CODE THE PIXEL

CakePHP 4 Authentication Using Auth Component (Advance) [Deprecated]

Asyraf Wahi Anuar - May 14, 2020
https://codethepixel.com/CakePHP-4-Authentication-Using-Auth-Component-Advance-Deprecated

This tutorial will show how to create an authentication using the Auth component for CakePHP 4. It includes the process of password hashing, registration with email verification, multi-field username/email login, forgot password, reset password and status management.-

Database scheme used in this tutorial:

```
CREATE TABLE `users` (
  `id` int(11) NOT NULL,
  `fullname` varchar(50) NOT NULL,
  `username` varchar(20) NOT NULL,
  `password` varchar(255) NOT NULL,
  `email` varchar(50) NOT NULL,
  `role` varchar(50) NOT NULL,
  `token` varchar(255) NOT NULL,
  `status` varchar(1) NOT NULL DEFAULT '0',
  `verified` int(1) NOT NULL DEFAULT 0,
  `created` datetime NOT NULL,
  `modified` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

ALTER TABLE `users`
  ADD PRIMARY KEY (`id`);

ALTER TABLE `users`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
COMMIT;
```

Authentication process flow:

Register a new account -> Receive account verification link via registered email -> Account activated

Request forgot password using email -> System will send reset password link -> Open link and input new password -> Login with the new password

Load Auth Component

File location: ...\src\Controller\AppController.php
Add the following codes to load the auth component in public function initialize as shown below.

```
public function initialize(): void
{
 parent::initialize();

 $this->loadComponent('RequestHandler');
 $this->loadComponent('Flash');
 $this->loadComponent('Auth', [ //load Auth component
  'authenticate' => [
   'Form' => [
    'finder' => 'auth'
   ]
  ],
 ]);
 //Allow non-auth page
 $this->Auth->allow(['login','add','forgotpassword','resetpassword','verificat
 //To call current auth fullname
 $this->set('fullname',$this->Auth->user('fullname'));
```

Email Transport Configuration

File location: ...\config\app_local.php
Set your email account as follows. This email configuration will be used to send emails to the new registered user and password reset requests.

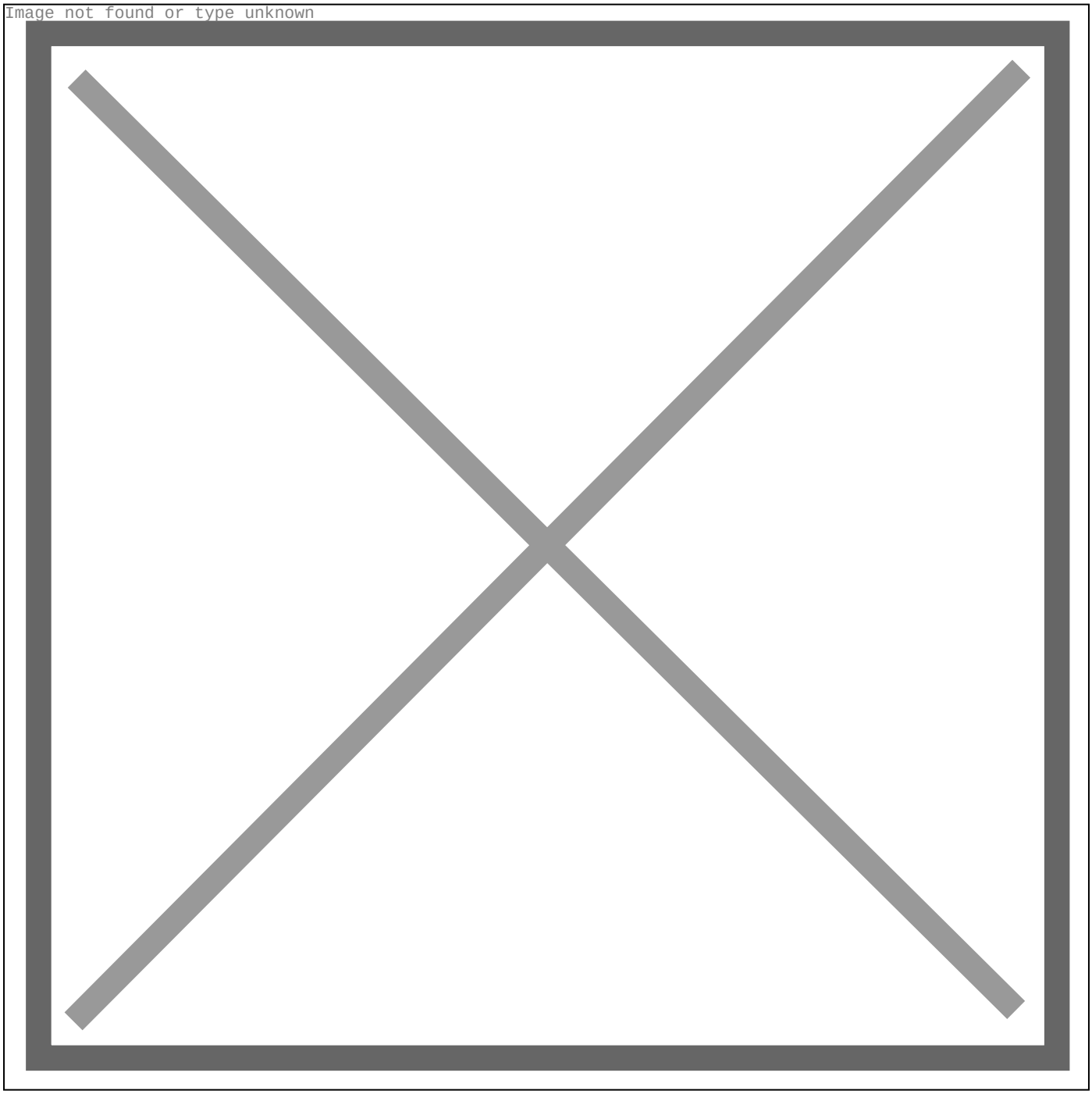
```
'EmailTransport' => [
 'smtp' => [
  'host' => 'YourHostName', //eg: codethepixel.com
  'port' => 26,
  'username' => 'EmailAddress', //eg: admin[at]codethepixel.com
  'password' => 'Secret', //email password
  'className' => 'Smtp',
  'client' => null,
  'url' => env('EMAIL_TRANSPORT_DEFAULT_URL', null),
 ],
],
```

