МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

о научно-исследовательской работе

Тема: «Разработка веб-платформы для обучения программированию на JavaScript с использованием микросервисной архитектуры»

Студент гр. 4304 _____ Михайлов А.А.

Преподаватель _____

Санкт-Петербург
2019

ЗАДАНИЕ

Студент Михайлов А.А.

Группа 4304

Тема работы: Разработка веб-платформы для обучения программированию на JavaScript с использованием микросервисной архитектуры

Дата сдачи реферата: 25.12.2019

Дата защиты реферата: 25.12.2019

Студент _____ Михайлов А.А.

Преподаватель _____

АННОТАЦИЯ

Цель работы – создание прототипа «песочницы», которая представляет собой симуляцию браузера внутри браузера, и которая будет использоваться в дальнейшем как ядро для разработки рабочей области студента.

В результате выполнения данной работы были рассмотрены популярные современные аналоги с целью выявления недостатков уже существующих решений, а также доказана актуальность данного направления в IT-индустрии. Были выбраны наиболее подходящие инструменты для дальнейшей разработки, спроектирована архитектура и реализован прототип «песочницы».

SUMMARY

The purpose of the work is to create a prototype of the «sandbox», which is a simulation of the browser inside the browser, and which will be used in the future as a kernel for developing the student’s workspace.

As a result of this work, popular modern analogues were considered in order to identify the shortcomings of existing solutions, and the relevance of this direction in the IT industry was also proved. The most suitable tools for further development were selected, the architecture was designed and a prototype of the sandbox was implemented.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Онлайн-обучение (e-learning, дистанционное обучение, электронное обучение) – это метод получения новых знаний с помощью Интернета в режиме реального времени. На данный момент индустрия e-learning одна из самых быстро развивающихся в мире технологий в сфере образования.

В силу своего удобства дистанционное образование становится все более популярной формой обучения. Процесс обучения представляет собой взаимодействие педагога, обучаемого и средств обучения. Возможности современных компьютерных средств и информационных технологий позволяют возложить на средства обучения часть функций преподавателя и часть функций обучаемого, принятых в классической форме обучения. Это и является главной причиной роста популярности данного формата обучения. Кроме того, обучение через Интернет прекрасно подходит для тех, кто живёт в отдалённых районах, а также для тех, кто в силу определённых причин не может посещать очную форму обучения. Несомненными преимуществами дистанционных курсов обучения в режиме «онлайн» также являются:

- Возможность для обучающегося самостоятельно выстраивать график обучения, а также определять продолжительность занятий.
- Свободный выбор. Учащийся выбирает любой из доступных курсов обучения, а также самостоятельно планирует время, место и продолжительность занятий.
- Доступность. Независимо от географического положения и времени учащийся имеет доступ к образовательному ресурсу и материалам курса.
- Технологичность – использование в образовательном процессе новейших достижений информационных и телекоммуникационных технологий.

В качестве инструмента для удаленного обучения можно использовать

соответствующее веб-приложение, которое будет предоставлять студентам удаленный доступ ко всем лекционным материалам и практическим заданиям. Преимуществом такого веб-приложения будет наличие автоматизации проверки практических заданий.

Веб-приложение — это клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Одним из преимуществ такого подхода является тот факт, что клиенты не зависят от конкретной операционной системы пользователя, поэтому веб-приложения являются межплатформенными службами [1].

Веб-приложение состоит из клиентской и серверной частей, тем самым реализуя технологию «клиент-сервер». Клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него. Серверная часть получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP [1].

Само веб-приложение может выступать в качестве клиента других служб, например, базы данных или другого веб-приложения, расположенного на другом сервере [1].

Цель данной работы состоит в реализации веб-приложения на основе микросервисной архитектуры для динамичного онлайн-обучения студентов основам веб-разработки: HTML, CSS, JavaScript. Для достижения цели необходимо решить следующие **задачи**:

- Провести обзор предметной области.
- Рассмотреть существующие решения и провести сравнительный анализ.
- Сформировать перечень требований к разрабатываемому продукту.
- Спроектировать архитектуру веб-приложения и структуру БД.

