View Component of a Web-based IDE to build Web Applications on CakePHP

December 13, 2010
Swathi Vegesna

Committee Members
Dr. Chris Pollett
Dr. Sami Khuri
Dr. T.Y.Lin

Agenda

- Goal
- Tools used
- Design
- Features
- Implementation
- Conclusion

Goal

Build the View Component of a Web-based IDE that enables users to create a Web application in PHP on the CakePHP framework

Motivation – Why Web-based IDE?

- Current available IDE’s
  - Desktop-Based
    - Needs software Installation
    - Maintenance costs (updating softwares)
    - No remote access to code
  - Lacks implementation of development Framework
    - No MVC architecture
    - No consistent Folder and File structure
    - No proper toolbar to build the Views

Web-Based IDE on CakePHP

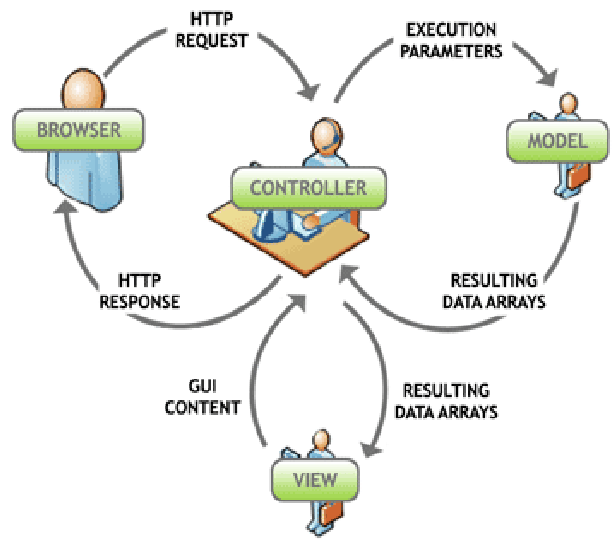
- Browser-based code development environment
- Provides Remote Access
- Reduces hardware costs and management overhead
- CakePHP based Web applications
- Consistent and Well organized
- Eases development of Web applications

Tools & Softwares

- Web Server : Apache
- Database: MySQL
- Language: PHP
- FrameWorks: CakePHP, jQuery
- Text Editor: CKEditor
- Debug Tool: FireBug
- Browsers: compatible with Firefox, IE 8, Chrome, Safari

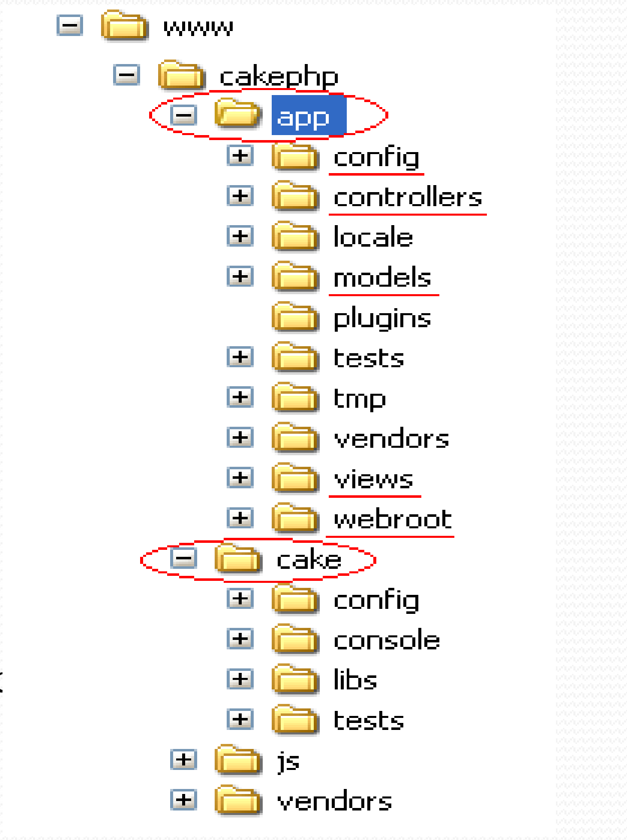
CakePHP

- One Among top 5 open source frameworks for PHP after Yii, Zend and CodeIgniter
- Design Patterns
  - Model-View-Controller
  - Object-Relational Mapping
  - Active Record pattern
  - Front Controller pattern
- Turns an application into a maintainable, modular and rapidly developed package.
- Application consistency and logicality



Building Web applications in CakePHP

- Folder Structure
  - Cake
  - app
    - Models
    - Controllers
    - Views
    - Webroot
    - Config
- Naming conventions
  - Filenames are underscored class names
  - Model class names should be singular and CamelCased.
  - Controller class names should be plural, CamelCased, an
    end in Controller.
  - Views are named after the controller functions they display (using underscores).



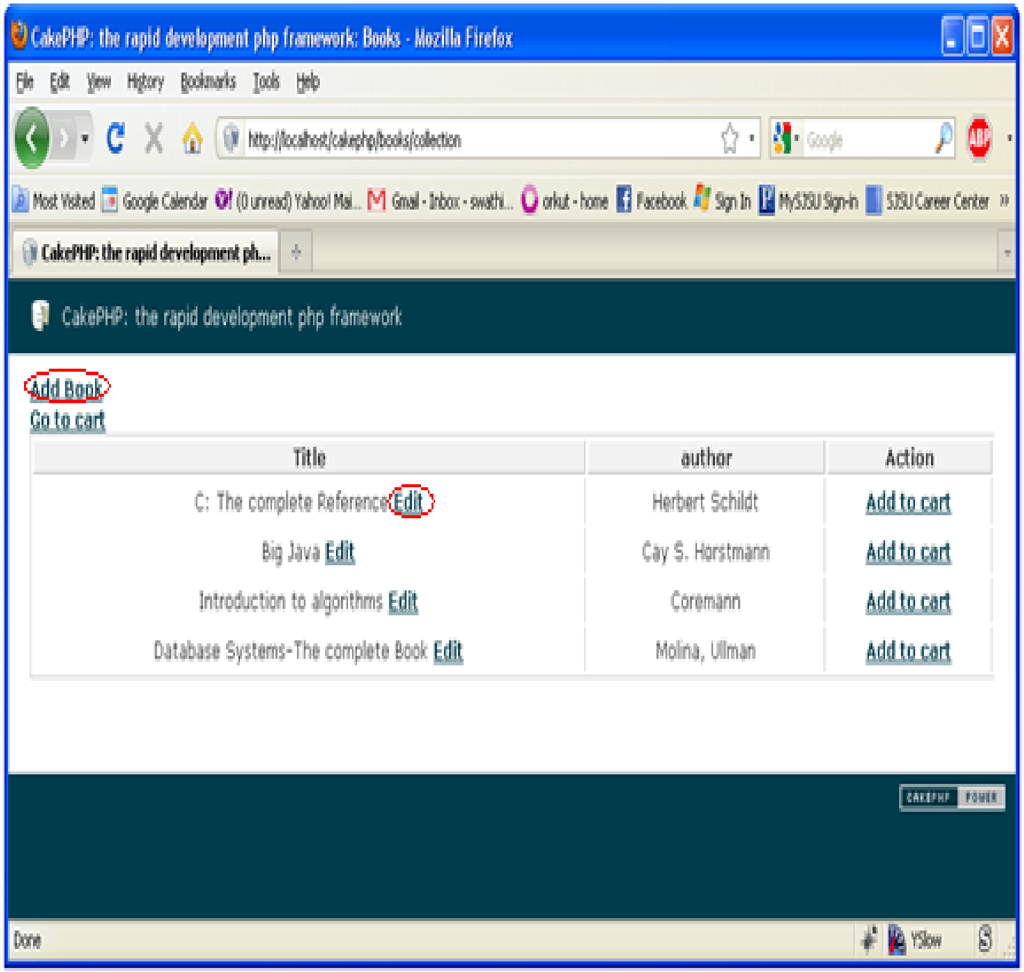
```
www
  cakephp
    app
      config
      controllers
      locale
      models
      plugins
      tests
      tmp
      vendors
      views
      webroot
    cake
      config
      console
      libs
      tests
    js
    vendors
```

Preliminary Work

- Book collection Website
  - Experiment with CakePHP to build a Web application.
- Performance Tests on JavaScript Frameworks
  - Select the best JavaScript framework.
- Draggable and Droppable classes
  - Experiment with jQuery library.
- Layout of the IDE
  - Build a prototype of the GUI.

