Φόρμες , Δομή, Εμφάνιση και λίγες Βάσεις

Δομή

Θα πρέπει να έχουμε κατα νου ότι τα views τα οποία θα παράγουμε είναι html σελίδες. Οπότε θα πρέπει κάθε σελίδα της εφαρμογής μας να αποτελείται από Views τα οποία όταν συντεθούν θα παράγουν μια σωστά δομημένη html σελίδα, με header και body.

Μια καλή πρακτική είναι να έχουμε ένα view για:

- **header** όπου εκεί θα μαζεύουμε όλη τηνπληροφορία που θέλουμε να εμφανίζεται σε όλες τις σελίδες (θα δηλώνουμε τα css, τη jquery, άλλα javascripts πιθανότατα, τα css που θα χρησιμοποιήσουμε, ίσως ένα βασικό μενού της εφαρμογής μας κτλ)
- **footer** όπου θα βάζουμε κι εκεί ίσως κάποια javascripts και ότι άλλο θέλουμε να εμφανίζεται στο «κάτω μέρος» της εφαρμογής μας
- **views** τα οποία θα συνθέτουν μαζί με τα header και footer τις οθόνες της εφαρμογής μας

Δείτε εδώ πως γράφουμε σωστά κώδικα στο CodeIgniter.

Codeigniter Forms

Για να διαχειριστούμε φόρμες στο Codeigniter χρησιμοποιούμε το form helper.

Πρέπει να κάνουμε autoload τον form helper.

Προκειμένου να χρησιμοποιούμε κάποιες χρήσιμες συναρτήσεις που αφορούν τη διαχείριση των url πρέπει να κάνουμε autoload τον url helper.

Παράδειγμα:

Ας υποθέσουμε ότι θα έχουμε δομήσει την εφαρμογή μας ώστε να έχει ένα header, ένα footer και μια login_form.

- Δημιουργήστε ένα view με όνομα `header.php` και βάλτε τον εξής κώδικα:

```
<html>
    <head>
        <meta charset="UTF-8">
        <title>Welcome to CodeIgniter</title>
    </head>
    <body>
        <div id="header">
            <h1> Codeigniter First Tutorial </h1>
        </div>
```

- Δημιουργήστε ένα view με όνομα `footer.php` και βάλτε τον εξής κώδικα:

```
Page rendered in <strong>{elapsed_time}
</body>
</html>
```

Δημιουργήστε έναν controller, ο οποίος θα χειριστεί τα δεδομένα της φόρμας

Το view για τη φόρμα θα είναι:

```
<?php
echo validation_errors();
echo form_open('main/check');
```

```
$uname = array(
              'name'        => 'username',
              'id'          => 'username',
              'placeholder' => 'type your username',
              'maxlength'   => '100',
              'size'        => '50'
            );

echo form_label('Username','username');
echo form_input($uname);


//echo form_input('username');
//echo form_error('username');

$pass = array(
              'name'        => 'password',
              'type' => 'password',
              'id'          => 'password',
              'placeholder' => 'type your password',
              'maxlength'   => '100',
              'size'        => '50'
            );

echo form_label('Password','password');
echo form_input($pass);

echo form_submit('submit','submit');

echo form_close();
?>
```

Form Validation

Πρέπει να κάνουμε autoload το form_validation library

Για να κάνουμε form validation, πρέπει να έχουμε 3 πράγματα:

- ★ Ένα view το οποίο να περιέχει τη φόρμα
- ★ Ένα view το οποίο να περιέχει ένα success μήνυμα για την υποβολή της φόρμας
- ★ Έναν controller ο οποίος να λαμβάνει και να επεξεργάζεται τα υποβαλλόμενα δεδομένα

Για να εμφανίζουμε τα validation errors στη φόρμα, πρέπει στο view της φόρμας να έχουμε τη δήλωση:

```
echo validation_errors();
```

Στον controller ο έλεγχος για το αν η φόρμα είναι valid γίνεται με τον εξής κώδικα:

```
if ($this->form_validation->run() == FALSE)
            {
                  $this->load->view('myform');
            }
            else
            {
                  $this->load->view('formsuccess');
            }
```

