# ЛАБОРАТОРНАЯ РАБОТА №10

# «Исследование механизма сессий в PHP»

## Цель работы

Исследовать механизмы сохранения данных в процессе работы пользователя с сайтом, изучить возможности авторизации пользователей с использованием механизма сессий PHP, приобрести практические навыки использования переменных, сохраняющих свое значение при переходе от одной страницы сайта к другой.

## Порядок выполнения работы

1. В соответствии с рекомендациями п. 3.2.7 реализовать зону администрирования сайта и переместить в нее страницы «Редактор блога» и «Загрузка сообщений гостевой книги».

2. Реализовать накопление информации о посетителях (см п. 3.2.6) страниц пользовательского раздела сайта в разработанной для этого таблице базы данных. Структура информации:

• Дата и время посещения;

• IP-адрес компьютера посетителя;

• Имя хоста компьютера посетителя;

• Название браузера, который использовал посетитель.

3. Добавить в зону администрирования сайта страницу «Статистика посещений», отображающую информацию о посетителях сайта. Реализовать постраничное отображение информации в порядке убывания даты и времени посещения.

4. Добавить в зону администратора форму входа, содержащую поля логин и пароль. Логин и пароль админстратора хранить к коде программы. Проверку пароля осуществлять сравнением зашифрованного пароля с вычисленным заранее хешем верного пароля.

5. На всех страницах зоны администратора реализовать проверку авторизации и в случае, если посетитель не авторизован как администратор –перенаправлять пользователя на страницу авторизации.

6. В пользовательском разделе сайта разработать страницу «Регистрация пользователя», содержащую форму для ввода (все поля обязательны к заполнению):

• ФИО пользователя;

• e-mail пользователя;

• логин пользователя;

• пароль пользователя.

Добавлять введенную пользователем информацию в разработанную таблицу базы данных. Предусмотреть проверку на невозможность добавления двух одинаковых логинов.

7. Добавить в меню пользовательского раздела сайта ссылку «Войти» на форму авторизации. Форма должна содержать строку ввода логина, строку ввода пароля, кнопку "Войти", а также ссылку «Регистрация» (на страницу «Регистрация пользователя»). После успешной авторизации в меню сайта вместо ссылки «Войти» должна отображаться ссылка "Выйти", при нажатии которой происходит выход авторизированного пользователя (завершение текущей сессии). После успешной авторизации на каждой странице сайта (возможно в области меню) должно отображаться "Пользователь: ФИО" (ФИО – Фамилия Имя Отчество пользователя).

8. Реализовать отображение результатов тестирования на странице «Тест по дисциплине» только авторизованным пользователям.

