Министерство образования Республики Беларусь
Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационных компьютерных систем

Дисциплина "Объектно-ориентированное программирование"

К защите допустить:
Руководитель курсовой работы
старший преподаватель
кафедры
_____ А.В.Михалькевич
16.07.2025

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

Разработка сайта по подбору питания и тренировок.

БГУИР КР 1-40 05 01-10 № 121 ПЗ

Студент (подпись студента)

Курсовая работа
представлена на проверку
16.07.2025
_____
(подпись студента)

2025

Реферат

БГУИР КР 1-40 05 01-10 № 121 ПЗ, гр. 814303

, Разработка сайта по подбору питания и тренировок., Минск: БГУИР - 2025.

Пояснительная записка 38414 с., 4 рис., 0 табл.

Ключевые слова: сайт, Laravel, MVC, тренировки, курсы

Предмет Объектно-ориентированное программирование, А.В.Михалькевич

В данной курсовой работе поставлена цель создать удобный формат онлайн-курсов с понятным пользовательским интерфейсом и выбором тех тренировок, которые будут каждому по душе. Для достижения данной цели следует разработать несколько курсов, которые будут отражать цели пользователей, а также удобный интерфейс, в котором сможет разобраться как ребенок, так и пожилой человек. При разработке использовался веб-фреймворк Laravel, архитектурный шаблон MVC, а также Bootstrap шаблоны. Данный проект предназначен для людей, желающих вести здоровый образ жизни. https://github.com/LizaShukhta/online-course

-

Содержание

Введение

В настоящее время интернет позволяет развиваться всем сферам жизни начиная от магазинов до социальных сетей. Множество людей выбирают виртуальную жизнь реальной. Они знакомятся в интернете, делают покупки, учатся, играют в игры, посещают разные достопримечательности. Данный список можно продолжать бесконечно, однако следует отметить, что все эти действия чаще всего люди выполняют за экранами телефонов и компьютеров, находясь дома. Именно это и является мотивацией для многих тренеров начинать свою деятельность в онлайн форме, ведь физическая активность очень важна для правильной жизнедеятельности организма человека. Если человек ведет довольно активную жизнь, это еще не гарант хорошего самочувствия, поэтому различного вида тренировки должны присутствовать в жизни каждого человека наравне с правильным питанием и хорошим психологическим состоянием. Существует неизмеримое количество видеотренировок, сайтов по подбору питания, блогов людей, ведущих здоровый образ жизни, но для пользователя довольно сложно разобраться в этой информации, именно по этой причине огромную популярность набирают онлайн-курсы, которые позволяют совмещать в себе и тренировки, и питание, и круглосуточную поддержку тренеров. Онлайн-курсы для ведения здорового образа

жизни отличный вариант для людей, работающих из дому, для тех, кому не хватает времени на тренажерный зал, а также для людей, которым сложно общаться с незнакомыми людьми при личных встречах. Данные курсы охватывают огромную аудиторию людей разных возрастов, физических форм, полов и видов деятельности. Однако есть то, что их объединяет, а именно, желание быть здоровым. В данной курсовой работе поставлена цель создать удобный формат онлайн-курсов с понятным пользовательским интерфейсом и выбором тех тренировок, которые будут каждому по душе. Для достижения данной цели следует разработать несколько курсов, которые будут отражать цели пользователей, а также удобный интерфейс, в котором сможет разобраться как ребенок, так и пожилой человек.

1 ОПИСАНИЕ ПРОЕКТА

Курсовой проект представляет собой веб-сайт, который позволяет пользователям вести здоровый образ жизни посредством курсов – определённых тренировочных программ.

Каждый курс направлен на достижение 3 первостепенных целей: набор мышечной массы, снижение жировой массы, а также поддержание текущей формы с улучшением таких качеств, как выносливость, сила, скорость и мобильность. Авторизированным пользователям дается доступ к определённым видам тренировок, которые разрабатываются на основе данных целей.

Курс представляет собой совокупность тренировок, а также плана питания.

Каждая тренировка состоит из нескольких упражнений, направленных на развитие разных групп мышц.

Пользовательский интерфейс сайта очень минималистичен, за счет чего получить доступ к курсам может любой желающий.

