

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ
ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ им. А. И. ГЕРЦЕНА»



Основная профессиональная образовательная программа
Направление подготовки 09.03.01 Информатика и вычислительная техника
Направленность (профиль) «Технологии разработки программного обеспечения»
форма обучения – очная

Выпускная квалификационная работа

Разработка электронного образовательного ресурса «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта»

Обучающегося 4 курса
Косыгина Кирилла Сергеевича

Научный руководитель:
Кандидат физ.-мат. наук, доцент
Жуков Николай Николаевич

Санкт-Петербург

2022

Содержание

ВВЕДЕНИЕ.....	3
1 ПРОБЛЕМАТИКА РАЗРАБОТКИ ЭОР.....	5
1.1 Определение и сущность ЭОР.....	5
1.2 Особенности использования ЭОР.....	6
1.2.1 Общие требования к ЭОР.....	8
1.2.2 Требования к представлению учебного материала.....	9
1.3 Обзор инструментов для создания ЭОР.....	11
1.3.1 LMS Moodle.....	12
1.3.2 CMS-базированные решения.....	13
1.3.3 Прочие инструменты.....	14
1.4 Этапы разработки ЭОР.....	15
1.5 Технологические особенности разработки ЭОР.....	16
2 РАЗРАБОТКА ЭОР.....	19
2.1 Адаптация требований к ЭОР.....	19
2.2 Проектирование ЭОР.....	20
2.2.1 Структура ЭОР.....	21
2.2.2 Выбор технологий.....	25
2.2.3 Структура программного проекта.....	27
2.3 Создание ЭОР.....	27
2.3.1 Подготовка рабочего окружения.....	27
2.3.2 Наполнение ресурса.....	28
2.3.3 Введение стимулирующих элементов.....	32
2.4 Контейнеризация ресурса.....	33
ЗАКЛЮЧЕНИЕ.....	36
СПИСОК ЛИТЕРАТУРЫ.....	38
ПРИЛОЖЕНИЯ	

ВВЕДЕНИЕ

Непрерывная интеграция и непрерывная доставка (англ. Continuous Integration и Continuous Delivery или CI/CD) — это практики разработки программного обеспечения, ориентированные на автоматизацию сборки и выпуска обновлений программного продукта в целях повышения его качества. Практики CI/CD получили настолько широкое распространение, что к разработчикам программного обеспечения сегодня выдвигаются новые требования: начинающие разработчики должны владеть основами CI/CD и уметь поддерживать существующие пайплайны, а более опытные разработчики должны уметь самостоятельно внедрять CI/CD в циклы разработки ПО [14]. Отсюда необходимость в учебных материалах по этой теме как для первичного освоения материала, так и для повышения квалификации. Учитывая специфику и навыки целевой аудитории (начинающие специалисты в области информационных технологий), можно утверждать, что электронный образовательный ресурс (ЭОР) — это наиболее подходящий вид образовательного ресурса в данной ситуации.

Следовательно, объект исследования данной работы — применение компьютерных и цифровых технологий в обучении.

Предмет исследования — разработка электронного образовательного ресурса практической направленности, предназначенного для изучения практик CI/CD в среде GitHub.

Целью данной работы является разработка электронного образовательного ресурса «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта».

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить сущность ЭОР и особенности их разработки.
2. Провести анализ инструментов, предназначенных для разработки ЭОР.
3. Адаптировать требования к разрабатываемому ЭОР с учётом особенностей изучения практик CI/CD.

4. Спроектировать ЭОР «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта».
5. Создать ЭОР в соответствии с разработанным планом.

В работе выдвигается следующая гипотеза: возможно разработать электронный образовательный ресурс практической направленности «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта».

При проведении исследования применяются следующие методы научного исследования: анализ, синтез, дедуктивный и индуктивный методы, наблюдение, сравнение, абстрагирование.

Теоретическая значимость проводимого исследования заключается в том, что при разработке электронного образовательного ресурса будут рассмотрены и обобщены теоретические аспекты разработки подобных ресурсов: будут изучены основные требования к ЭОР, методологии адаптации под специфические требования и типовые инструменты разработки ЭОР. Практическая значимость исследования состоит в том, что разработанный ресурс может быть использован учащимися для освоения практик CI/CD или применён при разработке других учебных материалов.

1 ПРОБЛЕМАТИКА РАЗРАБОТКИ ЭОР

Информатизация образовательного процесса является одной из целей современного российского образования [14].

Современные преподаватели должны уделять особое внимание информационно–коммуникационным технологиям, потому что современные технологии способствуют оптимизации учебного процесса.

Как показывают психолого-педагогические исследования, именно использование электронных образовательных ресурсов в образовательном процессе позволяет педагогу реализовать на практике инновационные идеи и направления индивидуализации и информатизации образования [10].

Активное взаимодействие пользователя с электронным учебным продуктом является главным преимуществом и стратегической задачей образования [13]. Следовательно, уровень интерактивности, другими словами – уровень активности пользователя при работе с ресурсом служит одним из важнейших показателей качества ЭОР.

1.1 Определение и сущность ЭОР

Электронный образовательный ресурс (ЭОР) – это учебные материалы, для воспроизведения которых используются электронные устройства. Иногда, чтобы выделить подмножество ЭОР, воспроизводимых на компьютере, используют понятие «цифровые образовательные ресурсы» (ЦОР). В данной работе автор следует межгосударственному стандарту ГОСТ 7.23-2001 и использует общий термин «электронные» и аббревиатуру ЭОР вместо «цифровые» и ЦОР, соответственно [4].

Структура, предметное содержание, методы и средства разработки и применения ЭОР определяются его функциональным назначением и спецификой применения в конкретных системах. ЭОР, прошедший редакционно-издательскую обработку, имеющий выходные сведения и предназначенный для

распространения в неизменном виде, является электронным изданием. Термин «Электронные образовательные ресурсы» (ЭОР) объединяет весь спектр средств обучения, которые разработаны и воспроизводятся на базе компьютерных технологий.

