ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»

(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

РАЗРАБОТКА САЙТА «ЛЕКТОРИЙ НИУ БЕЛГУ»

Выпускная квалификационная работа

обучающегося по направлению подготовки 38.03.05 «Бизнес-информатика»

очной формы обучения, группы 12001507

Величко Алексея Владимировича

Научный руководитель:

ст. пр.

Немцев Сергей Николаевич

БЕЛГОРОД 2019

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На сегодняшний день большую популярность приобрели электронные образовательные ресурсы. Процесс информатизации современного общества сделал необходимость создания образовательными учреждениями собственных web-ресурсов, представляющих собой коммуникационный центр, который позволяет производить всевозможные манипуляции с информацией, направленной на решение проблем образовательного характера.

Исходя из этого, образовательное учреждение, стремящееся быть конкурентоспособным, а также желающее иметь привлекательный имидж и эффективную систему работы с информацией для обеспечения внутренних потребностей образовательного учреждения, сталкивается с проблемой создания своего интернет-представительства, а именно с созданием собственного web-ресурса. Именно здесь можно сообщить всё, что может оказаться полезным или интересным для родителей, нынешних и будущих учеников.

Объектом исследования выпускной квалификационной работы является процесс сопровождения образовательной деятельности Белгородского государственного национального исследовательского университета (далее – НИУ «БелГУ»).

Предметом исследования выступают средства информационной поддержки НИУ «БелГУ».

Целью выпускной квалификационной работы является повышение доступности дистанционных образовательных услуг при помощи сайта «Лекторий НИУ БелГУ».

Для достижения поставленной цели были поставлены следующие задачи:

1) изучить понятие и классификации web-сайтов;

2) провести анализ технологий, используемых при создании web-сайтов;

3) провести анализ деятельности НИУ «БелГУ» и на его основе разработать требования к будущему сайту;

4) разработать сайт «Лекторий НИУ БелГУ» и разместить его в сети;

5) оценить эффективность внедрения сайта.

1 Теоретические основы разработки web-сайта

1.1 Понятие и классификация Web-сайтов

Web-сайтом называется совокупность документов, а также других различных материалов организации или частных лиц, хранящаяся на web-сервере по определенному адресу, которым может являться какое-либо доменное имя или IP-адрес [30, с 21]. Изначально предполагается, что web-сайт располагается в компьютерной сети интернет. Каждый сайт в интернете представляет собой часть одной большой системы, именуемой всемирной паутиной. Для того, чтобы клиенты имели прямой доступ к сайтам, был разработан протокол HTML. Web-сайты в узких кругах нередко называют интернет-представительством организации или конкретного человека. В первую очередь это связано с тем, что в наши дни своя страница в сети скорее является обыденностью, а не чем-то инновационным и редким. Интернет-представительством может являться как отдельный сайт, так и определенная страница в составе других сайтов. Ярким примером интернет-представительств служат социальные сети, которые сами по себе не представляют конкретного человека или компанию, а являются совокупностью страниц.

Для того чтобы получить доступ к web-сайту, размещенному в сети интернет, необходимо открыть на компьютере (подключенном к сети) специальную программу, именуемую браузером, в котором необходимо указать адрес сайта. Как уже было упомянуто, адрес может представлять собой как доменное имя, так и IP-адрес, состоящий из набора цифр.

Чаще всего каждому web-сайту присвоено только одно доменное имя, так как доменное имя является основным идентификатором сайта во всемирной сети интернет. Однако есть исключения. Например, одному сайту может быть присвоено несколько доменных адресов или же наоборот несколько сайтов могут иметь одно доменное имя. Такую возможность

обычно используют крупные сайты для того, чтобы дать пользователю интуитивное и логическое понимание того, что каждый домен необходим для обозначения разного вида услуг, предоставляемых главным сайтом. В качестве примера можно привести сайт компании Google, который включает в себя различные виды услуг, такие как электронная почта, новостной сайт и электронные карты. Также отдельно можно выделить доменные имена, предназначенные для разных стран. В качестве примера, можно привести тот же Google, которому принадлежат такие адреса как google.fr, google.xu, логически являющиеся одним сайтом, но переведенным на разные языки. Доменные имена также могут иметь несколько уровней. Например, site.com является доменом первого уровня, а blog.site.com – второго и т.д.

Интернет-службы используют архитектуру клиент-сервер, т.е. если вы – пользователь, клиентская программа на вашем устройстве должна подключиться к серверу и отправить ему запрос на получение информации. Программа, находящаяся на сервере, в свою очередь, вернет эту информацию и будет ждать следующего запроса.

По доступности сервисов:

– открытые – все сервисы полностью доступны для любых пользователей;

– полуоткрытые – для доступа к информации необходима регистрация (обычно бесплатная);

– закрытые – полностью закрытые служебные сайты организаций (в том числе корпоративные сайты), личные сайты частных лиц. Такие ресурсы доступны для определенного круга лиц. Доступ новых пользователей обычно даётся через приглашения.

По природе содержимого:

– статические – всё содержимое заранее подготавливается. Пользователю предоставляются файлы в том виде, в котором они находятся на сервере;

– динамические – содержимое генерируется специальными скриптами (программами) на основе других данных из любого источника.

По физическому расположению:

– внешние сайты сети Интернет;

– локальные сайты – доступны только в пределах локальной сети.

По схеме представления информации, её объёму и категории решаемых задач можно выделить следующие типы web-ресурсов:

– корпоративные сайты;

– интернет-магазины;

– информационные сайты;

– сайты-форумы;

– web-сервисы.

Корпоративный сайт – это сайт компании, ее официальное виртуальное представительство в сети Интернет. На корпоративном сайте размещается полный объем информации о самой фирме, о сфере ее деятельности, предлагаемой продукции и услугах. Часто на корпоративном сайте размещают каталог производимой продукции и дополнительные сервисы – форум, опросы, рассылки и т. д.

Корпоративный сайт позволяет выдавать онлайн расчёты стоимости, производить продажу продукции, проводить маркетинговые исследования и рекламные акции, рассылать посетителям новости, собирать отзывы клиентов, устраивать голосования и многое другое.

Особое внимание уделяется дизайну, ведь корпоративный сайт – это в первую очередь имиджевый инструмент. Он должен соответствовать стилю компании, вызывать доверие у потенциальных клиентов, партнеров и прессы.

В современном мире наличие корпоративного сайта является неотъемлемым для любой серьезной компании. Ведь сайт такого типа – это очень эффективное средство для предоставления в интернете информации об организации.

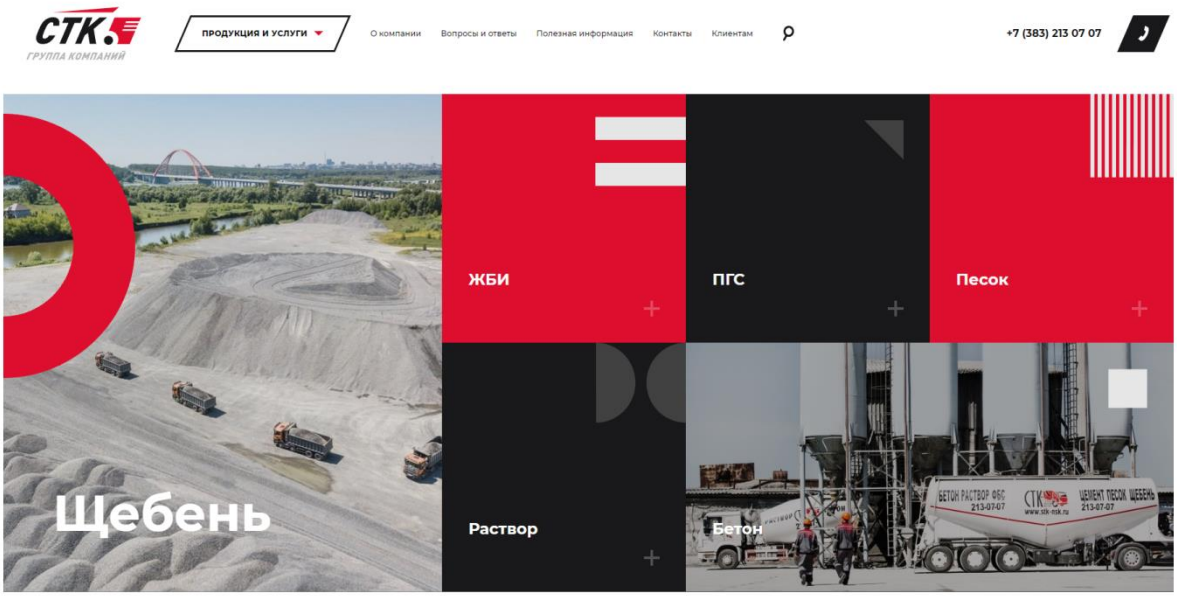

Рисунок 1 – Пример корпоративного сайта компании «СТК»

Интернет магазин – (англ. online shop) интерактивный web сайт, рекламирующий товар или услугу, принимающий заказы на покупку, предлагающий пользователю выбор варианта расчета, способа получения заказа и выписывающий счет на оплату [5, с. 85].

Клиент интернет магазина, просматривая ассортимент, представленный на сайте, выбирает подходящий ему товар и добавляет его в так называемую «корзину». Затем, по такому же принципу, покупатель может добавить еще несколько товаров или удалить их. После этого, когда покупатель определился с выбором, ему предлагается перейти к оформлению заказа.

В заявке на покупку товара кроме информации о выбранном товаре содержится также информация и о способах оплаты. Интернет-магазин подразумевает различные способы оплаты, пользователь выберет наиболее удобный для него. Это могут быть электронные деньги, оплата банковской картой или банковским переводом, а также оплата наличными при получении товара курьером или в почтовом отделении. Однако во избежание случаев мошенничества интернет магазины не используют, например, переводы Western Union, так как такой платеж нельзя отменить или опротестовать. Также не рекомендуется использовать оплату через SMS на короткий номер с

неизменяемой суммой платежа, потому что в этом случае также невозможно будет опротестовать или отменить платеж.

Доставка заказа может осуществляться либо собственной курьерской службой магазина, либо при помощи других организаций, предоставляющих курьерские услуги. Также возможен вариант доставки заказа покупателю по почте посылкой или бандеролью. В случае если заказанный товар электронный (фотография, программное обеспечение, лицензионные ключи к нему, текст, электронная книга и т.д.), то его доставка осуществляется по электронным каналам, например, по электронной почте или посредством доступа в защищенную область сайта.

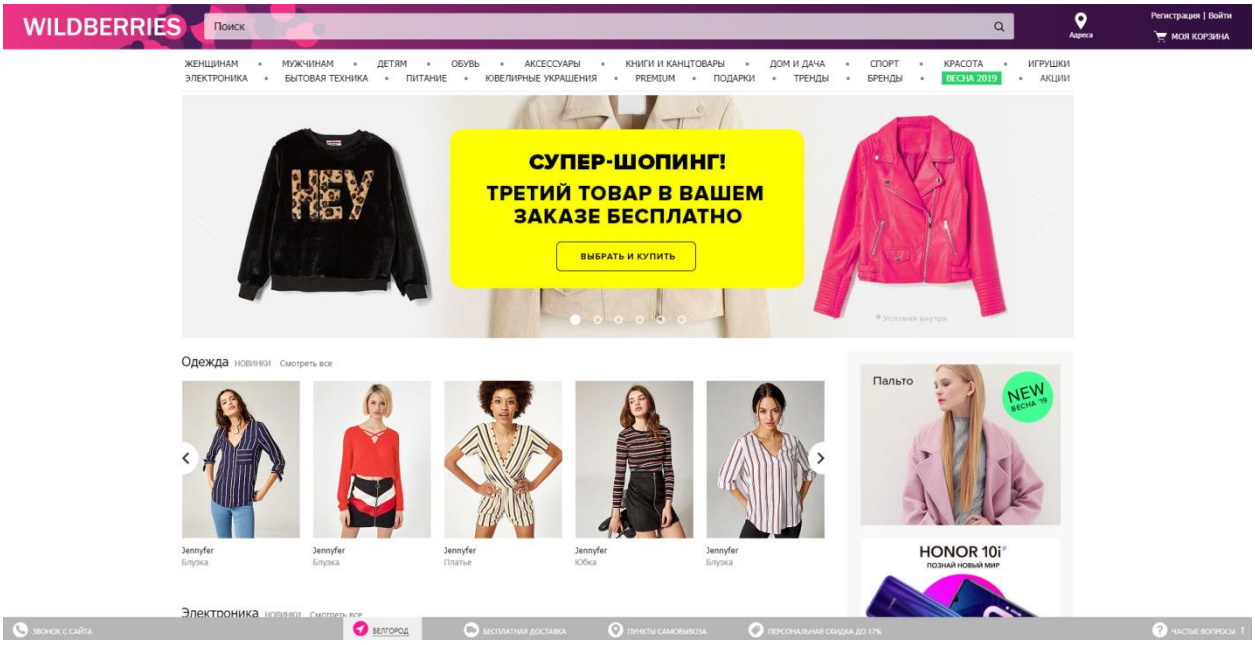

Рисунок 2 – Главная страница интернет магазина «Wildberries»

Информационный сайт – ресурс, на котором опубликованы инфоматериалы определенной тематики, предназначенные для ознакомления с ними читателей [5, с. 111].

Инфосайт может решать разные задачи. Например, общественная организация может использовать такой ресурс для размещения информации о деятельности объединения. Коммерческая компания может вести свой личный блог и рассказывать о своей деятельности. Простой обыватель может

создать такой сайт, чтобы делиться своими мыслями, советами с другими пользователями сети и, к тому же, зарабатывать при этом.

Контент на информационном сайте должен быть актуальным. Его нужно постоянно обновлять, дополнять и следить за качеством. Это требует времени, сил и денег.

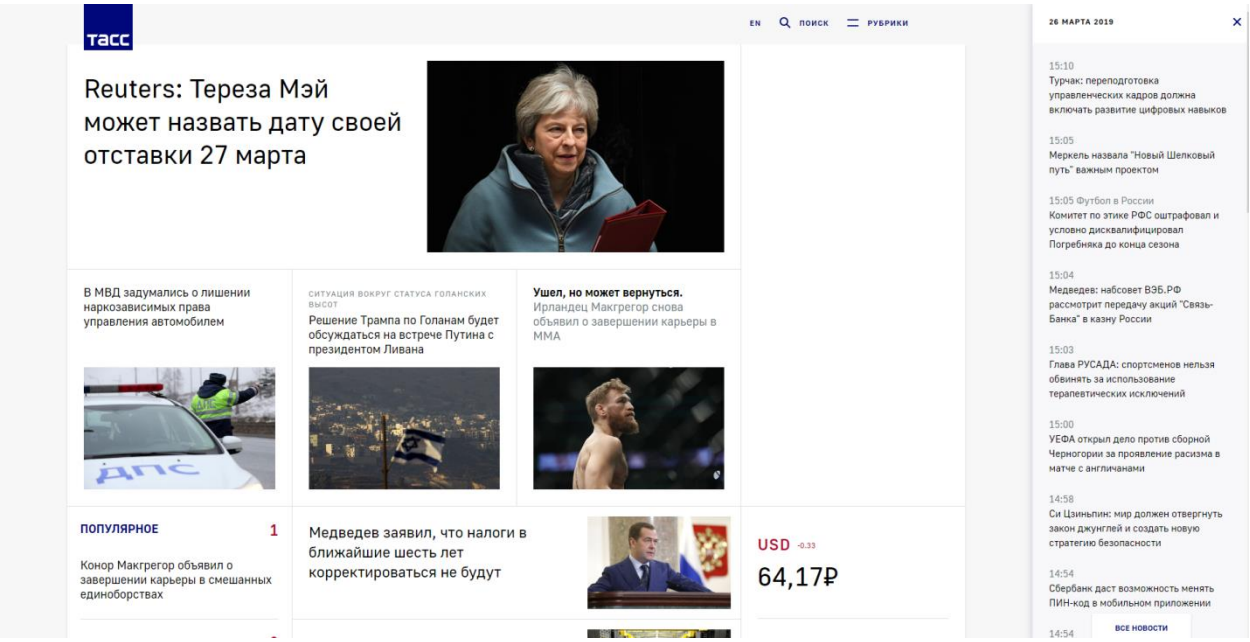

Рисунок 3 – Главная страница информационного ресурса «ТАСС»

Форум – это сайт, на котором создаются темы, в которых посетители общаются между собой, обсуждая какую-то тематику. Форумы бывают различных тематик: автомобильные, бухгалтерские, игровые, инвестиционные, трейдинговые, банковские, детские и т.д. [5, с. 151].

Основным отличием форума от других сервисов общения, например, того же самого чата является его выдержанность во времени при ответе на сообщения, т.е. человек может долго думать, прежде чем напишет ответ. Хотя бывают и исключения.

Форумы представляет собой сообщество вокруг данного сайта и его товаров и услуг. Такие сообщества благоприятно влияют на интерактивность сайта и повышают уровень цитируемости сайта в поисковых системах, что повышает популярность всего ресурса.