Посетив главную страницу сайта, пользователю описываются основные аспекты, из которых состоят курсы.

Пользователи, желающие начать ведение здорового образа жизни, нажав на кнопку «Начать!» пересылаются на меню выбора, состоящего из 3 иконок, которые соответствуют 3 курсам.

Когда пользователь делает выбор курса, и, если он не авторизовался на сайте, ему предлагается это сделать. После авторизации, пользователя пересылает на страницу, где он может ввести первоначальные данные, такие как рост и вес, чтобы в дальнейшем отслеживать прогресс.

Далее пользователь может вернуться на страницу с курсами и выбрать тот, который ему подходит.

После этого ему на странице курса будет продемонстрированы упражнения и план питания.

Пользовательский интерфейс первоначальной страницы представлен на рисунке 1.

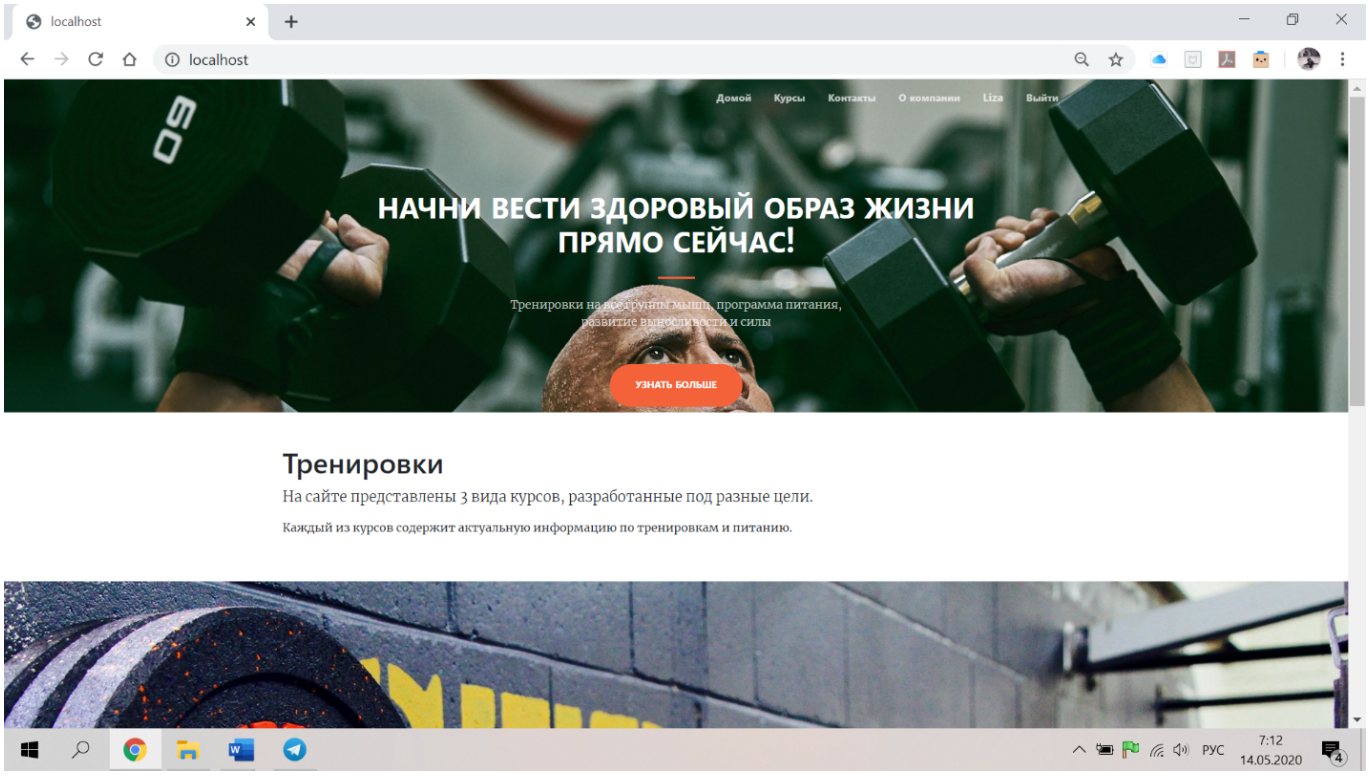


Рисунок 1 - Пользовательский интерфейс домашней страницы

Пользовательский интерфейс курсов представлен на рисунке 2.