ЭОР может быть представлен в трёх формах [3].

1) Текстографические ЭОР. Самый простой и очевидный тип ЭОР. Такие ресурсы отличаются от книг формой, в которой представлены тексты и иллюстрации: материал отображается на экране компьютера, а не на бумаге, но его можно легко распечатать, т.е. перенести на бумагу.

2) Текстографические ЭОР с навигацией по тексту. Информация, представленная в виде книги, подразумевает, что страницы читаются последовательно, осуществляется линейная навигация; при этом довольно часто в учебном тексте встречаются термины или ссылки на другой раздел того же текста. В таких случаях книга не очень удобна: нужно искать пояснения в других местах, перелистывать множество страниц и регулярно терять контекст. В ЭОР навигацию можно реализовать гораздо комфортнее: указать незнакомый термин и тут же получить его определение в небольшом дополнительном окне, мгновенно сменить содержимое экрана при указании так называемого ключевого слова и т. д. В данном случае навигация по тексту является нелинейной, то есть можно просматривать фрагменты текста в произвольном порядке.

3) ЭОР, целиком состоящие из визуального или звукового материала: фильмы, аудиозаписи и т. п. Такие ресурсы невозможно представить в полиграфическом издании.

1.2 Особенности использования ЭОР

Использование электронных образовательных ресурсов в учебном процессе – это обязательная часть работы современного педагога. Крайне сложно вести уроки в соответствии с ФГОС, не прибегая к современным методам и средствам обучения, особенно – в условиях дистанционного обучения.

Во-первых, электронные средства обучения представляют любую информацию в более наглядном виде и дают ученикам наиболее полное представление об изучаемых объектах и явлениях. Во-вторых, они обладают большим мотивирующим потенциалом: ученикам нравится получать новую информацию при помощи современного оборудования, самостоятельно изучать те или иные темы, проверять себя и получать обратную связь. Наконец, электронный ресурс обладает большими возможностями по организации больших массивов данных. Следовательно, ЭОР способны предоставить ученику гораздо больше информации, чем традиционные ресурсы, при этом вся текстовая, визуальная, звуковая информация будет компактно размещаться на одном цифровом устройстве.

Также ЭОР позволяют легко решать задачи по приведению учебного процесса к стандартам ФГОС, а именно:

1. Предоставить максимум информации по изучаемой теме, представленной в самых различных видах – текст, картинка, звук, видео, мультимедиа, интерактивная игра и т.д.
2. Дать обучающимся возможность самостоятельно постигать, изучать новые темы, подбирать и анализировать информацию.
3. Быстро и легко организовать работу в парах и группах
4. Индивидуально подходить к каждому ученику, подбирая уникальные задания для каждого и помогая выстраивать индивидуальные образовательные траектории.

Важно отметить, что применение учащимися в ходе обучения таких программ, как PowerPoint, Notepad, Scribus и т.п., не является обращением к ЭОР. Учащийся обращается к служебной компьютерной программе для выполнения определённых задач и создания конкретного продукта – презентации, макета брошюры и т. д.; образовательного потенциала у этих сервисов, как правило, нет. С другой стороны, если учащийся создаёт викторину, которую решают другие, – тогда можно сказать, что он принимает участие в создании

своего локального, неофициального обучающего продукта – но не в полной мере образовательного ресурса.

1.2.1 Общие требования к ЭОР

Качество ЭОР — это степень соответствия совокупности характеристик, присущих ЭОР, требованиям.

В соответствии с ГОСТ Р 52653-2006 ЭОР должен включать в себя образовательный контент, программные компоненты и метаданные.

Образовательный контент – это организованная предметная информация, используемая в образовательном процессе.

Программные компоненты реализуют интерактивный режим работы пользователя с контентом.

Метаданные – это структурированные данные, предназначенные для описания характеристик ЭОР.

Перед этапом создания ЭОР требуется разработать сценарный план, который должен включать в себя следующие компоненты:

1. Имя ЭОР.
2. Тип ЭОР.
3. Краткое описание содержания ЭОР.
4. Количество сцен (страниц) и их описание.
5. Перечень используемых инструментальных средств.
6. Значения уровней интерактивности и мультимедийности.
7. Описание методов взаимодействия пользователя с контентом.
8. Описание алгоритма верного прохождения контрольных заданий.
9. Указание ПО, необходимого для воспроизведения ЭОР.

Качество ЭОР определяют [5]:

1. Содержательные характеристики – определяют качество, достаточность и степень проработки учебного материала, представленного в ЭОР.

2. Мультимедийность – свойство, определяющее качество форм представления информации, используемых в ЭОР.
3. Интерактивность – свойство, определяющее характер и степень взаимодействия пользователя с элементами ЭОР.
4. Модифицируемость – свойство, определяющее возможность внесения изменений в содержание и программные решения ЭОР.

1.2.2 Требования к представлению учебного материала

При оформлении ЭОР необходимо придерживаться следующих правил:

1. Единый стиль оформления контента в рамках ЭОР.
2. Удобство работы с оглавлением ЭОР и словарём, содержащим основные понятия.
3. Представление текстового учебного материала должно быть предельно лаконично.
4. Оформление не должно отвлекать пользователя от содержательной составляющей, однако должно качественно предоставлять все необходимые средства управления.
5. Обоснованность применения мультимедиа и графической информации.
6. Представление визуальных компонентов с глубиной цвета, минимально достаточной для кодирования используемого в них количества цветов.
7. Рациональное использование пространства визуальных компонентов.
8. Удобство и наглядность навигации, простота и оперативность переходов к требуемым разделам.
9. Интерфейс должен быть дружелюбным (наличие справки, «всплывающих» подсказок и т.п.).