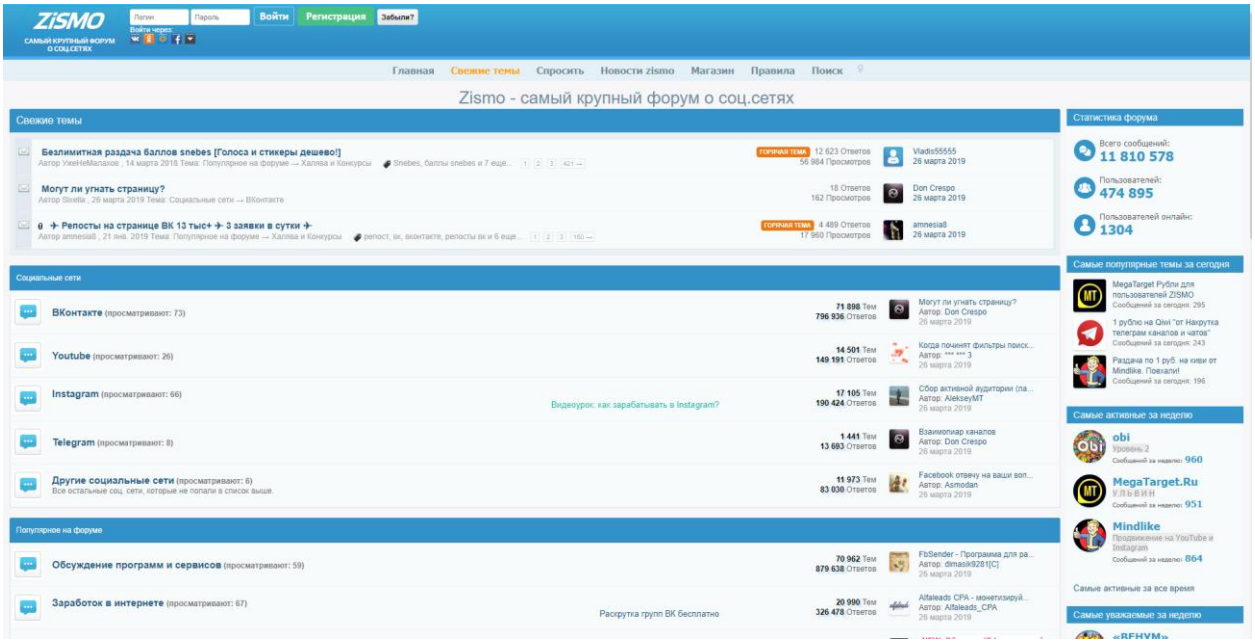

Рисунок 4 – Главная страница форума о социальных сетях «Zismo»

Web-сервис (служба) – программа, которая организовывает взаимодействие между сайтами. Информация с одного портала передается на другой [5, с. 183].

Например, есть авиакомпания. У нее много рейсов, соответственно, много билетов. Информацию через web-службу она передает сайту-агрегатору тур-путешествий. Пользователь, который заходит на агрегатор, сможет прямо там купить билеты этой авиакомпании.

Другой пример web-сервисов – это сайт отслеживания погоды, который содержит сведения о метеоусловиях в конкретном городе или по стране в целом. Данная информация также часто используется сторонними приложениями.

Информация в интернете разнородна. Сайты управляются разными системами. используются разные протоколы передачи и шифрования. Web-сервисы упрощают обмен информацией между разными площадками.

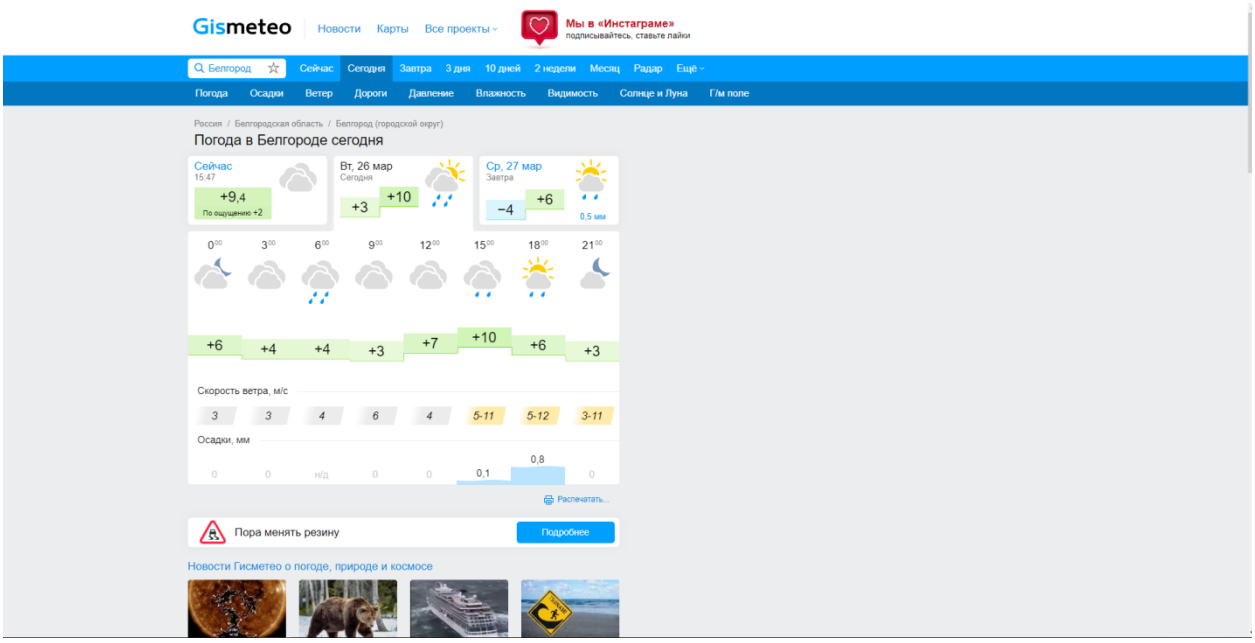

Рисунок 5 – Главная страница web-сервиса «Gismeteo»

1.2 Анализ технологий создания web-сайтов

Для создания современных сайтов обычно используются: HTML, CSS, JavaScript, XML, PHP, а также другие средства и программы.

Язык гипертекстовой разметки (HTML) Hyper Text Markup Language стандартный язык разметки документов во Всемирной паутине. Интерпретируется специальными приложениями, которые называются браузерами или web-обозревателями (web browser), после чего обработанная информация отображается в отформатированном виде на конечном устройстве. HTML состоит из специальных команд (тегов), которые предназначены для форматирования содержимого html-документа браузерами. У каждого такого тега есть свой набор параметров, которые называются атрибутами данного тега. При помощи языка HTML можно создать статический web-сайт, т.е. сайт, состоящий из статических html-страниц. Однако у таких сайтов есть определенные недостатки: для наполнения сайта информацией требуется вносить изменения в файлы сайта, что требует от администратора определенных навыков и умений, отсутствие взаимодействия с пользователем [29, с. 36].

CSS (каскадные таблицы стилей) – формальный язык описания внешнего вида документа. Каскадные таблицы стилей определяют свойства элементов и указывают, в каком виде элементы будут отображены на экране пользователя.

Существуют несколько стандартов CSS. CSS-1 – первый стандарт таблиц стилей. Данный стандарт допускает использование CSS-селекторов, осуществляющие выбор того или иного элемента или группы элементов, для применения определенного стиля, осуществляет создание блоков для работы с размерами шрифтов, отступов, полей и рамок, выполняет позиционирование блоков, позволяет управлять фоном и стилями текста, а также создавать нумерованные и маркированные списки.

CSS-2 – второй стандарт каскадных таблиц стилей. Данный стандарт основан на CSS-1 и дополняет его некоторыми новыми свойствами, например, изменилась блоковая модель, благодаря чему стало возможным размещать блоки на странице с более четкой точностью, а также делать их скрытыми [29, с. 157].

На сегодняшний день разработана новая спецификация каскадных таблиц стилей CSS-3. Она значительно расширяет функционал старого стандарта. В CSS-3 была введена так называемая модуляризация – теперь появилась возможность разделять таблицы стилей на модули. Также появилась возможность применения большого количества функций: создание градиентов, теней, фильтров, анимации и визуальных эффектов, скругление углов. К сожалению, даже по сей день, еще не все браузеры поддерживают CSS-3, поэтому сайт следует адаптировать под все браузеры.

JavaScript – язык сценариев или скриптов. Скрипт представляет собой программный код – набор инструкций, который не требует предварительной обработки (например, компиляции) перед запуском. Код JavaScript интерпретируется движком браузера во время загрузки web-страницы. Интерпретатор браузера выполняет построчный анализ, обработку и выполнение исходной программы или запроса [25, с. 314].

Области применения языка JavaScript:

– создание интерактивных страниц, содержимое которых может меняться в зависимости от пользователя без перезагрузки страницы;

– отправка данных на сервер без перезагрузки страницы;

– проверка правильности заполнения пользователем форм до пересылки их на сервер;

– решение «локальных» задач с помощью сценариев и некоторые другие сферы.

XML (eXtensible Markup Language) – это расширяемый язык разметки, предназначенный для описания в текстовой форме структурированных данных. Этот текстовый формат, во многом схожий с HTML, разработан специально для хранения и передачи данных.

XML позволяет описывать и передавать такие структурированные данные, как:

– отдельные документы;

– метаданные, описывающие содержимое какого-либо узла Internet;

– объекты, содержащие данные и методы работы с ними;

– отдельные данные;

– всевозможные web-ссылки на информационные и людские ресурсы в Интернете.

Язык XML легко читаем и довольно прост для понимания. Исходный текст XML-документа представляет собой набор XML-элементов, каждый из которых содержит начальный и конечный тэги. Каждая пара тэгов представляет часть данных. То есть, как и HTML, язык XML для описания данных использует тэги. Однако, в отличие от HTML, XML позволяет применять неограниченный набор пар тэгов, каждая из которых представляет не то, как заключенные в нее данные должны выглядеть, а то, что они означают.

Для разработки web-сайта «Лекторий НИУ БелГ» был выбран язык программирования PHP, поэтому рассмотрим его возможности, достоинства и недостатки подробнее.

PHP (Hypertext PreProcessor, препроцессор гипертекста) – язык программирования, исполняемый на стороне web-сервера, спроектированный в качестве инструмента создания динамических и интерактивных web-сайтов. Данный язык оказался достаточно гибким и мощным, поэтому приобрёл большую популярность и используется в проектах любого масштаба: от простого блога до крупнейших web-приложений в Интернете [29, с. 193].

Задачей языка PHP является обеспечение эффективной связи web-ресурса с сервером и базами данных. При этом данный язык необычайно прост в освоении и применении. По сути, все, что необходимо знать для начала изучения PHP и программирования на нем, это самый простой язык web-разметки HTML. PHP прекрасно сочетается с HTML-кодом. Однако для выполнения сценариев одного браузера недостаточно. Необходим web-сервер. Код, написанный на PHP, направлен на выполнение двух задач: HTML-часть отвечает за внешний вид и отображение информации, а PHP-часть, интегрированная в HTML, обеспечивает возможности интерактивности и динамику.

Преимущества PHP:

– является свободным программным обеспечением, распространяемым под особой лицензией (PHP license) и не требует специализированного программного обеспечения;

– имеет достаточно низкий порог вхождения;

– имеет широкое коммьюнити разработчиков

– поддерживает большинство современных СУБД;

– имеется огромное количество готовых библиотек и расширений языка;

– может использоваться в изолированной среде;

– предлагает нативные средства организации web-сессий, программный интерфейс расширений;

– является довольно полной заменой проприетарной среды ASP (Active Server Pages) от Microsoft;

– может быть развёрнут почти на любом сервере;

– портирован под большинство аппаратных платформ и операционных систем.

Недостатки PHP:

– не подходит для разработки десктопных приложений или системных компонентов;

– имеет слабые средства для работы с исключениями;

– глобальные параметры конфигурации влияют на базовый синтаксис языка, что затрудняет настройку сервера и разворачивание приложений;

– объекты передаются по значению, что смущает многих программистов, привыкших к передаче объектов по ссылке, как это делается в большинстве других языков;

– web-приложения, написанные на PHP, довольно часто имеют проблемы с безопасностью.

По данным StackOverflow, язык PHP на начало 2019 года занимает 9 место из 25 в рейтинге популярности языков программирования среди разработчиков, полный рейтинг представлен на рисунке 6. Количество опрошенных респондентов – 73428 человек.

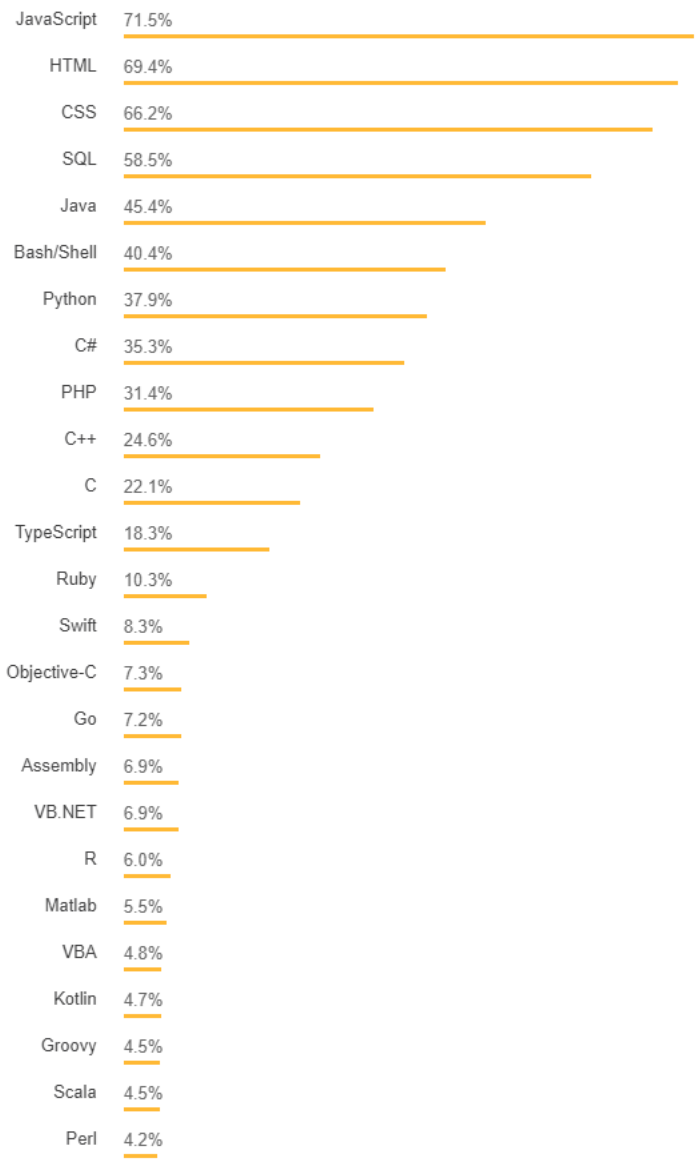

Рисунок 6 – Рейтинг популярности языков программирования от StackOverflow на начало 2019 года

Таким образом, мы изучили понятие web-сайта, особенности его функционирования и виды, рассмотрели наиболее популярные языки программирования, их достоинства и недостатки, исходя из чего выбрали наиболее подходящий язык для реализации web-ресурса «Лекторий НИУ БелГУ».

2 Анализ образовательной деятельности организации

2.1 Краткая характеристика образовательной деятельности НИУ «БелГУ»

Белгородский государственный национальный исследовательский университет – это научно-образовательный комплекс, обеспечивающий качественную подготовку кадров широкому спектру медицинских, социальных, гуманитарных, экономических, управленческих, филологических, физико-математических и информационных специальностей в центральной части России. С 1996 года является государственным университетом, а в 2010 получил статус «Национальный исследовательский университет». На текущий момент ректором университета является профессор, доктор политических наук Полухин Олег Николаевич. Юридический адрес: Россия, Белгород, индекс – 308015, улица Победы 85. Сайт университета расположен по адресу: http://www.bsu.edu.ru.

НИУ «БелГУ» состоит из 7 институтов и 3 факультетах, не входящих в состав институтов, 2 колледжа, Старооскольский филиал; 105 кафедр, 24 из которых базовые; более 600 предприятий, являющимися базами практик; около 1100 докторов и кандидатов наук, 14 академиков и членов-корреспондентов наук.

Структура учебных единиц НИУ «БелГУ»:

- институт юридический;
- институт педагогический;
    1) факультет физической культуры;
    2) факультет дошкольного, начального и специального образования;
    3) факультет историко-филологический;
    4) факультет математики и естественнонаучного образования;
    5) факультет иностранных языков;

6) факультет психологии;

– институт медицинский;

1) медицинский колледж;

2) центр дополнительного медицинского и фармацевтического образования, аккредитации и сертификации;

– институт фармации, химии и биологии;

– институт межкультурной коммуникации и международных отношений;

– институт экономики и управления;

– институт наук о земле;

– институт общественных наук и массовых коммуникаций;

– институт инженерных и цифровых технологий;

– подготовительный факультет;

– инжиниринговый колледж.