Book collection Website

- Goal
  - Build model component
  - Get Model- Controller interactivity
  - Get View-Controller interactivity
- Web Pages Built
  - Home page
  - Registration page
  - Login page
  - Book collection page
  - Book Cart Page


CakePHP: the rapid development php framework

Add Book
Go to cart

Title	author	Action
C: The complete Reference Edit	Herbert Schildt	Add to cart
Big Java Edit	Cay S. Horstmann	Add to cart
Introduction to algorithms Edit	Coremann	Add to cart
Database Systems-The complete Book Edit	Molina, Ullman	Add to cart

Book Collection Page

Implementation

Model Class

```
class Book extends AppModel {
    var $name = 'Book';
}
```

Controller Class

```
class BooksController extends AppController {
      var $name = 'PartnerLocators';
      function index() {
        $this->set('books', $this->Book->find('all'));
      }
}
```

View

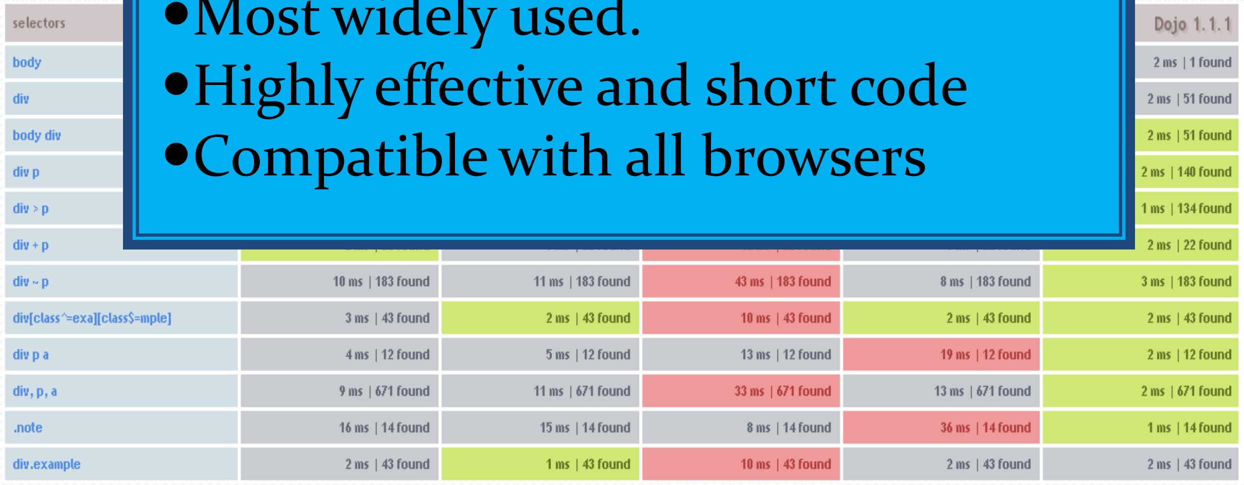
```
<?php foreach ($books as $book): ?>
        <tr>
                <td><?php echo $book[Book'][Title']; ?></td>
                <td> <?php echo $book['Book']['Author']; ?></td>
         </tr>
 <?php endforeach; ?>
```

Performance Tests on JavaScript Frameworks

Frameworks	Grade	Performance score	HTTP requests	Weight
jQuery	A	84	10	10.2K
YUI	B	76	14	97.2 K
DOJO				
Prototyp				

jQuery

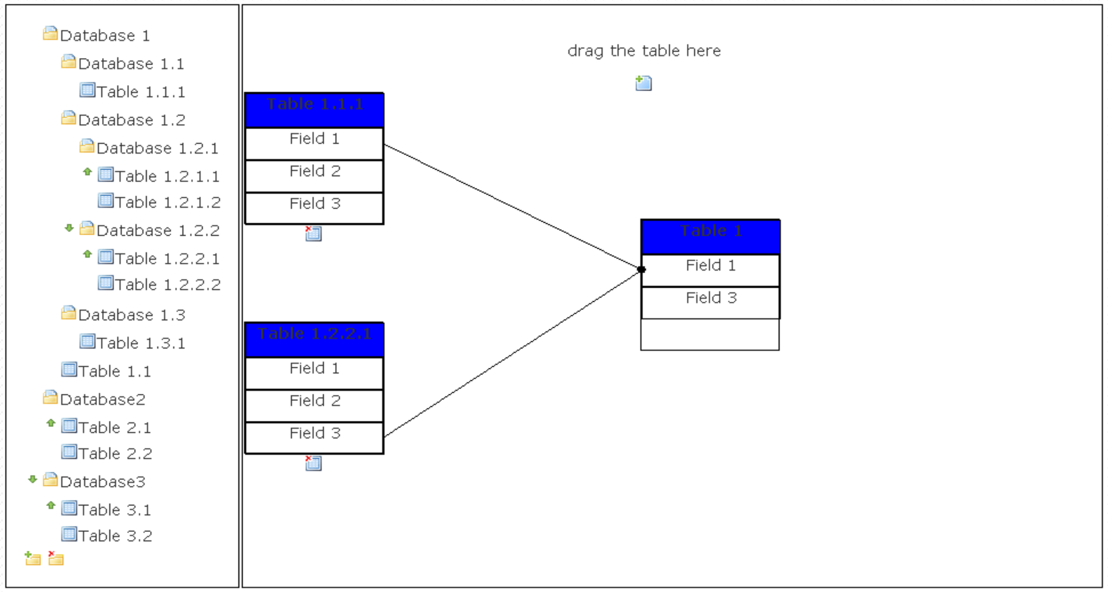
- Fast and less weight library
- Most widely used.
- Highly effective and short code
- Compatible with all browsers



selectors					Dojo 1.1.1
body					2 ms | 1 found
div					2 ms | 51 found
body div					2 ms | 51 found
div p					2 ms | 140 found
div > p					1 ms | 134 found
div + p					2 ms | 22 found
div ~ p	10 ms | 183 found	11 ms | 183 found	43 ms | 183 found	8 ms | 183 found	3 ms | 183 found
div[class^=exa][class$=mple]	3 ms | 43 found	2 ms | 43 found	10 ms | 43 found	2 ms | 43 found	2 ms | 43 found
div p a	4 ms | 12 found	5 ms | 12 found	13 ms | 12 found	19 ms | 12 found	2 ms | 12 found
div, p, a	9 ms | 671 found	11 ms | 671 found	33 ms | 671 found	13 ms | 671 found	2 ms | 671 found
.note	16 ms | 14 found	15 ms | 14 found	8 ms | 14 found	36 ms | 14 found	1 ms | 14 found
div.example	2 ms | 43 found	1 ms | 43 found	10 ms | 43 found	2 ms | 43 found	2 ms | 43 found

Slick Speed Test (Slick speed tool)