Также электронный образовательный ресурс может содержать интерактивные элементы. Выделяют четыре уровня интерактивности:

1. Первый уровень интерактивности: условно-пассивные формы. Ресурс не предоставляет никаких интерактивных элементов. Например, пользователь читает текст, просматривает изображения и видео, прослушивает аудиозаписи.

2. Второй уровень интерактивности: активные формы. Ресурс предоставляет элементарные интерактивные элементы (нажатие на клавиши или клики мыши). На данном уровне можно встретить навигацию по гиперссылкам, викторины и задания с выбором ответа, различные просмотрщики сложных объектов (трёхмерных объектов, больших изображений и т.п.).

3. Третий уровень интерактивности: деятельностные формы. Учащиеся взаимодействуют с учебными объектами по заранее определённым алгоритмам; при этом производится оценка ошибок и отклонений. Активные формы, в отличие от деятельностных, имеют меньшую степень свободы. Деятельностные формы предоставляют учащимся выбор последовательности действий, ведущих к учебной цели; каждый шаг может иметь множество возможных решений. При этом пользователи должны в любом случае прийти к единственно верному решению, т.е. сценарий учебной задачи определён заранее. К деятельностным формам можно отнести: задания с вводом ответа, перемещение объектов с целью установления их соотношений и иерархий, изменение параметров процессов и объектов и т.д.

4. Четвёртый уровень интерактивности: исследовательские формы. Учащиеся должны быть источником событий. События могут вызывать изменения сущности, внешнего вида, параметров, характеристик изучаемых объектов, процессов или явлений. Главное отличие форм четвёртого уровня от рассмотренных ранее заключается в том, что состояния изучаемых объектов и процессов четвёртой формы не могут быть определены заранее.

Таким образом, формы первого, второго и третьего уровней являются «детерминированными»: все варианты действий пользователя заранее предусмотрены, существует только одно решение, которое считается верным. Формы четвёртого уровня – «недетерминированные»: при создании электронного

образовательного ресурса определены только исходные элементы содержимого и параметры процессов. В этом случае определить или предугадать все действия или ошибки пользователя невозможно.

Уровень интерактивности ресурса определяется применяемыми формами взаимодействия учащегося с образовательным контентом. Если интерактив базируется на детерминированных формах, тогда электронный образовательный ресурс должен содержать не менее четырёх различных форм взаимодействия, при этом:

ЭОР относится к первому уровню интерактивности, если в нем используется менее двух различных форм взаимодействия второго и третьего уровней;

ЭОР относится ко второму уровню интерактивности, если в нем используется две и более различных форм взаимодействия второго уровня, либо одна форма третьего уровня и одна или более – второго уровня;

ЭОР относится к третьему уровню интерактивности, если в нем используется две и более различных форм взаимодействия третьего уровня.

Достаточным условием отнесения ЭОР к четвёртому уровню является отсутствие детерминированности действий пользователя при манипуляциях с элементами ресурса.

Оценка уровня интерактивности исходит исключительно из взаимодействия пользователя с содержательными элементами, при этом операции с манипуляторами не учитываются. Создание ЭОР без интерактивного контента, то есть контента, который нельзя отнести ни к одному из указанных уровней интерактивности, не допускается.

1.3 Обзор инструментов для создания ЭОР

Значительно ускорить процесс разработки ЭОР можно с помощью специализированных программ.

Простые средства публикации ЭОР, основанные на использовании приложений Adobe Acrobat или Microsoft Office (Word, Excel, PowerPoint) наиболее удобны при создании и публикации электронных учебников и методических рекомендаций к ним. Для разработки анимаций в рамках ЭОР может использоваться Adobe Flash или Adobe Animate CC.

Для создания ЭОР в виде программного продукта могут быть использованы различные объектно-ориентированные языки программирования (C++, C#, Visual Basic .NET, Java, Delphi и др.) При проектировании программы рекомендуется использование инструментов UML-моделирования (например, Sparx Enterprise Architect, Magic Draw и др.)

Создание специальных приложений, позволяющих сделать администрирование учебных курсов простым и технологичным, стало органическим развитием идеи дистанционного обучения и произошло во второй половине двадцатого века в академическом секторе.

Электронное обучение вышло за пределы университетов и стало неотъемлемой частью образования и повышения квалификации во всех сферах человеческой деятельности с появлением и развитием Интернета в 1990-е годы. В 2004 году появился стандарт SCORM, который позволяет использовать совместно материалы различных курсов.

1.3.1 LMS Moodle

Moodle — это система управления курсами (электронное обучение), также известная как система управления обучением или виртуальная обучающая среда. Название системы является аббревиатурой от англ. Modular Object-Oriented Dynamic Learning Environment (модульная объектно-ориентированная динамическая обучающая среда). Система представляет собой свободное (распространяющееся по лицензии GNU GPL) веб-приложение, предоставляет возможность создавать сайты для онлайн-обучения. Первая версия была представлена 20 августа 2002 года.

СДО Moodle занимает примерно 18 % рынка в США.

Платформа предоставляет пространство для совместной работы учителей и студентов. В Moodle доступны различные возможности для отслеживания успеваемости учащихся, а также есть поддержка массовой регистрации с безопасной аутентификацией.

Система имеет гибкий интерфейс с возможностью конфигурирования макетов и дизайна отдельных страниц. Платформу можно интегрировать с большим количеством программного обеспечения, включая инструменты для общения, совместной работы, управления документами и другие приложения для повышения производительности. Moodle имеет открытый исходный код; большая часть системы написана на языке программирования PHP.

1.3.2 CMS-базированные решения

Система управления содержимым (англ. Content management system, CMS, система управления контентом) — информационная система или компьютерная программа, используемая для обеспечения и организации совместного процесса создания, редактирования и управления содержимым, иначе — контентом (от англ. Content).

CMS обычно состоит из двух основных компонентов: приложения для управления контентом (CMA) в качестве внешнего пользовательского интерфейса, позволяющего пользователю добавлять, изменять и удалять контент с веб-сайта без вмешательства вебмастера, и приложение доставки контента (CDA), которое компилирует контент и обновляет веб-сайт.