В рамках концепции непрерывного образования НИУ «БелГУ» осуществляет обучение по образовательным программам высшего образования и среднего профессионального образования, дополнительным профессиональным программам послевузовского медицинского и фармацевтического образования в интернатуре. Образовательная деятельность в ВУЗе ведется по 180 направлениям подготовки бакалавриата, магистратуры и специалитета, около 300 дополнительных профессиональных программ повышения квалификации и переподготовки.

Главным информационным порталом, регулирующим учебную деятельности студентов университета, является официальный сайт НИУ «БелГУ», указанный выше. На нем расположена вся необходимая информация для осуществления учебного процесса студентами, аспирантами, выпускниками: расписание занятий, основные образовательные программы, компьютерный каталог научной библиотеки, план университетских мероприятий, информация о преподавателях и др.

Кратко рассмотрим автоматизированные системы, используемые в образовательной деятельности НИУ «БелГУ»:

– «ИнфоБелГУ: Учебный процесс»: предназначена для автоматизации управления учебным процессом, планирования и организации проведения учебной деятельности подразделениями университета, основана на модульной динамической учебной среде «Moodle»;

– «ИнфоБелГУ: Университет»: автоматизированная система учета обучающихся, кадрового учета работников, архивные сведения системы управления учебным процессом;

– «ИнфоБелГУ: Социально-воспитательная работа»: автоматизированная система управления социально-воспитательной работой, предназначенная для автоматизации деятельности кураторов академических групп и проведения анализа данных, показателей и результатов социально-воспитательной деятельности подразделений университета;

– СЭО «Пегас»: система электронного обучения «Пегас» на основе модульной динамической учебной среды «Moodle», предназначенная для автоматизации процессов управления обучением, предоставления доступа к электронному образовательному контенту и реализации электронных образовательных технологий;

– «ИнфоБелГУ: Документооборот/Дело»: автоматизированная система управления документооборотом на основе САДЭД «Дело», предназначенная для автоматизации делопроизводства и обеспечения электронного документооборота университета;

– «ИнфоСЭД: Управление документами»: автоматизированная информационная система управления проектной деятельностью и организации контроля исполнения поручений;

– «Антиплагиат.Вуз»: информационная система анализа текстов на наличие заимствований, предназначенная для проверки письменных работ студентов университета с целью выявления заимствований с учетом результатов, выданных системой;

– «МегаПро: Библиотека»: автоматизированная интегрированная библиотечная система, предназначенная для комплексной автоматизации библиотечных процессов, управления информационными ресурсами и организации доступа к ним на основе web-технологий;

– информационная система мониторинга использования оборудования (ИСМИО) предназначена для аналитического учета приобретенного учебного, научного и опытно-экспериментального оборудования, ведения электронных журналов учета рабочего времени оборудования и оценки затрат на содержание и эксплуатацию данного оборудования.

Также, в процессе обучения студенты могу воспользоваться электронными услугами корпоративной библиотечной системы НИУ «БелГУ», например, заказ книг в электронном каталоге научной библиотеки НИУ «БелГУ», услугой межбиблиотечного абонемента (МБА) «Доставка Документов» (ДД), подразумевающей доставку сканированных текстов пользователям посредством Интернет-технологий, а также справочной службой библиотеки, где можно узнать наличие интересующей книги или уточнить различные исторические факты. Кроме того, в образовательном процессе студентов ВУЗа для осуществления обратной связи с преподавателями используется электронная почта НИУ «БелГУ».

Ниже представлен полный перечень web-ресурсов интрасети «БелГУ», используемые при осуществлении образовательной деятельности:

– web-сайт «Научная библиотека им. Н.Н. Страхова НИУ «БелГУ»;

– web-сайт «Система электронного обучения НИУ «БелГУ» «ПЕГАС»;

– электронная версия университетской газеты «ВЕСТИ БелГУ»;

– web-сайт «Студенческий портал»;

– web-сайт юридического института;

– web-сайт педагогического института;

– web-сайт факультета физической культуры;

– web-сайт факультета дошкольного, начального и специального образования;

– web-сайт историко-филологического факультета;

– web-сайт факультета математики и естественнонаучного образования;

– web-сайт факультета иностранных языков;

– web-сайт факультета психологии;

– web-сайт кафедры педагогики;

– web-сайт «Медицинский колледж БелГУ»;

– web-сайт института межкультурной коммуникации и международных отношений;

– web-сайт факультета горного дела и природопользования;

– web-сайт института экономики и управления;

– web-сайт «Высшая школа управления»;

– web-сайт факультета журналистики;

– web-сайт социально-теологического факультета;

– web-сайт института инженерных и цифровых технологий;

– web-сайт «Инжиниринговый колледж»;

– web-сайт «Центр профессиональной карьеры и организации практик»;

– web-сайт «Центр компьютерного обучения БелГУ»;

– web-сайт «Научно-образовательный и инновационный Центр «Наноструктурные материалы и нанотехнологии»;

– web-сайт «Федерально-региональный центр аэрокосмического и наземного мониторинга объектов и природных ресурсов»;

– web-сайт «Региональный центр интеллектуальной собственности (РЦИС)»;

– web-сайт «Учебно-спортивный комплекс С. Хоркиной»;

– web-сайт «Поликлиника НИУ «БелГУ»;

– web-сайт профсоюзного комитета;
– web-сайт студенческого научного общества;
– web-сайт 130-летия «БелГУ» (2006 г.);
– web-сайт 140-летия «БелГУ» (2016 г.);
– web-сайт геолого-минералогического музея
– web-сайт музея истории;
– web-сайт «Курская битва глазами современников и потомков»;
– web-сайт «Огненная Дуга»;

2.2 Анализ и моделирование бизнес-процессов НИУ «БелГУ»

Основным бизнес-процессом организации является предоставление образовательных услуг. Схема IDEF0 данного бизнес-процесса представлена на рисунке 7.

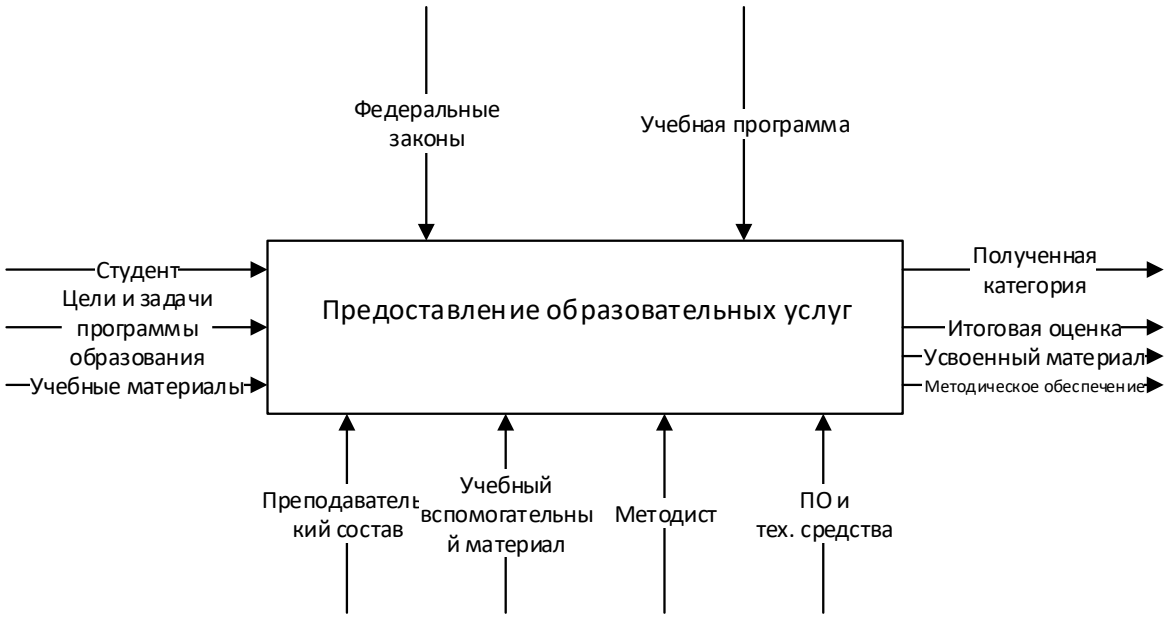

Рисунок 7 – Бизнес-процесс предоставления образовательных услуг

На входе: студент и цели и задачи программы образования. Управляющие воздействия: федеральные законы и учебная программа. Механизмы: преподавательский состав, учебный вспомогательный материал, программное обеспечение и технические средства. На выходе: итоговая оценка, полученная категория, методическое обеспечение и усвоенный материал. Декомпозиция данного процесса представлена на рисунке 8.

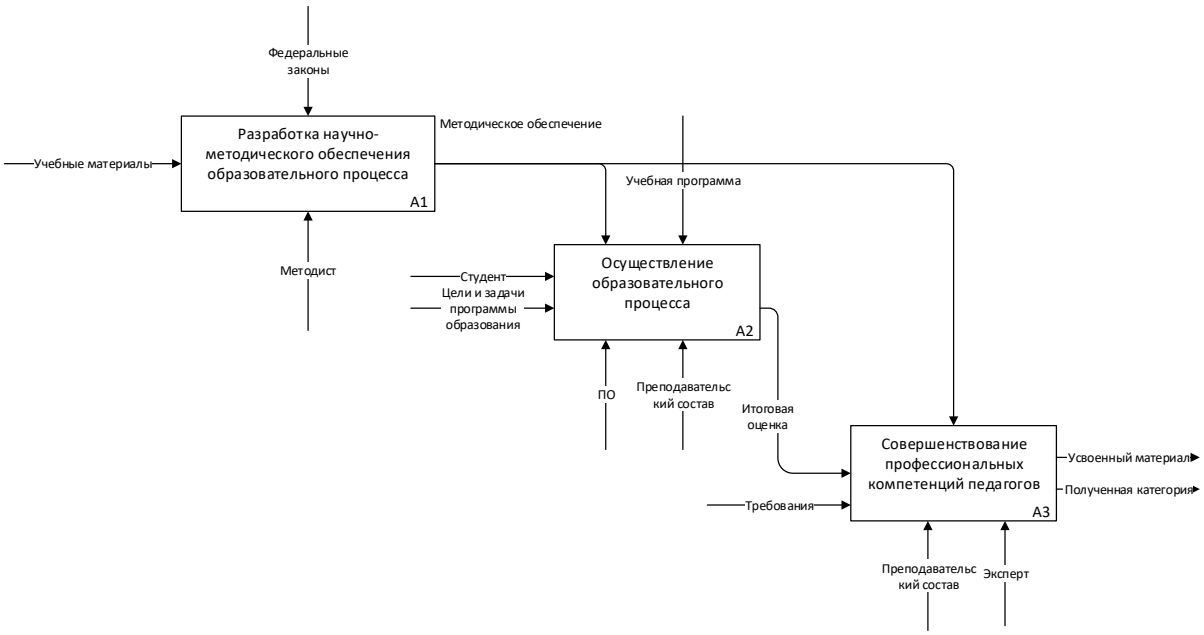

Рисунок 8 – Декомпозиция бизнес-процесса предоставления образовательных услуг

На входе процесса «Разработка научно-методического обеспечения образовательного процесса» учебные материалы, управляющим воздействием являются федеральные законы, а механизмом – методист. На выходе данного процесса – методическое обеспечение.

На входе следующего процесса «Осуществление образовательного процесса» студент и цели и программы образования. Управляющие воздействия – учебная программа и методическое обеспечение, механизмы – программное обеспечение и преподавательский состав, на выходе – итоговая оценка.

На входе процесса «Совершенствование профессиональных компетенций педагогов» требования ВУЗа и итоговые оценки студентов. Управляющее воздействие – методическое обеспечение, механизмы – преподавательский состав и эксперт в данной области, на выходе – усвоенные знания и полученная категория.

Также, для координации работы образовательного процесса учреждение проводит планирование учебного процесса, то есть распределение нагрузки между преподавателями. Диаграмма DFD данного бизнес-процесса представлена на рисунке 9.

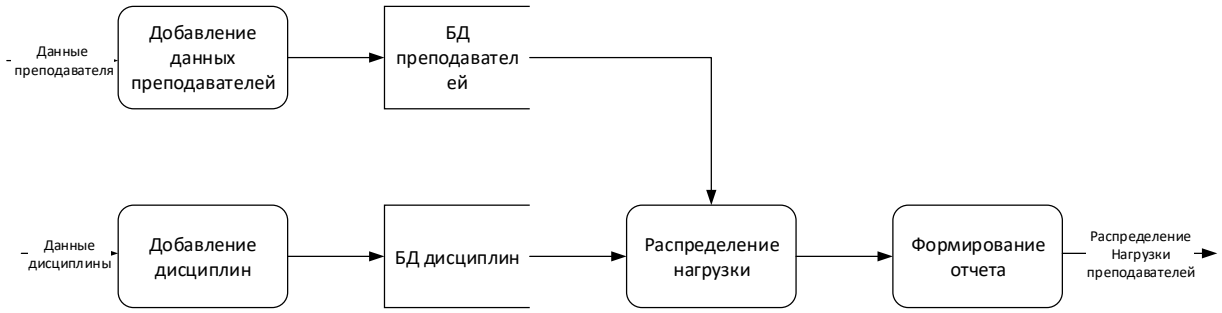

Рисунок 9 – Бизнес-процесс планирования учебного процесса ВУЗа

На основе информации о дисциплинах и преподавателях происходит заполнение соответствующими данными накопителей данных «База данных преподавателей» и «База данных дисциплин», после чего запускается процесс «Распределение нагрузки», затем «Формирование отчета», в котором отображаются окончательные данные касательно распределения нагрузки по дисциплинам между преподавателями.

Таким образом, мы изучили особенности образовательной деятельности НИУ «БелГУ», рассмотрели, какие автоматизированные системы и web-ресурсы используются в ВУЗе, а также разобрали и отобразили с помощью диаграмм IDEF0 и DFD основные бизнес-процессы, протекающие в данной организации.

3 Разработка сайта «Лекторий НИУ БелГУ»

3.1 Разработка требований к сайту «Лекторий НИУ БелГУ»

Наименование предметного продукта – «Лекторий НИУ БелГУ». Данный интернет ресурс разрабатывается для Белгородского Государственного национального исследовательского университета.

Назначение сайта «Лекторий НИУ БелГУ»:

– предоставление пользователям информации в рамках учебной программы (лекций) в видео-формате.

Целевой аудиторией сайта являются пользователи преимущественно 16-30 лет, проживающие на территории России, обучающиеся в НИУ «БелГУ», либо желающие стать студентом данного ВУЗа. Доступ к лекциям имеют как преподаватели и студенты НИУ «БелГУ», так и обычные пользователи.

При разработке указанного web-ресурса должны использоваться цвета, совпадающие с цветовой гаммой официального сайта НИУ «БелГУ», основные разделы должны быть доступны с первой страницы, следует избегать большого объема текстовой информации. Оформление должно быть разработано в консервативном ключе.

Для разрабатываемого web-ресурса должны быть доступны 2 класса пользователей: гость и администратор.

Гость – неавторизованный пользователь, обладающий правами:

– раздел «Лекции» – просмотр;
– раздел «Предметы» – просмотр;
– раздел «Курсы» – просмотр;
– раздел «О проекте» – просмотр;
– Видео на сайте – просмотр;
– Видео на сайте – поделиться в соц. сетях;
– Обратная связь – заполнение соответствующей формы на сайте.

На рисунке 10 представлена диаграмма прецедентов, отображающая возможности гостя на сайте.

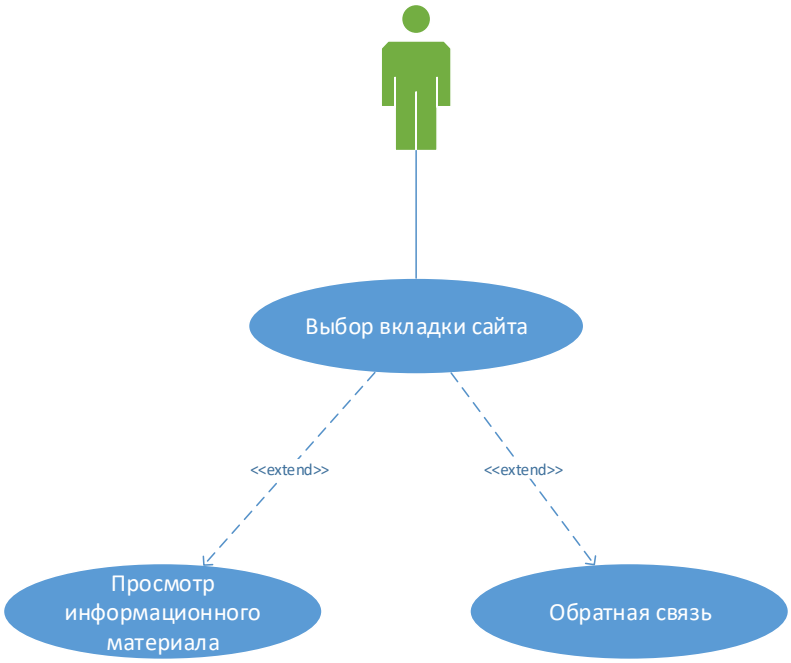

Рисунок 10 – Диаграмма прецедентов, отображающая действия гостя на сайте

Администратор сайта обладает всеми правами обычного пользователя, а также имеет доступ к панели управления сайта:

– раздел «Лекции» – добавление, редактирование, удаление, просмотр;

– раздел «Предметы» – добавление, редактирование, удаление, просмотр;

– раздел «Курсы» – добавление, редактирование, удаление, просмотр;

– Видео на сайте – добавление, редактирование, удаление, просмотр;

– Обратная связь – просмотр обращений, ответ на обращение.