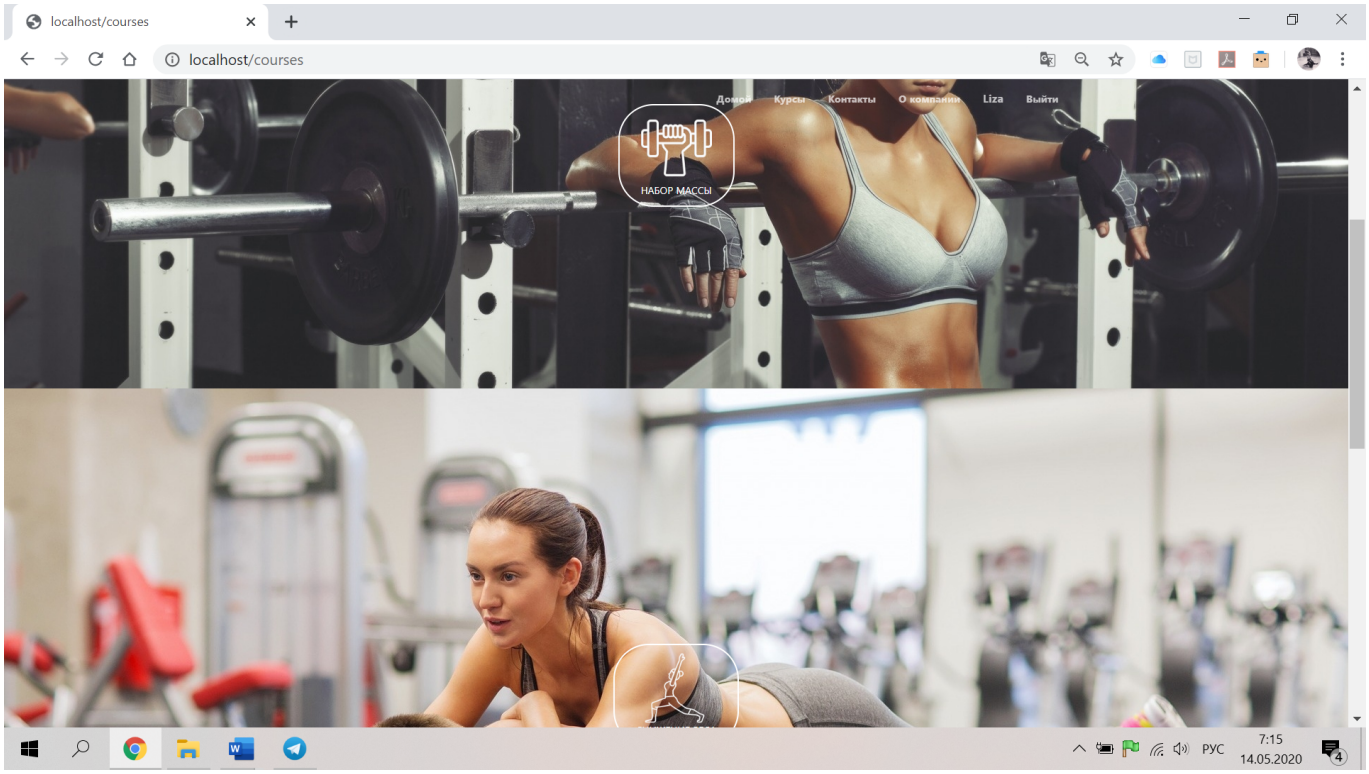


Рисунок 2 - Пользовательский интерфейс страницы курсов

Пользовательский интерфейс курса представлен на рисунке 3.