Основные функции CMS:

- предоставление инструментов для создания содержимого, организация совместной работы над содержимым;
- управление содержимым: хранение, контроль версий, соблюдение режима доступа, управление потоком документов;
- публикация содержимого;

- представление информации в виде, удобном для навигации, поиска.

В системе управления содержимым могут находиться самые различные данные: документы, фильмы, фотографии, номера телефонов, научные данные и так далее. Такая система часто используется для хранения, управления, пересмотра и публикации документации. Контроль версий является одной из важных возможностей, когда содержимое изменяется группой лиц.

В общем случае системы управления содержимым делятся на системы управления корпоративным контентом (англ. Enterprise Content Management System) — для работы с содержимым внутри какой-либо организации и системы управления веб-содержимым (англ. Web Content Management System) для поддержки работы веб-сайта.

Возможность разместить в сети Интернет и, при необходимости, редактировать документы, мультимедиа и клиентские скрипты JavaScript позволяет утверждать, что системы управления веб-содержимым могут быть использованы в качестве систем управления обучением.

По данным лаборатории W3Techs, в 2015 году 25 % всех сайтов работали под управлением системы управления содержимым WordPress, в августе 2018 — более 30 %, а в марте 2020 — уже более 41 % (доля рынка систем управления контентом составляет 63 %).

Функционал WordPress можно расширять при помощи платных и бесплатных плагинов. При этом среди бесплатных плагинов существует множество решений, позволяющих превратить WordPress в систему управления обучением [18].

При проведении исследования было рассмотрено пять наиболее популярных (фильтрация по убыванию популярности в магазине приложений) бесплатных плагинов, добавляющих в WordPress функционал системы управления обучением: LifterLMS, Sensei LMS, Tutor LMS, LearnPress, MasterStudy. Все рассмотренные плагины предоставляют возможность создавать курсы, главы (уроки) и викторины [1].

1.3.3 Прочие инструменты

Отличительной чертой рассмотренных ранее инструментов является то, что они предлагают пользователю «некую свободу действий». Например, WordPress имеет открытый исходный код, который можно модифицировать и расширять любыми способами, которые допускает лицензия. Важно отметить, что среди рассмотренных плагинов расширять и модифицировать возможно только код плагина «Sensei LMS»; все остальные плагины, даже имея открытый исходный код, в соответствии с лицензией, не могут быть модифицированы. Система Moodle также имеет открытый исходный код, который при необходимости можно изменять. Отдельно стоит отметить, что рассмотренные системы — крайне популярны; популярность, в свою очередь, гарантирует большое количество уже готовых расширений и поддержку со стороны сообщества и разработчиков.

Помимо инструментов с открытым исходным кодом существует множество других проприетарных решений, например, iSpring Suite – расширение для PowerPoint. iSpring Suite позволяет создавать электронные тесты, диалоговые тренажёры, озвучивать слайды, записывать скринкасты, монтировать видео или создавать обучающие игры [8]. NeoBook Professional – система для разработки электронных публикаций и презентаций. Everest — система, разработанная специально для образовательных приложений, в том числе и для дистанционного обучения и т.д. и т.д.

1.4 Этапы разработки ЭОР

Процесс разработки ЭОР состоит из двух основных этапов: подготовительного и компоновки.

На первом этапе (подготовительном) производится:

- подбор источников и формирование основного содержания;
- структуризация материала и разработка оглавления или сценария;
- переработка текста и формирование основных разделов;

- выбор, создание и обработка материала для мультимедийного воплощения (видеосюжеты, звуковое сопровождение, графические изображения).

На втором этапе производится компоновка (сборка в единое целое) всех отобранных и разработанных частей ЭОР (информационных, обучающих, контролирующих) для предъявления обучающимся в соответствии с задуманным автором сценарием.

1.5 Технологические особенности разработки ЭОР

Содержание ЭОР должно соответствовать уровню получаемого образования. В настоящее время разработка ЭОР должна быть ориентирована на получение заданных программой дисциплины компетенций. На подготовительном этапе ведётся подбор или разработка исходных материалов для ЭОР (текстов, графических иллюстраций, анимационных, аудио и видеофрагментов и т. д.), включая разработку или приобретение, при необходимости, пакетов учебных прикладных программ. На этом этапе обычно используют программные средства общего назначения: текстовые и графические редакторы, аниматоры, программы оцифровки аудио/видео, инструментальные среды программирования и т.п.

В структуре ЭОР принято выделять введение и основную часть, которая состоит из разделов, глав, тем. Введение является важным элементом ЭОР, поскольку в нем обосновывается актуальность данного ЭОР и определяются уровень образования и аудитория, на которые рассчитан данный ресурс. При формировании содержания рекомендуется разделить его на две части: основную часть, обязательную для изучения, и дополнительную – вариативную, для углублённого изучения материала, расширения кругозора, повышения мотивации.

При разработке структуры и содержания ЭОР необходимо учитывать следующие принципы и технологические особенности [17]:

1. Принцип приоритетности педагогического подхода. Принцип реализуется через постановку образовательной цели и разработку

содержания образовательной деятельности на основе одного или комбинации из нескольких дидактических подходов: системного, синергетического, проблемного, алгоритмического, программированного, проектного, эвристического и т.д. Системный подход означает, что целесообразно разрабатывать комплексные пособия, включающие как лекционный материал, семинарские занятия, так и комбинированные уроки (например, практики для гуманитарных и общеспециальных дисциплин).