- Разработать UI/UX-дизайн приложения.
- Составить программу обучения.
- Реализовать продукт, удовлетворяющий всем указанным требованиям.
- Провести тестирование.

Объектом исследования является обучение основам веб-разработки.

Предметом исследования является инструмент, позволяющий проводить обучение основам веб-разработки онлайн и автоматизировать сам процесс.

1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Актуальность тематики.

В настоящее время слушателей топ-10 мировых ресурсов в области онлайн-образования – около 20 млн. человек, что является огромной аудиторией. При этом, по информации Ambient Insight Research, совокупные инвестиции в эти ресурсы превысили $300 млн., а общий объем инвестиций в онлайн-образование только в 2000-2013 годах составил $8,5 млрд.

Согласно существующим оценкам аналитиков, текущий объем мирового рынка онлайн-образования оценивается более чем в $40 млрд. По прогнозу в 2019 году эта цифра превысит $70 млрд. Такой бурный рост на фоне стагнации остальной экономики привел к активному интересу IT компаний и части преподавательского состава, настроенной на инновации, к новому обучению. Фактически сейчас наблюдается массовая миграция курсов из оффлайна в онлайн. В России появилось множество проектов, связанных с переводом учебных курсов в онлайн-режим. Центры онлайн-образования открываются как при вузах, так и внутри IT компаний, создавших системы, поддерживающие процесс онлайн-образования.

1.2 Обзор аналогов и их сравнение

Список аналогичных ресурсов, которые занимаются онлайн обучением в сфере веб-разработки:

- Яндекс.Практикум.
- HtmlAcademy.
- GeekBrains.
- Codebra.
- JsExpert.
- OTUS.

Перечисленные выше ресурсы имеют разную тематику и ориентированы на

определенный круг пользователей. И чтобы провести анализ того, насколько эти ресурсы клиентоориентированны и удобны в использовании, можно выделить следующие критерии для оценивания с точки зрения клиента:

- Специализированность.
- Дизайн.
- Доступность.
- Автономность.
- Общая оценка.

Результаты сравнения ресурсов по критериям представлены в табл. 1.1. Критерий оцениваются в баллах от 1 до 10. Специализированность оценивается, как «Да» или «Нет».

Таблица 1.1 – Сравнение ресурсов в сети Интернет

	Специализированность	Дизайн	Доступность	Автономность	Общая оценка
Яндекс.Практикум	Нет	4	4	6	5
HtmlAcademy	Да	10	8	8	9
GeekBrains	Нет	8	2	2	4
Codebra	Да	1	9	10	7
JsExpert	Да	7	4	5	6
OTUS	Нет	9	7	6	7

На основе полученных результатов можно сделать следующие выводы:

- Можно заметить, что половина ресурсов не имеет конкретной тематики.

- Также у очень многих ресурсов довольно низкая доступность – то есть курсы не являются бесплатными и зачастую стоят довольно много.
- Общий недостаток большинства ресурсов – низкая автономность – то есть частичное или полное отсутствие возможности студентов проходить обучение, не контактируя с преподавателями.

1.3 Вывод

Был рассмотрена актуальность тематики онлайн-образования, а также проведен обзор существующих ресурсов, предоставляющих доступ к учебным материалам и практическим заданиям.

Большинство ресурсов являются платными, причем некоторые – достаточно дорогие. Только у нескольких платных ресурсов есть пробная версия. Единственный полностью бесплатный ресурс является устаревшим и имеет плохое оформление и дизайн. Как платные, так и бесплатные решения имеют существенные проблемы с автоматизацией проверки практических занятий. А для некоторых ресурсов такая функциональность и вовсе не предусмотрена, а обучение проводиться в форме онлайн-лекций.

2. ОПИСАНИЕ МЕТОДА РЕШЕНИЯ

2.1 Создание прототипа «песочницы»

2.1.1 Определение понятия «песочница»

Песочница (sandbox) – в компьютерной безопасности это механизм для безопасного исполнения программ.