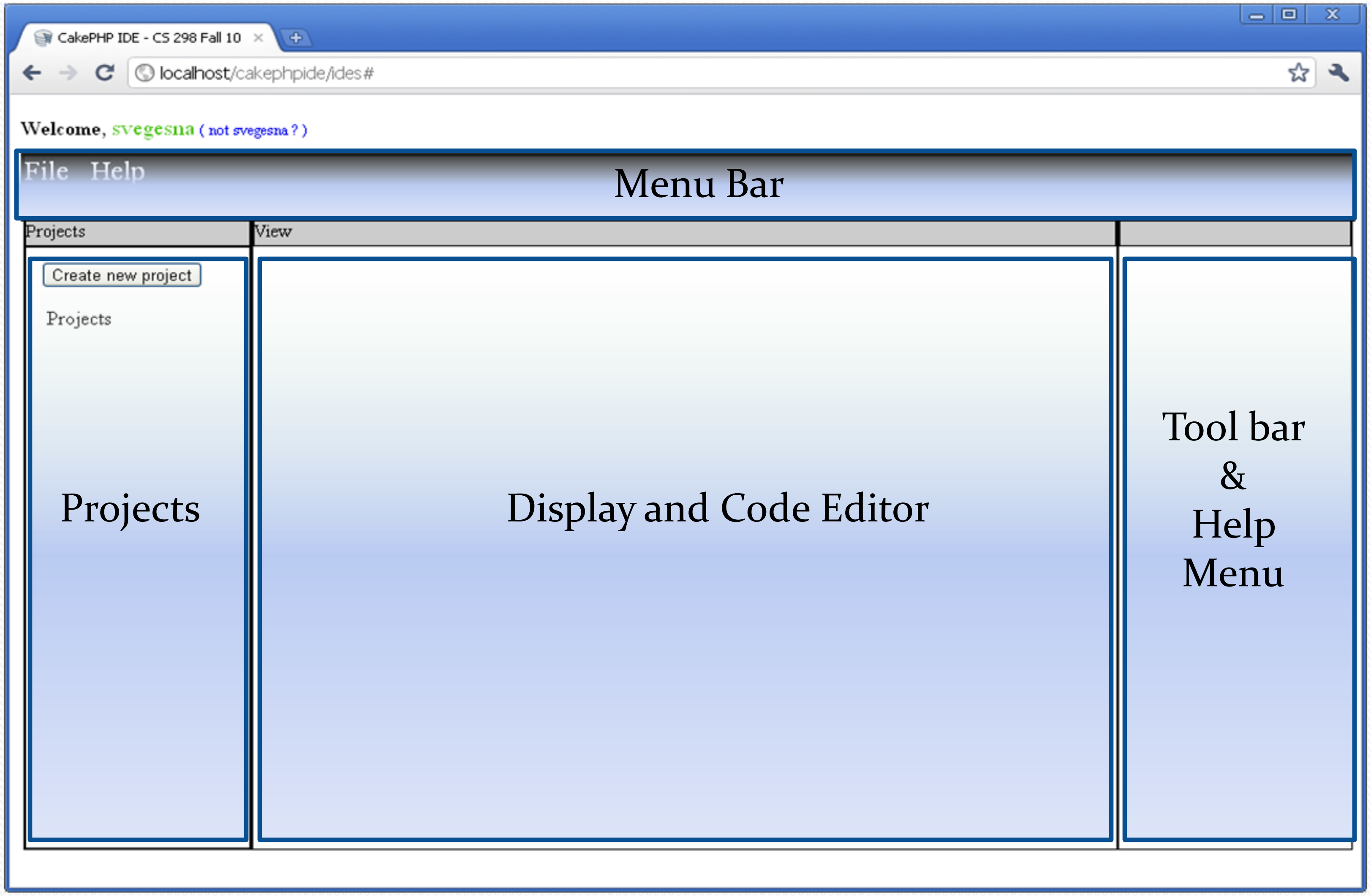
Draggable and Droppable classes



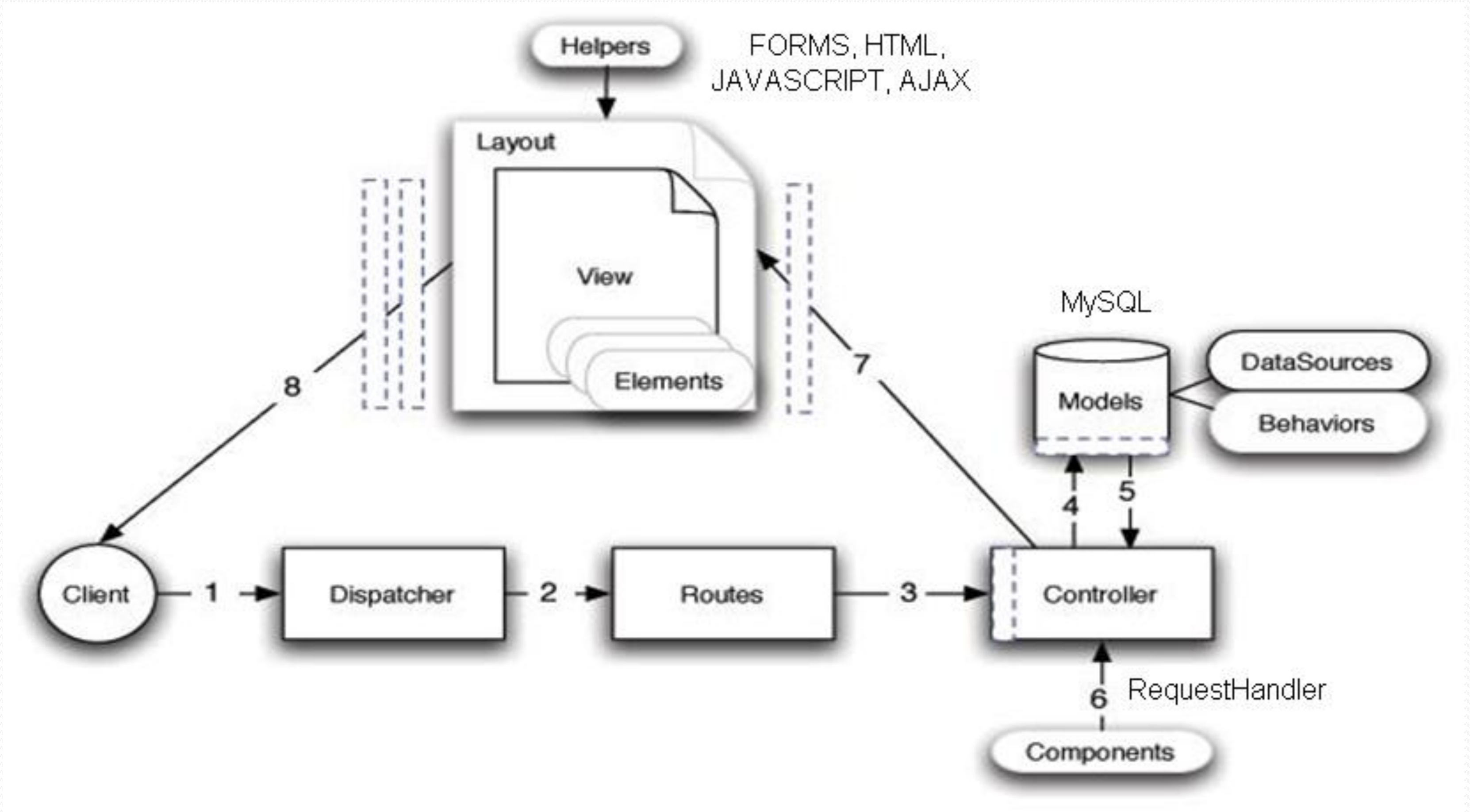
```
$(".tab").draggable({
            helper:'clone',
            cursor: 'move'
});
```

```
$("#area").droppable({
    accept: '.tab',
    drop: function(event, ui) {
        var d = ui.draggable.attr("id");
        createTable(d);
    }});
```