2. Принцип модуля. Материал должен быть разбит на разделы, состоящие из модулей. Каждый модуль должен быть минимален по объёму, но замкнут по содержанию.
3. Принцип полноты. Каждый модуль должен иметь следующие компоненты: теоретическое ядро, контрольные вопросы по теории и примеры. Иногда полезно давать исторический комментарий или хронологическую картину развития конкретного направления.
4. Принцип наглядности. Иллюстративного материала должно быть как можно больше и он должен быть представлен в каждом модуле. При отборе и подготовке иллюстраций следует выбирать такие, которые выполняют не рекламную или развлекательную роль, а обучающую функцию.

Следует стремиться к максимальному использованию иллюстраций в местах, трудных для понимания учебного материала; для обобщений и систематизации тематических смысловых блоков; для общего оживления всего учебного материала и рассредоточенного по всему полю текста как печатного, так и электронного (гипертекста).

На втором этапе компоновку электронных материалов в ЭОР можно осуществить путём прямого программирования сценария обучения на каком-либо алгоритмическом языке: Java, PHP, Python и т.п. В этом случае роль навигатора в процессе обучения выполняет сценарий, в то время как при использовании только

HTML эту роль, как и в традиционных учебниках, выполняет оглавление. Применение языков программирования позволяет реализовать практически любые дидактические методики автора и разработчиков. Однако этому подходу присущи и существенные недостатки, такие как:

- высокая трудоёмкость процесса разработки ЭОР;
- необходимость привлечения профессиональных программистов;
- невозможность внесения изменений без привлечения программистов;
- существенная зависимость дидактического качества сценария обучения от педагогической квалификации разработчиков.

2 РАЗРАБОТКА ЭОР

2.1 Адаптация требований к ЭОР

Методология Agile чётко определяет принципы гибкой разработки программного обеспечения, но не предлагает никаких конкретных инженерных и практических методов. Культура DevOps является сугубо практическим ответвлением методологии Agile и, таким образом, предлагает недостающие решения и методы; в свою очередь, непрерывная интеграция и непрерывное развёртывание являются проявлениями DevOps [6]. Отсюда можно сделать вывод, что материалы, используемые для изучения методологии CI/CD должны иметь практическую направленность.

На сегодняшний день существует большое количество инструментов и сервисов для реализации CI/CD. В каждой программной среде выработался свой набор инструментов [2]. При этом инструменты одной среды идейно схожи с инструментами другой среды. Например, для написания автоматизированных тестов в Java-среде используется библиотека «JUnit». Эта библиотека была первой в своём роде и получила настолько широкое распространение, что разработчики других сред начали копировать её интерфейсы. Так PHP-разработчики скопировали успешные интерфейсы JUnit и назвали получившийся продукт «PHPUnit». Похожим образом поступили разработчики Python, разработав модуль «unittest», который нарушает стандарты PEP 8, копируя оригинальные интерфейсы из JUnit [16]. Другой пример – статические анализаторы кода: «PHPStan» – для PHP, «Pylint» – для Python, а в случае Java – встроенный компилятор. Поэтому разрабатываемый образовательный ресурс должен быть нацелен на то, чтобы донести до учащихся факт существования схожих классов инструментов и общие принципы работы с ними, но не рассматривать каждый конкретный инструмент в деталях.

Другая важная проблема заключается в том, что рассмотрение инструментов без повторений и реального примера ведёт к быстрому забыванию материала. Отсюда следует, что образовательный ресурс должен предлагать учащимся, как минимум, третий уровень интерактивности.

Одно из возможных решений заключается в следующем: учащиеся, руководствуясь предоставленными теоретическими материалами, дополняют рассмотренные примеры, и таким образом реализуют полноценный CI/CD-пайплайн в контексте современного веб-проекта. При этом результат их деятельности, во-первых, становится виден сразу, что повышает мотивацию, во-вторых, сохраняется в справочной форме. Например, при изучении возможностей GitHub Actions по параллелизации задач, неоднократное повторение и демонстрация возможностей сервиса отложит в головах учащихся факт существования и сферы применения параллелизации в сервисе GitHub Actions; если учащиеся забудут конкретные параметры, они всегда смогут вернуться к файлу с конфигурацией и воспользоваться им как справочником.

Разработка типового веб-проекта не входит в материалы курса и является отдельной обширной темой, рассматриваемой сразу в нескольких дисциплинах. Следовательно, учащиеся должны иметь возможность быстро, а главное, не вдаваясь в подробности, запустить некий веб-проект, к которому и будет применена методология CI/CD. Учащиеся могут воспользоваться проектами, которые они разработали ранее; или – уже существующими демонстрационными приложениями, например, «Symfony Demo Application». Последний вариант является более предпочтительным, поскольку демонстрационные приложения обычно задействуют более широкий спектр технологий: они покрыты тестами, включают линтеры, сканеры зависимостей и т.д.

2.2 Проектирование ЭОР

Перед началом разработки электронного образовательного ресурса необходимо подготовить сценарий и подобрать инструменты [7].

2.2.1 Структура ЭОР

Разрабатываемый электронный образовательный ресурс представляет собой цикл статей, которые можно разбить на три основных раздела:

1. Введение.
2. Непрерывная интеграция (Continuous Integration).
3. Непрерывная доставка и развёртывание (Continuous Delivery/Deployment).

Во введении рассматривается роль методологии CI/CD в современных циклах разработки ПО, её связь с DevOps и Agile, преимущества и недостатки, общие понятия. Основная задача введения – показать значимость и, как следствие, вездесущность методологии CI/CD. Этот раздел содержит исключительно теоретические материалы: текст и изображения. Раздел оканчивается викториной. Викторина предоставляет обратную связь, незамедлительно обозначая неверные ответы и пояснения к ним.

Второй раздел рассматривает непрерывную интеграцию и имеет практическую направленность. Порядок рассмотрения инструментов и, как следствие, глав определяется типовым сценарием работы с настоящим проектом. По этой же причине реальная задача и её практическое решение предшествуют теории. Детальная информация о втором разделе представлена в таблице 1.