Обычно песочница – это жестко контролируемый набор ресурсов, предназначенный для исполнения гостевой программы, например, место в памяти или на диске. Доступ к сети, возможность сообщения с главной ОС или считывание информации с устройства ввода зачастую либо эмулируют, либо ограничивают. По сути – песочница это пример виртуализации. Часто песочницы используют при разработке ПО для запуска еще «сырого» кода, который возможно может повредить систему либо сложную конфигурацию. Песочницы копируют самые основные элементы среды, для которой был написан код, и позволяют разработчикам безопасно экспериментировать.

В нашем случае песочница – это симуляция браузера внутри браузера, которая позволит пользователю на странице веб-приложения писать HTML, CSS и JS-код, который будет отображаться на этой же странице веб-приложение в специальной области, которая и будет симулировать поведение браузера.

2.1.2 Реализация прототипа

На рисунке 2.1 представлен прототип песочницы.

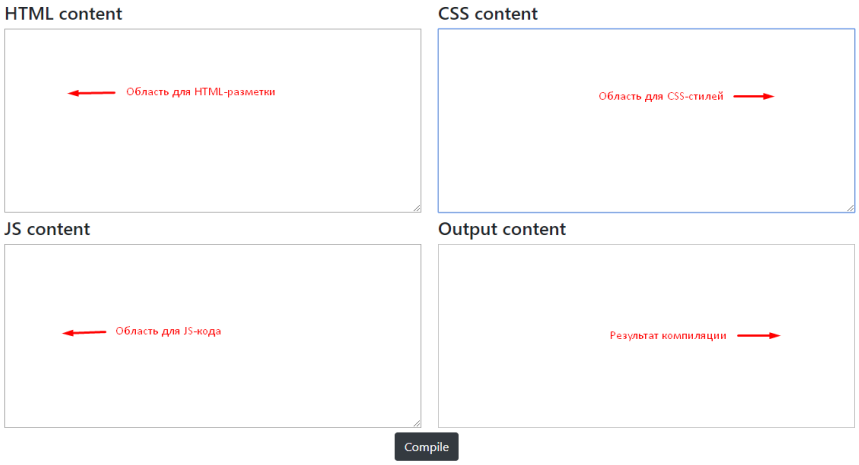

Рисунок 2.1. Прототип песочницы

На рисунке 2.2 представлен прототип песочницы с содержимым внутри блоков «HTML content», «CSS content» и «JS content».

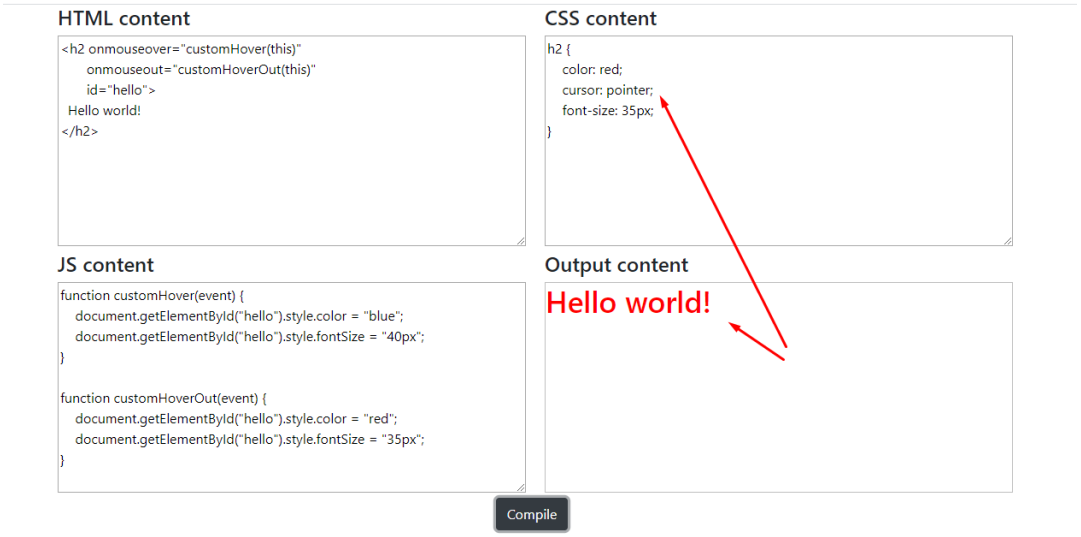

Рисунок 2.2. Прототип песочницы с содержимым внутри блоков

На рисунке 2.2 в блоке «Output content» отображается разметка из блока «HTML content», к котором применен стиль из блока «CSS content». Именно поэтому текст имеет красный текст. К данному тексту привязан JS-скрипт из блока «JS content», который исполнится при наведении на него курсора мыши.

