**РЕСПУБЛИКА БЕЛАРУСЬ**

**КОМИТЕТ ПО ОБРАЗОВАНИЮ МИНГОРИСПОЛКОМА**

учреждение образования

МИНСКИЙ ГОСУДАРСТВЕННЫЙ КОЛЛЕДЖ ЭЛЕКТРОНИКИ

##### Группа 45 ТП

## Отчет

## по лабораторной работе № 24

**"Создание программной системы, использующей шаблон MVC"**

дисциплина

“Технология разработки программного обеспечения”

**Исполнитель: Сушко А. Ю.**

**Руководитель: Комиссарова Я. О.**

**Минск, 2021**

**Цель работы:** научиться применять шаблон MVC для создания программной системы.

**Вариант 11**

**Выполнение заданий:**

*Задание 1*: Изучить примеры реализации шаблона MVC. Применить шаблон MVC для реализации проекта.

Model:

<?php

namespace application\core;

**abstract** **class** Model {

$news1='This is first news of the day';

$news2 = 'This is second news of the day';

**public** **function** \_\_construct(){

$this->news1=$news1;

$this->news2 = $news2;

}

}

View:

<?php

namespace application\core;

**class** View{

**public** $path;

**public** $route;

**public** $layout='default';

**public** **function** \_\_construct($route) {

$this->route = $route;

$this->path =$route['controller'].'/'.$route['action'];

}

**public** **function** render($title, $vars = []){

extract($vars)

if(file\_exists('application/views/'.$this->path.'.php')){

ob\_start();

require 'application/view/'.$this->path.'.php';

$content = ob\_get\_clean();

require 'application/view/layout/'.$this->layout.'.php';

}else{

echo "layout not found";

}

}

}

Controller:

<?php

namespace application\core;

use application\core\View;

**abstract** **class** Controller{

**public** $route;

**public** $view;

**public** **function** \_\_construct($route) {

$this->route = $route;

$this->view = new View($route);

$this->model = $this->loadModel($route['controller']);

}

**public** **function** loadModel($name){

$path = 'application\models\\'.ucfirst($name);

if(class\_exists($path)){

return new $path()

}

}

}

Router:

<?php

namespace application\core;

**class** Router {

**protected** $routes = [];

**protected** $params = [];

**function** \_\_construct() {

$arr = require 'application/config/routes.php';

*//debug($arr);*

foreach ($arr as $key => $val){

$this->add($key,$val);

}

}

**public** **function** add($route, $params){

$route = '#^'.$route.'$#';

$this->routes[$route]=$params;

}

public function match(){

//debug($\_SERVER);

$url = trim($\_SERVER['REQUEST\_URI'],'/');

//debug($url);

foreach($this->routes as $route => $params){

if(preg\_match($route,$url,$matches)){

$this->params=$params;

return true;

}

}

return false;

}

public function run(){

if($this->match()){

$path = 'application\controllers\\'.ucfirst($this->params['controller']).'Controller.php';

if (class\_exists($path)){

$action = this->$params['action'].'Action';

if(method\_exists($path,$action)){

$controller = new $path($this->params);

$controller->$action();

}else{

echo 'action not found';

}

}else{

echo 'not found controller: '.$path;

}

} else {

echo 'not found';