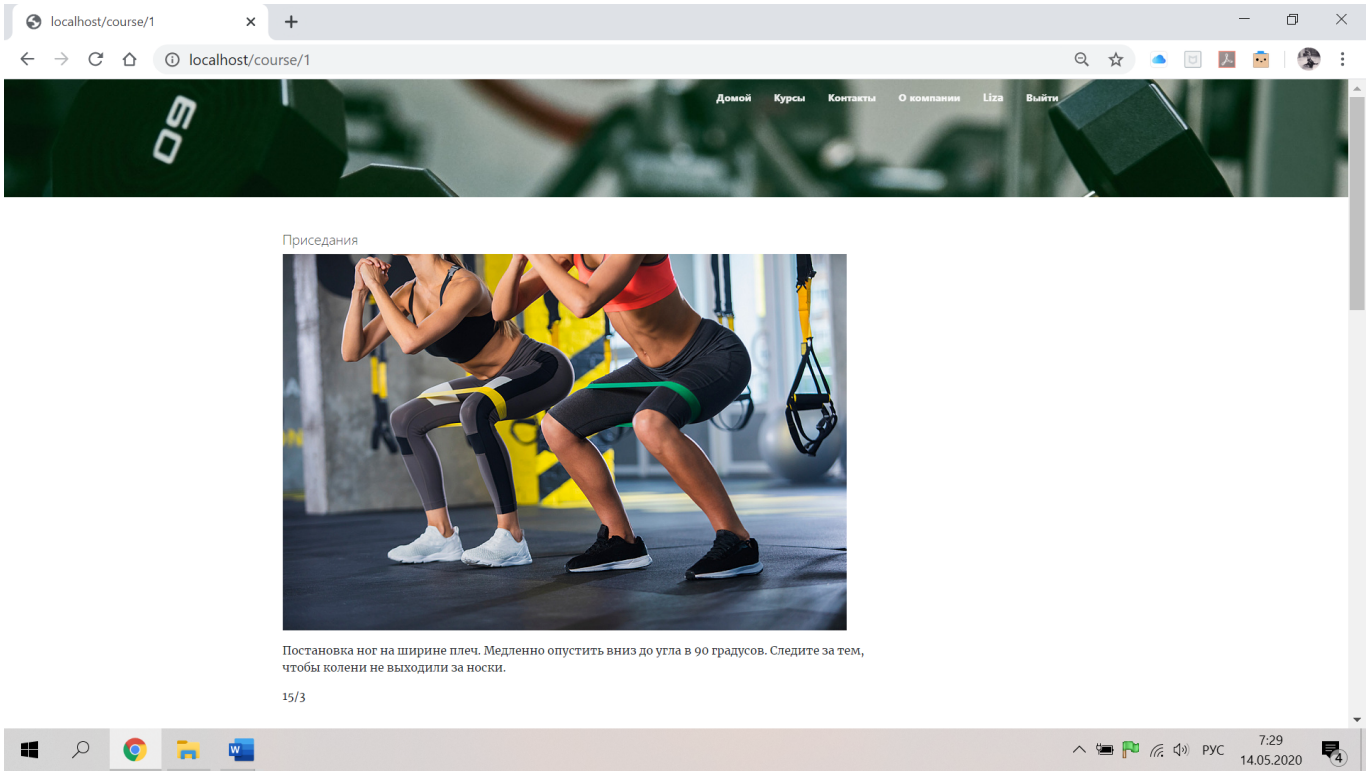


Рисунок 3 - Пользовательский интерфейс курса

Проект был реализован с помощью использования бесплатного веб-фреймворка с открытым кодом, предназначенным для разработки с использованием архитектурной модели MVC – Laravel.

Также был использован фреймворк Bootsrap - свободный набор инструментов для создания сайтов и веб-приложений.

В качестве языков программирования были использованы CSS - формальный язык описания внешнего вида документа, написанного с использованием языка разметки, HTML - стандартизированный язык разметки документов во Всемирной паутине, PHP - скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений.

Немалую роль при разработке веб-сайта сыграли такие инструментарии, как Windows 10 - операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT. и Notepad++ - свободный текстовый редактор с открытым исходным кодом для Windows с подсветкой синтаксиса большого количества языков программирования и разметки, а также языков описания аппаратуры VHDL и Verilog.

2 ОБОСНОВАНИЕ ВЫБОРА ТЕХНОЛОГИЙ

Для реализации проекта использовался бесплатный веб-фреймворк с открытым кодом, предназначенный для разработки с использованием архитектурной модели MVC – Laravel.