or you can use a Gmail account. However, it required you to allow less secure app access in your Google account setting to allow your Gmail to be accessed from your localhost. Refer to the Google announcement here.-

```
'EmailTransport' => [
 'gmail' => [
  'host' => 'smtp.gmail.com',
  'port' => 587,
  'username' => 'YourGmailAddress', //eg: sample[at]gmail.com
  'password' => 'Secret', //email password
  'className' => 'Smtp',
```

```
  'tls' => true,
  'client' => null,
  'url' => env('EMAIL_TRANSPORT_DEFAULT_URL', null),
 ],
],
```

Image not found or type unknown



Configure Handler

File location: ...\src\Controller\UsersController.php

The aim of this authentication tutorial is to register a new user, validate the registration using registered email, activate the account, authenticate (login), logout, forgot password, reset password and activate/deactivate an account. To hash the password input and use the email

component, load the codes as follow at the beginning of the file:

```
use Cake\Mailer\Email;
use Cake\Mailer\Mailer;
use Cake\Mailer\TransportFactory;
use Cake\Auth\DefaultPasswordHasher;
use Cake\Utility\Security;
use Cake\ORM\TableRegistry;
```

Public Function Add

File location: ...\src\Controller\UsersController.php

Inside the class UsersController extends AppController { you need to create a few public functions to satisfy all the aforementioned features in the authentication. First, modify the public function add as follows (It also recommend to change the add to register) to hash the password input, create registration token, set the status and verified value and email the verification link. By default, the database should accept the registration with status==0 and verified==0.

```
public function add()
{
 $user = $this->Users->newEmptyEntity();

 if($this->request->is('post')){
  $userTable = TableRegistry::get('Users');

  $hasher = new DefaultPasswordHasher();
  $fullname = $this->request->getData('fullname');
  $email = $this->request->getData('email');
  $password = $this->request->getData('password');
  $token = Security::hash(Security::randomBytes(32));

  $user = $userTable->newEntity($this->request->getData());

  if($userTable->save($user)){
   $user->fullname = $fullname;
   $user->email = $email;
   $user->password = $hasher->hash($password);
   $user->token = $token;
   $user->status = '0';
   $user->verified = '0';

   $this->Flash->success(__('The user has been registered.'));
   $mailer = new Mailer('default');
   $mailer->setTransport('smtp'); //your email configuration name
   $mailer->setFrom(['noreply[at]codethepixel.com' => 'myCake4'])
   ->setTo($email)
   ->setEmailFormat('html')
   ->setSubject('Verify New Account')
   ->deliver('Hi <br/>Please confirm your email link below<br/><a href="http:/

   return $this->redirect(['action' => 'index']);
  }
  else
```

```
  {
   $this->Flash->error(__('Registration failed, please try again.'));
  }
 }
 $this->set(compact('user'));
}
```