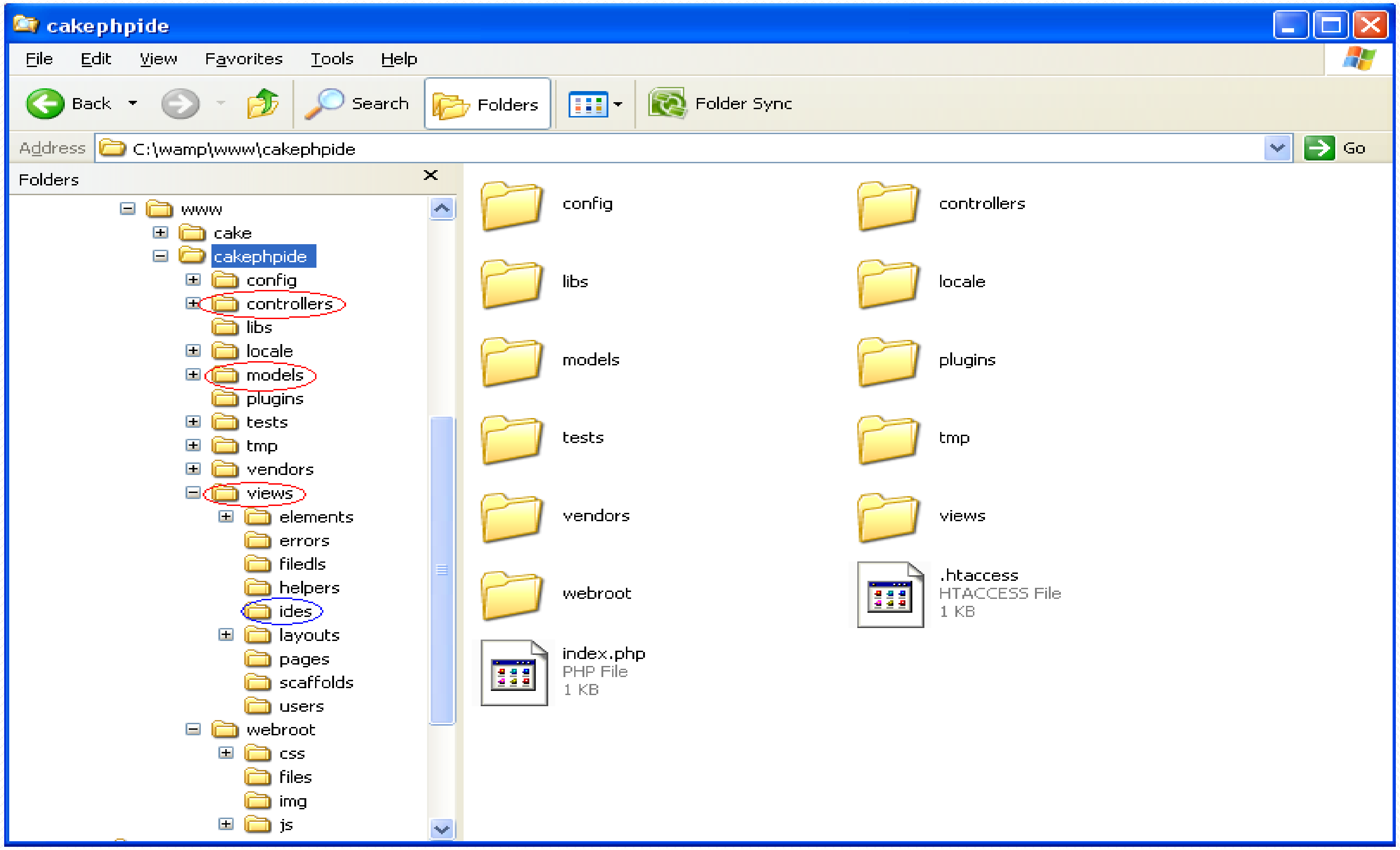
Layout of the IDE



Architecture (MVC)



Design



Folder Structure

Naming Conventions

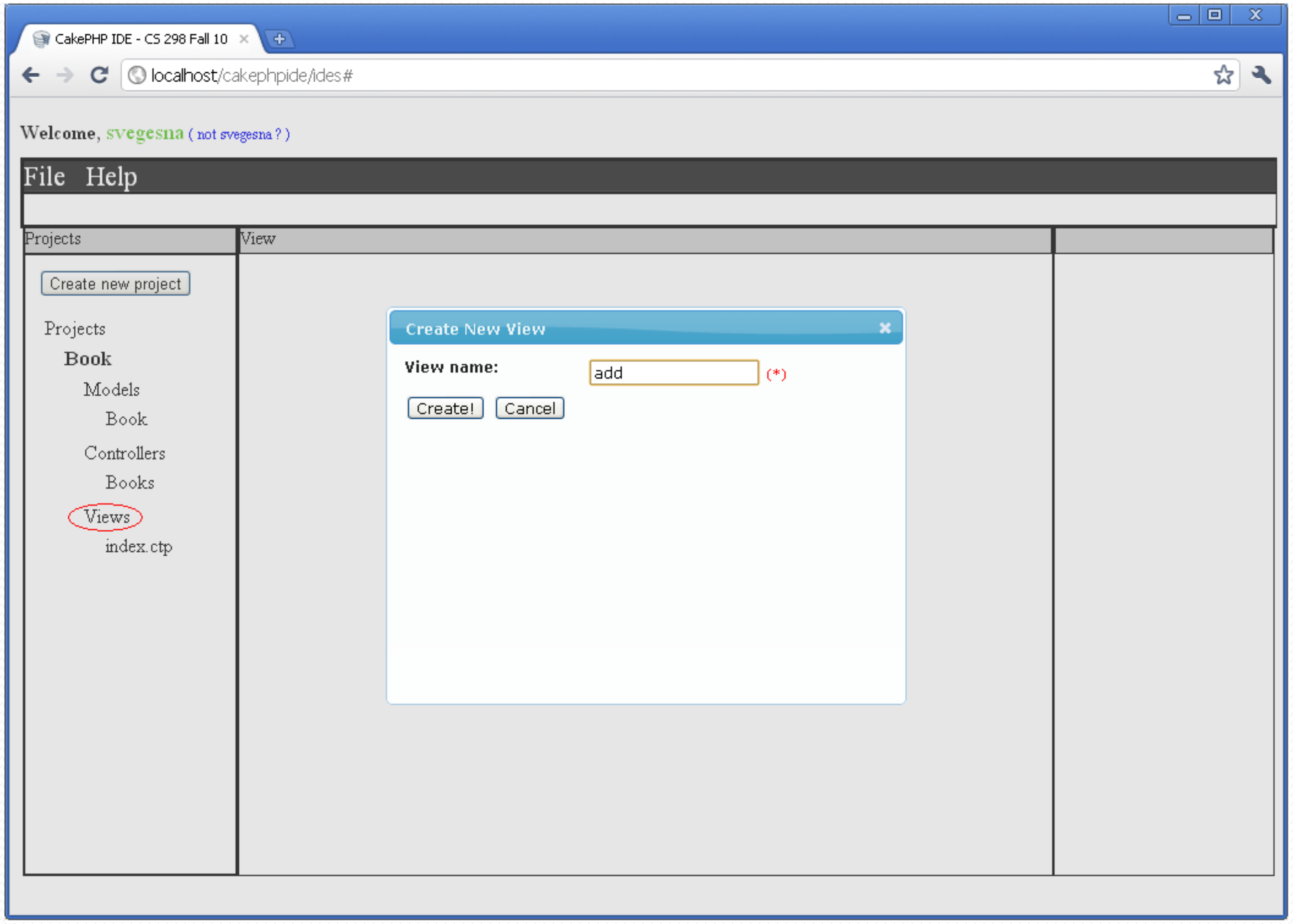
- Model class names should be singular and CamelCased.
  - Ide → ide.php
  - Table name → ides
- Controller class names should be plural, CamelCased, and end in Controller.
  - IdesController → ides_controller.php
- Views are named after the controller functions they display (using underscores).
  - getFile( ) →/cakephpide/views/ides/get_file.ctp

Features

Different features of the View Component of Web-based IDE include:

- Create Views
- Edit Views
- Design and Edit modes
- Form build Toolbar
- Preview of Web application