На рисунке 2.3 представлен прототип песочницы, когда на него наведен курсор мыши (использован эффект hover).

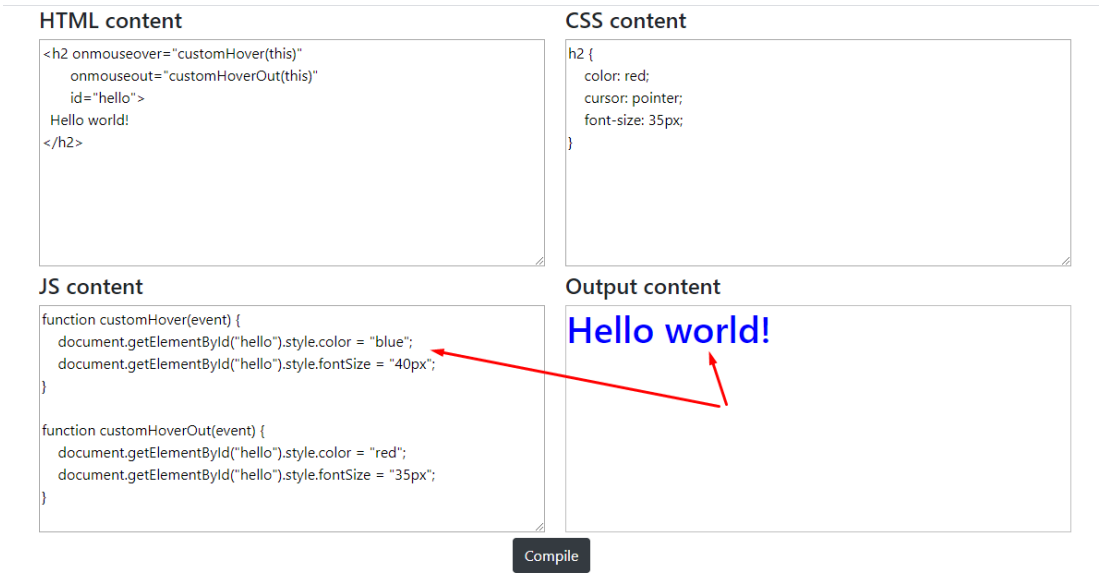

Рисунок 2.3. Прототип песочницы с примененным hover-эффектом

2.2 Описание архитектуры приложения

2.2.1 Описание используемых технологий

Разрабатываемый инструмент представляет собой веб-приложение. В качестве языка программирования (ЯП) был выбран язык Java. Было принято решение использовать язык Java. Этот выбор довольно просто обосновать, потому что Java имеет действительно большое число различных преимуществ[3, 4]:

- Java – это объектно-ориентированный язык программирова-ния. Это позволяет с легкостью моделироваться системы любой сложности и расширять уже существующие;
- Данный язык программирования имеет довольно низкий порог вхождения;
- Just-In-Time компилятор Java позволяет добиться действитель-но высокой производительности;
- Данный язык очень распространен в мире ИТ, что позволяет без особых проблем привлечь к разработке новых специали-стов. Абсолютное большинство всех известных проблем, с ко-торыми можно встретиться в процессе разработки, уже решены и решения эти с легкостью можно найти в сети Интернет;
- Обширный стек технологий;
- Пожалуй, одно из самых больших преимуществ языка Java над остальными – платформонезависимость. Любая Java-программа компилируется в байт-код, который может выпол-няться на любой машине независимо от ее платформы с помо-щью Java Virtual Machine (JVM);

После того, как сам язык был выбран, необходимо определиться с

остальным стеком технологий, которые будут использоваться в процессе разработки. Для реализуемого программного продукта выбор СУБД не критично важен, поэтому остановиться можно на любой из них. Выбор пал на MongoDB, так как имеет внутреннюю поддержку для реактивного стиля программирования, который будет использоваться при реализации.