На рисунке 11 представлена соответствующая диаграмма прецедентов, отображающая возможности администратора сайта.

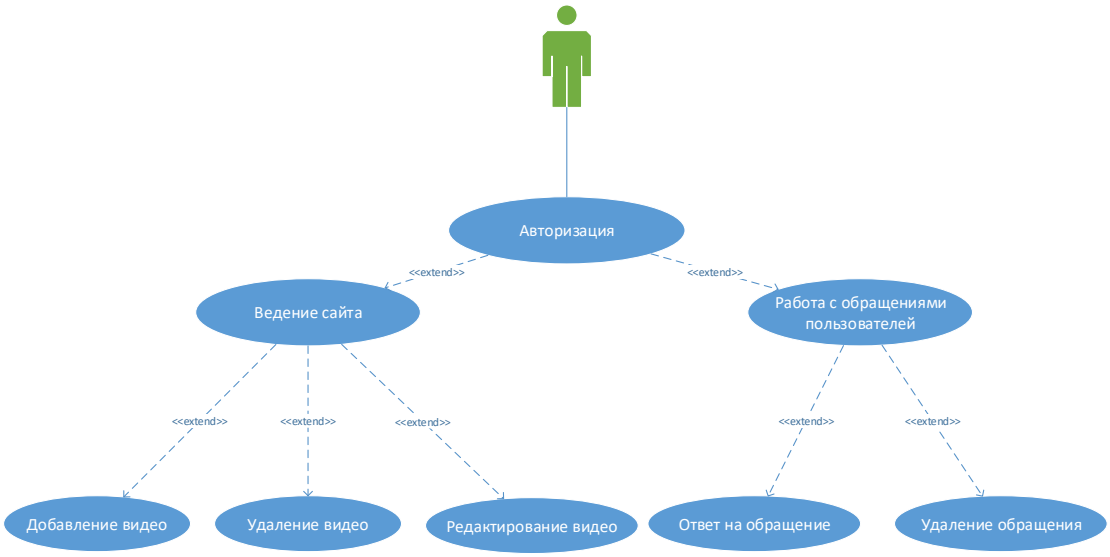

Рисунок 11 – Диаграмма прецедентов, отражающая действия администратора сайта

Для удобного пользования сайтом, ресурс должен иметь функцию фильтрации и поиска видеозаписей. Фильтрация должна включать в себя фильтрацию по дате размещения видеозаписи, фильтрацию по преподавателю и фильтрацию по дисциплине.

Также информационный ресурс должен иметь функцию личного кабинета и обеспечивать возможность выполнения функций авторизации администратора для загрузки видео на сайт, просмотра и редактирования содержимого сайта, считывания информации из базы данных (БД), а также отображения названия сайта, копирайта и комментариев разработчика.

Выходные данные Интернет-ресурса должны быть организованы в виде отдельных таблиц подключенной базы данных. Видеофайлы лекций, размещенных на сайте, должны храниться на видео-хостинге YouTube (Web-сайт – http://youtube.com).

Ниже представлен прототип структуры сайта, общая структура и состав раздоев сайта в дальнейшем могут быть изменены и дополнены (Рисунок 13).

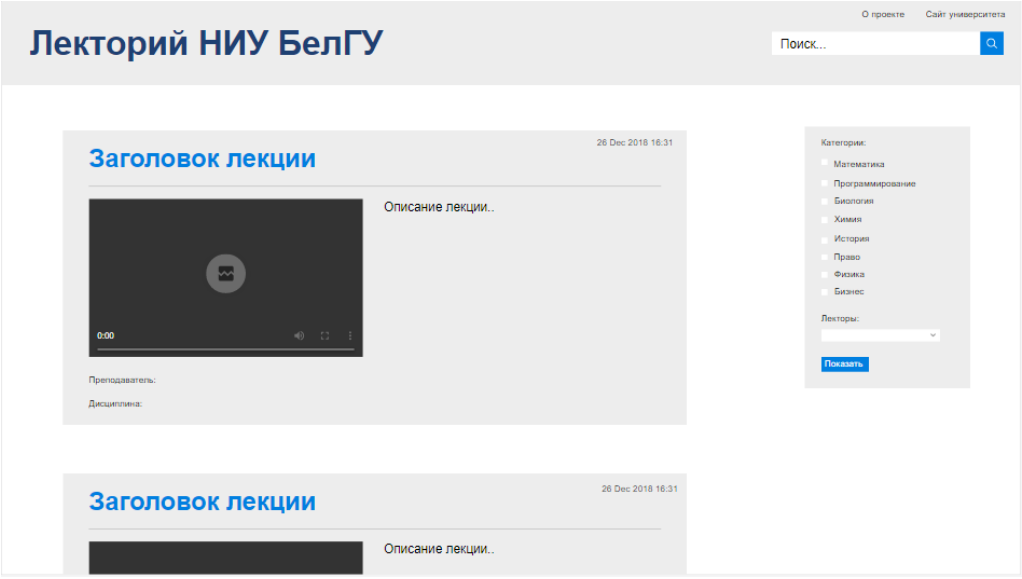

Рисунок 13 – Предполагаемый внешний вид главной страницы сайта

Как видно на рисунке 13, сверху будет расположена шапка сайта со строкой поиска и ссылками. В правой части сайта находится панель навигации для удобного пользования ресурсом. Область контента выполнена в форме блога с заголовками записи, описанием, датой, именем преподавателя и названием сайта. При нажатии на заголовок лекции, пользователь попадает на страницу с видеолекцией, прототип которой представлен на рисунке 14.

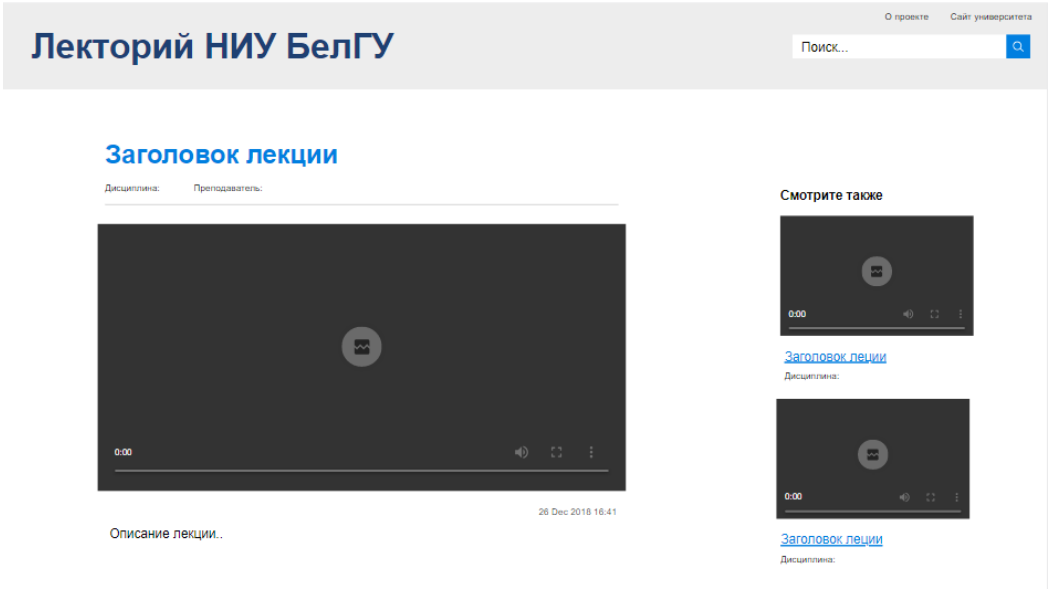

Рисунок 14 – Прототип страницы с видео-лекцией

На рисунке 14 отображена компоновка контента на странице. Видно, что при переходе на страницу с лекцией, шапка сайта остается статичной, а панель навигации пропадает. Вместо нее система будет предлагать пользователю лекции, которые могут его заинтересовать.

При нажатии на ссылку «О проекте» в шапке сайта, пользователь должен попадать на страницу с описанием проекта (прототип страницы представлен на рисунке 15), а при нажатии на ссылку «Сайт университета», пользователь попадет на внешний сайт Белгородского Государственного национального исследовательского университета (web-адрес – http://bsu.edu.ru).

Рисунок 15 – Прототип страницы «О проекте»

Требования к программному обеспечению со стороны серверной части:

– операционная система семейства Unix (Linux, FreeBSD и пр.);
– web-сервер Apache 1.3.18 и выше;
– Nginx, модуль mod_accel для Apache;
– набор библиотек и утилит ffmpeg;
– PHP 7.1.3 и выше (должен быть собран как модуль Apache);

– СУБД MySQL 5.7 и выше (предпочтительно: поддержка формата InnoDB);

– модули PHP: Mcrypt, FTP, ffmpeg-php;

– библиотеки PHP: Composer, GeoIP;

– возможность доступа к localhost по FTP протоколу;

– 2 пользователя БД.

Клиентская часть:

– любой из перечисленный браузеров (указана минимальная версия) с включенным интерпретатором JavaScript: Internet Explorer 6, Mozilla 1.6 (Firefox 1.0), Opera 9, YandexBrowser, GoogleChrome, Safari;

– Adobe Flash Player версии 9 и выше.

Требования к техническому обеспечению серверной части:

– компьютер с процессором уровня Intel Core i3 с частотой 3 ГГц;

– оперативная память 2 Гб (рекомендуется от 4 Гб);

– SSD-накопитель со свободным объемом от 1 Гб;

– точные технически характеристики сервера могут быть уточнены после завершения системы и обширного тестирования всех модулей сайта.

Клиентская часть:

– компьютер с процессором Pentium IV 1.5ГГц (рекомендуется от 2 ГГц);

– оперативная память 512 Мб (рекомендуется от 1 Гб).

План-график реализации данного проекта поэтапно представлен в таблице 1.

Таблица 1 – План-график реализации web-ресурса НИУ «БелГУ»

Этап	**Дата начала**	**Дата окончания**
1	2	3
Сбор необходимой информации	01.05.19	02.05.19

Продолжение таблицы 1

1	2	3
Составление списка требований и ограничений	02.05.19	04.05.19
Разработка макета сайта (внешний вид)	04.05.10	10.05.19
Создание прототипа базы данных	10.05.19	10.05.19
Создание представление сайта (верстка)	10.05.19	19.05.19
Создание базы данных (миграций)	19.05.19	20.05.19
Создание моделей, контроллеров и маршрутов	20.05.19	25.05.19
Тестирование	25.05.19	31.05.19
Размещение на хостинге	01.06.19	01.06.19
Наполнение контентом	01.06.19	15.06.19

3.2 Разработка сайта «Лекторий НИУ БелГУ»

Как было указано ранее, для разработки сайта «Лекторий НИУ БелГУ» был выбран язык программирования PHP с использованием фреймворка Laravel. Программный код страниц сайта представлен в приложении А. Рассмотрим подробнее возможности и достоинства Laravel.

Laravel – это фреймворк для web-приложений с выразительным и элегантным синтаксисом. Он позволит упростить решение основных наболевших задач, таких как аутентификация, маршрутизация, сессии и кэширование.

Laravel построен на концепции MVC (Model-View-Controller). MVC — это паттерн проектирования web-приложений, который включает в себя

несколько более мелких шаблонов. При использовании MVC модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента, благодаря чему модификация одного из них оказывает минимальное воздействие на остальные или не оказывает его вовсе [10, с. 28].

Возможности данного фреймворка:

– пакеты – 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;

– 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 – блоки кода, которые выполняются при генерации представления (шаблона);

– инверсия управления – позволяет получать экземпляры объектов по принципу обратного управления. Также может использоваться для создания и получения объектов-одиночек;

– 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 ошибок;

– страничный вывод – упрощает генерацию страниц, заменяя различные способы решения этой задачи единым механизмом, встроенным в Laravel.

Основные преимущества Laravel:

– широкий функционал: разработка сайтов любого уровня возможна благодаря огромной функциональности. Используя данный фреймворк, можно создавать проекты, предоставляющие возможность интеграции необходимого функционала в соответствии с индивидуальными требованиями и особенностями конкретного бизнеса

– возможность создать гибкую панель администратора: можно реализовать наиболее удобный вариант администрирования ресурса, создавая индивидуальную панель управления под задачи конкретного приложения;

– высокий уровень безопасности: получить несанкционированный доступ к базе данных, созданной при помощи Laravel, крайне сложно. Высокий уровень безопасности гарантирует надежную защиту от SQL-инъекций, атак типа XSS, CSRF;

– регулярные обновления: исходный код изменяется с учетом нововведений в PHP и потребностей программистов. Свежие релизы помогают устранять ранее существовавшие проблемы и уязвимости и делать фреймворк еще более удобным;

– популярность и активное сообщество: наличие большого коммьюнити открывает простор для динамичной коммуникации, обмена личным опытом и мнениями, решения всевозможных вопросов, связанных с проектированием и поддержкой Интернет-ресурсов [41].

Разработка сайта начиналась с создания представлений (View). Представление (View) отвечает за отображение информации (визуализацию), одни и те же данные могут представляться различными способами, например, коллекцию объектов можно представить как в табличном виде, так и списком.

Итак, главная страница сайта состоит из 5 блоков: шапка, строка меню, новые курсы, последние добавленные лекции и футер. Окончательный внешний вид главной страницы сайта представлен на рисунке 17.

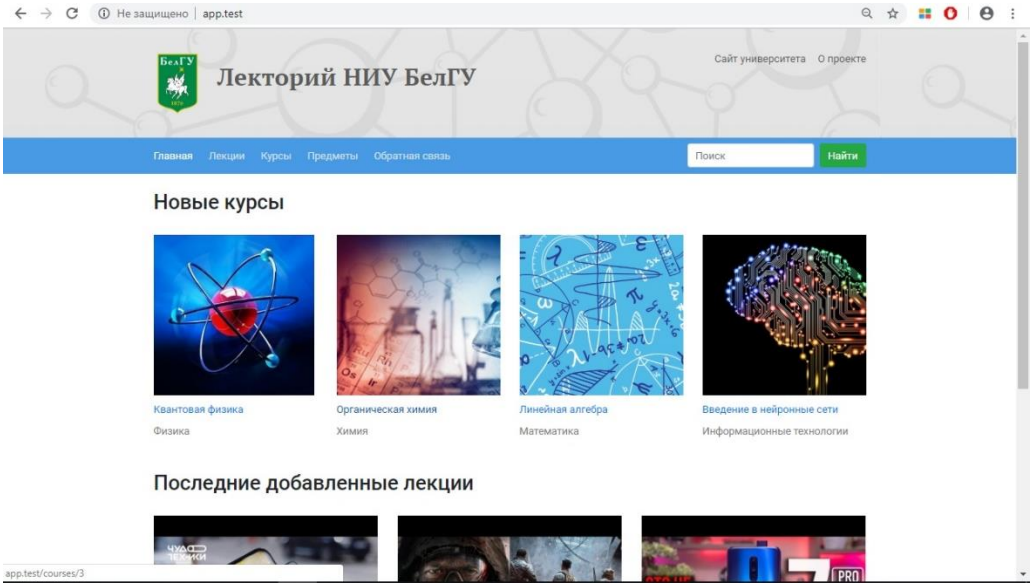

Рисунок 17 – Внешний вид главной страницы сайта

Раздел «Лекции» содержит в себе список всех добавленных лекций. Внешний вид раздела представлен на рисунке 18.