## Ход работы

### Реализация раздела администратора представлена на рисунке 3.1.

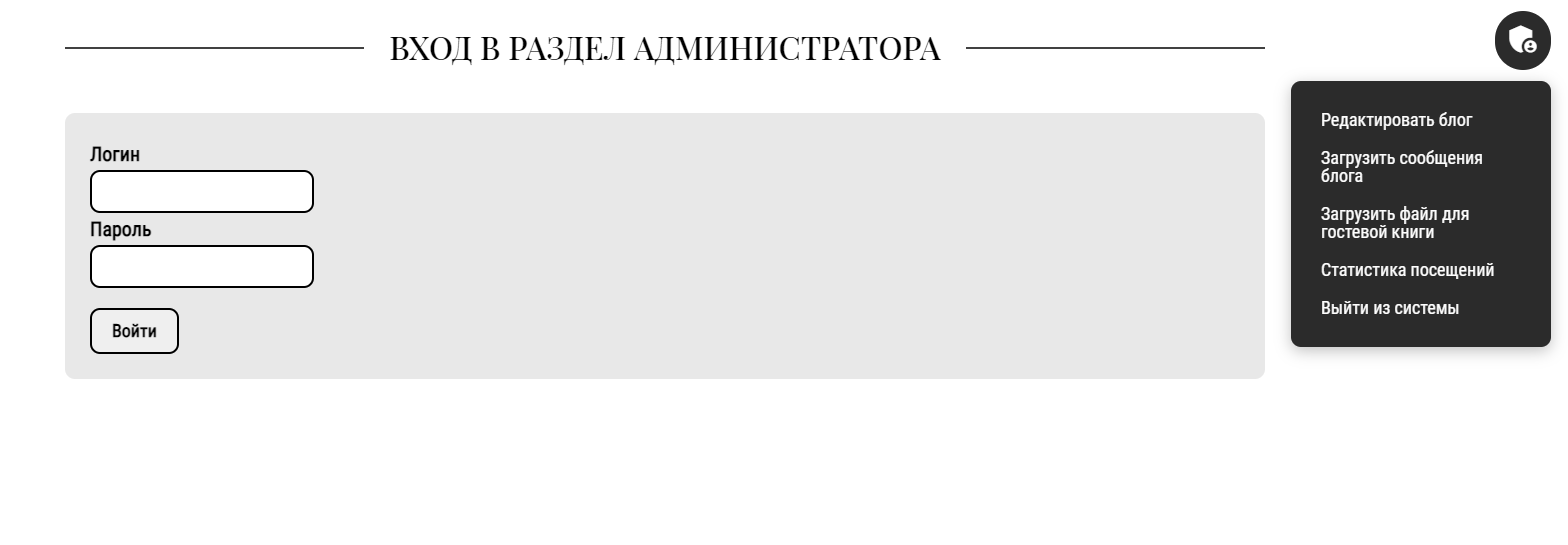


Рисунок 3.1 – Реализация раздела администратора

### На рисунке 3.2 представлена разработанная страница «Статистика посещений». Полна реализация кода представлена в приложении А

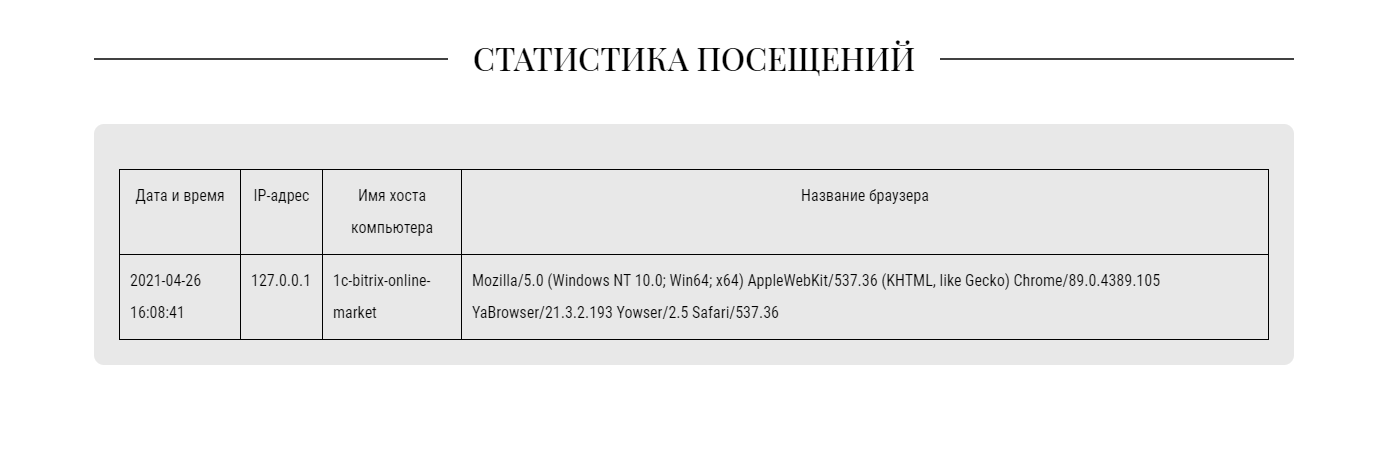


Рисунок 3.2 – Страница «Статистика посещений»