Θέτοντας κανόνες για το validation

Για να θέσουμε κανόνες, χρησιμοποιούμε τη συνάρτηση
`$this->form_validation->set_rules();`

στην οποία καθορίζουμε 3 παραμέτρους:

- ➔ το όνομα του πεδίου της φόρμας
- ➔ ένα «ανθρώπινο» όνομα για το πεδίο, το οποίο θα περιληφθεί στο μήνυμα σφάλματος
- ➔ τους κανόνες [1]

παράδειγμα:

```
$this->form_validation->set_rules('username', 'Username',
'required|min_length[5]|max_length[12]|is_unique[users.username]');
$this->form_validation->set_rules('password', 'Password', 'required|matches[passconf]');
$this->form_validation->set_rules('passconf', 'Password Confirmation', 'required');
$this->form_validation->set_rules('email', 'Email', 'required|valid_email|is_unique[users.email]');
```

θέτοντας δικούς μας κανόνες

Για να το κάνουμε αυτό, πρέπει να δημιουργήσουμε μια δική μας συνάρτηση, της οποίας το όνομα να ξεκινάει με το callback_

[1] Για μια πλήρη λίστα με έτοιμους κανόνες δέιτε την ενότητα "Rule Reference" από το form_validation library

παράδειγμα:

```
$this->form_validation->set_rules('username', 'Username', 'callback_username_check');
….

public function username_check($str)
       {
              if ($str == 'test')
              {
                     $this->form_validation->set_message('username_check', 'The %s
field can not be the word "test"');
                     return FALSE;
              }
              else
              {
                     return TRUE;
              }
       }
```

θέτοντας δικά μας μηνύματα σφάλματος

Για να το κάνουμε αυτό, χτησιμοποιούμε τη συνάρτηση:

```
$this->form_validation->set_message('rule', 'Error Message');
```

παράδειγμα:

```
$this->form_validation->set_message('required', 'Your custom message here');
```

Εμφανίζοντας ανεξάρτητα μηνύματα σφάλματος

Η `validation_errors()` εμφανίζει όλα τα σφάλματα μαζί. Αν θέλουμε δίπλα σε κάποιο πεδίο να εμφανίζονται τα αντίστοιχα σφάλματα, θα πρέπει να χρησιμοποιήσουμε τη `form_error()`, παράδειγμα `form_error('username');`

Δημιουργήστε το μενού σας

Πηγαίνετε στο header.php και βάλτε αμέσως μετά το <body> δυο links τα οποία να δείχνουν:

- στην αρχική σας σελίδα
- στη σελίδα με τις πληροφορίες των χρηστών

Hint: Χρησιμοποιήστε τη συνάρτηση anchor() η οποία παρέχεται μέσω του url helper.

```
anchor('controller/function', 'Link Title', 'title="Title Page"')
```

Εμφάνιση

Μπορούμε να δώσουμε στυλ στην εφαρμογή μας χρησιμοποιώντας css, jquery και javascript. Μια καλή πρακτική είναι να δημιουργήσουμε ένα φάκελο στο επίπεδο του root folder του codeingiter (εκεί που είναι και ο application).

css

Θα πρέπει να χρησιμοποιήσουμε το header.php που φτιάξαμε πριν και να δηλώσουμε το css εκεί. Ας υποθέσουμε ότι έχουμε φτιάξει τον φάκελο css και το αρχείο main.css. Για να το συνδέσουμε θα βάλουμε στο <header> το εξής:

```
<link rel="stylesheet" href="<?php echo base_url(); ?>css/main.css">
```

Μπορείτε αν θέλετε να χρησιμοποιήσετε το bootstrap το οποίο είναι ένα front-end framework για responsive εφαρμογές (έχει css, javascripts και themes).