}

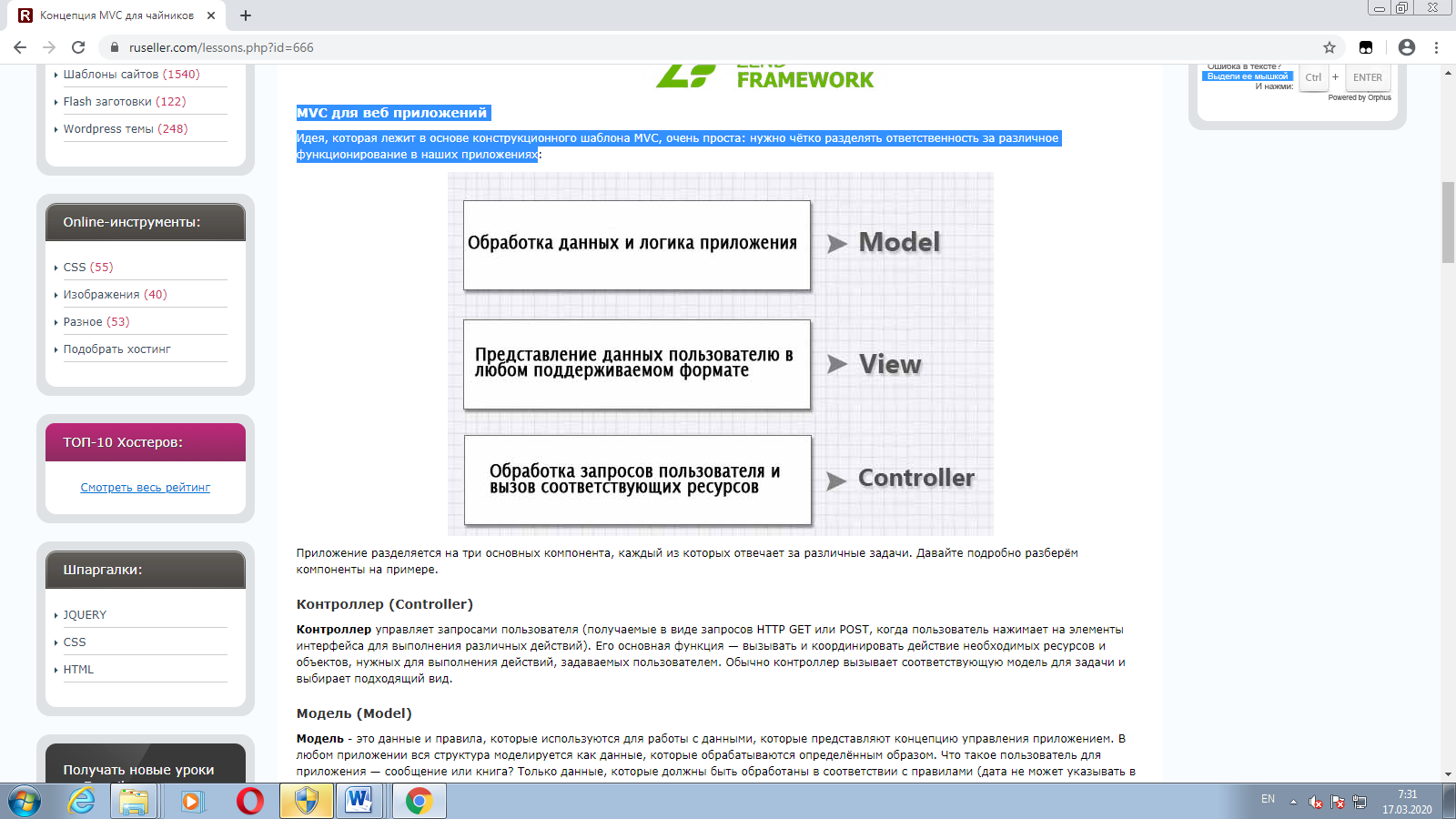
}

}

*Ответы на контрольные вопросы:*

1. Каково назначение шаблон MVC?

Идея, которая лежит в основе конструкционного шаблона MVC, очень проста: нужно чётко разделять ответственность за различное функционирование в приложениях (рисунок 24.1):



1. Для чего предназначен Контроллер?

**Контроллер** управляет запросами пользователя (получаемые в виде запросов HTTP GET или POST, когда пользователь нажимает на элементы интерфейса для выполнения различных действий). Его основная функция — вызывать и координировать действие необходимых ресурсов и объектов, нужных для выполнения действий, задаваемых пользователем. Обычно контроллер вызывает соответствующую модель для задачи и выбирает подходящий вид.

1. Что такое модель и как она связана с Контроллером?

**Модель** - это данные и правила, которые используются для работы с данными, которые представляют концепцию управления приложением. В любом приложении вся структура моделируется как данные, которые обрабатываются определённым образом.

Модель даёт контроллеру представление данных, которые запросил пользователь (сообщение, страницу книги, фотоальбом, и тому подобное). Модель данных будет одинаковой, вне зависимости от того, как мы хотим представлять их пользователю. Поэтому мы выбираем любой доступный вид для отображения данных.

1. Что обеспечивает Вид?

**Вид** обеспечивает различные способы представления данных, которые получены из модели. Он может быть шаблоном, который заполняется данными. Может быть несколько различных видов, и контроллер выбирает, какой подходит наилучшим образом для текущей ситуации.