Create View



On a Right Click on "Views folder", creation of a new view

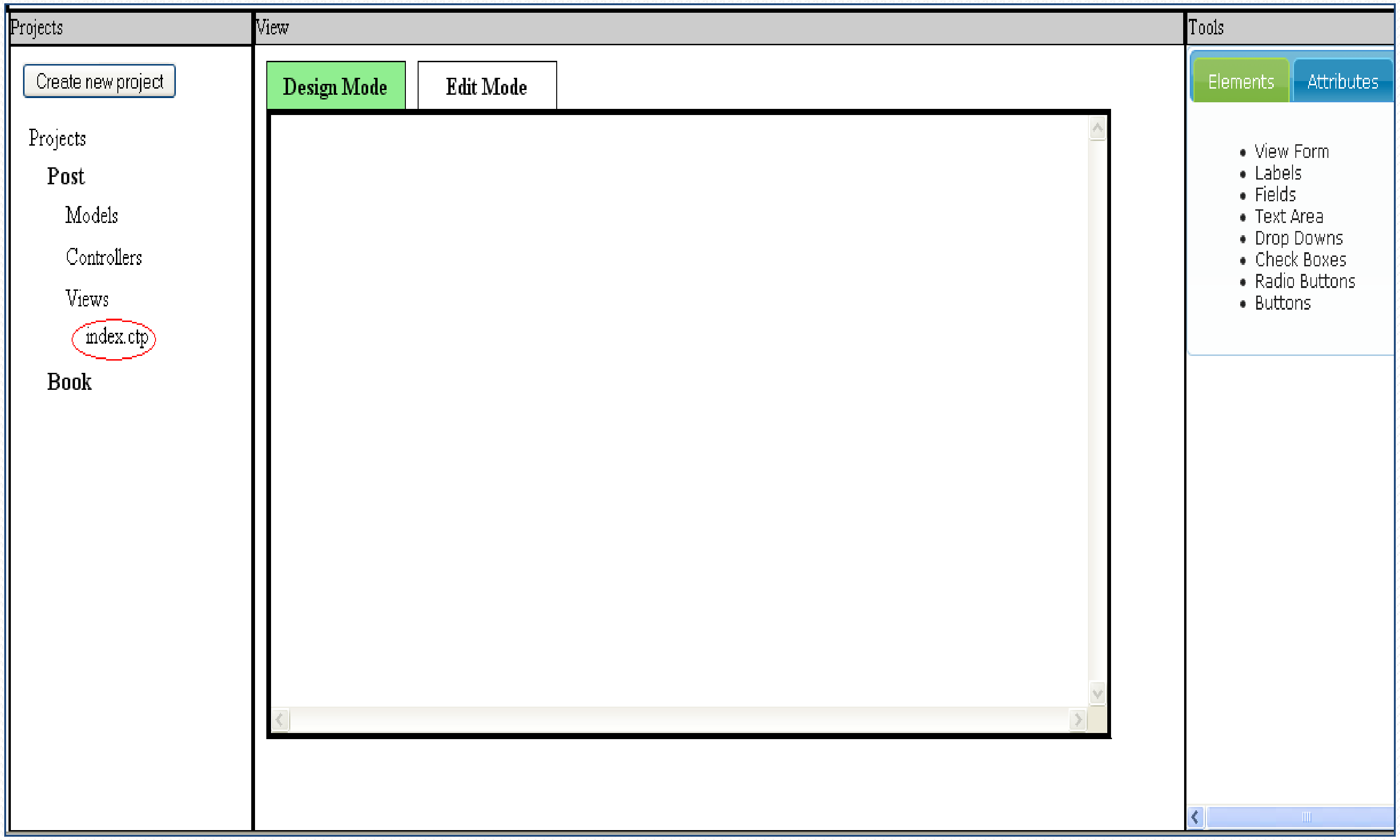
Implementation

```
function createview(){
                /*Get the data from the  view form*/
                 $this_project = $this->Ide->find('first', array('fields' => array('Ide.id, Ide.project_name,
Ide.project_path'), 'conditions' => array('Ide.id' => $project_id)));
                if($this_project){
                              if(!$this->Ide->is_view_available($project_id, $view_name){
                                return("-1");
                              }else{
                                 $this->create_view_skeleton_file($view_name, $view_filename);
                                 return (int) $this->Ide->ViewComponent->save();
                              }}
                    else  return 0;
  }
```

Create File

- Ajax call to the create the file
- Check for existence
- If exists complain else create a view file

Edit View



On a Right Click on “Views file”, opens the view file to edit

Implementation

```
function read_view_file(project_name, component_name){
        /* On successful ajax call */
          {
               msg_div = htmlspecialchars_decode(msg);
               msg_text = msg.replace(/\n/gi, "<br>");
               msg_text = msg.replace(/\s/gi, "&nbsp;");
               $('#viewarea').html(' ').html(msg_div);            // design mode
               $('#edit_panel textarea#myvoiceid').val(msg_text);    // edit mode
          }
}
```

Read File

- Ajax call to the file and fetch the data
- Modify the data according
- Display the data both in design and edit modes

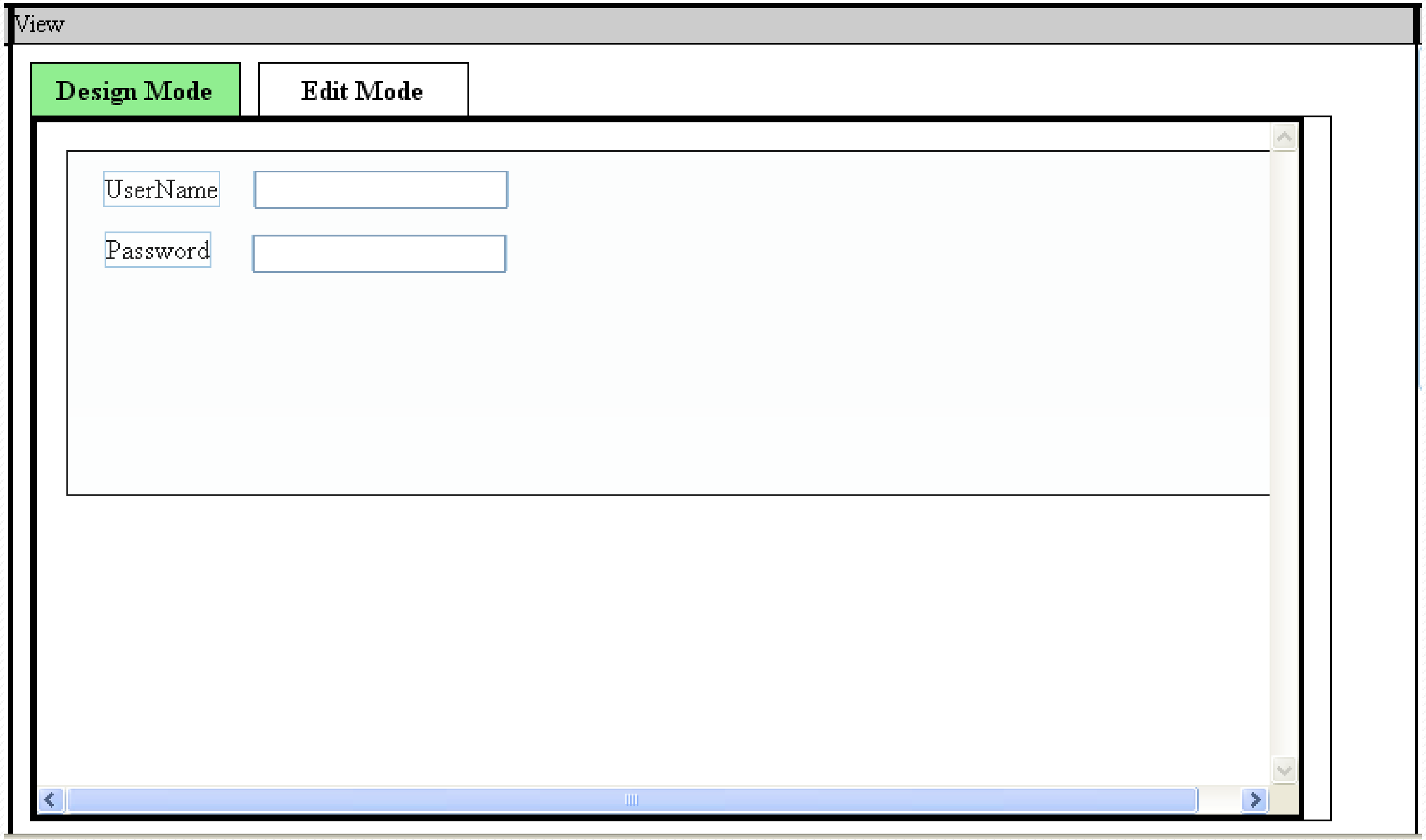
Implementation

```
function save_view_file(){
        var fileio = $('#edit_panel textarea#myvoiceid').data('fileio');
        var content = encodeURIComponent($('.editpanel textarea#myvoiceid').val());
        var component_type = fileio.component_type;
        var component_filename = fileio.component_filename;
        $.ajax({'type' : 'POST',
              'url' : write_file_url,
              data : 'component_filename='+component_filename+'&content='+content,
              success: function(msg){}
         });
}
```