### На рисунке 3.3 представлена разработанная страница «Вход в раздел администратора». Полный код представлен в приложении А

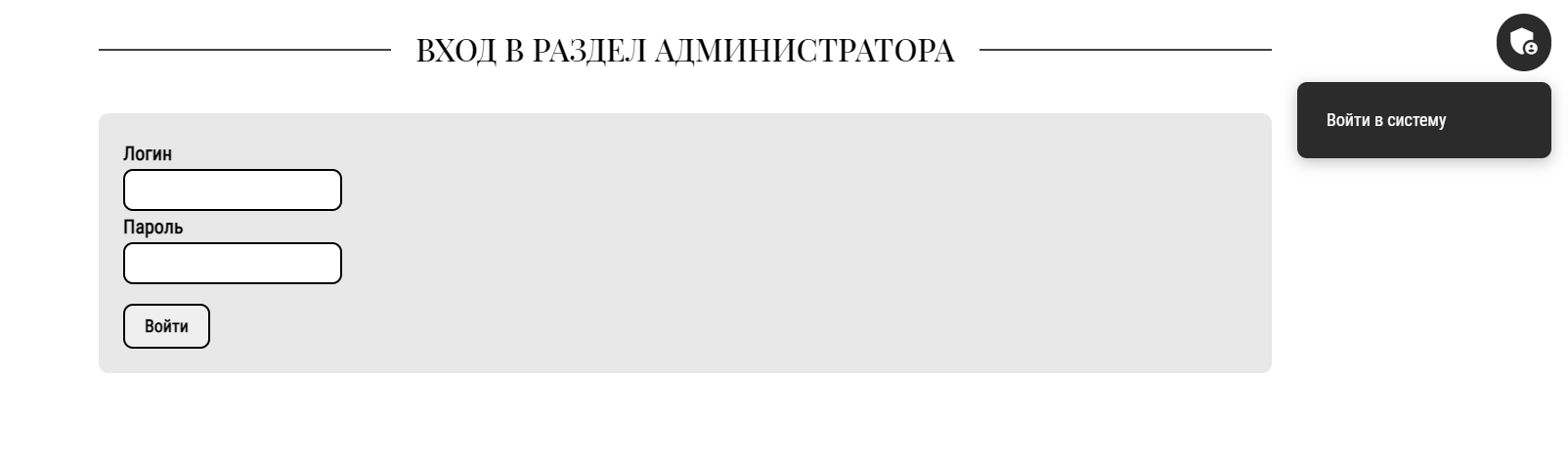


Рисунок 3.3 – Страница «Вход в раздел администратора»

### При попытке получить доступ к страницам из раздела администратора неавторизованным пользователям появится ошибка (Рисунок 3.4)