Таблица 1 – Главы второго раздела

№ п/п	Название главы	Рассматриваемые инструменты и понятия
1	SSH: Защищённое соединение	SSH, ssh-keygen, асинхронное шифрование, ssh-agent, конфигурирование SSH-ключей в GitHub и GitLab
2	Размещение проекта в GitHub	Проект «Symfony Demo Application», форк репозитория, основы работы с Git, настройки GitHub Actions

Продолжение таблицы 1

3	Директория .github: первый пайплайн	Механизм событий и подписчиков, пайплайн (workflow), задача (job), action-приложение, отчёт по пайплайну, отлавливание ошибок в пайплайнах
4	Кеш GitHub: ускорение пайплайнов	Менеджер зависимостей (на примере Composer), .lock-файлы, ограничения GitHub Actions, кеширование (приложение actions/cache), язык выражений, контексты, функции
5	PHPUnit: автоматизированное тестирование	TDD и Test First Design, PHPUnit, автоматизированные тесты, семантическое версионирование
6	Линтеры: как не забыть про точку с запятой	Линтер, DI-контейнер, параллелизация, создание action-приложений, теги Git
7	PHP CS Fixer: стандарты написания кода	Стандарты написания кода, стандартные рекомендации (на примере PSR), параметризация action-приложений, PHP CS Fixer
8	PHPStan: типобезопасность без компиляции	Статический анализатор кода (на примере PHPStan)
9	Symfony CLI: поиск уязвимостей в зависимостях	Автоматизированная проверка рабочего окружения, проверка зависимостей проекта на уязвимости, периодические события GitHub Actions

В первой главе учащиеся знакомятся с протоколом SSH на прикладном уровне. Учащиеся должны сгенерировать пару ключей, настроить SSH-агент и добавить сгенерированный публичный ключ в настройки GitHub.

Во второй главе учащиеся размещают копии проекта «Symfony Demo Application» в их GitHub-репозитории.

В третьей главе учащиеся создают примитивный пайплайн. Для этого они создают новую Git-ветку, в которой будет происходить дальнейшая работа, знакомятся с основными понятиями пайплайна и создают две версии пайплайна: один, завершающийся успешно, второй – с ошибкой. Глава завершается викториной, посвящённой основным понятиям GitHub Actions (workflow, event, job, action) и основам синтаксиса конфигурационных файлов.

В четвёртой главе учащиеся настраивают кеш зависимостей проекта, предварительно установив их. Учащиеся должны добавить в пайплайн два новых шага: установку зависимостей и их кеширование.

В пятой главе учащиеся знакомятся с автоматизированными тестами на примере фреймворка «PHPUnit». Учащиеся должны добавить в пайплайн новый шаг, отвечающий за тестирование.

В шестой главе учащиеся знакомятся с линтерами как классом утилит и основами разработки action-приложений. Учащиеся должны добавить 4 однотипных шага в пайплайн; после этого они должны вынести повторяющиеся элементы в отдельное приложение и разбить одну задачу на две меньших. Глава завершается викториной; вопросы викторины посвящены структуре файлов конфигурации пайплайнов и action-приложений (язык выражений, контексты, возможные опции и т.п.).

В седьмой главе учащиеся знакомятся с классом утилит, отвечающих за проверку кода на соответствие стандартам форматирования. Учащиеся должны выпустить новую параметризованную версию action-приложения, разработанного в предыдущей главе, и добавить новый шаг в пайплайн.

В восьмой главе учащиеся знакомятся со статическими анализаторами кода. Для этого учащиеся должны установить и настроить утилиту «PHPStan» и добавить новый шаг в пайплайн.

В последней главе второго раздела учащиеся знакомятся с утилитой `Symfony CLI`. Учащиеся должны выпустить новую версию приложения, разработанного в шестой главе, добавив в него проверку рабочего окружения и создать новый пайплайн, запускаемый периодически, главной задачей которого является поиск уязвимостей в зависимостях проекта.

В конце раздела располагается обобщающая викторина. Вопросы этой викторины призваны закрепить все навыки, полученные в ходе освоения материалов раздела, и имеют дистинктивный характер. Последнее качество вынуждает учащихся выявить и запомнить различия между рассмотренными инструментами.

Изменения в очерёдности глав – невозможны. Порядок определён сценарием и взаимосвязью между рассматриваемыми технологиями. Например, SSH рассматривается и настраивается в самом начале, потому что учащиеся в следующих главах должны взаимодействовать с репозиториями GitHub по этому протоколу [11].

Третий раздел рассматривает непрерывную доставку и непрерывное развёртывание и также имеет практическую направленность. Со списком рассматриваемых понятий и инструментов можно ознакомиться в таблице 2.

Таблица 2 – Главы третьего раздела

№ п/п	Название главы	Рассматриваемые инструменты и понятия
1	Подготовка сервера	Хостинг, ACL, Nginx, CGI (на примере PHP-FPM)
2	Rsync: ручная доставка	Rsync, GitHub Secrets, sshpass
3	Deployer: автоматическая доставка и развёртывание	.env-файлы, Deployer
4	Теги и релизы: порядок в пайплайне	GitHub flow, Personal Access Token, релизы GitHub

В первой главе учащиеся подготавливают сервер, на котором будет развёрнут проект. Для этого они должны подготовить хостинг, установить требуемые программы и утилиты и настроить веб-сервер Nginx (для статических файлов и CGI-скриптов) [9].

Во второй главе учащиеся загружают проект на сервер при помощи утилиты Rsync. Для этого они должны настроить секреты GitHub и добавить новую задачу, состоящую из нескольких шагов и выгружающую файлы проекта по rsync.

В третьей главе учащиеся загружают файлы на сервер в автоматизированном режиме при помощи утилиты Deployer. Учащиеся должны поменять рабочее окружение приложения, настроить утилиту Deployer и обновить задачу, добавленную в предыдущей главе. После этого учащиеся вносят изменения в файлы проекта и наблюдают автоматизированный процесс доставки и развёртывания.