Save File

- Get the code form the edit mode
- the data through Ajax call
- Write into the file

Design mode of the Views



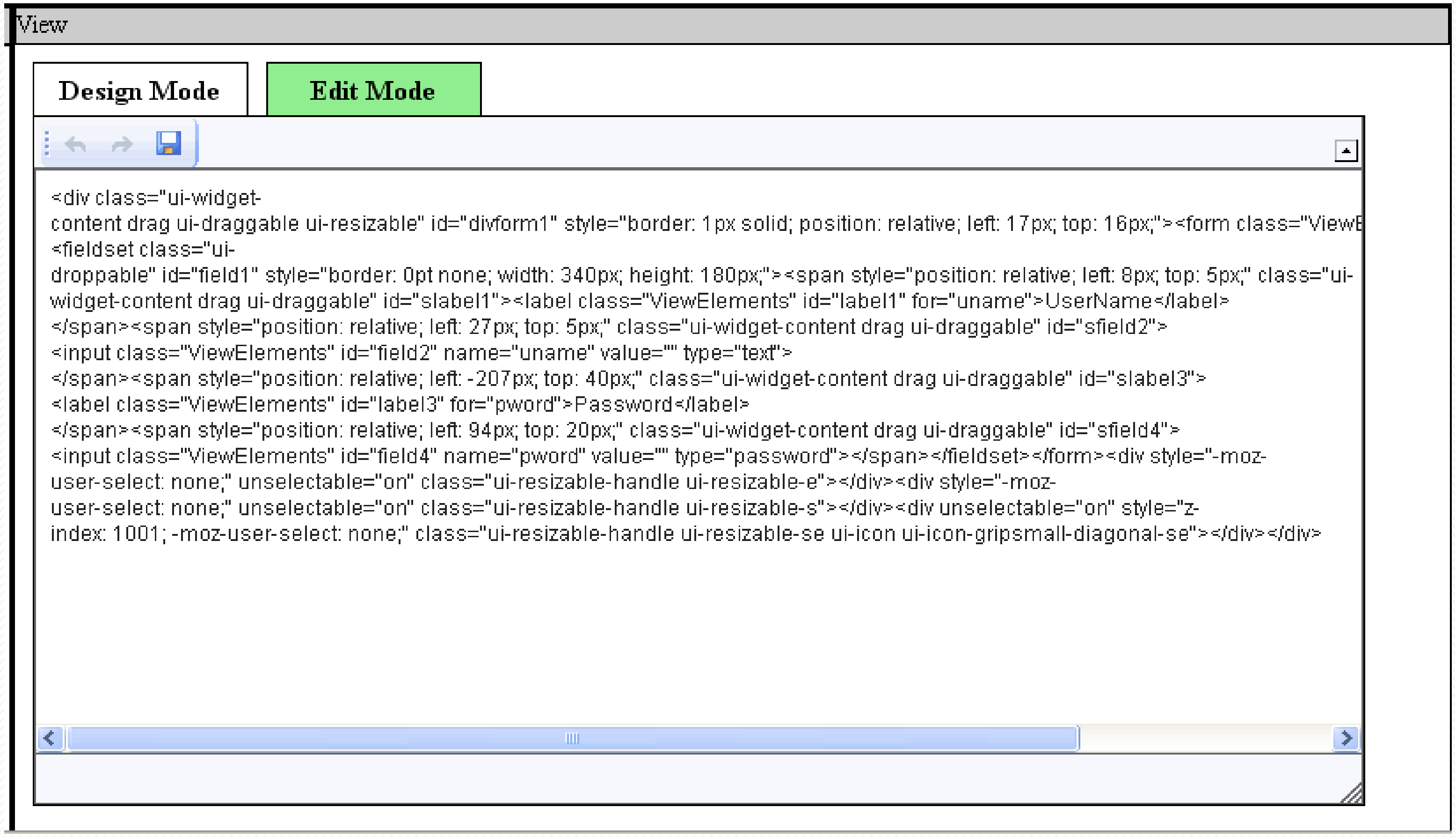
Design mode – Using Toolbar to build the views

Implementation

```
$("#design_tab").click(function() {
            $(‘.formelements').draggable ({
                        helper:'clone',
                        cursor: 'move'
            });
            var h = htmlspecialchars_decode($('.editpanel textarea#myvoiceid').val());
            h = h.replace("&nbsp;", " ");
            $('#viewarea').html(' ').html(h);
});
```

- All the elements are made draggable
- Gets the content from the edit mode
- Displays the content

Edit mode of the Views



```
<div class="ui-widget-
content drag ui-draggable ui-resizable" id="divform1" style="border: 1px solid; position: relative; left: 17px; top: 16px;"><form class="ViewE
<fieldset class="ui-
droppable" id="field1" style="border: 0pt none; width: 340px; height: 180px;"><span style="position: relative; left: 8px; top: 5px;" class="ui-
widget-content drag ui-draggable" id="slabel1"><label class="ViewElements" id="label1" for="uname">UserName</label>
</span><span style="position: relative; left: 27px; top: 5px;" class="ui-widget-content drag ui-draggable" id="sfield2">
<input class="ViewElements" id="field2" name="uname" value="" type="text">
</span><span style="position: relative; left: -207px; top: 40px;" class="ui-widget-content drag ui-draggable" id="slabel3">
<label class="ViewElements" id="label3" for="pword">Password</label>
</span><span style="position: relative; left: 94px; top: 20px;" class="ui-widget-content drag ui-draggable" id="sfield4">
<input class="ViewElements" id="field4" name="pword" value="" type="password"></span></fieldset></form><div style="-moz-
user-select: none;" unselectable="on" class="ui-resizable-handle ui-resizable-e"></div><div style="-moz-
user-select: none;" unselectable="on" class="ui-resizable-handle ui-resizable-s"></div><div unselectable="on" style="z-
index: 1001; -moz-user-select: none;" class="ui-resizable-handle ui-resizable-se ui-icon ui-icon-gripsmall-diagonal-se"></div></div>
```

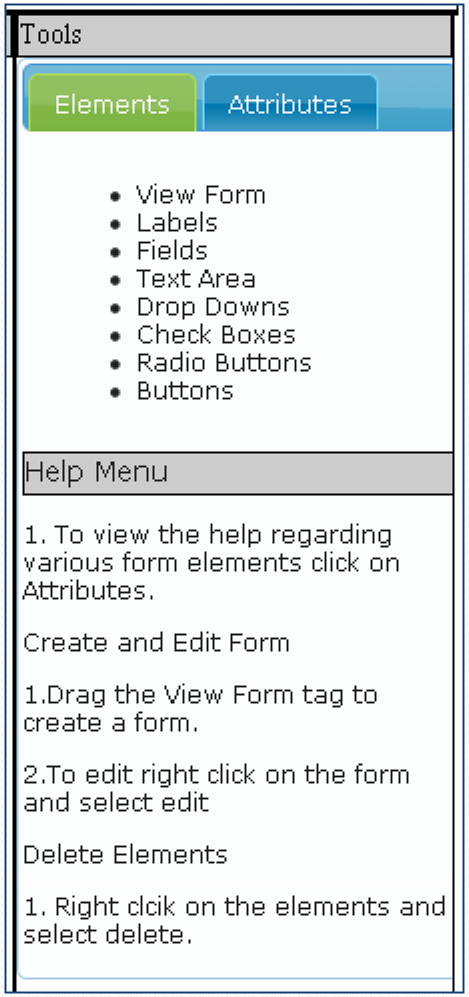
Edit mode – Can modify the view code and save the file

Implementation

```
$("#edit_tab").click(function() {
        $(‘.formelements').draggable("destroy");
        var h = htmlEntities($('#viewarea').html());
        h = h.replace("&nbsp;", " ");
        h = h.replace(/\>\n/gi, ">");
        h = h.replace(/\>/gi, ">\n");
        h = h.replace(/\s/gi, "&nbsp;");
        $('.editpanel textarea#myvoiceid').val(h);
});
```

- Destroys the draggable nature of the elements
- Fetches the modified content from the design mode
- Replaces the HTML entities
- Displays the code in the Text editor

Form build Toolbar



- View Form
- Labels
- Fields
- Text Area
- Drop Downs
- Check Boxes
- Radio Buttons
- Buttons

Help Menu

1. To view the help regarding various form elements click on Attributes.

Create and Edit Form

1.Drag the View Form tag to create a form.