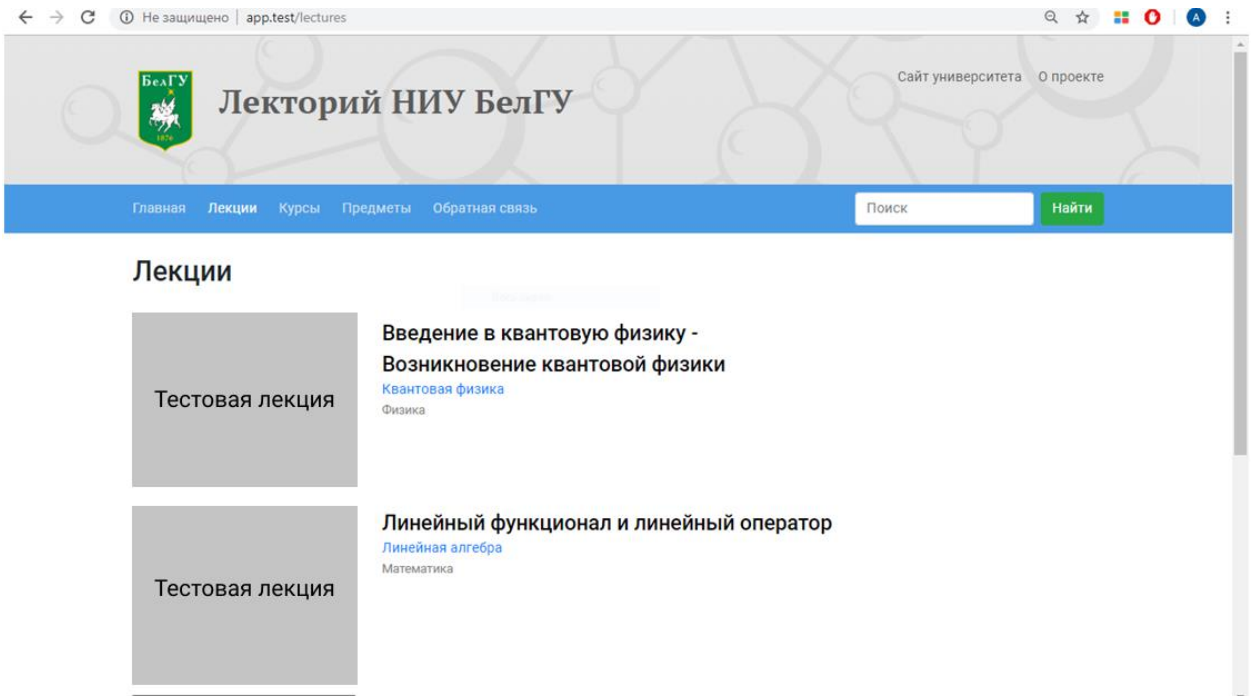

Рисунок 18 – Внешний вид раздела «Лекции»

Страница конкретной видео-лекции содержит само видео, название курса, предмет, имя преподавателя, виджеты социальных сетей, а также блоки «Следующая лекция» и «Предыдущая лекция» для удобной навигации по сайту. Внешний вид страницы представлен на рисунке 19.

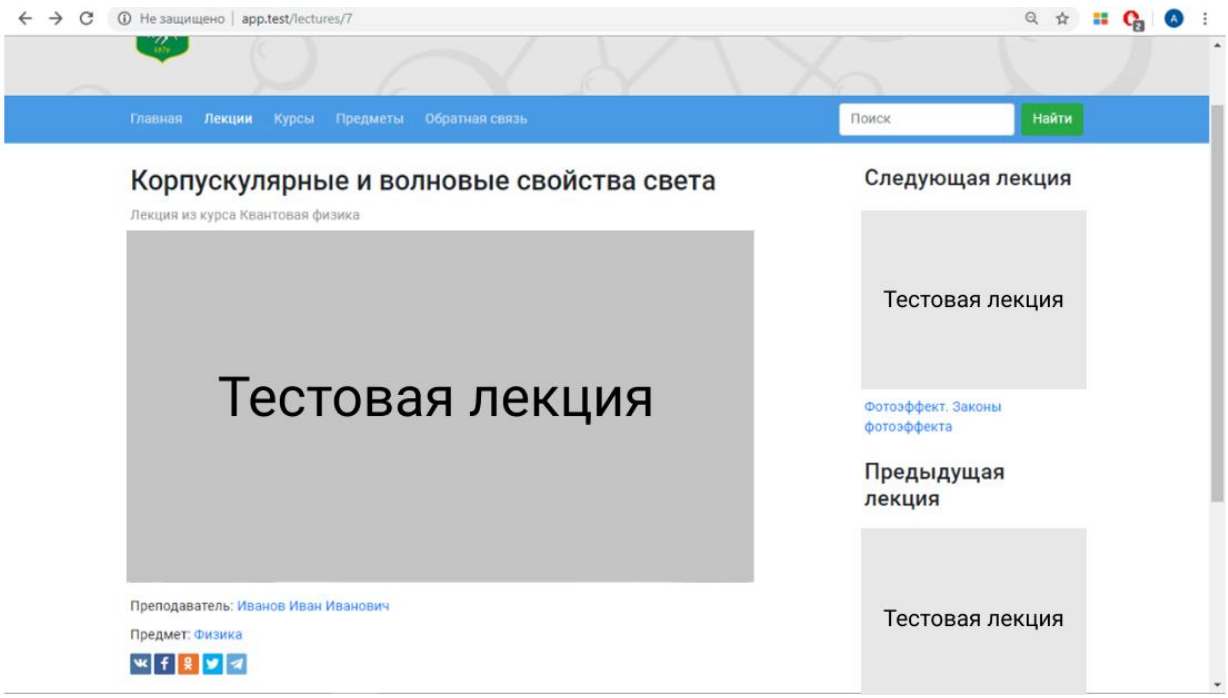

Рисунок 19 – Внешний вид страницы с видео-лекцией

Раздел «Курсы» содержит список всех добавленных курсов. Представленный список курсов можно отфильтровать по выбранным предметам при помощи формы, расположенной в правой части сайта. Внешний вид раздела «Курсы» представлен на рисунке 20.

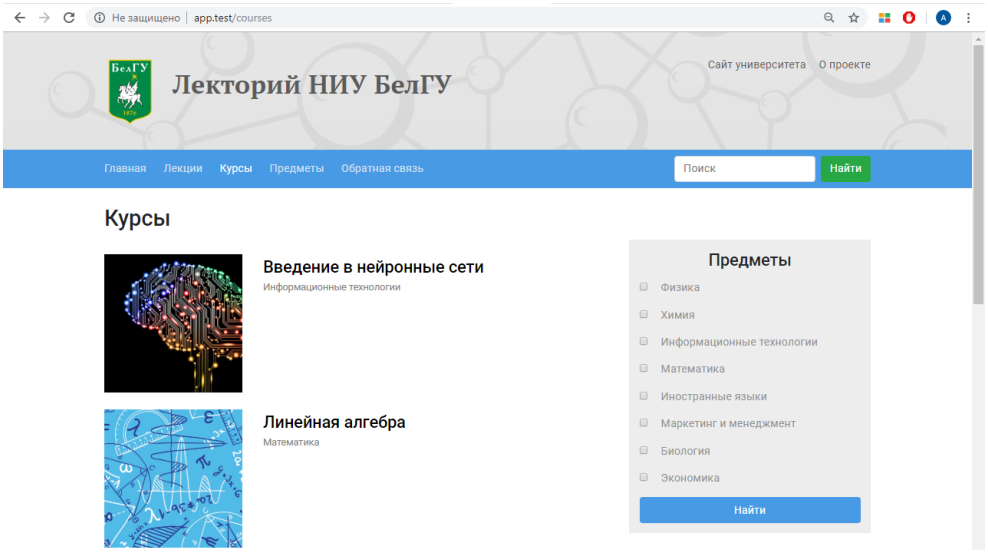

Рисунок 20 – Внешний вид раздела «Курсы»

Раздел предметы содержит список добавленных предметов. В скобках после названия предметов указано количество курсов по данной тематике. Внешний вид раздела «Предметы» представлен на рисунке 21.

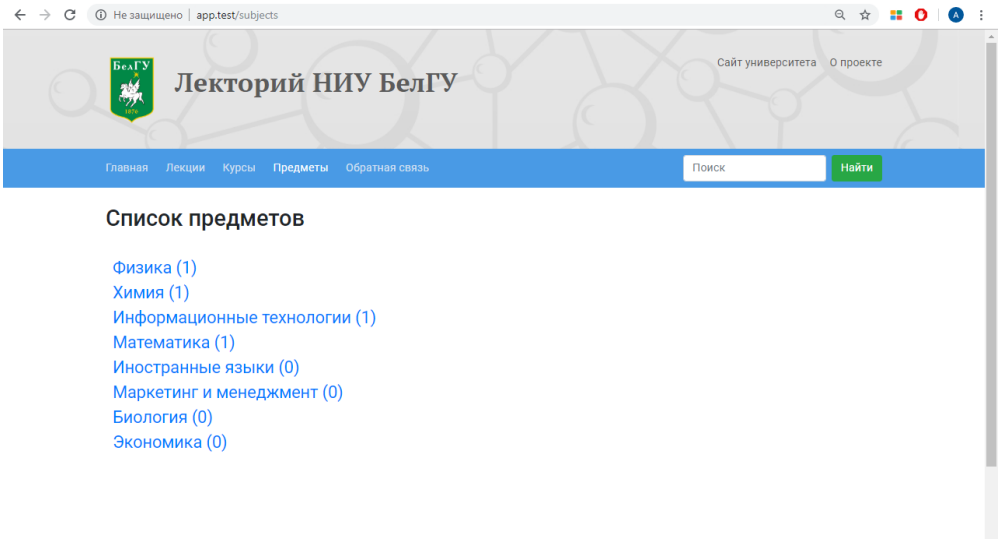

Рисунок 21 – Внешний вид раздела «Предметы»

Раздел «Обратная связь» содержит форму обратной связи, где пользователь может оставить свои пожелания и замечания. Внешний вид раздела «Обратная связь» представлен на рисунке 22.

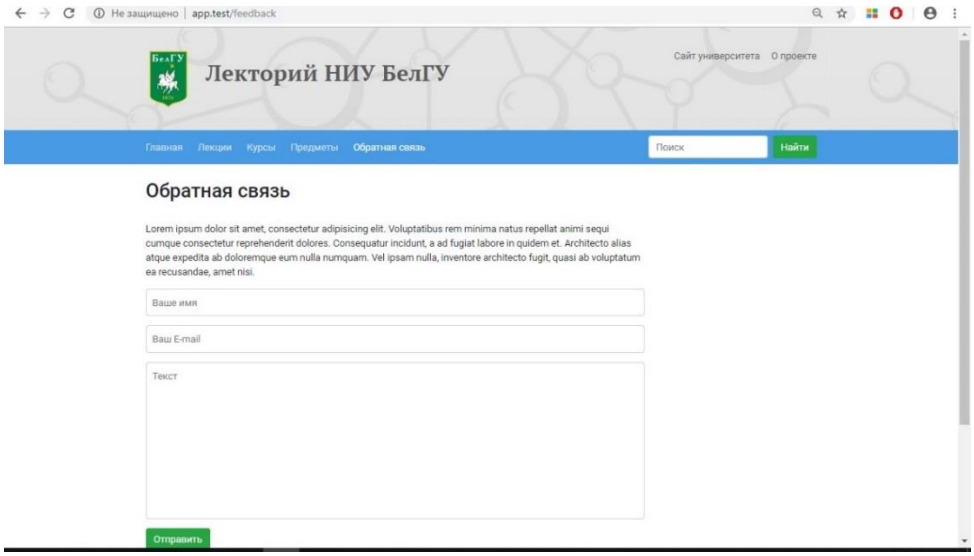

Рисунок 22 – Внешний вид раздела «Обратная связь»

Раздел «О проекте» содержит информацию о проекте для пользователя сайта. Внешний вид раздела представлен на рисунке 23.

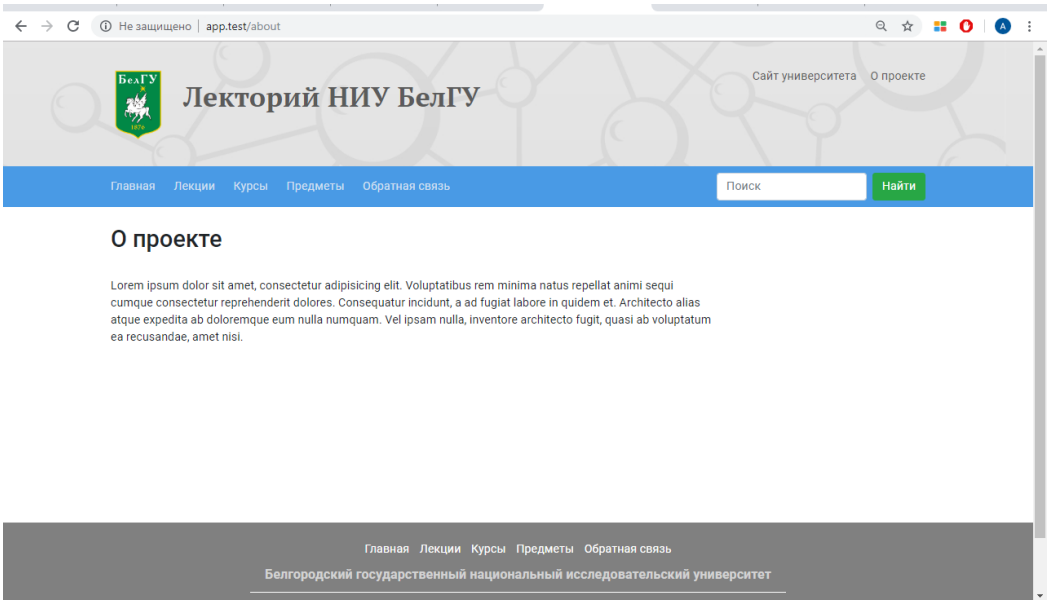

Рисунок 23 – Внешний вид раздела «О проекте»

Рассмотрим внешний вид панели управления сайтом.

По адресу «/login» находится страница входа в панель управления сайтом (рисунок 24). Если же авторизация на сайте уже была произведена, то произойдет перенаправление на главную страницу панели управления, который представлен разделом «Видео» (рисунок 25).

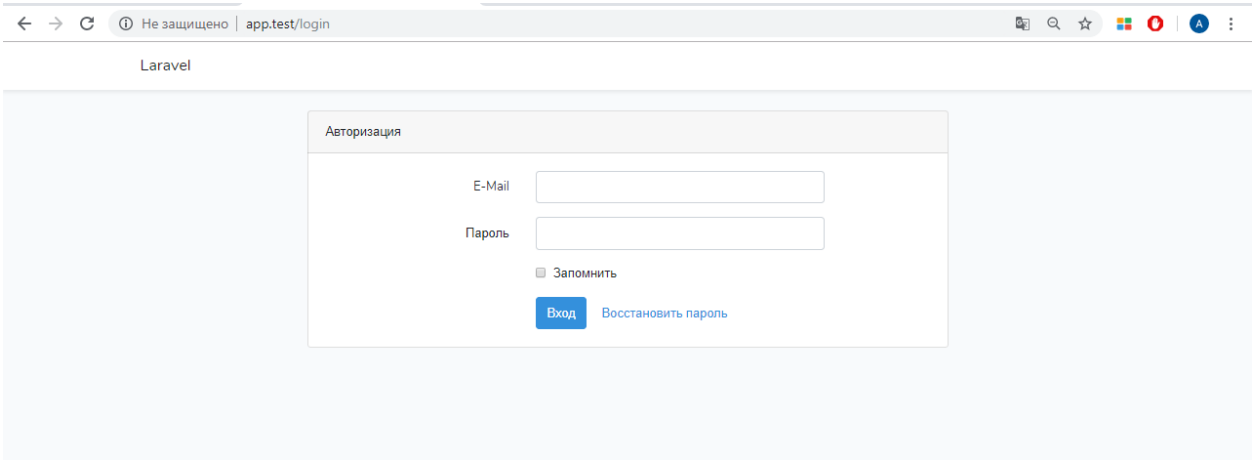

Рисунок 24 – Страница входа в панель управления

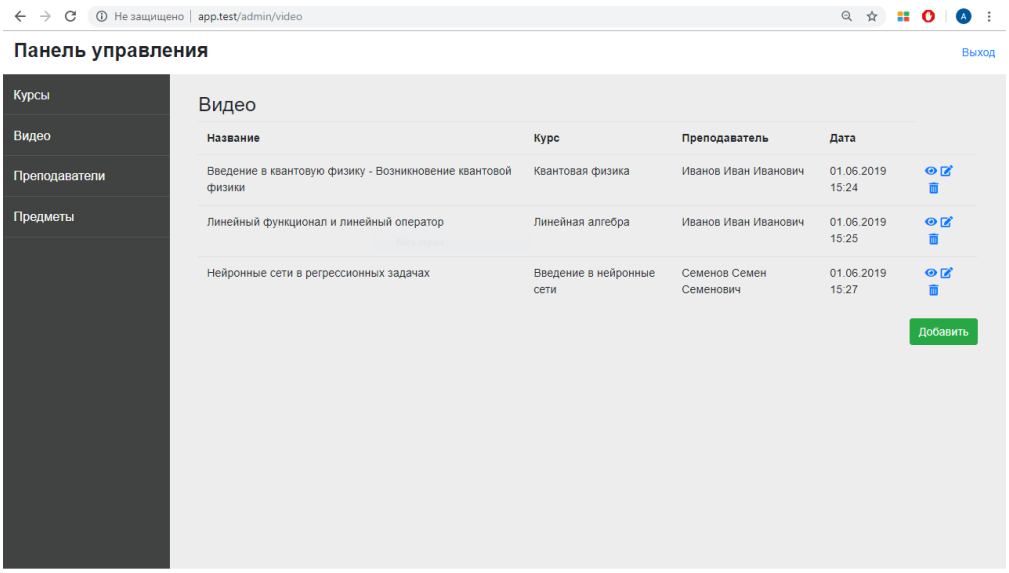

Рисунок 25 – Главная страница панели управления

В представленном разделе выводится список всех лекций, добавленных на сайт, а также присутствуют кнопки добавления, редактирования и удаления видео из базы данных. При нажатии на кнопку «Добавить» на

экране отобразится форма, представленная на рисунке 26. Остальные разделы панели администратора построены по такому же принципу, как и раздел «Видео».