Public Function Verification

File location: ...\src\Controller\UsersController.php

Once the registration was successful, the user will receive an email from the system that contains the verification link. If the user tries to log in, the system will reject because the account is not activated yet. Using the verification link, the user account will be activated. The public function verification will be used to execute this process. This function will find the account based on token and activate the registered account by changing the verified and status == 1.

```
public function verification($token)
{
 $userTable = TableRegistry::get('Users');
 $verify = $userTable->find('all')->where(['token'=>$token])->first();
 $verify->verified = '1';
 $verify->status = '1';
 $userTable->save($verify);
 $this->Flash->success(__('Your email has been verified, and please login now.
 return $this->redirect(['controller' => 'Users', 'action' => 'login']);
}
```

Public Function Login

File location: ...\src\Controller\UsersController.php

Next, create the login and logout function. When the system received a username and password, it will authenticate the user based on the username and password. It also includes the user status check. If the user account status and verification is equal to (==) 0, the user is unable to log in. The only user with status == 1 (active) can log in to the system.

```
public function login()
{
 if($this->request->is('post')){
  $user = $this->Auth->identify();
  if($user){
   $this->Auth->setUser($user);
   if($user['status'] == 0)
   {
    $this->Flash->error("Sorry, your account is inactive.");
    return $this->redirect(['controller' => 'Users', 'action' => 'logout']);
   }
   if($user['verified'] == 0)
   {
    $this->Flash->error("Sorry, your account is not verified.");
    return $this->redirect(['controller' => 'Users', 'action' => 'logout']);
```

```
   }
   return $this->redirect(['controller'=>'Users','action'=>'index']);
  }else {
   $this->Flash->error("Incorrect username or password !");
  }
 }
}

public function logout(){
 return $this->redirect($this->Auth->logout());
}
```