Причинами этого являются следующие ключевые особенности. Пакеты (англ. packages) — позволяют создавать и подключать модули в формате Composer к приложению на Laravel. Многие дополнительные возможности уже доступны в виде таких модулей. Eloquent ORM —

реализация шаблона проектирования ActiveRecord на PHP. Позволяет строго определить отношения между объектами базы данных. Стандартный для Laravel построитель запросов Fluent поддерживается ядром Eloquent. Логика приложения — часть разрабатываемого приложения, объявленная либо при помощи контроллеров, либо маршрутов (функций-замыканий). Синтаксис объявлений похож на синтаксис, используемый в каркасе Sinatra. Обратная маршрутизация связывает между собой генерируемые приложением ссылки и маршруты, позволяя изменять последние с автоматическим обновлением связанных ссылок. При создании ссылок с помощью именованных маршрутов Laravel автоматически генерирует конечные URL. REST-контроллеры — дополнительный слой для разделения логики обработки GET- и POST-запросов HTTP. Автозагрузка классов — механизм автоматической загрузки классов PHP без необходимости подключать файлы их определений в include. Загрузка по требованию предотвращает загрузку ненужных компонентов; загружаются только те из них, которые действительно используются. Составители представлений (англ. view composers) — блоки кода, которые выполняются при генерации представления (шаблона). Инверсия управления (англ. Inversion of Control) — позволяет получать экземпляры объектов по принципу обратного управления. Также может использоваться для создания и получения объектов-одиночек (англ. singleton). Миграции — система управления версиями для баз данных. Позволяет связывать изменения в коде приложения с изменениями, которые требуется внести в структуру БД, что упрощает развёртывание и обновление приложения. Модульное тестирование (юнит-тесты) — играет очень большую роль в Laravel, который сам по себе содержит большое число тестов для предотвращения регрессий (ошибок вследствие обновления кода или исправления других ошибок). Страничный вывод (англ. pagination) — упрощает генерацию страниц, заменяя различные способы решения этой задачи единым механизмом, встроенным в Laravel. Поддержка noSQL СУБД Redis. Множество готовых админ-панелей, шаблонов и CRUD. Шаблонизатор Blade (по умолчанию). Возможность подключать CSS шаблоны.

В качестве языка программирования был использован php. Вот перечень преимуществ, которые делают широко применимым его в веб-разработке: разработка с помощью PHP дает много возможностей. При должном уровне владения, с помощью шаблонизатора можно создавать не только сценарии для веб-приложений, но и полноценные программы. Существуют решения, позволяющие создавать мобильные приложения на PHP; изучение PHP не требует много времени. Это одновременно и плюс, и минус. Ведь основательное знание требует практики, но об этом позже; кроссплатформенность. PHP может быть запущен в любой операционной системе, включая юниксоиды; поддержка веб-серверов. Сложно найти тот, который бы не работал с PHP; бесплатное распространение. Возможно, PHP не был так популярен для создания web-приложений, если бы не был бесплатным, как и большинство инструментов для работы с ним. Аналоги, которые, в основном, могут выполнить ту же работу, обычно стоят дороже; имеет достаточную произвольность для web-разработки. Конечно, такие базовые языки, как C-семейство, работают быстрее, но для веба это не критично; наличие учебных материалов. Все знают о «косяках» PHP лишь потому, что разработку, в основном, ведут с его помощью. Попробуйте найти в Google недостатки «Virtual Reality Modeling

Language». Будет сложно, ведь его мало кто знает. Зато основу недостатков «препроцессора» уже все выучили наизусть из-за широкой используемости языка. Именно потому, если у вас что-то не получается, всегда можно заглянуть в поисковик: с вашей проблемой, вероятнее всего, кто-то уже сталкивался; непрерывное развитие. То, что сегодня о шаблонизаторе знают так много, означает лишь одно: с недостатками, рано или поздно, справятся.

Около 80% всех существующих web-приложений было создано на шаблонизаторе, и естественно, что в них были найдены ошибки. К тому же, низкий порог входа позволяет новичкам создавать масштабные продукты. Их «поделки» редко отличаются качеством, но все-же работают.