2.To edit right click on the form and select edit

Delete Elements

1. Right clcik on the elements and select delete.

Tool bar and Help Menu for View component

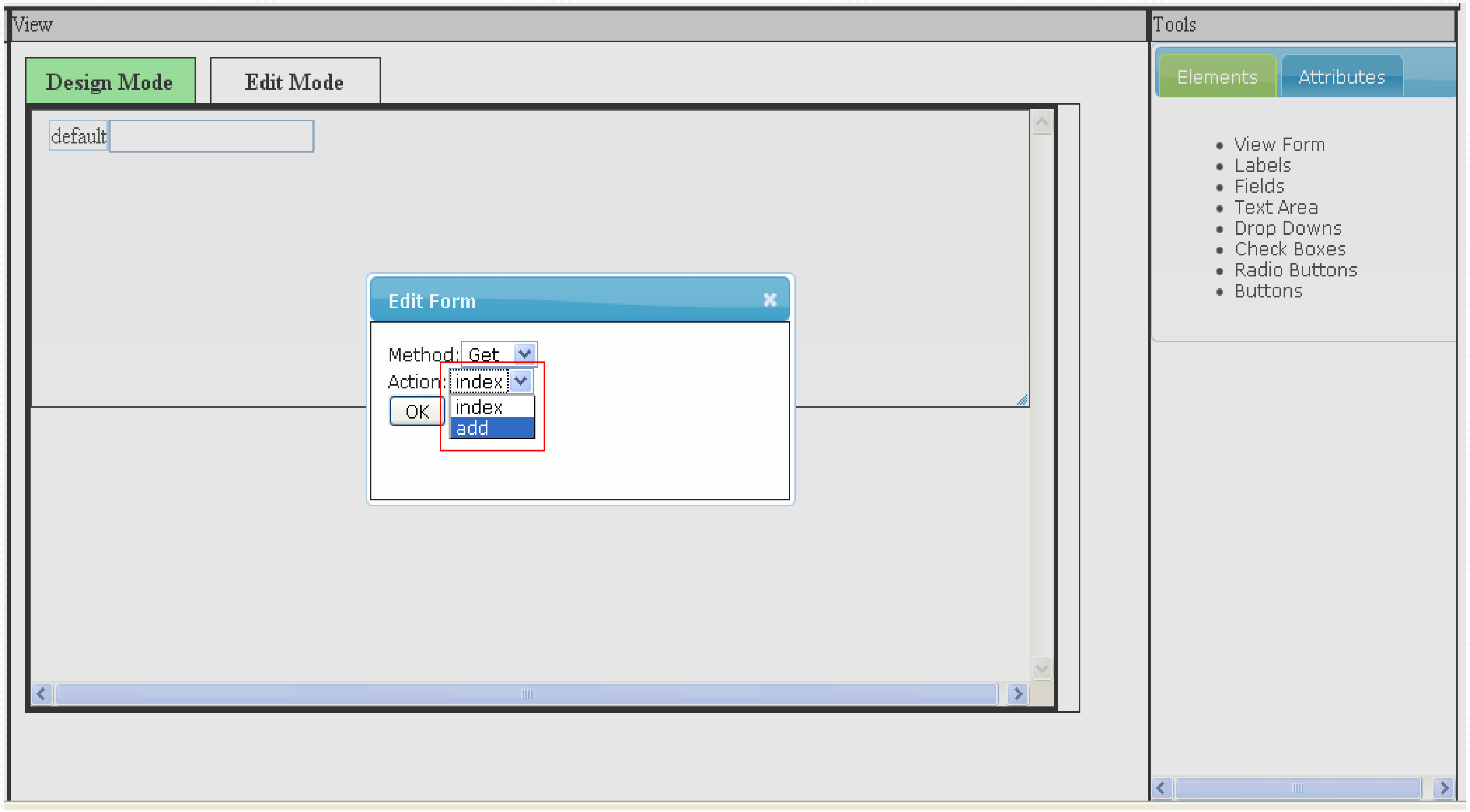
Implementation

```
$(".formelements").draggable({
     helper:'clone',
     cursor: 'move'
});
```

- Create the draggable objects of the elements in the toolbar
- Create the droppable element of the dic in the design mode
- On drop event implemented the features
  - Create a view element
  - Make them draggle
  - Right click implementation

```
$(".viewFile").droppable({
     accept: '.formelements',
     drop: function(event, ui) {
     $('.viewFile').append(form);
     $(".drag").draggable({
               cursor: 'move',
               containment: "parent“
      });
      append_rightClick();
      $("#divform"+formId).resizable();
      $("#field"+formId).droppable({
               accept: '.formElements',
               drop: function(event, ui){
                    addElement(d, fId);
                    append_rightClick();
               }
       });
      }
});
```

View-Controller Interactivity



Edit Form– Displaying various Controller functions for the form action

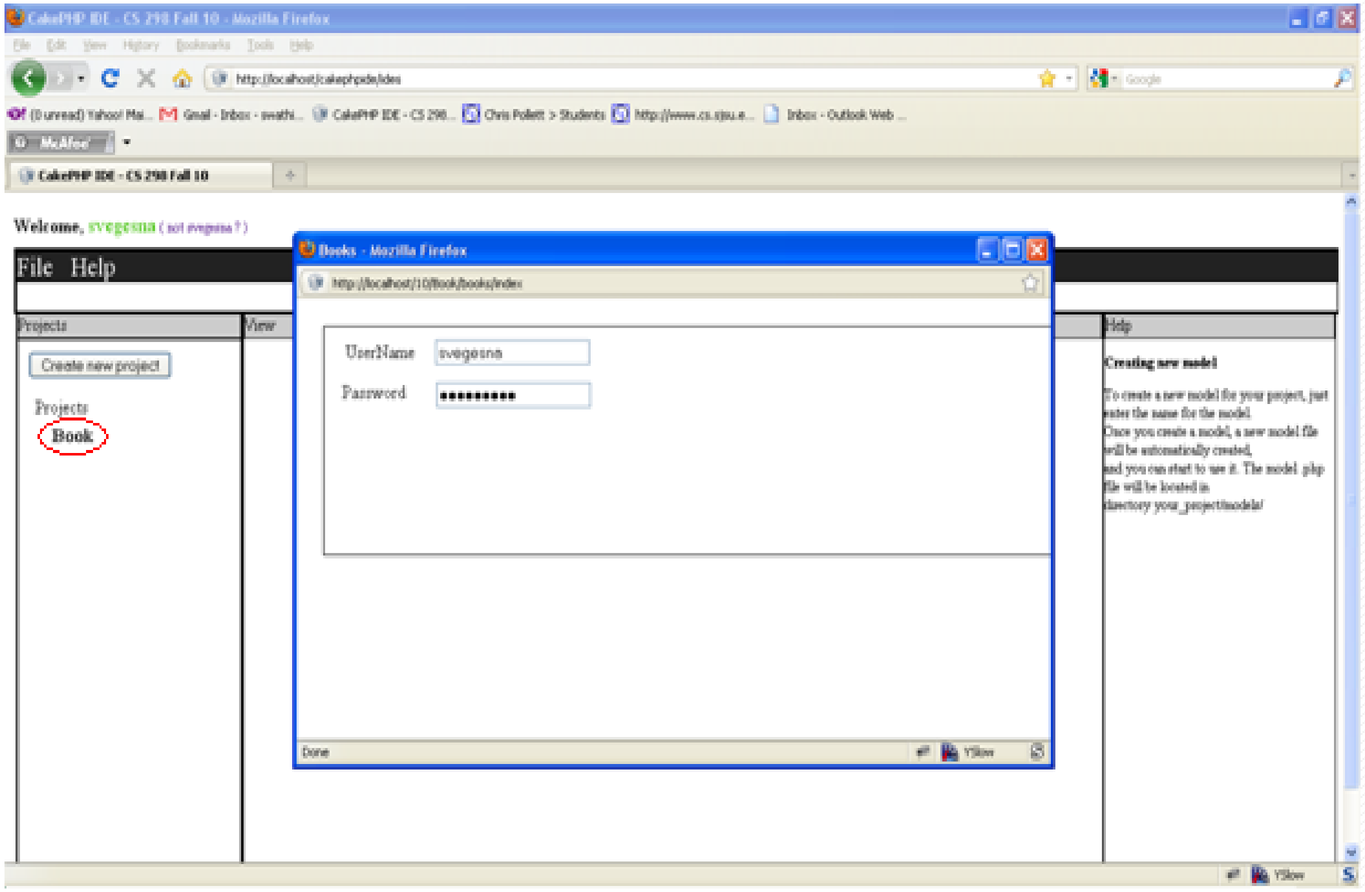
Implementation

```
function getformfunctions(){
        /*get the data from the view form*/
        $all_contents = file($cfile_path);
        $obj = $this->instantiate_controller($project_name, $controller_name, $controller_filename);
        if($obj){
                results = get_class_methods(get_class($obj));
        }
        print(json_encode($results));
}
```

