 'Bootstrap.Html'],  
    'Modal' => ['className' => 'Bootstrap.Modal'],  
    'Navbar' => ['className' => 'Bootstrap.Navbar'],  
    'Paginator' => ['className' => 'Bootstrap.Paginator'],  
    'Panel' => ['className' => 'Bootstrap.Panel']  
];
```

CakePHP3.x Layout:**Install Package node package:**

```
$ cd D:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
```

create file call name package.json and then adding for below content

```
{
  "name": "demo_cakephp_crud",
  "version": "1.0.0",
  "description": "Demo for CakePHP 3.5.x with CRUDE and Material Design for Bootstrap (MBD)",
  "main": "index.js",
  "directories": {
    "test": "tests"
  },
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "dev": "echo \"xxx\""
  },
  "author": "Sarawutt.b",
  "license": "ISC",
  "devDependencies": {
    "bootstrap": "^3.3.7",
    "bower": "^1.8.0",
    "font-awesome": "^4.7.0",
    "infinite-scroll": "^3.0.0",
    "mdbootstrap": "4.3.2",
    "toastr": "^2.1.2"
  }
}
```

And then run:

\$ npm install

```

MINGW64/d:/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
jk_antikorn@DEKANTIKORN MINGW64 ~
$ cd D:
jk_antikorn@DEKANTIKORN MINGW64 /d
$ cd wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot/
jk_antikorn@DEKANTIKORN MINGW64 /d/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
$ npm install
npm WARN deprecated bower@1.8.2: ...psst! Your project can stop working at any moment because its dependencies can change. Prevent this by migrating to Yarn: https://bower.io/blog/2017/how-to-migrate-away-from-bower/
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN demo_cakephp_crud@1.0.0 No repository field.
added 9 packages in 26.09s
jk_antikorn@DEKANTIKORN MINGW64 /d/wamp/www/DEMO_CAKEPHP3_CRUD/www/webroot
$
  
```

Create file signin.ctp in src/Template/Layout/signin.ctp and adding for the content:

```

<?php
/**
 * CakePHP(tm) : Rapid Development Framework (http://cakephp.org)
 * Copyright (c) Cake Software Foundation, Inc. (http://cakefoundation.org)
 *
 * Licensed under The MIT License
 * For full copyright and license information, please see the LICENSE.txt
 * Redistributions of files must retain the above copyright notice.
 *
 * @copyright Copyright (c) Cake Software Foundation, Inc. (http://cakefoundation.org)
 * @link http://cakephp.org CakePHP(tm) Project
 * @since 0.10.0
 * @license http://www.opensource.org/licenses/mit-license.php MIT License
 */

$cakeDescription = 'CakePHP: the rapid development php framework';

?>

<!DOCTYPE html>

<html>

    <head>
  
```

```

<?php echo $this->Html->charset(); ?>

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<title> <?php echo $cakeDescription; ?>: <?php echo $this->fetch('title'); ?></title>

<?php echo $this->Html->meta('icon'); ?>


<!-- Font Awesome -->

<?php echo $this->Html->css('/node_modules/font-awesome/css/font-
awesome.min.css');?>

<!-- Bootstrap core CSS -->

<?php echo $this->Html->css('/node_modules/mdbootstrap/css/bootstrap.min.css'); ?>


<!-- Material Design Bootstrap -->

<?php echo $this->Html->css('/node_modules/mdbootstrap/css/mdb.min.css'); ?>

<!-- Your custom styles (optional) -->

<?php echo $this->Html->css('/node_modules/mdbootstrap/css/style.css'); ?>


<!-- Toastr Modern Notification for MDB -->

<?php echo $this->Html->css('/node_modules/toastr/build/toastr.min.css'); ?>


<!-- JQuery -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/jquery-3.2.1.min.js'); ?>


<!-- Bootstrap core JavaScript -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/bootstrap.min.js'); ?>


<?php echo $this->fetch('meta'); ?>

<?php echo $this->fetch('css'); ?>

<?php echo $this->fetch('script'); ?>

</head>

<body>

```



```

<main>

    <?php echo $this->Flash->render(); ?>

    <?php echo $this->fetch('content'); ?>

</main>


<!-- SCRIPTS -->

<!-- Bootstrap tooltips -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/tether.min.js'); ?>

<!-- MDB core JavaScript -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/mdb.min.js'); ?>