Также использовалась база данных MySQL, чьими преимуществами являются: простота в использовании. MySQL достаточно легко инсталлируется, а наличие множества плагинов и вспомогательных приложений упрощает работу с базами данных, обширный функционал. Система MySQL обладает практически всем необходимым инструментарием, который может понадобиться в реализации практически любого проекта, безопасность. Система изначально создана таким образом, что множество встроенных функций безопасности в ней работают по умолчанию, масштабируемость. Являясь весьма универсальной СУБД, MySQL в равной степени легко может быть использована для работы и с малыми, и с большими объемами данных, скорость. Высокая производительность системы обеспечивается за счет упрощения некоторых используемых в ней стандартов.

В качестве локального сервера использовался OpenServer, так он имеет ряд преимуществ: поставляется в 3 версиях, наличие бесплатных программ, частное обновление программы, портативная версия программы, богатые возможности программы, отличная документация и форум поддержки.

Также использование PhpMyAdmin, имеющий следующие достоинства: создавать и удалять базы данных, создавать, копировать, удалять, переименовывать и изменять таблицы, осуществлять сопровождение таблиц, удалять, править и добавлять поля, выполнять SQL-запросы, в том числе пакетные SQL-запросы, управлять ключами, загружать текстовые файлы в таблицы.

3 ИНСТРУМЕНТАРИЙ

Notepad++ - текстовый редактор, предназначенный для программистов и всех тех, кого не устраивает скромная функциональность входящего в состав Windows Блокнота.

Основные особенности программы: подсветка текста и возможность сворачивания блоков, согласно синтаксису языка программирования; поддержка большого количества языков (C, C++, Java, XML, HTML, PHP, Java Script, ASCII, VB/VBS, SQL, CSS, Pascal, Perl, Python, Lua, TCL, Assembler); WYSIWYG (печатаешь и получаешь то, что видишь на экране); настраиваемый пользователем режим подсветки синтаксиса; авто-завершение набираемого слова; одновременная работа с множеством документов; одновременный просмотр нескольких документов; поддержка регулярных выражений Поиска/Замены; полная поддержка перетягивания фрагментов текста; динамическое изменение окон просмотра; автоматическое

определение состояния файла; увеличение и уменьшение; заметки; выделение скобок при редактировании текста; запись макроса и его выполнение.

Git - распределённая система управления версиями.

Преимуществами являются: распределённая разработка; транзакционный подход в управлении пакетами; простота управления исходным кодом; простота и удобство создания патчей; интеграция в VCS апстрима, в том числе и по истории; прозрачная сборка как локально, так и на сервере (в последнем случае нужно лишь создать подписанный тег и задать команду на сборку по ssh); быстрое бэкпортирование; малый трафик при постановке на сборку в репозиторий; разнообразные проверки собираемых пакетов.

Разработка веб-сайта происходила на операционной системе Windows 10 - операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT. Плюсами являются: разнообразие лицензионных программ предназначенных для использования на данной ОС, удобна в использовании на десктопе.

4 АРХИТЕКТУРНЫЙ ШАБЛОН ПРОЕКТИРОВАНИЯ MVC

Идея, которая лежит в основе конструкционного шаблона MVC, очень проста: нужно чётко разделять ответственность за различное функционирование в наших приложениях.

Приложение разделяется на три основных компонента, каждый из которых отвечает за различные задачи.

Контроллер (Controller)

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.

Модель (Model)

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. Что такое пользователь для приложения — сообщение или книга? Только данные, которые должны быть обработаны в соответствии с правилами (дата не может указывать в будущее, e-mail должен быть в определённом формате, имя не может быть длиннее X символов, и так далее).

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.

Модель содержит наиболее важную часть логики нашего приложения, логики, которая решает задачу, с которой мы имеем дело (форум, магазин, банк, и тому подобное). Контроллер содержит в основном организационную логику для самого приложения (очень похоже на

ведение домашнего хозяйства).

Вид (View)

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.

Веб приложение обычно состоит из набора контроллеров, моделей и видов. Контроллер может быть устроен как основной, который получает все запросы и вызывает другие контроллеры для выполнения действий в зависимости от ситуации.