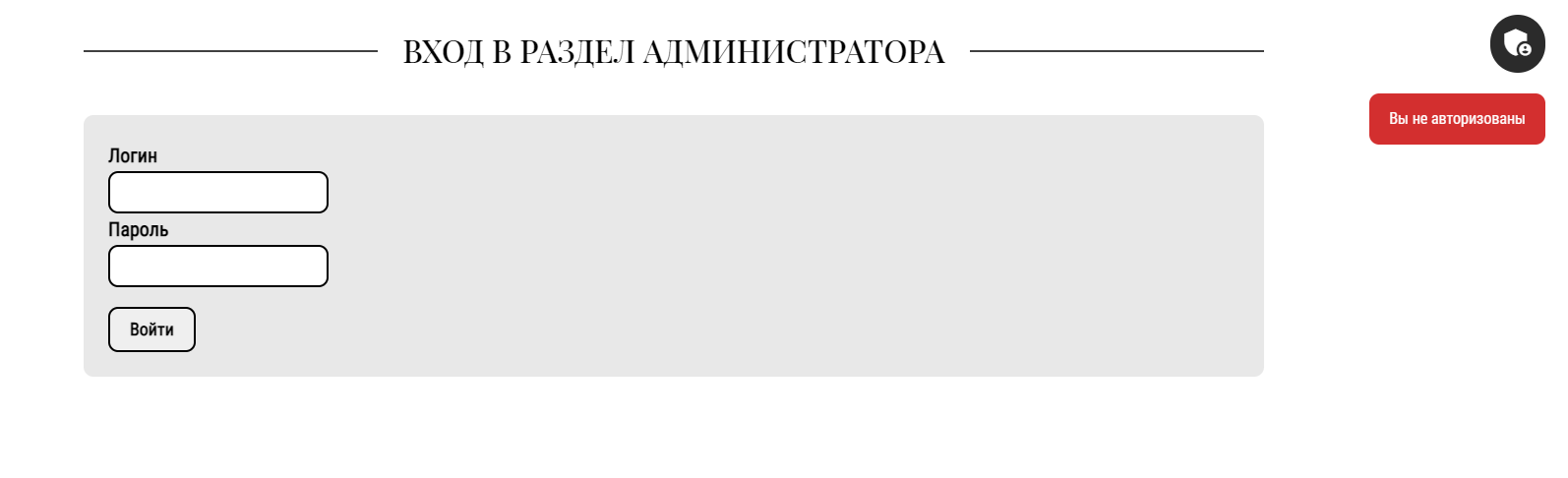


Рисунок 3.4 – Ошибка при попытке получить доступ к страницам из раздела администратора неавторизованным пользователям

### На рисунке 3.5 представлена разработанная страница регистрации пользователя. Если пользователь с таким логин уже существует появится сообщение с ошибкой (рисунок 3.6)



Рисунок 3.5 – Страница регистрации пользователя



Рисунок 3.6 – Ошибка при попытке зарегистрироваться с одинаковым логином

### На рисунке 3.7 представлена разработанная страница входа пользователя. В шапке сайта появится приветственное сообщение, если пользователь успешно вошёл в систему (Рисунок 3.8)