<!--toastr modern notification-->

<?php echo $this->Html->script('/node_modules/toastr/build/toastr.min.js'); ?>


<!--Load initial for infinite-scroll-->

<?php echo $this->Html->script('/node_modules/infinite-scroll/dist/infinite-
scroll.pkgd.min.js');?>

<!--EEN SCRIPT SECTION-->

</body>

</html>

```

Create file default.ctp in src/Template/Layout/default.ctp and adding for the content:

```
<?php
/**
 * CakePHP(tm) : Rapid Development Framework (http://cakephp.org)
 * Copyright (c) Cake Software Foundation, Inc. (http://cakefoundation.org)
 *
 * Licensed under The MIT License
 * For full copyright and license information, please see the LICENSE.txt
 * Redistributions of files must retain the above copyright notice.
 *
 * @copyright Copyright (c) Cake Software Foundation, Inc. (http://cakefoundation.org)
 * @link http://cakephp.org CakePHP(tm) Project
 * @since 0.10.0
 * @license http://www.opensource.org/licenses/mit-license.php MIT License
 */
$cakeDescription = 'CakePHP: the rapid development php framework';
?>
<!DOCTYPE html>
<html>
<head>
<?php echo $this->Html->charset(); ?>
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<title><?php echo $cakeDescription; ?>: <?php echo $this->fetch('title'); ?></title>
<?php echo $this->Html->meta('icon'); ?>

<!-- Font Awesome -->
<?php echo $this->Html->css('/node_modules/font-awesome/css/font-awesome.min.css');?>

<!-- Bootstrap core CSS -->
<?php echo $this->Html->css('/node_modules/mdbootstrap/css/bootstrap.min.css'); ?>
```

```

<!-- Material Design Bootstrap -->

<?php echo $this->Html->css('/node_modules/mdbootstrap/css/mdb.min.css'); ?>

<!-- Your custom styles (optional) -->

<?php echo $this->Html->css('/node_modules/mdbootstrap/css/style.css'); ?>


<!-- Toastr Modern Notification for MDB -->

<?php echo $this->Html->css('/node_modules/toastr/build/toastr.min.css'); ?>


<!-- JQuery -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/jquery-3.2.1.min.js'); ?>


<!-- Bootstrap core JavaScript -->

<?php echo $this->Html->script('/node_modules/mdbootstrap/js/bootstrap.min.js'); ?>


<?php echo $this->fetch('meta'); ?>

<?php echo $this->fetch('css'); ?>

<?php echo $this->fetch('script'); ?>

</head>

<body>

<header>

<!--Navbar-->

<nav class="navbar navbar-toggleable-md navbar-dark bg-primary">

    <div class="container">

        <button class="navbar-toggler navbar-toggler-right" type="button" data-
toggle="collapse" data-target="#collapseEx2" aria-controls="collapseEx2" aria-expanded="false"
aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <a class="navbar-brand" href="#">

            <strong><?php echo $this->fetch('title'); ?></strong>

        </a>

```

```

<div class="collapse navbar-collapse" id="collapseEx2">
  <ul class="navbar-nav mr-auto">
    <li class="nav-item active">
      <a class="nav-link waves-effect waves-light">Home <span class="sr-
only">(current)</span></a>
    </li>
    <li class="nav-item">
      <a class="nav-link waves-effect waves-light">Features</a>
    </li>
    <li class="nav-item">
      <a class="nav-link waves-effect waves-light">Pricing</a>
    </li>
    <li class="nav-item btn-group">
      <a class="nav-link dropdown-toggle waves-effect waves-light"
id="dropdownMenu1" data-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">Dropdown</a>
      <div class="dropdown-menu" aria-labelledby="dropdownMenu1">
        <a class="dropdown-item waves-effect waves-light">Action</a>
        <a class="dropdown-item waves-effect waves-light">Another action</a>
        <a class="dropdown-item waves-effect waves-light">Something else here</a>
      </div>
    </li>
  </ul>
  <form class="form-inline waves-effect waves-light">
    <input class="form-control" placeholder="Search" type="text">
  </form>
</div>
</div>
</nav>
<!--/.Navbar-->
</header>

```