Разберём пример

Предположим, нам надо разработать онлайновый книжный магазин. Пользователь может выполнять следующие действия: просматривать книги, регистрироваться, покупать, добавлять пункты к текущему заказу, создавать или удалять книги (если он администратор). Давайте посмотрим, что произойдёт, когда пользователь нажмёт на категорию фэнтези для просмотра названий книг, которые имеются в нашем магазине.

У нас есть определённый контроллер для обработки всех действий, связанных с книгами (просматривать, редактировать, создавать и так далее). Давайте назовем его books_controller.php в нашем примере. Также нам нужна модель, например, book_model.php, которая обрабатывает данные и логику, связанные с позицией в магазине. В заключение, нам нужно несколько видов для представления данных, например, список книг, страница для редактирования и так далее.

Следующий рисунок показывает, как обрабатывается запрос пользователя для просмотра списка книг по теме фэнтези:

Контроллер (books_controller.php) получает запрос пользователя (запрос HTTP GET или POST). Мы можем организовать центральный контроллер, например, index.php, который получает запрос и вызывает books_controller.php.

Контроллер проверяет запрос и параметры, а затем вызывает модель(book_model.php), запрашивая у неё список доступных книг по теме фэнтези.

Модель получает данные из базы (или из другого источника, в котором хранится информация), применяет фильтры и необходимую логику, а затем возвращает данные, которые представляют список книг.

Контроллер использует подходящий вид для представления данных пользователю. Если запрос приходит с мобильного телефона, используется вид для мобильного телефона; если пользователь использует определённое оформление интерфейса, то выбирается соответствующий вид, и так далее.

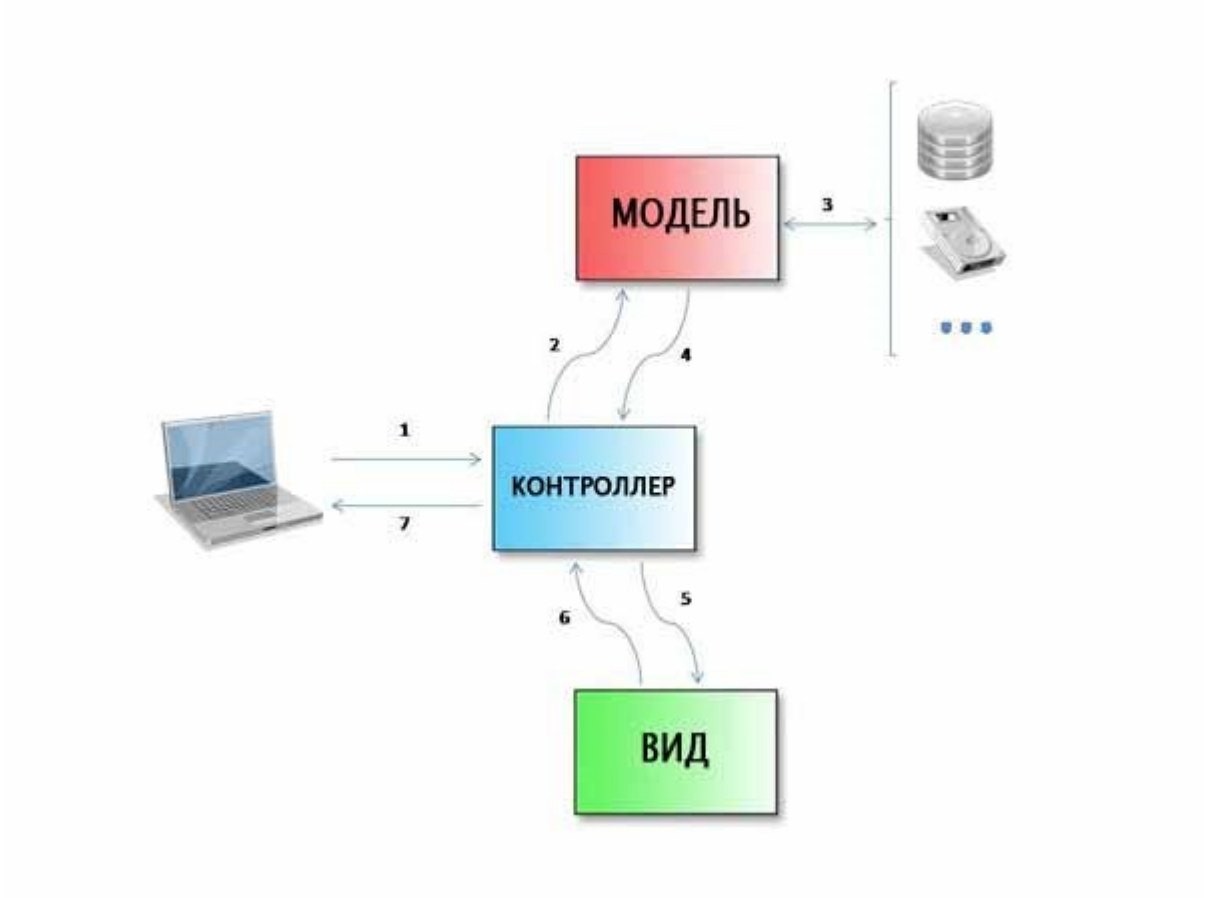
В чем преимущества?