Προσοχή! Αν θέλετε να χρησιμοποιήσετε το bootstrap θα πρέπει να βάλετε στο header της html την jquery πριν τα scripts του bootstrap.

jquery

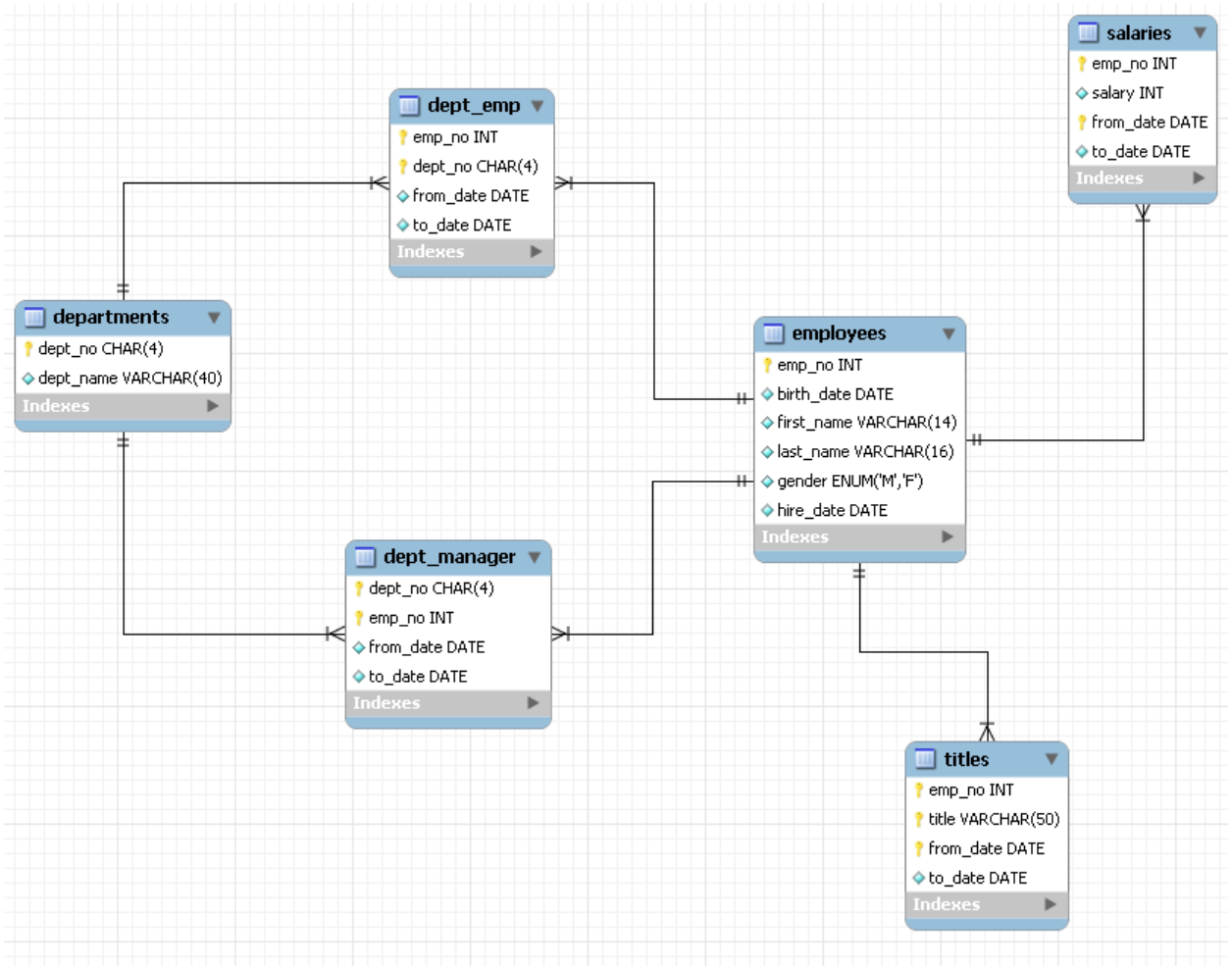
Για να χρησιμοποιήσουμε τη jquery στο codeigniter, την κατεβάζουμε και τη βάζουμε σε ένα φάκελο (π.χ. δημιουργώ έναν φάκελο themes/js και τη βάζω εκεί - στο επίπεδο του φακέλου

application)[2].