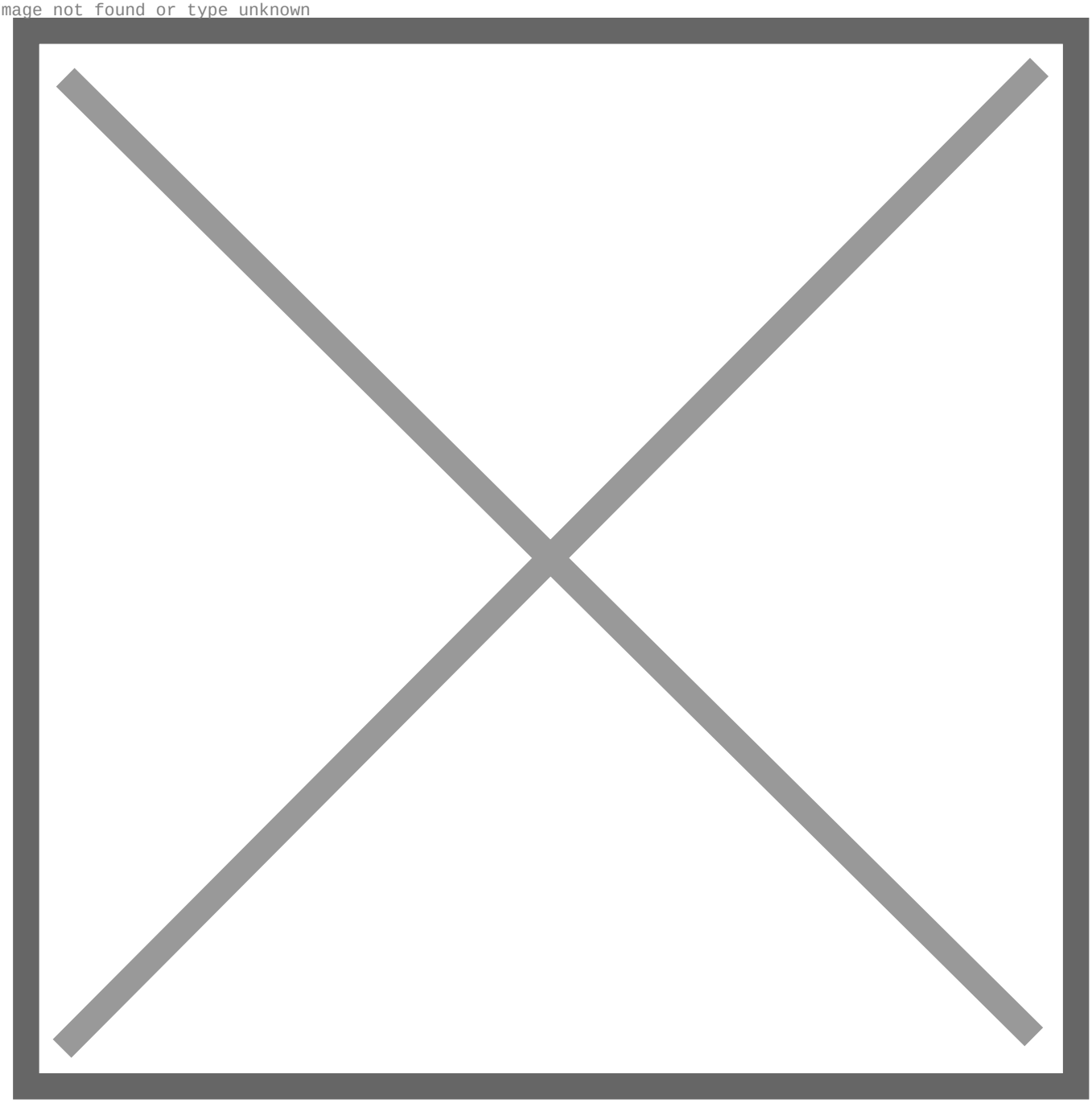
Login Page

File location: ...\templates\Users\login.php

Create the login form in login.php (create a new file) which contain the following codes:

```
<?php echo $this->Flash->render() ?>
<?= $this->Form->create() ?>
<?= $this->Form->control('username'); ?>
<?= $this->Form->control('password'); ?>
<?= $this->Form->submit() ?>
<?= $this->Form->end() ?>
<!-- button for new account registration-->
<?php echo $this->Html->link('Register',['controller' => 'users', 'action' =>
```

Image not found or type unknown



Public Function Edit

File location: ...\src\Controller\UsersController.php

At this phase, you should be able to register a new account with the hashed password that will be stored in your database and able to login into the system once it is activated. Don't forget to modify your edit function to hash the new received password.

```
public function edit($id = null)
{
 $user = $this->Users->get($id, [
  'contain' => [],
 ]);
 if ($this->request->is(['patch', 'post', 'put'])) {
```

```
  $user = $this->Users->patchEntity($user, $this->request->getData());
  $hasher = new DefaultPasswordHasher();
  $mypass = $this->request->getData('password');
  $user->password = $hasher->hash($mypass);
  if ($this->Users->save($user)) {

   $this->Flash->success(__('The user has been saved.'));

   return $this->redirect(['action' => 'index']);
  }
  $this->Flash->error(__('The user could not be saved. Please, try again.'));
 }
 $this->set(compact('user'));
}
```

Multi-field Authentication (username/email)

File location: ...\src\Model\Table\UsersTable.php

If you want to log in using username or password, simply add the following codes to UsersTable.php:

```
public function findAuth(Query $query, array $options)
{
    $query->where([
            'OR' => [
                'username' => $options['username'],
                'email' => $options['username']
            ]], [], true);
    return $query;
}
```