Самое очевидное преимущество, которое мы получаем от использования концепции MVC — это чёткое разделение логики представления (интерфейса пользователя) и логики приложения.

Поддержка различных типов пользователей, которые используют различные типы устройств является общей проблемой наших дней. Предоставляемый интерфейс должен различаться, если запрос приходит с персонального компьютера или с мобильного телефона. Модель

возвращает одинаковые данные, единственное различие заключается в том, что контроллер выбирает различные виды для вывода данных.

Помимо изолирования видов от логики приложения, концепция MVC существенно уменьшает сложность больших приложений. Код получается гораздо более структурированным, и, тем самым, облегчается поддержка, тестирование и повторное использование решений.



5 РАЗРАБОТКА МОДУЛЯ ЗАПОЛНЕНИЯ ДАННЫХ ПОЛЬЗОВАТЕЛЯ

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Http\Requests\PeopleRequest;

use Auth;

use App\Person;

class HomeController extends Controller
```

```
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
       // $this->middleware('auth');
    }

    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Contracts\Support\Renderable
     */
    public function index()
    {
            $objs=Person::orderBy('id', 'DESC')->paginate(10);
        return view('home', compact('objs'));
    }

  public function getDelete($id=null)
    {
            $obj=Person::find($id);
            $obj->delete();
            return redirect()->back();
    }

  public function postIndex(PeopleRequest $r)
    {
            $r['user_id']=(Auth::guest())?'0':Auth::user()->id;
            Person::create($r->all());
            return redirect()->back();/* redirect(' '); */
    }
}



 Route::group(['middleware'=>['auth']], function(){
  Route::get('home/delete/{id}', 'HomeController@getDelete');
  Route::post('/home', 'HomeController@postindex');
```

```
});
```

Фасад в Laravel представляет собой единую точку доступа к цельному пакету, который состоит из нескольких классов (или любой другой цельной структуре, состоящей из частей). То есть, вместо того чтобы оперировать внутренними классами пакета, мы оперируем фасадом, который транслирует нужный функционал из нужных классов или классов пакета. Тем самым достигается инкапсуляция пакета, и уход от лишних зависимостей.

В формате Laravel фасад, по сути, это обертка, дающая более удобный доступ к классу. Вернее, даже не к классу, а к его алиасу в IoC. То есть, позволяет сжать синтаксис вызова класса, но имеет мало общего с моим представлением о фасаде.

Данный шаблон является структурным и отвечает за построение удобных в поддержке иерархий классов.

Laravel уже поставляется со множеством готовых фасадов, которые определены в файле config/app.php, в элементе массива aliases.

Фасады мы можем использовать в своих классах, предварительно подключив их с помощью конструкции use, после чего идет имя фасада.

Заключение

В современном мире создание веб-сайтов играет огромную роль на IT-пространстве. Эта необходимость связана со значением Интернета в жизни человека. В настоящее время, использование различных онлайн-курсов распространено повсеместно и плотно вошло в повседневную жизнь пользователей. В процессе выполнения курсового проекта были получены навыки работы с Laravel. Познакомилась и изучила языки программирования: CSS, HTML, php. Также ознакомилась с некоторыми шаблонами проектирования практических задач.

Список использованных источников

1. [url] **Электронный ресурс учебных дисциплин** http://erud.by/

Приложения