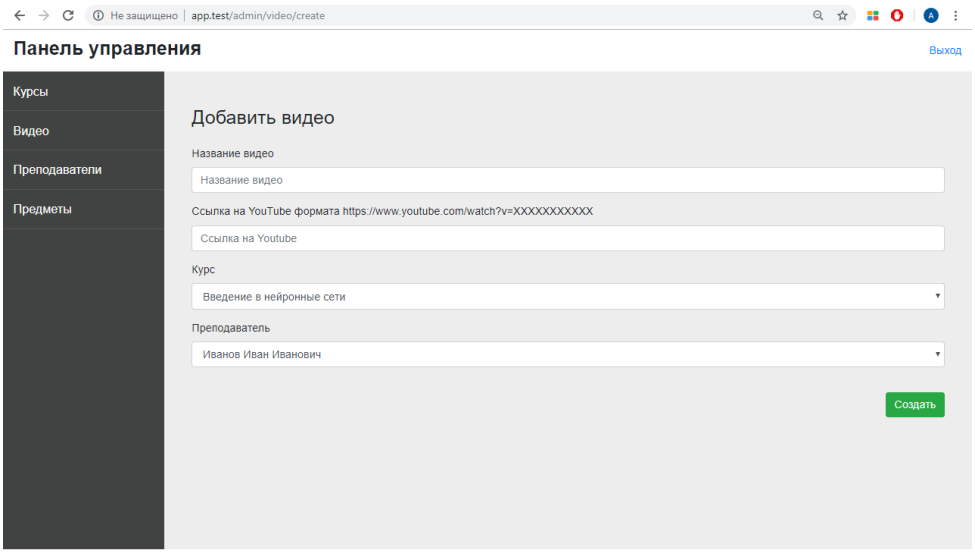

Рисунок 26 – Форма добавления нового видео

Также, стоит отметить, что, благодаря использованию библиотеки Bootstrap 4, сайт корректно отображается на всех устройствах. Пример отображения сайта на смартфоне представлен на рисунке 27.

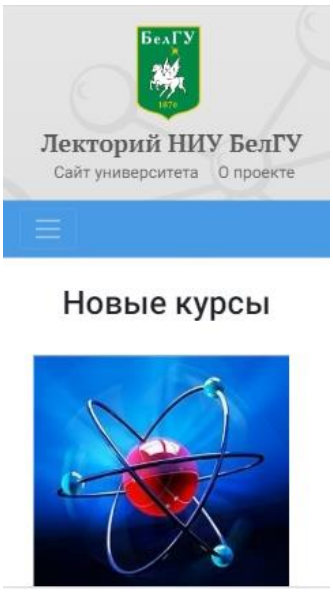

Рисунок 27 – Отображение сайта на мобильном устройстве

Следующим этапом разработки сайта является создания базы данных и ее структуры.

Разработка базы данных в Laravel осуществляется при помощи миграций (Migrations). Миграции – это тип контроля версий для базы данных. Они позволяют команде изменять схему базы данных и оставаться в курсе текущего состояния схемы. Миграции, как правило, сопряжены с Schema Builder, чтобы легко управлять схемой приложения. Миграции позволяют описывать структуру базы данных и вносить изменения прямо в Laravel без помощи средств phpMyAdmin.

Полученная в ходе разработки структура базы данных представлена на рисунке 28.

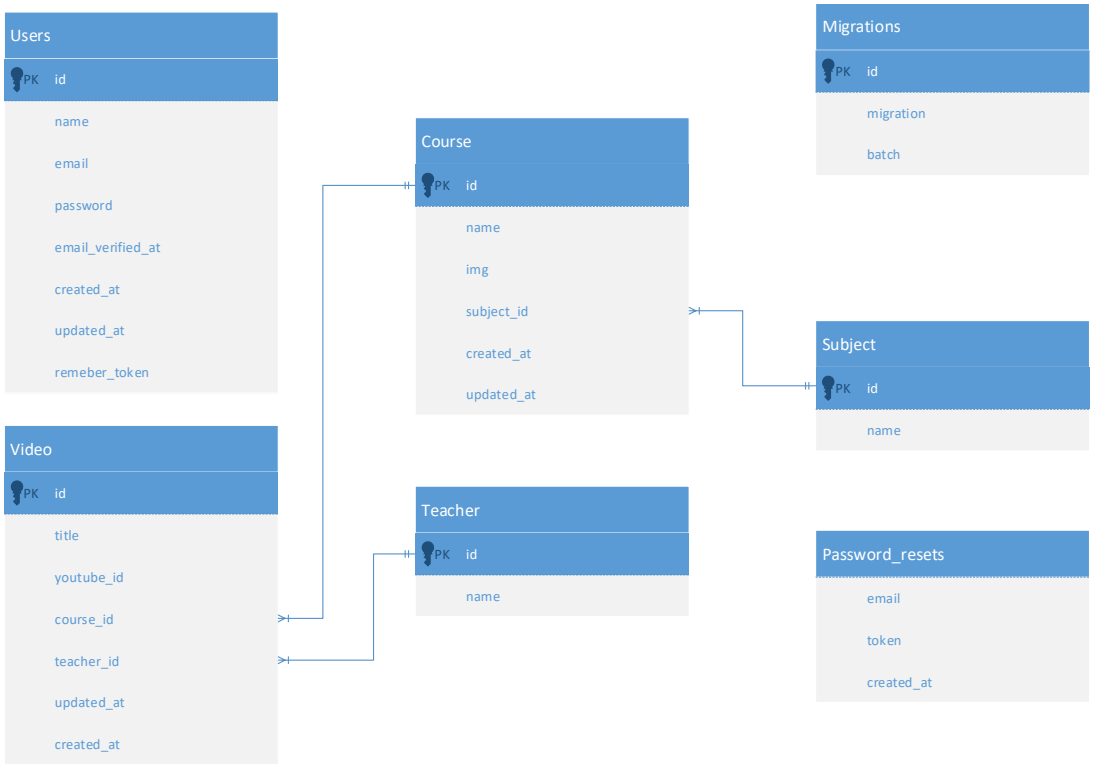

Рисунок 28 – База данных сайта «Лекторий НИУ Белгу»

Представленная база данных состоит из 7 таблиц, а именно:

– таблица «Video» хранит в себе информацию о всех лекциях, представленных на сайте. Таблица состоит из 7 полей: id, title, youtube_id,

course_id, teacher_id, created_at, updated_at. Связана с таблицами «Course» и «Teacher» при помощи связи «Один-ко-многим»;

– таблица «Course» содержит в себе информацию о представленных на сайте курсах. Таблица состоит из 6 полей id, name, img, subject_id, created_at, updated_at. Связана с таблицей «Subjects» при помощи связи «Один-ко-многим»;

– таблица «Teacher» содержит информацию о преподавателях. Таблица состоит из 2-х полей: id и name;

– таблица «Subject» содержит информацию о предметах. Таблица состоит из 2-х полей: id и name;

– таблица «Migrations» содержит информацию о миграциях сайта. Таблица состоит из 3 полей: id, migration, batch;

– таблица «Users» содержит информацию об администраторах сайта. Таблица состоит из 8 полей: id, name, password, email, email_verified_at, remember_token, created_at, updated_at;

– таблица «Password_resets» хранит в себе информацию о сбросах пароля администратора. Состоит из 3 полей: email, token, created_at.

Следующим этапом разработки сайта является создание моделей.

Модель (Model) – предоставляет собой объектную модель некой предметной области, включает в себя данные и методы работы с этими данными, реагирует на запросы из контроллера, возвращая данные и/или изменяя своё состояние, при этом модель не содержит в себе информации, как данные можно визуализировать, а также не «общается» с пользователем напрямую.

В ходе разработки сайта было создано 5 моделей:

– модель «Video». Представляет собой модель видео-лекции;

– модель «Course». Представляет собой модель курса видео-лекций;

– модель «Subject». Представляет собой модель предмета;

– модель «Teacher». Представляет собой модель преподавателя;

– модель «User». Представляет собой модель администратора.

Все модели наследуются от общей модели фреймворка, находящейся в каталоге Illuminate\Database\Eloquent\Model и, соответственно, наследуют все методы данной модели.

Каждая созданная модель привязана к конкретной таблице в БД при помощи переменной public $table. Также в описании моделей проведены связи, аналогичные связям между таблицами в БД. Сделано это при помощи методов belongsTo () и hasMany ().

Следующим шагом является создание контроллеров. Контроллер (Controller) обеспечивает связь между пользователем и системой, использует модель и представление для реализации необходимой реакции на действия пользователя, как правило, на уровне контроллера осуществляется фильтрация полученных данных и авторизация (проверяются права пользователя на выполнение действий или получение информации).

В Laravel все контроллеры находятся в папке App\Http\Controllers и наследуются от одного общего класса Controller.

Таким образом, для представленного сайта было создано 6 контроллеров:

– VideoController. Отвечает за добавление, удаление, редактирование и вывод видеолекций на сайт;

– CourseController. Отвечает за добавление, удаление, редактирование и вывод курсов на сайт;

– SubjectController. Отвечает за добавление, удаление, редактирование и вывод предметов на сайт;

– TeacherController. Отвечает за добавление, удаление, редактирование и вывод преподавателей на сайт;

– FeedbackController. Обрабатывает форму обратной связи на сайте путем отправки обращений на указанную почту;

– SearchController. Отвечает за модуль поиска на сайте.

В заключительном этапе разработки были прописаны маршруты приложения (роуты). в Laravel маршрутизация предназначена для направления запроса к соответствующему контроллеру.

Маршрутизация происходит путем обращения к классу Route и вызова соответствующего методы get() или post(), в аргументах которого указывается сам адрес маршрута и действие, которое будет происходить при переходе по данному маршруту.

После завершения этапа разработки было проведено тестирование сайта, которое позволило выявить и исправить ошибки в работе ресурса, после чего сайт был выгружен на хостинг университета.

3.3 Оценка эффективности внедрения сайта «Лекторий НИУ БелГУ»

Эффективность системы – это свойство системы выполнять поставленную цель в заданных условия использования с определенным качеством.

Показатели эффективности характеризуют степень приспособленности системы к выполнению поставленных задач и являются обобщающими показателями оптимальности функционирования сайта.

В данном случае цель разработки сайта и деятельность организации-заказчика (НИУ «БелГУ») носят некоммерческий характер, поэтому расчет показателей экономической эффективности не представляется возможным.

Ожидаются следующие эффекты от реализации проекта:

– уменьшение аудиторной нагрузки преподавателей;

– повышение популяризации и конкурентоспособности ВУЗа при помощи сайта «Лекторий НИУ БелГУ»;

– повышение качества образования (дистанционного и заочной формы обучения) посредством предоставления круглосуточного всеобщего доступа к обучающим видео-лекциям от преподавателей НИУ «БелГУ»;

– повышение лояльности к ВУЗу будущих абитуриентов за счет возможности ознакомиться с преподавателями и оценить материал курсов, не являясь студентом.

Процесс создания сайта «Лекторий НИУ БелГУ» можно разделить на 3 этапа:

– проектирование сайта
– разработка программной части сайта
– ввод в эксплуатацию

Опишем каждый этап разработки более подробно. Этап проектирования сайта состоит из проектирования базы данных сайта, и прототипирования интерфейса. Разработку программной части сайта можно разделить на создание пользовательского интерфейса (front-end) и логики сайта (back-end). Ввод в эксплуатацию включает в себя размещение сайта на хостинге, наполнение контентом и тестирование.

Исходя из проделанной работы по созданию сайта, была составлена смета с указанием трудоемкости работы в часах и стоимостью часа работы.

Таблица 2 – Смета затрат

Наименование работ	Стоимость часа работы (руб.)	Количество часов	Итого (руб.)
1	2	3	4
Проектирование сайта			
Проектирование базы данных	300	15	4500
Прототипирование интерфейса	250	20	5000

Продолжение таблицы 2

1	2	3	4
Разработка программной части сайта			
Создание логики сайта	800	40	32000
Ввод в эксплуатацию			
Размещение сайта на хостинге	200	2	400
Наполнение контентом	250	8	2000
Тестирование сайта	150	16	2400
ИТОГО			62300

Итоговая стоимость проекта составила 62300 (шестьдесят две тысячи триста) рублей.

Как уже было отмечено ранее, цель разработки сайта носит некоммерческий характер, поэтому расчет показателей экономической эффективности не представляется возможным.

Таким образом, в данной главе мы разобрали цель и назначение web-ресурса «Лекторий НИУ БелГУ», составили список требований к нему и рассмотрели ограничения, создали макет внешнего вида сайта и прототип базы данных, обосновали выбор языка программирования, после чего, опираясь на все выше перечисленное, разработали непосредственно сайт, наполнили его контентом и разместили его в сети Интернет, предварительно обозначив ожидаемый эффект от его функционирования.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была достигнута поставленная цель – повышена доступность дистанционных образовательных услуг при помощи сайта «Лекторий НИУ БелГУ», для этого были решены следующие задачи:

– изучено понятия и классификация web-сайтов. Мы узнали, что web-сайтом называется совокупность документов, а также других различных материалов организации или частных лиц, хранящаяся на web-сервере по определенному адресу, которым может являться какое-либо доменное имя или IP-адрес, а также то, что web-ресурсы делятся по доступности сервисов, по природе содержимого, по физическому расположению, по схеме представления информации. По доступности сервисов web-ресурсы делятся на открытые, полуоткрытые и закрытые. По природе содержимого web-ресурсы бывают статические и динамические. По физическому расположению: внешние сайты сети Интернет и локальные. По схеме представления информации web-ресурсы подразделяют на корпоративные сайты, Интернет-магазины, информационные сайты, сайты-форумы, web-сервисы. Рассмотрели примеры конкретных web-ресурсов;

– проведен анализ технологий, используемых при разработке web-сайтов. При решении этой задачи были рассмотрены особенности следующих языков программирования: HTML, CSS, JavaScript, XML, PHP. Так как для реализации сайта «Лекторий НИУ БелГУ» был выбран язык PHP, его возможности, достоинства и недостатки были разобраны подробнее остальных. Также был исследован рейтинг наиболее популярных языков программировании в среде разработчиков на начало 2019 года по данным StackOverflow;

– проведен анализ образовательной деятельности НИУ «БелГУ». Была кратко рассмотрена история и структура ВУЗа, изучен перечень использующихся в университете автоматизированных систем и web-

ресурсов, после чего разобраны и смоделированы посредством диаграмм IDEF0 и DFD основные бизнес-процессы, протекающие в организации – процессы предоставления образовательных услуг и планирования учебного процесса;

– основываясь на результатах вышеописанных задач, был сформирован список требований и ограничений к разрабатываемому сайту. Были определены назначение сайта и его целевая аудитория, требования к базе данных, функционалу и внешнему виду web-ресурса, составлен план-график реализации проекта. «Лекторий НИУ БелГУ» – Интернет ресурс, разрабатываемый для Белгородского государственного национального исследовательского университета, главным назначением которого является предоставление пользователям информации в рамках учебной программы в видео формате;

– web-ресурс «Лекторий НИУ БелГУ» разработан и выгружен в Интернет. Сайт написан на языке PHP с использованием фреймворка Laravel. Доступ к сайту имеет любой пользователь, проживающий на территории России. Web-ресурс выполнен в цветах официального сайта ВУЗа, состоит из 6 страниц, не считая страниц отдельно взятых лекций. Сайт выгружен на хостинг НИУ «БелГУ»;

– оценена эффективность реализации проекта. Так как сайт разработан не в коммерческих целях, экономическую эффективность оценить не представляется возможным, поэтому был рассмотрен ожидаемый социальный и образовательный эффекты.

Разработанный web-ресурс является законченным программным продуктом и готов к использованию.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Бабич, А. В. Введение в UML [Текст] / А. В. Бабич – М.: НОУ ИНТУИТ, 2016 г. – 209 с.