Οπότε τη δηλώνω ως εξής:

```
<script src="<?php echo base_url(); ?>themes/js/jquery-1.10.2.min.js"></script>
```

Βάσεις Δεδομένων



[2] ίσως χρειαστεί και το jquery-1.10.2.min.map

Υπάρχει μια sample βάση στον server 83.212.105.20

Μπορείτε να έχετε πρόσβαση σε αυτή είτε μέσω:

- PhpMyadmin στη διεύθυνση http://83.212.105.20/phpmyadmin/
- με το MySQL-Workbench
- με mysql-client από command-line

Τα στοιχεία σύνδεσης είναι:

```
Host: 83.212.105.20

Port: 3306

username: employee

database Name: employee

password: employ33
```

Προσοχή: Η συγκεκριμένη βάση έχει αρκετά δεδομένα. Βάλτε `limit` στις αναζητήσεις σας.

Σύνδεση Codeigniter με τη βάση δεδομένων

Στο αρχείο `application/config/database.php` βάζουμε τις παραμέτρους σύνδεσης με τη βάση.

Από τη στιγμή που θέλουμε να επικοινωνήσουμε από την εφαρμογή μας με τη βάση, θα πρέπει να κάνουμε load το database class (`$this->load->database();`). Αν θέλουμε να το έχουμε εξαρχής φορτωμένο, μπορούμε να το δηλωουμε στο application/config/autoload.php στο `$autoload['libraries'].`

Χρήσιμη κλάση είναι η Active Records Class που μας παρέχει ευκολίες για sql queries.

Ας δημιουργήσουμε ένα Model το οποίο να διαβάζει τα δεδομένα από τη βάση.

Δημιουργήστε ένα αρχείο με όνομα `user_model.php` μέσα στον κατάλογο `application/models`

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class User_model extends CI_Model {
```

```
    function __construct() {
        parent::__construct();
    }

    public function getusersinfo() {

        $query = $this->db->get('users');

        if ($query->num_rows() > 0) {
            $data = $query->result_array();
            return $data;
        } else {
            return FALSE;
        }
    }

}
```

Στη συνέχεια πηγαίνετε στο autoload και στο `$autoload['model']` προσθέστε το `user_model`.

```
$autoload['model'] = array('user_model');
```

Μετά δημιουργήστε μια function στον main controller η οποία να παίρνει τα δεδομένα από το Model και να καλεί ένα view το οποίο να εμφανίζει τα δεδομένα του πίνακα.

SQL κώδικας για τη δημιουργία ενός πίνακα και εισαγωγή τιμών σε αυτόν

```
CREATE TABLE IF NOT EXISTS `users` (
  `id` int(5) NOT NULL AUTO_INCREMENT,
  `username` varchar(65) COLLATE utf8_unicode_ci NOT NULL,
  `password` varchar(65) COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(30) COLLATE utf8_unicode_ci NOT NULL,
  `state` tinyint(1) NOT NULL,
  `code` varchar(5) COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`id`),
  KEY `username` (`username`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=42 ;

--
-- Άδειασμα δεδομένων του πίνακα `users`
--

INSERT INTO `users` (`username`, `password`, `email`, `state`, `code`) VALUES
('Jason', 'ffeerffdf', 'jason@hua.gr', 1, '98968'),
('Δημήτρης', 'rewrewvvsre', 'jim@hua.gr', 0, '01989'),
('Gas', 'ffdfefefefe', 'gas@hua.gr', 1, '98945');
```

Δημιουργία μιας συνάρτησης η οποία παίρνει δεδομένα από τη βάση και τα στέλνει σε ένα view (users.php).

```
public function get_users() {
       $data['users'] = $this->user_model->getusersinfo();
       $this->load->view('header');
       $this->load->view('users', $data);
       $this->load->view('footer');
       }
```

Δημιουργία ενός απλού View το οποίο παρουσιάζει τα δεδομένα τα οποία έρχονται από τον controller ο οποίος τα παίρνει από τη βάση.