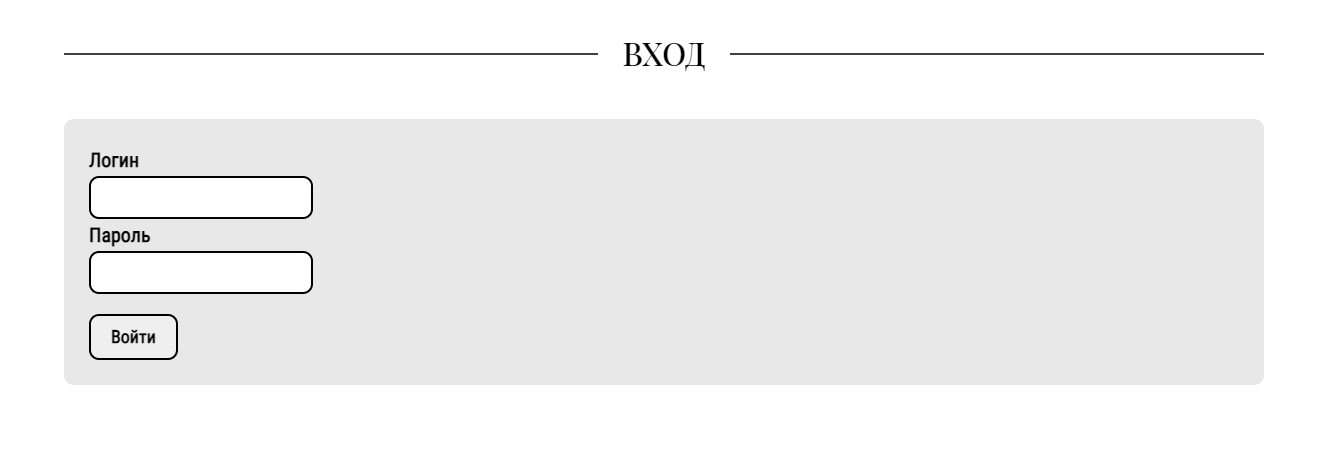


Рисунок 3.7 – Страница входа пользователя

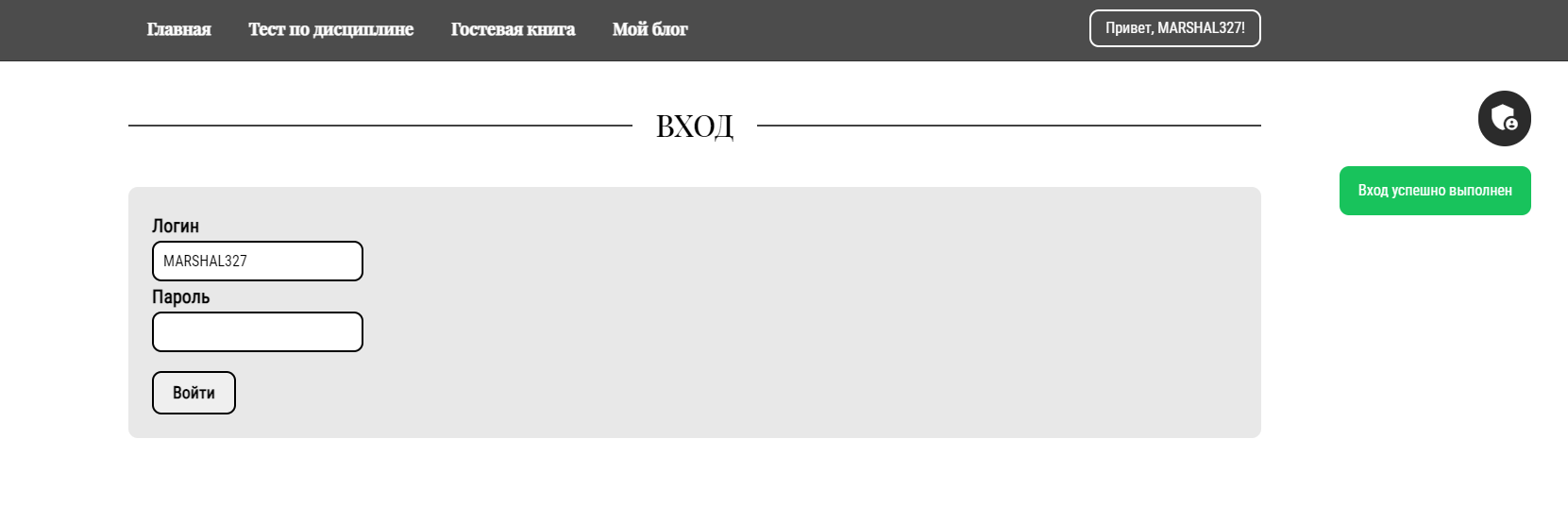


Рисунок 3.8 – Сообщение об успешном входе и приветственное сообщение

### Если пользователь вошёл в систему ему станут доступны данные о прохождении теста (Рисунок 3.9), в противном случае он увидит сообщение (Рисунок 3.10)