2 Белоногов, Г. Г. Автоматизация процессов накопления, поиска и обобщения информации [Текст] / Г. Г. Белоногов, А. П. Новоселов – М.: Наука, 2017 г. – 256 с.

3 Веллинг, Л. Разработка WEB-приложений с помощью PHP и MySQL, 2-е издание: Пер. с англ. [Текст] / Л. Веллинг – М.: Вильямс, 2009 г. – 800 с.

4 Вендров, А. М. Современные технологии создания программного обеспечения [Текст] / А. М. Вендров – М.: Красный квадрат, 2016 г. – 304 с.

5 Глушаков, С. В. Программирование Web-страниц [Текст]: учебное пособие / С. В. Глушаков – М.: Фолио, 2005 г. – 390 с.

6 Головчинер, М. Н. «Базы данных. Основные понятия, модели данных, процесс проектирования» [Текст] / М. Н. Головчинер – М.: Альфа, 2009 г. – 318 с.

7 Гончаров, А. В. Самоучитель HTML [Текст] / А. В. Гончаров – СПб: Питер, 2014 г. – 240 с.

8 Гультяев, А. К. Уроки Web-мастера. Технология. Дизайн. Инструменты [Текст] / А. К. Гультяев, В. А. Машин – М.: Корона-Принт, 2016 г. – 448 с.

9 Дейт, К. Дж. Введение в системы баз данных, 6-е издание [Текст] / К. Дж. Дейт – СПб: Вильямс, 2008 г. – 848 с.

10 Дронов, В. А. Laravel. Быстрая разработка современных динамических Web-сайтов на PHP, MySQL, HTML и CSS [Текст] / В. А. Дронов – М.: БХВ-Петербург, 2018 г. – 228 с.

11 Зандстра, М. PHP. Объекты, шаблоны и методики программирования [Текст] / М. Зандстра – М.: Вильямс, 2017 г. – 480 с.

12 Ипатова, Э. Р. Методологии и технологии системного проектирования информационных систем [Текст]: учебник / Э. Р. Ипатова – М.: Флинта, 2016 г. – 318 с.

13 Калянов, Г. Н. CASE-технологии: Консалтинг в автоматизации бизнес-процессов, 3-е издание [Текст] / Г. Н. Калянов – М.: Горячая линия-Телеком, 2008 г. – 320 с.

14 Кожемякин, А. А. HTML и CSS в примерах. Создание Web-страниц [Текст] / А. А. Кожемякин – М.: Альтекс-А, 2014 г. – 416 с.

15 Костенко, К. К. PHP. Web-профессионалам [Текст] / К. К. Костенко Красноярск: BHV, 2011 г. – 208 с.

16 Кутикова, К. В. Методика проектирования информационных систем для сферы государственных и муниципальных услуг [Текст] / К. В. Кутикова. – М.: Синергия, 2014 г. – 92 с.

17 Кушнарев, Л. И. Методические рекомендации по дипломному проектированию [Текст] / Л. И. Кушнарев – М.: ФГОУ ВПО МГАУ, 2015 г. – 114 с.

18 Леонтьев, Б. В. Web-Дизайн: Тонкости, хитрости и секреты [Текст]/ Леонтьев Б. В. – М.: Майор, 2011 г. – 170 с.

19 Лешек, А. М. Анализ и проектирование информационных систем с помощью UML 2.0 [Текст] / Лешек А. М. – М.: Вильямс, 2016 г. – 816 с.

20 Мальцева, С. В. Применение онтологических моделей для решения задач идентификации и мониторинга предметных областей [Текст] / С. В. Мальцева – М.: Дельта-М, 2017 г. – 411 с.

21 Моисеев, О. С. Использование портальных технологий в построении виртуальной медиатеки ВУЗа [Текст] / О. С. Моисеев – М.: ИТО, 2005 г. – 206 с.

22 Нидерст, Дж. Web-мастеринг для профессионалов. Настольный справочник [Текст] / Дж. Нидерст – СПб: Питер, 2011 г. – 240 с.

23 Никсон, Р. PHP. Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript [Текст] / Р. Никсон – СПб: Питер, 2011 г. – 496 с.

24 Олифер, В. Г. Компьютерные сети. Принципы, технологии, протоколы [Текст] / В. Г. Олифер – СПб: Питер, 2013 г. – 458 с.

25 Прохоренок, Н. М. HTML, JavaScript, PHP и MySQL. Джентельменский набор Web-мастера [Текст] / Н. М. Прохоренок – М.: Алфавит, 2013 г. – 912 с.

26 Раскин, Д. Интерфейс: новые направления в проектировании компьютерных систем / Раскин Д. – М.: Символ-плюс, 2017 г. – 272 с.

27 Селиванов, В. Г. Бизнес-процессы: Регламентация и управление [Текст] / В. Г. Селиванов – М.: ИНФРА-М, 2009 г. – 237 с.

28 Сост, П. В. Проектирование информационных систем [Текст]: учебно–методическое пособие для студентов очной и заочной форм обучения специальности прикладная информатика (в экономике) / П. В. Сост – Красноярск: КГТУ, 2018 г. – 68 с.

29 Ташков, П. А. Веб-мастеринг. HTML, CSS, JavaScript, PHP, CMS, AJAX, раскрутка [Текст] / П. А. Ташков – СПб: Питер, 2013 г. – 512 с.

30 Федорчук, А. С. Как создаются Web-сайты [Текст] / А. С. Федорчук –СПб.: Питер, – 2014 г. – 224 с.

31 Чипига, А. Ф. Информационная безопасность автоматизированных систем [Текст] / А. Ф. Чипига – М.: Гелиос АРВ, 2010 г. – 336 с.

32 Web-порталы: классификация и назначение [Электронный ресурс] – URL: https://compress.ru/article.aspx?id=10932 (дата обращения: 15.05.2019)

33 Белгородский государственный национальный исследовательский университет [Электронный ресурс] – URL: http://bsu.edu.ru/ (дата обращения: 21.05.2019)

34 Университет. Мое образование [Электронный ресурс] – Режим доступа: https://moeobrazovanie.ru/universitet.html

35 Основные характеристики Microsoft Access [Электронный ресурс] – URL: https://products.office.com/ru-ru/access (дата обращения: 28.05.2019)

36 20 инструментов для прототипирования [Электронный ресурс] – URL: https://medium.com/@denysergushkin/20-инструментов-для-прототипирования-от-быстрого-и-грязного-wireframe-к-функциональному-прототипу-210f223323fe (дата обращения: 22.05.2019)

37 Прототипирование сайтов [Электронный ресурс] – URL: http://www.castcom.ru/services/web-development/prototipirovanie-sajtov/ (дата обращения: 01.06.2019)

38 Прототипирование сайта. Методы прототипирования [Электронный ресурс] – URL: http://fotodizart.ru/prototipirovanie-sajtov.html (дата обращения: 03.06.2019)

39 Проектирование базы данных. Информационные технологии в менеджменте [Электронный ресурс] – URL: https://studme.org/62415/menedzhment/proektirovanie_bazy_dannyh (дата обращения: 28.05.2019)

40 Nginx: документация [Электронный ресурс] – URL: https://nginx.org/ru/ (дата обращения: 28.05.2019)

41 Laravel. Русскоязычное комьюнити [Электронный ресурс] – URL: http://laravel.su (дата обращения: 28.05.2019)

ПРИЛОЖЕНИЕ А

Листинг программного кода сайта «Лекторий НИУ БелГУ»

Содержимое файла main.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
<title>Лекторий</title>
<!-- Bootstrap -->
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootst
rap.min.css" integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm
" crossorigin="anonymous">
<!-- Fonts -->
<link
href="https://fonts.googleapis.com/css?family=PT+Serif:400,400i,
700,700i&amp;subset=cyrillic,cyrillic-ext" rel="stylesheet">
<link
href="https://fonts.googleapis.com/css?family=Cormorant+Unicase:
400,500,600,700&amp;subset=cyrillic" rel="stylesheet">
<link
href="https://fonts.googleapis.com/css?family=Roboto:400,500,700
&amp;subset=cyrillic" rel="stylesheet">
<!-- CSS -->
<link rel="stylesheet" type="text/css" href="{{
asset('css/style.css') }}" />
</head>
<body>
<div class="container-fluid no-margin">
<header>
<div class="container">
<div class="row header-top">
<div class="col-md-6 logo-box">
<a href="{{route('main.index')}}">
<img src="{{ asset('images/logo.png') }}" alt="">
<span>Лекторий НИУ БелГУ</span>
</a>
</div>
<div class="col-md-6 text-right link-box">
<a href="{{url('https://bsu.edu.ru')}}" target="_blank">Сайт
университета</a>
<a href="{{route('main.about')}}">О проекте</a>
</div>
</div>
```

```
</div>
<nav class="navbar navbar-dark navbar-expand-lg ">
<div class="container">
<button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse"
id="navbarSupportedContent">
<ul class="navbar-nav mr-auto">
<li class="nav-item active">
<a href="{{route('main.index')}}" class="nav-link {{
Request::is('/') ? 'active' : '' }}">Главная<span class="sr-
only">(current)</span></a>
</li>
<li class="nav-item">
<a  href="{{route('main.lectures')}}" class="nav-link {{
Request::is('lectures*') ? 'active' : '' }} ">Лекции</a>
</li>
<li class="nav-item">
<a href="{{route('main.courses')}}" class="nav-link {{
Request::is('courses*') ? 'active' : '' }} ">Курсы</a>
</li>
<li class="nav-item">
<a  href="{{route('main.subjects')}}" class="nav-link {{
Request::is('subjects*') ? 'active' : '' }} ">Предметы</a>
</li>
<li class="nav-item">
<a  href="{{route('main.feedback')}}" class="nav-link {{
Request::is('feedback*') ? 'active' : '' }} ">Обратная связь</a>
</li>
</ul>
<form class="form-inline my-2 my-lg-0"
action="{{route('main.search')}}" method="post">
{{ csrf_field() }}
<input class="form-control mr-sm-2" type="search"
placeholder="Поиск" aria-label="Search" name='search'>
<button class="btn my-2 my-sm-0" type="submit">Найти</button>
</form>
</div>
</div>
</nav>
</header>
<div class="row main no-margin">
@yield('content')
</div>
<footer>
<div class="container">
<div class="row justify-content-center">
<div class="col-md-8 text-center">
<div class="footer-links">
```

```
<a href="" class="{{ Request::is('lectures*') ? 'active' : '' }}
">Главная</a>
<a href="" class="{{ Request::is('lectures*') ? 'active' : '' }}
">Лекции</a>
<a href="" class="{{ Request::is('lectures*') ? 'active' : '' }}
">Курсы</a>
<a href="" class="{{ Request::is('lectures*') ? 'active' : '' }}
">Предметы</a>
<a href="" class="{{ Request::is('lectures*') ? 'active' : '' }}
">Обратная связь</a>
</div>
<p><b>Белгородский государственный национальный
исследовательский университет </b></p>
<hr>
<p>Россия, 308015, г. Белгород, ул. Победы, 85, </p>
<p>Тел. (4722) 30-12-21, 30-12-18</p>
<p>E-mail: news@bsu.edu.ru </p>
</div>
</div>
</div>
</footer>
</div>
<script src="{{ asset('js/jquery-3.3.1.min.js') }}"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstra
p.min.js" integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl
" crossorigin="anonymous"></script>
</body>
</html>
```

Содержимое файла admin.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Панель управления</title>
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/boo
tstrap.min.css">
<link rel="stylesheet" type="text/css" href="{{
asset('css/style.css') }}" />
<script
src="https://code.jquery.com/jquery-3.4.1.min.js"
integrity="sha256-CSXorXvZcTkaix6Yvo6HppcZGetbYMGWSFlBw8HfCJo="
crossorigin="anonymous"></script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/boots
trap.min.js" integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM
" crossorigin="anonymous"></script>
```

```
<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.8.2/css/all.css"
integrity="sha384-
oS3vJWv+0UjzBfQzYUhtDYW+Pj2yciDJxpsK1OYPAYjqT085Qq/1cq5FLXAZQ7Ay
" crossorigin="anonymous">
</head>
<body>
<div class="container-fluid no-margin">
<div class="row no-margin">
<div class="col-md-12 admin-header">
<div class="row align-items-center no-margin">
<div class="col-md-6">
<h1>Панель управления</h1>
</div>
<div class="col-md-6 text-right">
@if(Auth::check())
<a href="/logout">Выход</a>
@endif
</div>
</div>
</div>
<div class="col-md-2 admin-menu no-margin">
<ul>
<li><a href="{{route('course.index')}}">Курсы</a></li>
<li><a href="{{route('video.index')}}">Видео</a></li>
<li><a href="{{route('teacher.index')}}">Преподаватели</a></li>
<li><a href="{{route('subject.index')}}">Предметы</a></li>
</ul>
</div>
<div class="col-md-10 admin-content">
@yield('content')
</div>
</div>
</div>
</body>
</html>
```

Содержимое файла home.blade.php

```
@extends('layouts.app')
@section('content')
<div class="container">
<div class="row justify-content-center">
<div class="col-md-8">
<div class="card">
<div class="card-header">Dashboard</div>
<div class="card-body">
@if (session('status'))
<div class="alert alert-success" role="alert">
{{ session('status') }}
</div>
@endif
You are logged in!
```

```
</div>
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла video.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9 lession-block">
<h1 class="lession-block-title">{{$video->title}}</h1>
@if (!empty ($video->courses))
<span class="lession-block-course">Лекция из курса {{$video-
>courses->name}}</span>
@endif
<iframe width="720" height="405"
src="https://www.youtube.com/embed/{{$video->youtube_id}}"
frameborder="0" allow="accelerometer; autoplay; encrypted-media;
gyroscope; picture-in-picture" allowfullscreen></iframe>
<div class="row">
<div class="col-md-6">
@if(!empty($video->teachers->name))
<p>
Преподаватель:
<a href="">
{{$video->teachers->name}}
</a>
</p>
@endif
@if(!empty($video->courses->subjects->name))
<p>
Предмет: <a href="">{{$video->courses->subjects->name}}</a>
</p>
@endif
<script src="https://yastatic.net/es5-shims/0.0.2/es5-
shims.min.js"></script>
<script src="https://yastatic.net/share2/share.js"></script>
<div class="ya-share2" data-
services="vkontakte,facebook,odnoklassniki,twitter,telegram"
style="margin-top: 10px;"></div>
</div>
<div class="col-md-6 text-right">
</div>
</div>
</div>
<div class="col-md-3">
<div class="row">
@if(!empty($next))
<div class="col-md-12 next-lession-block">
```

```
<h2>Следующая лекция</h2>
<a href="{{route('main.lecture', $next->id)}}">
<img src="https://img.youtube.com/vi/{{$next-
>youtube_id}}/sddefault.jpg" width="100%">
{{$next->title}}
</a>
</div>
@endif
@if(!empty($previous))
<div class="col-md-12 next-lession-block">
<h2>Предыдущая лекция</h2>
<a href="{{route('main.lecture', $previous->id)}}">
<img src="https://img.youtube.com/vi/{{$previous-
>youtube_id}}/sddefault.jpg" width="100%">
{{$previous->title}}
</a>
</div>
@endif
</div>
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла subjects.blade.php

```
@extends ('main')
@section ('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9">
<h1>Список предметов</h1>
@if(!empty($subjects))
@foreach ($subjects as $subject)
<a href="{{route('main.subject', $subject->id)}}"
class="subject-link">
{{$subject->name}}
({{$subject->courses->count()}})
</a>
@endforeach
@endif
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла lecture.blade.php

```
@extends('main')
```

```
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9">
<h1>Лекции</h1>
@if(!empty($videos))
@foreach($videos as $video)
<div class="row course-preview">
<div class="col-md-4 course-box">
<a href="">
<img src="https://img.youtube.com/vi/{{$video-
>youtube_id}}/sddefault.jpg" width="100%">
</a>
</div>
<div class="col-md-8">
<a href="{{route('main.lecture', $video->id)}}" class="course-
box-name">{{$video->title}}</a>
<br>
<a href="{{route('main.course', $video->courses->id)}}">
@if(!empty($video->courses->name))
{{$video->courses->name}}
@endif
</a>
<br>
<span class="course-subject">
@if(!empty($video->courses->subjects->name))
{{$video->courses->subjects->name}}
@endif
</span>
</div>
</div>
@endforeach
@endif
</div>
<div class="col-md-3">
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла index.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<h1>Новые курсы</h1>
<div class="row justify-content-between">
@if(!empty($courses))
@foreach ($courses as $course)
<div class="col-md-3 course-box">
```

```
<a href="{{route('main.course', $course->id)}}">
<div class="course-img-wrap">
<img src="{{asset('storage/' . $course->img)}}" alt="">
</div>
<span class="course-name">{{$course->name}}</span>
<span class="course-subject">
@if(!empty($course->subjects->name))
{{$course->subjects->name}}
@endif
</span>
</a>
</div>
@endforeach
@endif
</div>
</div>
</div>
<div class="col-md-12 section">
<div class="container">
<h1>Последние добавленные лекции</h1>
<div class="row justify-content-between">
@if(!empty($videos))
@foreach($videos as $video)
<div class="col-md-4 course-box">
<a href="{{route('main.lecture', $video->id)}}">
<div class="lection-img-wrap">
<img src="https://img.youtube.com/vi/{{$video-
>youtube_id}}/sddefault.jpg" width="100%">
</div>
<span class="course-name">{{$video->title}}</span>
</a>
<a href="">
<span class="course-subject">
@if($video->subjects != null)
{{$video->subjects->name}}
@endif
</span>
</a>
</div>
@endforeach
@endif
</div>
</div>
</div>
@endsection
```