В последней главе третьего раздела учащиеся автоматизируют и совершенствуют выпуск релизов приложения. Для этого они создают новый Personal Access Token, используемый при API-вызовах к GitHub, выносят процессы доставки и развёртывания в отдельный пайплайн.

В третьем разделе порядок глав также не может быть изменён. Раздел завершается викториной, вопросы которой посвящены исключительно инструментам и понятиям, связанным с непрерывной доставкой и развёртыванием.

После поэтапного выполнения практических заданий второго и третьего разделов у учащихся должна получиться типовая конфигурация GitHub Actions. Получившиеся пайплайны не являются «оторванными от реальности тестовыми полигонами» и могут быть использованы при разработке настоящего веб-проекта.

2.2.2 Выбор технологий

Поскольку разрабатываемый ресурс представляет собой цикл статей, применяемые технологии должны, в первую очередь, позволить набирать и редактировать текст и изображения.

Вывод содержимого статей на экран является второй важнейшей проблемой: оно содержит много примеров кода и изображений, которые должны быть корректно форматированы в любом окружении.

Также должна присутствовать возможность создавать и редактировать интерактивные викторины.

Среди ранее рассмотренных инструментов этим требованиям отвечают система управления курсами Moodle и некоторые CMS-базированные решения. Также можно разработать новое решение на базе HTML и JavaScript. Рациональность применения того или иного инструмента определяется существующими, уже задействованными при обучении, технологиями. Гораздо удобнее и быстрее будет ввести CMS-базированное решение, если система Moodle ещё не используется в организации. Так как ресурс разрабатывается не для конкретной учебной организации, а для пользования всеми желающими, будет задействовано CMS-базированное решение.

Среди всех систем управления содержимым предпочтение отдано WordPress по нескольким причинам. Главная из них заключается в том, что WordPress – это самая популярная CMS с открытым кодом, появившаяся сравнительно давно (в 2003 году). Популярность, открытый код и возраст продукта обеспечивают его надёжность и расширяемость [19].

При разработке электронного образовательного ресурса были изучены и опробованы возможности нескольких популярных бесплатных плагинов, добавляющих функционал системы управления обучением в WordPress. Функционал всех опробованных плагинов мало различается. Руководствуясь субъективным мнением, автор отдал предпочтение плагину «LifterLMS». Определяющими факторами для автора стали такие параметры, как возможности текстового редактора, совместимость с популярными темами и возможность импорта и экспорта содержимого курса.

Для функционирования WordPress необходим сервер, умеющий работать с CGI, например Apache или Nginx. Также для хранения данных проекта необходима база данных MySQL или совместимая с ней база данных MariaDB.

2.2.3 Структура программного проекта

Любой проект, построенный на базе WordPress, представляет собой набор PHP-скриптов. Поэтому установка WordPress сводится к распаковке архива с исходным кодом проекта и запуску сервера, корневая директория которого указывает на распакованный дистрибутив. При этом необходимо учесть права доступа: у сервера должен быть доступ на чтение и запись ко всему содержимому распакованного архива.

Наполнение ресурса и модификация его содержимого не требуют вмешательства в исходный код проекта: файлы и их содержимое регулируются системой WordPress. Отсюда следует, что структура программного проекта представляет собой распакованное содержимое архива, базу данных и CGI-сервер.

2.3 Создание ЭОР

Для реализации разработанного проекта электронного образовательного ресурса необходимо подготовить рабочее окружение и наполнить ресурс содержимым.

2.3.1 Подготовка рабочего окружения

WordPress известен своей «лёгкой установкой за 5 минут». Многие веб-хостинги предлагают готовые инструменты для автоматической установки WordPress, например, Fantastico. Поскольку разрабатываемый ресурс не полагается на сторонний хостинг и должен быть готов к развёртыванию в любом окружении, будет проведена ручная установка и настройка.

Перед установкой необходимо проверить сервер на соответствие минимальным системным требованиям.

Установка состоит из следующих этапов:

1. Скачать и распаковать дистрибутив WordPress.
2. Создать базу данных для WordPress.
3. Переименовать (переместить) файл «wp-config-sample.php» в файл «wp-config.php».
4. Открыв «wp-config.php» в текстовом редакторе, задать настройки подключения к базе данных.
5. Разместить файлы дистрибутива на сервере.
6. Перейти по пути wp-admin/install.php (например, <http://example.com/wp-admin/install.php>).
7. Следовать дальнейшим инструкциям.

После настройки дистрибутива необходимо установить плагин «LifterLMS». Для этого нужно воспользоваться магазином расширений: перейти по пути «wp-admin/plugin-install.php», найти плагин (с помощью меню поиска или вручную) и следовать дальнейшим инструкциям по установке.

По умолчанию блоки с примерами кода в WordPress имеют скудное форматирование: в блоках сохраняются отступы, и к ним применяется моноширинный шрифт; при этом отсутствует подсветка синтаксиса, условное форматирование, индикации используемого языка и т.п. Значительно улучшить представление кода можно при помощи плагинов. Поэтому был установлен плагин «Prismatic», отвечающий за условное форматирование. В качестве библиотеки форматирования используется «prism.js».

2.3.2 Наполнение ресурса

После установки плагина «LifterLMS» выводится форма, предлагающая создать новый или импортировать уже существующий курс. Создать курс можно в любое другое время, воспользовавшись панелью администрирования и разделом «Courses».

Каждый курс должен иметь один или несколько тарифных планов. Управление планам осуществляется в настройках курса, которые выводятся в самом низу экрана при редактировании (см. рисунок 1).