Сборщиком проектов был выбран Maven. Выбор обусловлен тем, что это, де-факто, стандартный сборщик в мире Java Enterprise-приложений.

В качестве интегрированной среды разработки была выбрана Intellij IDEA от компании JetBrains, так как это самая удобная среда разработки.

В качестве основного фреймворка для разработки был выбран Spring Framework по нескольким причинам:

- Он содержит всю необходимую инфраструктуру для создания современного и высоконагруженного Enterprise-приложения;
- Гибкость конфигурации;
- Использование Spring позволяет масштабировать систему в будущем с минимальным количеством затрачиваемых усилий;
- Spring поощряет подход использования слабой связанности между компонентами. Как следствие:
    - Упрощение инициализации и настройки компонентов;
    - Упрощение модульного тестирования;
    - Упрощение разработки и дальнейшей поддержи приложения в целом;

Помимо всего этого, стоит также отметить, что Spring Framework – это, пожалуй, самое популярное решение в плане разработки enterprise-приложений на языке Java и Kotlin. Оно позволяет конфигурировать и разрабатывать приложения абсолютно любой сложности. У Spring Framework одно из самых больших сообществ разработчиков в мире, это по большей части связано с тем, что данный фреймворк является open-source проектом, что также является значительным плюсом. Spring обладает весьма развитой экосистемой, большим числом

независимых модулей, которые используются по мере необходимости[5]. Однако есть и ряд существенных минусов. Основные из них:

- Относительно высокий порог вхождения;
- Сложность конфигурации. Цена за предоставляемую гибкость. Поэтому Spring Framework нецелесообразно использовать для небольших проектов. Это также одна из причин высокого порога вхождения для новичков.

Ознакомиться с инфраструктурой Spring Framework можно на рисунке 2.4.