```

<main>

    <?php echo $this->Flash->render(); ?>

    <?php echo $this->fetch('content'); ?>

</main>


<footer class="page-footer blue center-on-small-only">

    <!--Footer Links-->

    <div class="container-fluid">

        <div class="row">


            <!--First column-->

            <div class="col-md-6">

                <h5 class="title">Footer Content</h5>

                <p>Here you can use rows and columns here to organize your footer content.</p>

            </div>

            <!--/.First column-->


            <!--Second column-->

            <div class="col-md-6">

                <h5 class="title">Links</h5>

                <ul>

                    <li><a href="#">Link 1</a></li>

                    <li><a href="#">Link 2</a></li>

                    <li><a href="#">Link 3</a></li>

                    <li><a href="#">Link 4</a></li>

                </ul>

            </div>

            <!--/.Second column-->

        </div>

    </div>

```

```

<!--/.Footer Links-->

<!--Copyright-->
<div class="footer-copyright">
    <div class="container-fluid">
        © 2015 Copyright: <a href="https://mdbootstrap.com"> MDBootstrap.com </a>
    </div>
</div>
<!--/.Copyright-->
</footer>

<!-- SCRIPTS -->
<!-- Bootstrap tooltips -->
<?php echo $this->Html->script('/node_modules/mdbootstrap/js/tether.min.js'); ?>
<!-- MDB core JavaScript -->
<?php echo $this->Html->script('/node_modules/mdbootstrap/js/mdb.min.js'); ?>

<!--toastr modern notification-->
<?php echo $this->Html->script('/node_modules/toastr/build/toastr.min.js'); ?>

<!--Load initial for infinite-scroll-->
<?php echo $this->Html->script('/node_modules/infinite-scroll/dist/infinite-
scroll.pkgd.min.js');?>
<!--EEN SCRIPT SECTION-->
</body>
</html>

```

Path Routing:

Open config/route/routes.php

Fine and change:

```
$routes->connect('/', ['controller' => 'Pages', 'action' => 'display', 'home']);
```

To:

```
$routes->connect('/', ['controller' => 'Users', 'action' => 'login']);
```

How to Authentication / Authorize (Login / Logout / Permission / ACL):

Adding this below to src/Model/Entity/User.php

```
use Cake\Auth\DefaultPasswordHasher;
```

Adding Password Hashing:

```
/**
 *
 * Function trigger save database
 * @author sarawutt.b
 * @param type $password
 * @return type
 */
protected function _setPassword($password) {
    if (strlen($password) > 0) {
        return (new DefaultPasswordHasher)->hash($password);
    }
}
```

Add this below to src/Controller/UsersController.php

```

/**
 *
 * Function initialize make for automatically trigger when contructure
 */
public function initialize() {
    parent::initialize();
    $this->Auth->allow('add', 'logout');
}

/**
 *
 * Function login
 * @author sarawutt.b
 * @return void redirect to home page if pass authentication
 */
public function login() {
    $this->viewBuilder()->layout('signin');
    if ($this->request->is('post')) {
        $user = $this->Auth->identify();
        if ($user) {
            $this->Auth->setUser($user);
            return $this->redirect($this->Auth->redirectUrl());
        }

        $this->Flash->error(__('Invalid username or password Please try again!'));
    }
}
/**

```



```

*
* Function Logout
* @author sarawutt.b
* @return void redirect to login page
*/
public function logout() {
    $this->Flash->success('You are now logged out. ');
    return $this->redirect($this->Auth->logout());
}

```

Create file src/Template/Users/signin.ctp and adding for the content:

```

<div class="container">
    <div class="row">
        <div class="col-md-offset-3 col-md-6 mt-4">

            <?php echo $this->Form->create(); ?>

            <div class="card">
                <div class="card-block">

                    <!--Header-->
                    <div class="form-header purple darken-4">
                        <h3><i class="fa fa-lock"></i> Login:</h3>
                    </div>

                    <!--Body-->
                    <div class="md-form">
                        <i class="fa fa-envelope prefix"></i>
                        <input type="text" id="form2" class="form-control" name="username">
                        <label for="form2">Your email</label>

```

```

</div>

<div class="md-form">
    <i class="fa fa-lock prefix"></i>
    <input type="password" id="form4" class="form-control" name="password">
    <label for="form4">Your password</label>
</div>

<?php echo $this->Form->button(__('Login'), ['class' => 'btn-deep-purple waves-effect
waves-light pull-right']); ?>
</div>

<!--Footer-->
<div class="modal-footer">
    <div class="options">
        <p>Not a member? <a href="/Users/add"><?php echo __('Sign Up'); ?></a></p>
        <p>Forgot <a href="/Users/forgotPassword"><?php echo __('Password ?');
?></a></p>
    </div>
</div>
</div>
<?php echo $this->Form->end(); ?>

</div>
</div>
</div>

```

Add this below line to src/Controller/AppController.php:

```

/**
 *
 * Initialization hook method.
 * Use this method to add common initialization code like loading components.
 * e.g. `$this->loadComponent('Security');`
 * @return void
 */
public function initialize() {
    parent::initialize();

    $this->loadComponent('RequestHandler');
    $this->loadComponent('Flash');
    $this->loadComponent('Auth', [
        'loginRedirect' => ['controller' => 'Users', 'action' => 'index'],
        'logoutRedirect' => ['controller' => 'Users', 'action' => 'login'],
        'authenticate' => [
            'Form' => ['fields' => ['username' => 'username', 'password' => 'password']]
        ],
        'loginAction' => ['controller' => 'Users', 'action' => 'login'],
        'authorize' => ['Controller'],
        'unauthorizedRedirect' => $this->referer()// If unauthorized, return them to page they were
just on
    ]);
}

/**

```

```

*

* Function beforeFilter trigger function catching automatically
* @author sarawutt.b
* @param Event $event
*/

public function beforeFilter(Event $event) {
    parent::beforeFilter($event);
    $this->Auth->allow(['login', 'logout']);
}

/**
*
* Function check authorize
* @author sarawutt.b
* @param type $user
* @return boolean
*/

public function isAuthorized($user) {
    return true;
}

```

UsersController.php

```

public function index() {
    $this->paginate = [
        'contain' => ['Faculties', 'Roles', 'NamePrefixes']
    ];
    $users = $this->paginate($this->Users);
    $this->set(compact('users'));
    $this->set('_serialize', ['users']);
}

```

src/Template/Users/index.ctp:

```

<div class="container">
    <div class="row">
        <div class="users index large-9 medium-8 columns content">
            <h3><?php echo __('Users') ?></h3>
            <table cellpadding="0" cellspacing="0" class="table">
                <thead>
                    <tr>
                        <th scope="col"><?php echo $this->Paginator->sort('faculty_id') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('role_id') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('username') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('first_name') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('last_name') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('email') ?></th>
                        <th scope="col"><?php echo $this->Paginator->sort('status') ?></th>
                        <th scope="col" class="actions"><?php echo __('Actions') ?></th>
                    </tr>
                </thead>
                <tbody>

```

```

<?php foreach ($users as $user): ?>

    <tr>

        <td><?php echo $user->has('faculty') ? $this->Html->link($user->faculty->name,
['controller' => 'Faculties', 'action' => 'view', $user->faculty->id]) : '' ?></td>

        <td><?php echo $user->has('role') ? $this->Html->link($user->role->name,
['controller' => 'Roles', 'action' => 'view', $user->role->id]) : '' ?></td>

        <td><?php echo h($user->username) ?></td>

        <td><?php echo h($user->first_name) ?></td>

        <td><?php echo h($user->last_name) ?></td>

        <td><?php echo h($user->email) ?></td>

        <td><?php echo h($user->status) ?></td>

        <td class="actions">

            <?php $this->Html->templates(['icon' => '<i class="fa fa-
{{type}} {{attrs.class}}"{{attrs}}></i>'];?>

            <?php echo $this->Html->link($this->Html->icon('search') . ' ' . __('View'), ['action'
=> 'view', $user->id], ['class' => 'btn btn-sm btn-info btn-rounded waves-effect waves-light', 'escape'
=> false]) ?>

            <?php echo $this->Html->link($this->Html->icon('pencil') . ' ' . __('Edit'), ['action' =>
'edit', $user->id], array('class' => 'btn btn-sm btn-warning btn-rounded waves-effect waves-
light', 'escape' => false)) ?>

            <?php echo $this->Form->postLink($this->Html->icon('trash') . ' ' . __('Delete'),
['action' => 'delete', $user->id], ['confirm' => __('Are you sure you want to delete # {0}?', $user->id),
'class' => 'btn btn-sm btn-danger btn-rounded waves-effect waves-light confirmModal action-
delete', 'escape' => false]) ?>

        </td>

    </tr>

<?php endforeach; ?>

</tbody>

</table>

<?php echo $this->element('common/paginator'); ?>

</div>

</div>

</div>

```

Upload File:

Open src/Controller/AppController.php and then adding three below method