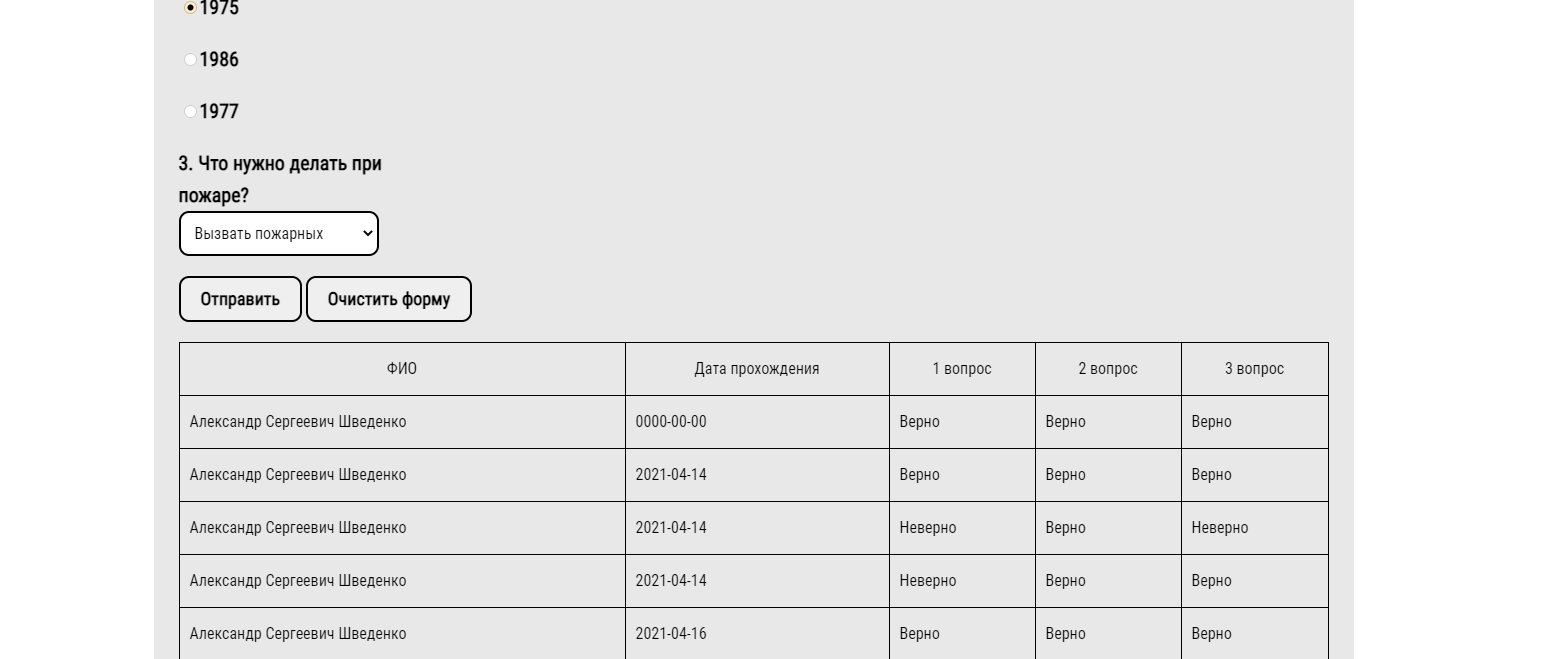


Рисунок 3.9 – Данные о прохождении теста

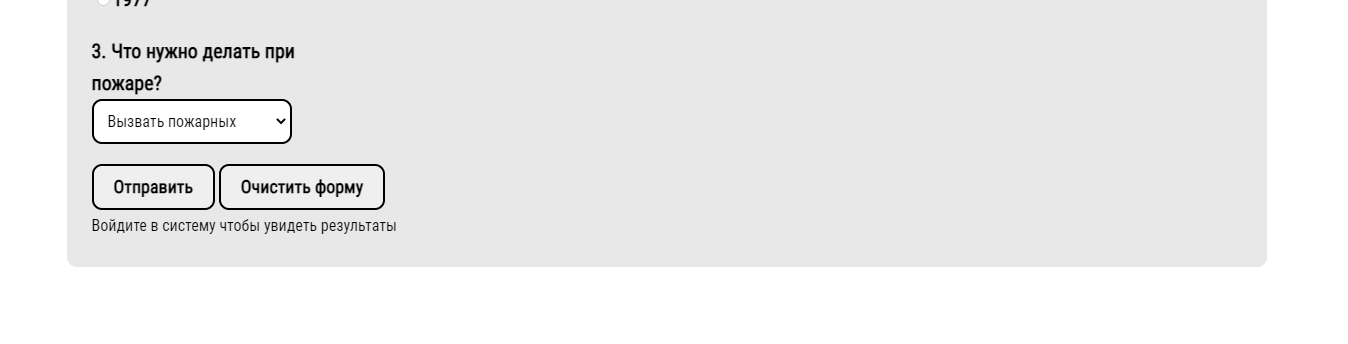


Рисунок 3.10 – Сообщение о необходимости входа

# Вывод

В ходе выполнения данной лабораторной работы были исследованы механизмы сохранения данных в процессе работы пользователя с сайтом, изучены возможности авторизации пользователей с использованием механизма сессий PHP, приобретены практические навыки использования переменных, сохраняющих свое значение при переходе от одной страницы сайта к другой. Реализованы административный раздел сайта, а также раздел для пользователей.

# Приложение А

Файл app/controllers/AdminPanelController.php

<?php  
namespace app\controllers;  
  
use app\core\Controller;  
use app\core\View;  
  
class AdminPanelController extends Controller  
{  
  
 public function indexAction()  
 {  
 $this->view->render("Администратор", [  
 "menuIndex" => 0  
 ]);  
 }  
  
 public function loginAction()  
 {  
 $errors = [];  
  
 if (!empty($\_POST)) {  
 $loginRes = $this->login();  
  
 if( !$loginRes ) $errors[] = "Неверный логин или пароль";  
 }  
  
 $this->view->render("Вход в раздел администратора", [  
 "menuIndex" => 0,  
 "errors" => $errors,  
 ]);  
 }  
  
 public function logoutAction(){  
 $\_SESSION['isAdmin'] = 0;  
 View::*redirect*("/");  
 }  
 public function login()  
 {  
 if (($\_POST["login"] == 'admin') && (md5($\_POST["password"]) == 'd8578edf8458ce06fbc5bb76a58c5ca4')) {  
 $\_SESSION['isAdmin'] = 1;  