```
<pre>
<?= var_dump($users);?>
</pre>
```

Άσκηση: Δείτε τι επιστρέφει η `$users` και τυπώστε τα πεδία του πίνακα ως εξής[3]:

```
id : 42

username : Jason

password : 712e4604fed93b06f72f8b6c419849ed

email : jason@hua.gr

state : 1

code : 98968
```

```
<?php foreach ($users as $user) {
   foreach ($user as $key=>$value) {
   echo $key . ' : '. $value . '<br/>';
   }
   echo '<hr>';
}
```

[3] Δείτε εδώ κι εδώ παραδείγματα για το πως τυπώνουμε arrays

```
?>
```

Active Records

Δείτε λεπτομερώς την Active Record class του CodeIgniter. Δείτε εδώ πως χειριζόμαστε τα αποτελέσματα από ένα query στη Βάση.

Selecting Data

```
$this->db->get('mytable');
```

είναι σαν

```
SELECT * FROM mytable
```

```
$query = $this->db->get('mytable', 10, 20);
```

παράγει το

```
SELECT * FROM mytable LIMIT 20, 10
```

Για να επιλέξουμε συγκεκριμένα δεδομένα χρησιμοποιούμε την get_where:

```
$query = $this->db->get_where('mytable', array('id' => $id), $limit, $offset);
```

Αν θέλουμε να δούμε το query που έτρεξε, χρησιμοποιούμε την **`$this->db->last_query();`**

Τυπώνοντας τα αποτελέσματα μιας βάσης[4]:

```
$query = $this->db->get('mytable');

foreach ($query->result() as $row)
```

[4] θεωρούμε ότι έχουμε έναν πίνακα mytable ο οποίος έχει μια στήλη title

```
{
    echo $row->title;
}
```

```
$query = $this->db->get('users');
```

Inserting Data

```
$data = array(
   'title' => 'My title' ,
   'name' => 'My Name' ,
   'date' => 'My date'
);

$this->db->insert('mytable', $data);

// Produces: INSERT INTO mytable (title, name, date) VALUES ('My title', 'My name', 'My date')
```

Updating Data

```
$data = array(
               'title' => $title,
               'name' => $name,
               'date' => $date
            );

$this->db->where('id', $id);
$this->db->update('mytable', $data);

// Produces:
// UPDATE mytable
// SET title = '{$title}', name = '{$name}', date = '{$date}'
// WHERE id = $id
```

Deleting Data

```
$this->db->delete('mytable', array('id' => $id));
```

```
// Produces:
// DELETE FROM mytable
// WHERE id = $id
```

Μπορούμε να σβήσουμε έναν πίνακα ως εξής:

```
$this->db->empty_table();
```

Transactions

Εδώ μπορείτε να δείτε τι κάνουμε στην περίπτωση που θέλουμε να κάνουμε Transcaction στη βάση, δηλαδή αν δεν τρέξουν όλα τα queries που θέλουμε να μη γίνει αλλλαγή στη βάση.

Άσκηση: αποθηκεύστε στη βάση τα πεδία της φόρμας που είχατε πριν.

 CodeIgniter : Database selecting Methods

Logging

Για να μπορέσουμε να κρατήσουμε στο Log διάφορα, προκειμένου να κάνουμε debugging θα πρέπει

- ★ να ορίσουμε στο αρχείο config/config.php το επίπεδο του log που θέλουμε:
  - 0 = Disables logging, Error logging TURNED OFF
  - 1 = Error Messages (including PHP errors)
  - 2 = Debug Messages
  - 3 = Informational Messages
  - 4 = All Messages
- ★ να δώσουμε δικαίωμα πρόσβασης στον χρήστη του apache web server, ο οποίος στο debian και στο ubuntu είναι ο www-data στον κατάλογο logs κάτω από τον κατάλογο applications. `chown www-data application/logs`

- ★ στο σημειο του κώδικα που θέλουμε να γράψουμε στο Log γράφουμε `log_message('type','message'),` παράδειγμα θέλουμε να τυπώσουμε το τελευταίο sql query Που έγινε στη βάση:

```
log_message('info', 'SQL '.  $this->db->last_query());
```

- ★ θα παρατηρήσουμε ότι κάτω από τον φάκελο logs θα έχει δημιουργηθεί ένα αρχείο με την τρέχουσα ημερομηνία σαν όνομα που έχει όλα τα μηνύματα που αποθηκεύονται στο Log[5].

Sessions

Για να δουλέψουμε με την session class, θα πρέπει να καλέσουμε τη session library[6]. Καλό είναι να τη δηλώσουμε στο autoload.php αρχείο.

Από τη στιγμή που δημιουργείται ένα session, μπορούμε να αποκτήσουμε πρόσβαση σε αυτό το αντικείμενο μέσω του $this->session.

Όταν καλείται μια σελίδα, η session class θα ελέγξει αν υπάρχουν έγκυρα δεδομένα συνόδου στο session cookie του χρήστη. Αν δεν υπάρχουν ή αν έχει λήξει η σύνοδος, μια νέα σύνοδος θα δημιουργηθεί και θα αποθηκευτεί στο cookie[7]. Αν η σύνοδος υπάρχει οι πληροφορίες ενημερώνονται κατάλληλα. Με κάθε update το session_id ξαναδημιουργείται

Παίρνοντας τα session δεδομένα

Αν θέλουμε να πάρουμε όλα τα δεδομένα του session, αυτό γίνεται ως εξής:

```
$this->session->all_userdata()
```

αν θέλουμε συγκεκριμένη μεταβλητή, τότε παίρνουμε την τιμή της ως εξής:

```
$this->session->userdata('item');
```

όπου item η session μεταβλητή.

Αποθηκεύοντας δεδομένα στο session

Για να αποθηκεύσουμε δεδομένα στο session, το κάνουμε ως εξής:

```
$this->session->set_userdata('some_name', 'some_value');
```

---

[5] Για να δημιουργηθεί το Log θα πρέπει ο web server user να έχει δικαίωμα εγγραφής στον κατάλογο application/logs (π.χ. στο debian/ubuntu αλλάζουμε τα δικαιώματα με `chown www-data:www-data logs`)
[6] Από τη στιγμή που κάνουμε load τη session class, το Codeigniter μας αναγκάζει να ορίσουμε encryption key στο config.php ώστε τα δεδομένα της συνόδου να κρυπτογραφούνται.
[7] το cookie ονομάζεται ci_session

ή μπορούμε πολλά μαζί, ως εξής:

```
$newdata = array(
                        'username'  => 'johndoe',
                        'email'     => 'johndoe@some-site.com',
                        'logged_in' => TRUE
                    );
      $this->session->set_userdata($newdata);
```

Σβήνοντας δεδομένα από το session

Για να «σβήσουμε» δεδομένα από το session, αυτό γίνεται ως εξής:

```
$this->session->unset_userdata('some_name');
```

Άσκηση : Αποθηκεύστε στο session την τιμή που διαβάζετε στο username στη φόρμα, που είδαμε προηγουμένως.

 Login mechanism with Codeigniter

Simple login with codeigniter
http://www.codefactorycr.com/login-with-codeigniter-php.html

Debugging στον server

Συνδεόμαστε με ssh ή με putty. Ο server έχει ip 62.217.125.30. Όνομα χρήστη: ellakuser, κωδικός: 311@k

Από terminal συνδεόμαστε ως εξής:

```
ssh ellakuser@62.217.125.30
```

Αφού συνδεθούμε, πηγαίνουμε στην εφαρμογή μας στον browser στη σελίδα που έχει πρόβλημα, πατάμε enter και μετά πηγαίνουμε στο terminal ή στο putty και πατάμε:

```
tail -f /var/log/apache2/error.log
```

έτσι βλέπουμε το αρχείο error.log και μας δείχνει real time τα τελευταία errors στον server.

Για να αποσυνδεθούμε πατάμε **Ctrl+C** για να βγούμε από το αρχείο και μετά exit.

Υλικό

Εδώ θα βρείτε το αρχείο με τον κώδικα που κάναμε μέχρι τώρα.
Προσοχή: Θα πρέπει να αλλάξατε τις ρυθμίσεις ώστε να βλέπει τη δική σας βάση.