Get controller Functions

- Read all the controller file
- Fetch all the class methods
- Send the list of controller functions

Preview of Web application


CakePHP IDE - CS 298 Fall 10 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/cakephpide/ides
Welcome, svegesna (not svegesna ?)
File Help
Projects
View
Help
Create new project
Projects
Book
Books - Mozilla Firefox
http://localhost/10/Book/books/index
Done
YSlow

UserName svegesna
Password ••••••••••

Creating new model

To create a new model for your project, just enter the name for the model.
Once you create a model, a new model file will be automatically created,
and you can start to use it. The model .php file will be located in
directory your_project/models/

On a Right Click on Project for a preview, new window with the web application is displayed.

Implementation

```
$("span.project").contextMenu({
    menu: 'projectMenu’ },
    function(action, el, pos) {
        if(action == 'preview'){
            popitup(url);
        }
});
```

```
$function popitup(url) {
        newwindow =window.open(url,'name',
'height=350,width=400');
        if (window.focus) {
                newwindow.focus()
        }
        return false;
 }
```

- Right click implementation on the project
  - Call the function to preview
- Preview of the web application
  - Open a new window to display the web application

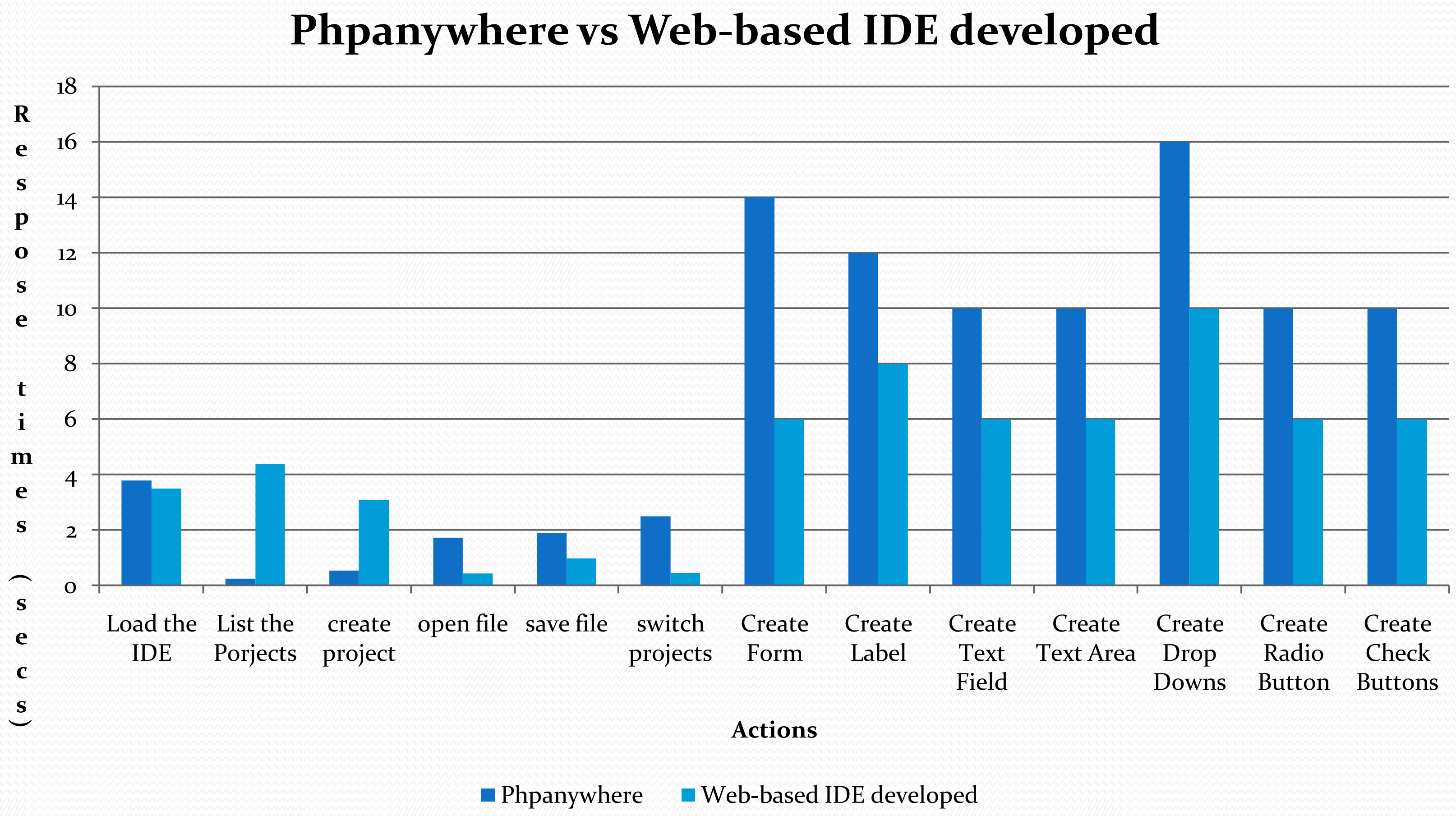
User Testing

- Feedback from 3 graduate students
  - GUI is very user friendly
  - Form Toolbar is a best feature
  - View-controller Interactivity is very challenging
  - Fast as no sluggish behavior in cursor movement
- Response to Feedback:
  - Modified Toolbar to display the form elements only when a form created.
  - Included help menu at bottom of the Toolbar.
  - Included a generic CSS file.

Comparison with Phpanywhere

Features	Proposed IDE	Phpanywhere
MVC pattern	Implemented	N/A
Framework	CakePHP	N/A
Tools to build Views	Available	N/A
Preview	Available	N/A
Tabs	N/A	Available
Code Indentation	N/A	Available
Syntax highlighting	N/A	Available

Performance Testing



Conclusion

View Component of the Web-based IDE developed

- Enables users to create CakePHP view templates
- Provides user-friendly GUI to edit the templates
- Eases creation of HTML Forms
- Establishes a View–Controller interaction

IDE developed eases the development of CakePHP based Web applications

Demo

References

CakePHP. Retrieved November 06, 2010, from http://cakephp.org/

Top 10 Ranking PHP Frameworks. Retrieved November 14, 2010 from http://www.phpframeworks.com/top-10-php-frameworks/

Model–View–Controller. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/Model%E2%80%93View%E2%80%93Controller

Web Server. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/Web_server

Apache HTTP server. Retrieved November 06, 2010 from http://httpd.apache.org/

MySQL. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/MySQL

PHP. Retrieved November 06, 2010 from http://www.php.net/

PhpMyAdmin. Retrieved November 06, 2010 from http://www.phpmyadmin.net/home_page/index.php

JQuery. Retrieved November 06, 2010 from http://jquery.com/

FireBug. Retrieved November 06, 2010 from http://getfirebug.com/

CKEditor. Retrieved November 06, 2010 from http://ckeditor.com/

Phpanywhere. Retrieved November 20, 2010 from http://phpanywhere.net/overview

Thank you

Questions