 return true;  
 }  
  
 $\_SESSION['isAdmin'] = 0;  
 return false;  
 }  
  
 public static function authenticate(){  
 if (!isset($\_SESSION['isAdmin']) || $\_SESSION['isAdmin'] == 0) {  
 setcookie("redirect", 1, 0, "/");  
 View::*redirect*("/admin/login");  
 die();  
 }  
 }  
}

Файл app/controllers/UserController.php

<?php  
  
  
namespace app\controllers;  
  
use app\core\Controller;  
use app\core\View;  
use app\models\UserModel;  
  
class UserController extends Controller  
{  
 private $errors = [];  
  
 public function loginAction(){  
 if( !empty( $\_POST ) ){  
 $this->errors = $this->model->Validate($\_POST);  
  
 if( count( $this->errors ) == 0 ){  
 $userData = UserModel::*find*($\_POST["login"], "login");  
 if( $userData != null ){  
 if( $userData->password == md5(md5("salt") . $\_POST["password"]) ){  
 $\_SESSION["user\_login"] = $\_POST["login"];  
 } else {  
 $this->errors[] = "Неверный пароль";  
 }  
 } else {  
 $this->errors[] = "Пользователь с таким логином не найден";  
 }  
 }  
 }  
  
 $this->view->render("Вход", [  
 "menuIndex" => 0,  
 "errors" => $this->errors  
 ]);  
 }  
  
 public function registerAction(){  
 if( !empty( $\_POST ) ){  
 $this->errors = $this->model->Validate($\_POST);  
  
 if( count( $this->errors ) == 0 ){  
 if( UserModel::*find*($\_POST["login"], "login") == null ){  
 $this->save();  
 } else {  
 $this->errors[] = "Пользователь с таким логином уже существует";  
 }  
 }  
 }  
  