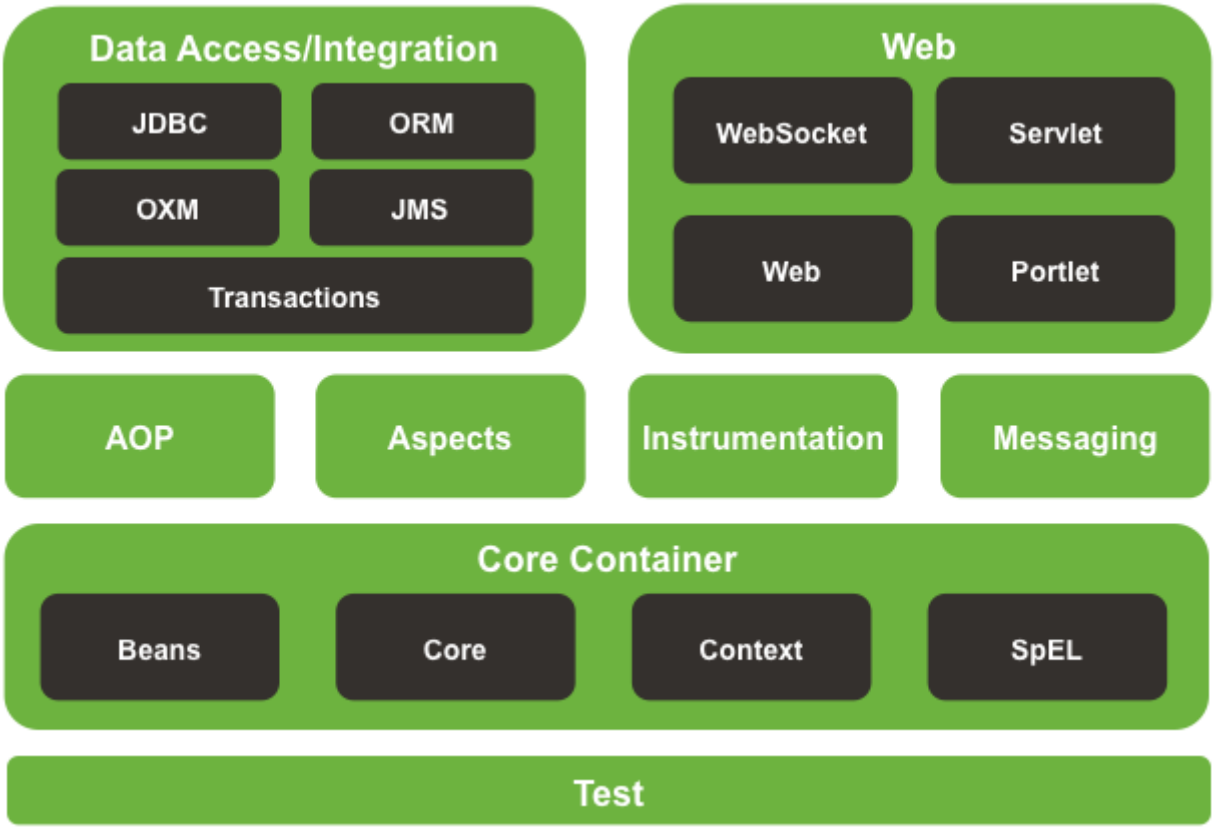

Рисунок 2.4. – Инфраструктура Spring Framework

В качестве контейнера сервлетов был выбран встроенный в Spring Jetty.

2.2.2. Архитектура приложения

Один из ключевых и самых основных принципов проектирования для достижения таких характеристик – разделение ответственности. Он базируется на том, что каждый отдельно взятый компонент или модуль должен быть ответственен только за одно конкретное свойство или функцию в приложении.

Это относится как к приложению целиком, так и к отдельным частям. В дальнейшем для реализации приложения будет использоваться микросервисная архитектура, где каждый сервис будет представлять из себя цельную с семантической точки зрения единицу. Каждый сервис будет спроектирован по принципу многоуровневой архитектуры.

Многоуровневая архитектура имеет следующие достоинства: абстракция, изоляция, более прозрачная организация, масштабируемость отдельных модулей, независимое тестирование функциональности отдельных модулей, возможность переиспользования модулей.

Чаще всего используются трехуровневая модель. Она состоит из уровня представления, уровня бизнес-логики и уровня доступа к данным. Задача уровня представления заключается в коммуникации с пользователем: преобразование данных для визуализации, предоставление интерфейса взаимодействия и последующая обработка действий пользователя. Бизнес-уровень содержит все функциональные алгоритмы, выступает в роли посредника между уровнем представления и уровнем доступа к данным. Последний же представляет из себя хранилище данных и набор унифицированных методов доступа к этим данным, известных как CRUD (Create, Update, Delete).

Более наглядно ознакомиться с данной моделью многоуровневой архитектуры можно на рисунке 2.5.

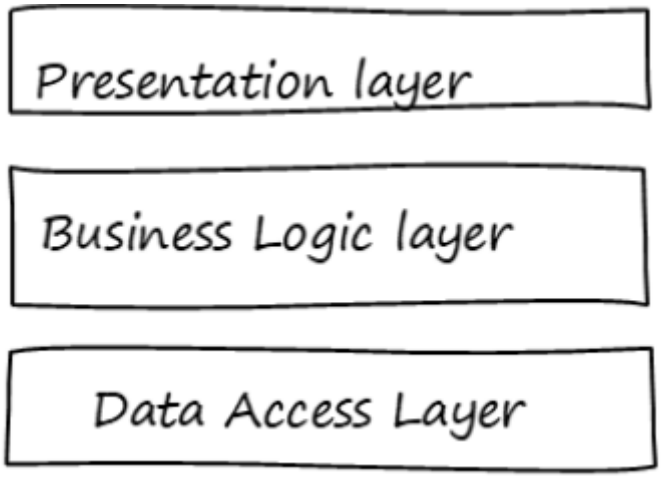

Рисунок 2.5. – Трехуровневая модель архитектуры

В реализуемом приложении используется именно трехуровневая модель

архитектуры.