```
/**
 *
 * This function for upload attachment excel file from view of information style list
 * @author sarawutt.b
 * @param string name of target upload path
 * @param array() file attribute option from upload form
 * @since 2017/10/30
 * @return array()
 */
function uploadFiles($folder, $file, $itemId = null) {
    $folder_url = WWW_ROOT . $folder;
    $rel_url = $folder;

    if (!is_dir($folder_url)) {
        mkdir($folder_url);
    }

    //Bould new path if $itemId to be not null
    if ($itemId) {
        $folder_url = WWW_ROOT . $folder . '/' . $itemId;
        $rel_url = $folder . '/' . $itemId;
        if (!is_dir($folder_url)) {
            mkdir($folder_url);
        }
    }

    //define for file type where it allow to upload
    $map = array(
        'image/gif' => '.gif',
```

```

'image/jpeg' => '.jpg',
'image/png' => '.png',
'application/pdf' => '.pdf',
'application/msword' => '.doc',
'application/vnd.openxmlformats-officedocument.wordprocessingml.document' => '.docx',
'application/excel' => '.xls',
'application/vnd.ms-excel' => '.xls',
'application/vnd.openxmlformats-officedocument.spreadsheetml.sheet' => '.xlsx'
);

//Bould file extension keep to the database
$userfile_extn = substr($file['name'], strrpos($file['name'], '.') + 1);
if (array_key_exists($file['type'], $map)) {
    $typeOK = true;
}

//Rename for the file if not change of the upload file makbe duplicate
$filename = $this->VERSION() . $map[$file['type']];
if ($typeOK) {
    switch ($file['error']) {
        case 0:
            @unlink($folder_url . '/' . $filename); //Delete the file if it already existing
            $full_url = $folder_url . '/' . $filename;
            $url = $rel_url . '/' . $filename;
            $success = move_uploaded_file($file['tmp_name'], $url);
            if ($success) {
                $result['urls'][] = '/' . $url;
                $result['upfilename'][] = $filename;
                $result['ext'][] = $userfile_extn;
                $result['origin_name'][] = $file['name'];
            }
        }
    }
}

```



```

        $result['type'][] = $file['type'];
    } else {
        $result['errors'][] = __("Error uploaded {$filename}. Please try again.");
    }

    break;

case 3:
    $result['errors'][] = __("Error uploading {$filename}. Please try again.");

    break;

case 4:
    $result['nofiles'][] = __("No file Selected");

    break;

default:
    $result['errors'][] = __("System error uploading {$filename}. Contact webmaster.");

    break;
}

} else {
    $permiss = "";

    foreach ($map as $k => $v) {
        $permiss .= "{$v}, ";
    }

    $result['errors'][] = __("{$filename} cannot be uploaded. Acceptable file types in : %s",
trim($permiss, ','));
}

return $result;
}

```

```
/**
```

```

*

* Function used fro generate _VERSION_

* @author sarawutt.b

* @return biginteger of the version number

*/

public function VERSION() {
    $parts = explode(' ', microtime());
    $micro = $parts[0] * 1000000;
    return(substr(date('YmdHis'), 2) . sprintf("%06d", $micro));
}

/**
*
* Function used for generate UUID key patern
* @author sarawutt.b
* @return string uuid in version
*/

function UUID() {
    return sprintf('%04x%04x-%04x-%04x-%04x%04x%04x', mt_rand(0, 0xffff), mt_rand(0,
0xffff), mt_rand(0, 0xffff), mt_rand(0, 0xffff) | 0x4000, mt_rand(0, 0x3fff) | 0x8000, mt_rand(0,
0xffff), mt_rand(0, 0xffff), mt_rand(0, 0xffff));
}

```

References:

GIT Repository for our demo

https://github.com/dkantikorn/DEMO_CAKEPHP3_CRUD

Clone for our repository

```
$ cd D:/wamp/www
```

```
$ git clone https://github.com/dkantikorn/DEMO\_CAKEPHP3\_CRUD.git
```

CakePHP3.x Document

CakePHP3.x Basic Connection

<https://book.cakephp.org/3.0/en/orm/database-basics.html>

CakePHP3.x Database Relation and Associations

<https://book.cakephp.org/3.0/en/orm/associations.html>

CakePHP3.x Database Query Builder

<https://book.cakephp.org/3.0/en/orm/query-builder.html>

CakePHP3.x Database Receiving Data

<https://book.cakephp.org/3.0/en/orm/retrieving-data-and-resultsets.html>

CakePHP3.x Baking Console

<https://book.cakephp.org/3.0/en/bake/usage.html>

Themes:

<https://mdbootstrap.com/components/>

Netbean Plugin:

<https://github.com/junichi11/cakephp3-netbeans>

<https://github.com/junichi11/cakephp3-netbeans/releases>

::END::