 $this->view->render("Регистрация", [  
 "menuIndex" => 0,  
 "errors" => $this->errors  
 ]);  
 }  
  
 public function logoutAction(){  
 $\_SESSION['user\_login'] = "";  
 View::*redirect*("/");  
 }  
  
  
 public static function isUserLogin(){  
 if( !isset($\_SESSION["user\_login"]) || $\_SESSION["user\_login"] == "" ){  
 return false;  
 } else return true;  
 }  
  
 function save(){  
 $user = new UserModel();  
  
 $user->fio = $\_POST["fio"];  
 $user->email = $\_POST["email"];  
 $user->login = $\_POST["login"];  
 $user->password = md5(md5("salt") . $\_POST["password"]);  
  
 $user->save();  
 }  
}

Файл app/controllers/VisitStatisticController.php

<?php  
  
namespace app\controllers;  
  
use app\core\Controller;  
use app\models\VisitStatisticModel;  
  
class VisitStatisticController extends Controller  
{  
 public function indexAction(){  
 AdminPanelController::*authenticate*();  
  
 if( VisitStatisticModel::*find*($\_SERVER['HTTP\_USER\_AGENT'], "browser\_name") == null ){  
 $this->save();  
 }  
  
 $allVisits = VisitStatisticModel::*all*("date", "ASC");  
  
 $this->view->render("Статистика посещений", [  
 "menuIndex" => 11,  
 "allVisits" => $allVisits  
 ]);  
 }  
  
 function save(){  
 $visitStatistic = new VisitStatisticModel();  
  
 $visitStatistic->date = date('Y-m-d H:i:s');  
 $visitStatistic->host\_name = gethostbyaddr($\_SERVER['REMOTE\_ADDR']);  
 $visitStatistic->ip\_address = $\_SERVER['REMOTE\_ADDR'];  
 $visitStatistic->browser\_name = $\_SERVER['HTTP\_USER\_AGENT'];  
  
 $visitStatistic->save();  
 }  
}

Файл app/models/AdminPanelModel.php

<?php  
  
  
namespace app\models;  
  
use app\core\BaseActiveRecord;  
  
class AdminPanelModel extends BaseActiveRecord  
{  
 public $adminItems;  
  
 public function \_\_construct()  
 {  
 $this->adminItems = [  
 "/blog/edit" => "Редактировать блог",  
 "/blog/load" => "Загрузить сообщения блога",  
 "/guest-book/load" => "Загрузить файл для гостевой книги",  
 "/visit-statistic" => "Статистика посещений",  
 "/admin/logout" => "Выйти из системы",  
 ];  
 }  
}

Файл app/models/UserModel.php

<?php  
  
namespace app\models;  
  
use app\core\BaseActiveRecord;  
  
class UserModel extends BaseActiveRecord  
{  
 protected static *$tablename* = 'users';  
 public $id;  
 public $fio;  
 public $email;  
 public $login;  
 public $password;  
 public function \_\_construct()  
 {  
 parent::*\_\_construct*();  
 $url = $\_SERVER["REQUEST\_URI"];  
 if( $url == "/user/login" ){  
 $this->validator->SetRule("login", "isEmpty", "Логин");  
 $this->validator->SetRule("password", "isEmpty", "Пароль");  
 } else if( $url == "/user/register" ) {  
 $this->validator->SetRule("fio", "checkFio");  
 $this->validator->SetRule("email", "isEmail");  
 $this->validator->SetRule("login", "isEmpty", "Логин");  
 $this->validator->SetRule("password", "isEmpty", "Пароль");  
 }  
 }  
 public function Validate($postData){  
 return $this->validator->Validate($postData);  
 }  
}

Файл app/models/VisitStatisticModel.php

<?php  
  
namespace app\models;  
  
use app\core\BaseActiveRecord;  
  
class VisitStatisticModel extends BaseActiveRecord  
{  
 protected static *$tablename* = 'visit\_statistics';  
  
 public $id;  
 public $date;  
 public $ip\_address;  
 public $host\_name;  
 public $browser\_name;  
}