Для реализации уровня доступа к данным используется паттерн проектирования DAO (Data Access Object). DAO – это объект, который предоставляет абстрактный интерфейс для работы с БД или каким-либо другим хранилищем данных. Использование данного паттерна предоставляет ряд возможностей работы с данными независимо от того, какой именно механизм хранения данных используется.

Уровень бизнес-логики реализован с помощью сервисов – набора интерфейсов, который используются уровнем представления для совершения манипуляций над данными.

Для реализации уровня представления был использован паттерн проектирования, известный как MVC (Model-View-Controller). Наглядно он продемонстрирован на рисунке 2.6.

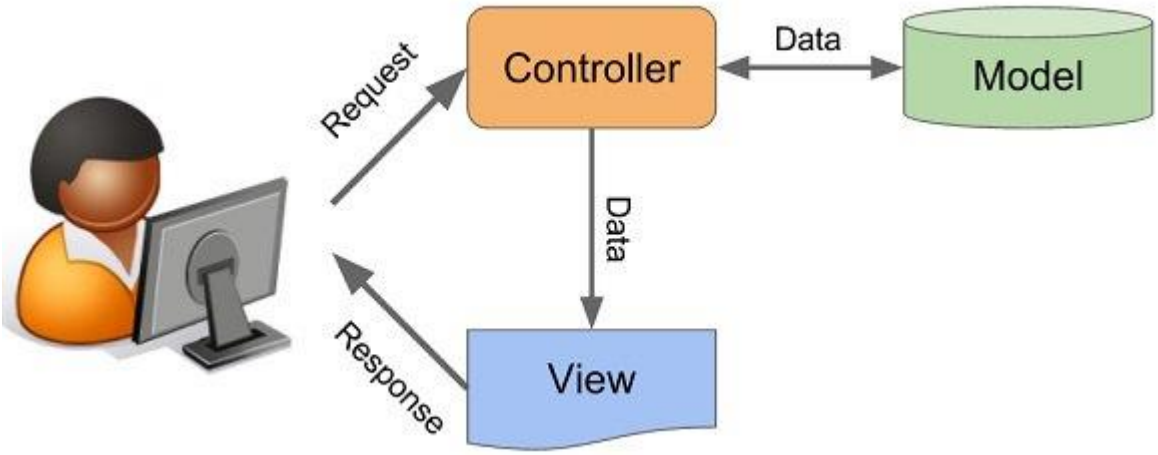

Рисунок 2.6. – Паттерн Model-View-Controller

Представление (View) отвечает за отображение данных модели пользователю. Пользователь взаимодействует с предоставляемым ему графическим интерфейсом, тем самым изменяя модель. Такие изменения обрабатываются контроллером.

Контроллер (Controller) – это интерпретатор действий пользователя, который с помощью интерфейса взаимодействия с уровнем бизнес-логики оповещает модель о необходимости изменения.

Модель (Model), в свою очередь, представляет из себя набор данных,

которые могут видоизменяться, реагируя на команды контроллера[2].

ЗАКЛЮЧЕНИЕ

В результате проведенной работы был выполнен начальный этап разработки веб-приложения для онлайн-обучения студентов основам веб-разработки: HTML, CSS, JavaScript. В частности, был осуществлен анализ предметной области, были рассмотрены популярные современные аналоги с целью выявления их недостатков, была доказана актуальность направления онлайн-образования в IT-индустрии, а также были выбраны наиболее подходящие инструменты для дальнейшей разработки и был реализован прототип «песочницы».

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Марко Беллиньясо. Разработка Web-приложений в среде ASP.NET 2.0: задача — проект — решение = ASP.NET 2.0 Website Programming: Problem - Design - Solution. — М.: Издательство Диалектика. - 2007. — С. 640.
2. Свободная энциклопедия Википедия. [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Model-View-Controller (дата обращения: 25.12.2019).
3. Б. Эккель Философия Java СПб.: Питер, 2003. 976 с.
4. Г. Шилдт Java 8. Полное руководство М.: Вильямс, 2015. 1376 с.
5. К. Уоллс Спринг в действии. 3-е полное изд. СПб: ДМК Пресс, 2013. 721 с.