Содержимое файла feedback.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
```

```
<div class="col-md-9 feedback-box">
<h1>Обратная связь</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Voluptatibus rem minima natus repellat animi sequi cumque
consectetur reprehenderit dolores. Consequatur incidunt, a ad
fugiat labore in quidem et. Architecto alias atque expedita ab
doloremque eum nulla numquam. Vel ipsam nulla, inventore
architecto fugit, quasi ab voluptatum ea recusandae, amet
nisi.</p>
<form action="{{route('main.feedback.send')}}" method="POST">
{{ csrf_field() }}
<input type="text" name="name" placeholder="Ваше имя"
required="">
<input type="email" name="email" placeholder="Ваш E-mail"
required="">
<textarea name="text" id="" cols="30" rows="10"
placeholder="Текст" required=""></textarea>
<button class="btn btn-success">Отправить</button>
</form>
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла courses.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9">
<h1>Курсы</h1>
@foreach($courses as $course)
<div class="row course-preview">
<div class="col-md-auto course-box">
<a href="">
<div class="course-img-wrap">
<img src="{{asset('storage/' . $course->img)}}" alt="">
</div>
</a>
</div>
<div class="col-md-8">
<a href="{{route('main.course', $course->id)}}" class="course-
box-name">{{$course->name}}</a>
<br>
<span class="course-subject">
@if(!empty($course->subjects->name))
{{$course->subjects->name}}
@endif
</span>
</div>
```

```
</div>
@endforeach
<div class="row justify-content-center">
<div class="col-md-auto paginate-box">{{$courses-
>links()}}</div>
</div>
</div>
<div class="col-md-3">
<div class="filter-box">
<h2>Предметы</h2>
<form action="{{route('main.course.filter')}}" method="POST">
{{ csrf_field() }}
@if (!empty($subjects))
@foreach($subjects as $subject)
<div class="form-group">
<input name='check{{$subject->id}}' type="checkbox"
value="{{$subject->id}}">
<label for="">{{$subject->name}}</label>
</div>
@endforeach
@endif
<button class="btn">Найти</button>
</form>
</div>
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла course.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9">
<h1>{{$course_name->name}}</h1>
@if(!empty($videos))
@foreach($videos as $video)
<div class="row course-preview">
<div class="col-md-4 course-box">
<a href="">
<img src="https://img.youtube.com/vi/{{$video-
>youtube_id}}/sddefault.jpg" width="100%">
</a>
</div>
<div class="col-md-8">
<a href="{{route('main.lecture', $video->id)}}" class="course-
box-name">{{$video->title}}</a>
<br>
<span>
```

```
@if(!empty($video->teachers->name))
{{$video->teachers->name}}
@endif
</span>
<br>
<span class="course-subject">
@if(!empty($video->courses->subjects->name))
{{$video->courses->subjects->name}}
@endif
</span>
</div>
</div>
@endforeach
@endif
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла about.blade.php

```
@extends('main')
@section('content')
<div class="col-md-12 section">
<div class="container">
<div class="row">
<div class="col-md-9 feedback-box">
<h1>О проекте</h1>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Voluptatibus rem minima natus repellat animi sequi cumque
consectetur reprehenderit dolores. Consequatur incidunt, a ad
fugiat labore in quidem et. Architecto alias atque expedita ab
doloremque eum nulla numquam. Vel ipsam nulla, inventore
architecto fugit, quasi ab voluptatum ea recusandae, amet
nisi.</p>
</div>
</div>
</div>
</div>
@endsection
```

Содержимое файла create.blade.php

```
@extends('admin')
@section('content')
<div class="container">
<div class="row">
<div class="col-md-12 col-md-offset-1">
<h3 style="margin: 25px 0;">Добавить видео</h3>
<form action="{{route('video.store')}}" method="POST">
{{ csrf_field() }}
<div class="form-group">
```

```
<div class="form-group">
<label for="title">Название видео</label>
<input type="text" class="form-control" name="title"
placeholder="Название видео" required>
</div>
<div class="form-group">
<label for="youtube_id">Ссылка на YouTube формата
https://www.youtube.com/watch?v=XXXXXXXXXXX</label>
<input type="text" class="form-control" name="youtube_id"
placeholder="Ссылка на Youtube" required>
</div>
<div class="form-group">
<label for="course_id">Курс</label>
<select class="form-control" name="course_id"  id="" style="">
@foreach ($courses as $course)
<option value="{{$course->id}}">{{$course->name}}</option>
@endforeach
</select>
</div>
<div class="form-group">
<label for="teacher_id">Преподаватель</label>
<select class="form-control" name="teacher_id"  id="">
@foreach ($teachers as $teacher)
<option value="{{$teacher->id}}">{{$teacher->name}}</option>
@endforeach
</select>
</div>
<button class="btn btn-success" style="margin-top:
20px;">Создать</button>
</div>
</form>
</div>
</div>
</div>
@endsection
```

Содержимое файла index.blade.php

```
@extends('admin')
@section('content')
<div class="container">
<div class="row">
<div class="col-md-12 col-md-offset-1">
<h2>Видео</h2>
<table class="table">
<thead>
<tr>
<th>Название</th>
<th>Курс</th>
<th>Преподаватель</th>
<th>Дата</th>
</tr>
</thead>
```

```
<tbody>
@foreach ($videos as $video)
<tr>
<td>{{$video->title}}</td>
<td>
@if (!empty($video->courses))
{{$video->courses->name}}
@endif
</td>
<td>
@if (!empty($video->teachers))
{{$video->teachers->name}}
@endif
</td>
<td>{{$video->updated_at->format('d.m.Y H:i')}}</td>
<td>
<a href="{{route('video.edit', $video->id)}}"><i class="fas fa-
eye"></i></a>
<a href="{{route('video.edit', $video->id)}}"><i class="fas fa-
edit"></i></a>
<form action="{{route('video.delete', $video->id)}}"
method="POST" class="delete-form">
{{ csrf_field() }}
{{ method_field('DELETE') }}
<button class="admin-delete"><i class="fas fa-trash-
alt"></i></button>
</form>
</td>
</tr>
@endforeach
</tbody>
</table>
<a href="video/create" class="btn btn-success">Добавить</a>
</div>
</div>
</div>
@endsection
```

Содержимое файла update.blade.php

```
@extends('admin')
@section('content')
<div class="container">
<div class="row">
<h3 style="margin: 25px 0;">Редактировать видео</h3>
<div class="col-md-12 col-md-offset-1">
<form action="{{route('video.update', $videos->id)}}"
method="POST">
{{ csrf_field() }}
<div class="form-group">
<div class="form-group">
<label for="title">Название видео</label>
```

```
<input type="text" class="form-control" name="title"
placeholder="Название видео" value="{{$videos->title}}">
</div>
<div class="form-group">
<label for="youtube_id">Ссылка на YouTube формата
https://www.youtube.com/watch?v=XXXXXXXXXXX</label>
<input type="text" class="form-control" name="youtube_id"
placeholder="Youtube ID" value="{{$videos->youtube_id}}">
</div>
<div class="form-group">
<label for="course_id">Курс</label>
<select class="form-control" name="course_id"  id="">
@foreach ($courses as $course)
<option value="{{$course->id}}"
@if($videos->course_id == $course->id) selected @endif
>{{$course->name}}</option>
@endforeach
</select>
</div>
<div class="form-group">
<label for="teacher_id">Преподаватель</label>
<select class="form-control" name="teacher_id"  id="">
@foreach ($teachers as $teacher)
<option value="{{$teacher->id}}" @if($videos->teacher_id ==
$teacher->id) selected @endif
>{{$teacher->name}}</option>
@endforeach
</select>
</div>
<button class="btn btn-success" style="margin-top:
20px;">Обновить</button>
</div>
</form>
</div>
</div>
</div>
@endsection
```

Содержимое файла video.php

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
class Video extends Model
{
public $table = 'video';
public function courses () {
return $this->belongsTo('App\Course', 'course_id');
}
public function teachers () {
return $this->belongsTo('App\Teacher', 'teacher_id');
}
public function addVideo ($r) {
$this->title = $r->get('title');
```

```
$url = $r->get('youtube_id');
$lp = explode('v=', $url);
$this->youtube_id = explode('&', end($lp))[0];
$this->course_id = $r->get('course_id');
$this->teacher_id = $r->get('teacher_id');
$this->save();
}
public static function updateVideo ($r, Video $video) {
$video->title = $r->get('title');
$video->youtube_id = $r->get('youtube_id');
$video->course_id = $r->get('course_id');
$video->teacher_id = $r->get('teacher_id');
$video->save();
}
}
```

Содержимое файла user.php

```
<?php
namespace App;
use Illuminate\Notifications\Notifiable;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Foundation\Auth\User as Authenticatable;
class User extends Authenticatable
{
use Notifiable;
protected $fillable = [
'name', 'email', 'password',
];
protected $hidden = [
'password', 'remember_token',
];
protected $casts = [
'email_verified_at' => 'datetime',
];
}
```

Содержимое файла teacher.php

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
class Teacher extends Model
{
public $table = 'teacher';
public $timestamps = false;
public function videos () {
return $this->hasMany('App\Video', 'teacher_id');
}
}
```

Содержимое файла subject.php

```
<?php
```

```
namespace App;
use Illuminate\Database\Eloquent\Model;
use app\Video;
class Subject extends Model
{
public $table = 'subject';
public $timestamps = false;
public function courses () {
return $this->hasMany('App\Course', 'subject_id');
}
}
```

Содержимое файла course.php

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\Storage;
class Course extends Model
{
public $table = 'course';
public function subjects () {
return $this->belongsTo('App\Subject', 'subject_id');
}
public function videos () {
return $this->hasMany('App\Video', 'id');
}
public function addCourse ($r) {
$this->name = $r->get('name');
$file = $r->file('img');
$this->img = $file->store('images/uploads', 'public');
$this->subject_id = $r->get('subject_id');
$this->save();
}
public static function updateCourse ($r, Course $course) {
$course->name = $r->get('name');
$file = $r->file('img');
if ($file) {
unlink(public_path() .'/storage/'. $course->img);
$course->img = $file->store('images/uploads', 'public');
}
$course->subject_id = $r->get('subject_id');
$course->save();
}
}
```

Содержимое файла videocontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Video;
use App\Subject;
```

```
use App\Teacher;
use App\Course;
class VideoController extends Controller
{
public function index () {
$courses = Course::orderBy('created_at', 'desc')->limit(4)-
>get();
$videos = Video::with('courses', 'teachers')-
>orderBy('created_at', 'desc')->limit(3)->get();
return view('content.index', ['videos'=>$videos, 'courses' =>
$courses]);
}
public function create () {
$courses = Course::get();
$teachers = Teacher::get();
return view('admin.video.create', ['courses' => $courses,
'teachers' => $teachers]);
}
public function edit ($id) {
$courses = Course::get();
$teachers = Teacher::get();
$videos = Video::with('courses', 'teachers')->find($id);
return view('admin.video.update', ['videos' => $videos,
'teachers' => $teachers, 'courses' => $courses]);
}
public function update (Request $request, $id) {
$video = Video::find($id);
Video::updateVideo($request, $video);
return redirect()->route('video.index');
}
public function store (Request $request) {
$video = new Video;
$video->addVideo($request);
return redirect()->route('video.index');
}
public function delete ($id) {
Video::find($id)->delete();
return redirect()->route('video.index');
}
public function category () {
$videos = Video::with('courses')->paginate(5);
return view('content.lecture', ['videos'=>$videos]);
}
public function show ($id) {
$video = Video::with('courses')->find($id);
$next = Video::where('id', '>', $video->id)->where('course_id',
'=', $video->course_id)->first();
$previous = Video::where('id', '<', $video->id)-
>where('course_id', '=', $video->course_id)->first();
return view('content.video', ['video'=>$video , 'next'=>$next ,
'previous'=>$previous]);
}
}
```

Содержимое файла teachercontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Video;
use App\Subject;
use App\Teacher;
class TeacherController extends Controller
{
public function index () {
$teachers = Teacher::with('videos')->get();
return view('admin.teachers.index', ['teachers'=>$teachers]);
}
public function create () {
return view('admin.teachers.create');
}
public function edit ($id) {
$teacher = Teacher::find($id);
return view('admin.teachers.update', ['teacher' => $teacher]);
}
public function update (Request $request, $id) {
$teacher = Teacher::find($id);
$teacher->name = $request->get('name');
$teacher->save();
return redirect()->route('teacher.index');
}
public function store (Request $request) {
$teacher = new Teacher;
$teacher->name = $request->name;
$teacher->save();
return redirect()->route('teacher.index');
}
public function delete ($id) {
Teacher::find($id)->delete();
return redirect()->route('teacher.index');
}
}
```

Содержимое файла subjectcontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Video;
use App\Subject;
use App\Teacher;
use App\Course;
class SubjectController extends Controller
{
public function index () {
$subjects = Subject::get();
return view('admin.subjects.index', ['subjects'=>$subjects]);
```

```
}
public function create () {
return view('admin.subjects.create');
}
public function edit ($id) {
$subject = Subject::find($id);
return view('admin.subjects.update', ['subject' => $subject]);
}
public function update (Request $request, $id) {
$subject = Subject::find($id);
$subject->name = $request->name;
$subject->save();
return redirect()->route('subject.index');
}
public function store (Request $request) {
$subject = new Subject;
$subject->name = $request->get('name');
$subject->save();
return redirect()->route('subject.index');
}
public function delete ($id) {
Subject::find($id)->delete();
return redirect()->route('subject.index');
}
public function category () {
$subjects = Subject::with('courses')->get();
return view('content.subjects', ['subjects'=>$subjects]);
}
public function show ($id) {
$courses = Course::with('subjects')->where('subject_id', '=',
$id)->paginate(5);
return view('content.courses', ['courses'=>$courses]);
}
}
```

Содержимое файла searchcontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Video;
use App\Subject;
use App\Teacher;
use App\Course;
class SearchController extends Controller
{
public function search (Request $request) {
$videos = Video::where('title', 'LIKE',  '%' . $request->search
. '%')->get();
return view ('content.lecture', ['videos'=>$videos]);
}
}
```

Содержимое файла feedbackcontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Mail;
use App\Mail\MailClass;
class FeedbackController extends Controller
{
public function index () {
return view('content.feedback');
}
public function send (Request $request) {
$name = $request->name;
$email = $request->email;
$text = $request->text;
Mail::to('test@mail.ru')->send(new MailClass($name, $email,
$text));
return redirect()->route('main.feedback');
}
}
```

Содержимое файла coursecontroller.php

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Video;
use App\Subject;
use App\Teacher;
use App\Course;
class CourseController extends Controller
{
public function index () {
$courses = Course::with('subjects')->get();
return view('admin.course.index', ['courses'=>$courses]);
}
public function create () {
$subjects = Subject::get();
return view('admin.course.create', ['subjects' => $subjects]);
}
public function edit ($id) {
$subjects = Subject::get();
$courses = Course::with('subjects')->find($id);
return view('admin.course.update', ['courses' => $courses,
'subjects' => $subjects]);
}
public function update (Request $request, $id) {
$course = Course::find($id);
Course::updateCourse($request, $course);
return redirect()->route('course.index');
}
public function store (Request $request) {
```

```
$course = new Course;
$course->addCourse($request);
return redirect()->route('course.index');
}
public function delete ($id) {
Course::find($id)->delete();
return redirect()->route('course.index');
}
public function category () {
$courses = Course::with('subjects')->paginate(5);
$subjects = Subject::get();
return view('content.courses', ['courses'=>$courses, 'subjects'
=> $subjects]);
}
public function show ($id) {
$videos = Video::with('courses')->where('course_id', '=', $id)-
>paginate(5);
$course_name = Course::find($id)->first();
return view('content.course', ['videos'=>$videos,
'course_name'=>$course_name]);
}
public function filter (Request $request) {
$categorys = array();
$subjects = Subject::all()->max('id');
for($i = 0; $i <= $subjects; $i++)
if($request->input('check'.$i))
array_push($categorys, $i);
$courses = Course::whereIn('subject_id', $categorys)-
>with('subjects')->paginate(5);
$subjects = Subject::get();
return view('content.courses', ['courses'=>$courses, 'subjects'
=> $subjects]);
dd($result);
}
}
```