Forgot Password

File location: ...\src\Controller\UsersController.php

To create the forgot password function, add the following code in the user controller. The function will request email and check, if the email field is empty or not found, it will flash an error and if it found the email address registered in the system, it will proceed to create a token and send a password reset link to the requested email.

```
public function forgotpassword()
{
if ($this->request->is('post')) {
  $email = $this->request->getData('email');
  $token = Security::hash(Security::randomBytes(25));

  $userTable = TableRegistry::get('Users');
   if ($email == NULL) {
    $this->Flash->error(__('Please insert your email address'));
   }
   if ($user = $userTable->find('all')->where(['email'=>$email])->first()) {
    $user->token = $token;
```

```
    if ($userTable->save($user)){
     $mailer = new Mailer('default');
     $mailer->setTransport('smtp');
     $mailer->setFrom(['noreply[at]codethepixel.com' => 'myCake4'])
     ->setTo($email)
     ->setEmailFormat('html')
     ->setSubject('Forgot Password Request')
     ->deliver('Hello<br/>Please click link below to reset your password<br/><
    }
    $this->Flash->success('Reset password link has been sent to your email ('.
   }
   if ($total = $userTable->find('all')->where(['email'=>$email])->count()==0)
    $this->Flash->error(__('Email is not registered in system'));
   }
 }
}
```

Forgot Password Request Page

File location: ...\templates\Users\forgotpassword.php

Create forgotpassword.php that contain the following codes to request email.

```
<?php echo $this->Flash->render() ?>
<?= $this->Form->create() ?>
<?= $this->Form->control('email'); ?>
<?= $this->Form->submit() ?>
<?= $this->Form->end() ?>
```

Reset Password

File location: ...\src\Controller\UsersController.php

Once the reset password link sends to the requested email, the user needs to click and input the new password. To reset the password, create the public function reset password as follows. This code will find the registered account based on the token and hash the new password and save it into the database.

```
public function resetpassword($token)
{
 if($this->request->is('post')){
  $hasher = new DefaultPasswordHasher();
  $newPass = $hasher->hash($this->request->getData('password'));

  $userTable = TableRegistry::get('Users');
  $user = $userTable->find('all')->where(['token'=>$token])->first();
  $user->password = $newPass;
  if ($userTable->save($user)) {
   $this->Flash->success('Password successfully reset. Please login using your
   return $this->redirect(['action'=>'login']);
  }
 }
}
```

Reset Password Page

File location: ...\templates\Users\resetpassword.php

Create resetpassword.php that contain the following codes to request a new password.

```
<?php echo $this->Flash->render() ?>
<?= $this->Form->create() ?>
<?= $this->Form->control('password'); ?>
<?= $this->Form->submit() ?>
<?= $this->Form->end() ?>
```

At this phase, you should be able to request forgot password and reset your password with your new password.

User Status

File location: ...\src\Controller\UsersController.php

Next is to manage the user account status (active or inactive). Active = 1 and Inactive = 0. Create the public function to change the status:

```
public function userStatus($id=null,$status)
{
 $this->request->allowMethod(['post']);
 $user = $this->Users->get($id);

 if($status == 1 )
  $user->status = 0;
 else
  $user->status = 1;

 if($this->Users->save($user))
 {
  $this->Flash->success(__('The users status has changed.'));
 }
 return $this->redirect(['action' => 'index']);
}
```

Activation/Deactivation Button

File location: ...\templates\Users\index.php

Create the button to manage the account activation and deactivation process. I generate the button on the index page as follows:

```
<?php if ($user->status == 1) : ?>
 <?= $this->Form->postLink(__('Deactivate'), ['action' => 'userStatus', $user
<?php else : ?>
 <?= $this->Form->postLink(__('Activate'), ['action' => 'userStatus', $user->i
<?php endif; ?>
```

Print Current Auth Fullname

Remember the $this->set('fullname',$this->Auth->user('fullname')); in the AppController.php ? That is used to read the current authenticated user information. To print the currently authenticated full name, simply add the following codes in any view.

```
Hello, <?= $fullname; ?>
```

If you need to print others current authenticated user information, simply add the following codes to the AppController.php (eg: role/status) and print using the same method as above.

```
$this->set('role',$this->Auth->user('role'));
$this->set('status',$this->Auth->user('status'));
```

Logout

The logout function has been integrated into the user controller after the login function. To log out, the user needs to be redirected to URL: myCake4/users/logout and it will call the public function logout and destroy the current session. To create a button to execute the logout process, add the following codes into any view.

```
<?php echo $this->Html->link('Logout',['controller' => 'users', 'action' => 'l
```

Now you've got the authentication with account verification, forgot password, reset password and account status management.

That all. Happy coding :)