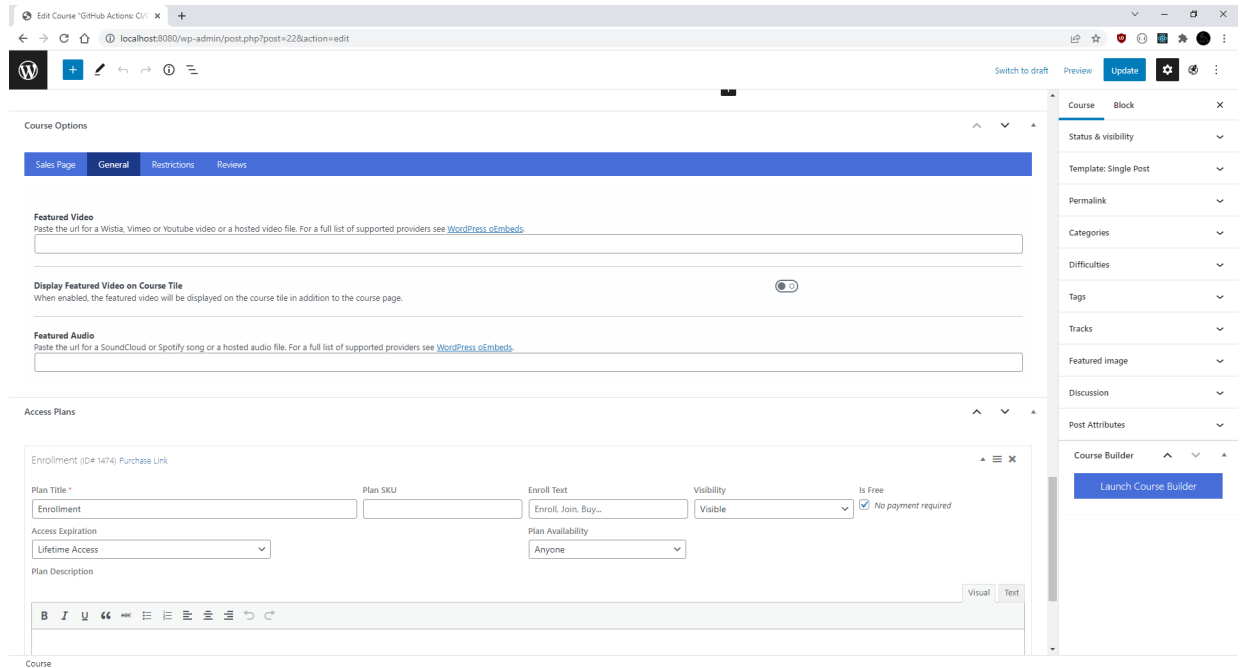


Рисунок 1 – Управление тарифными планами курса

Так как курс должен быть доступен всем, необходимо создать бесплатный план доступа, на который может записаться любой пользователь: выбрать значение «Anyone» в опции «Plan Availability» и выставить флаг «Is Free».

Создание разделов и глав осуществляется при помощи редактора курса (Builder). В соответствии с разработанным планом были созданы три раздела (см. рисунок 2).

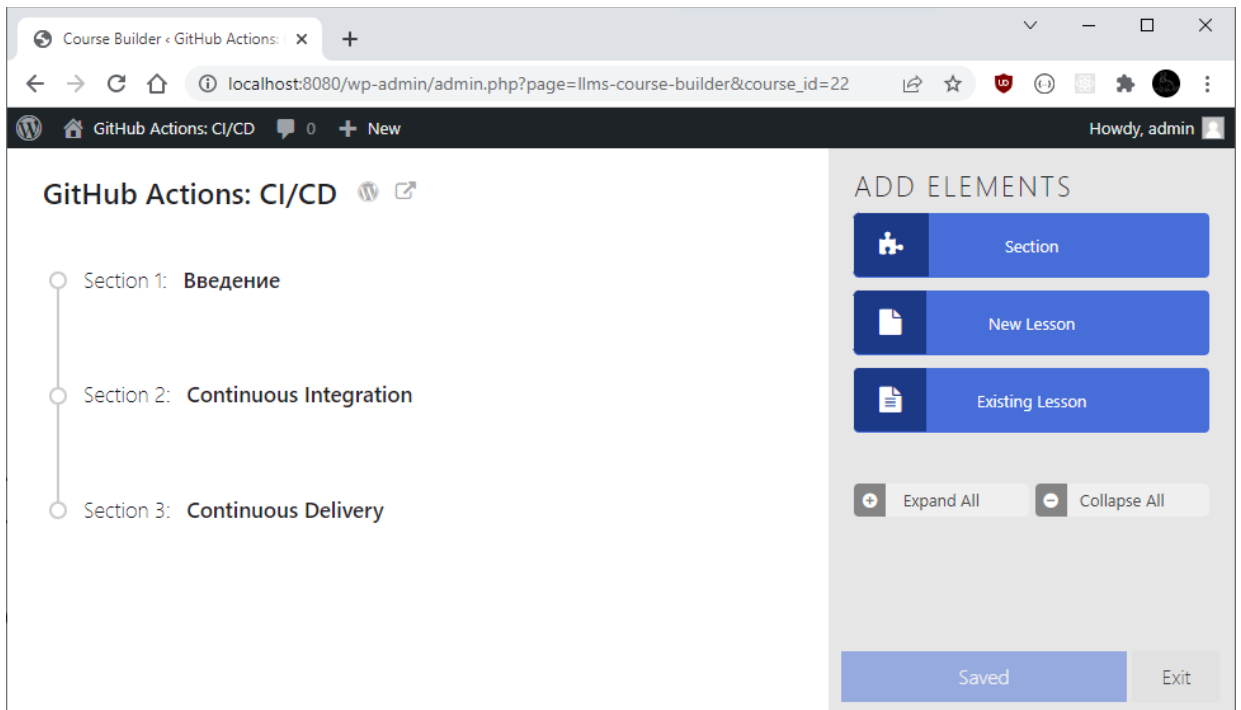


Рисунок 2 – Разделы курса

Также в соответствии с разработанным планом, в каждом разделе были созданы главы (см. рисунок 3).

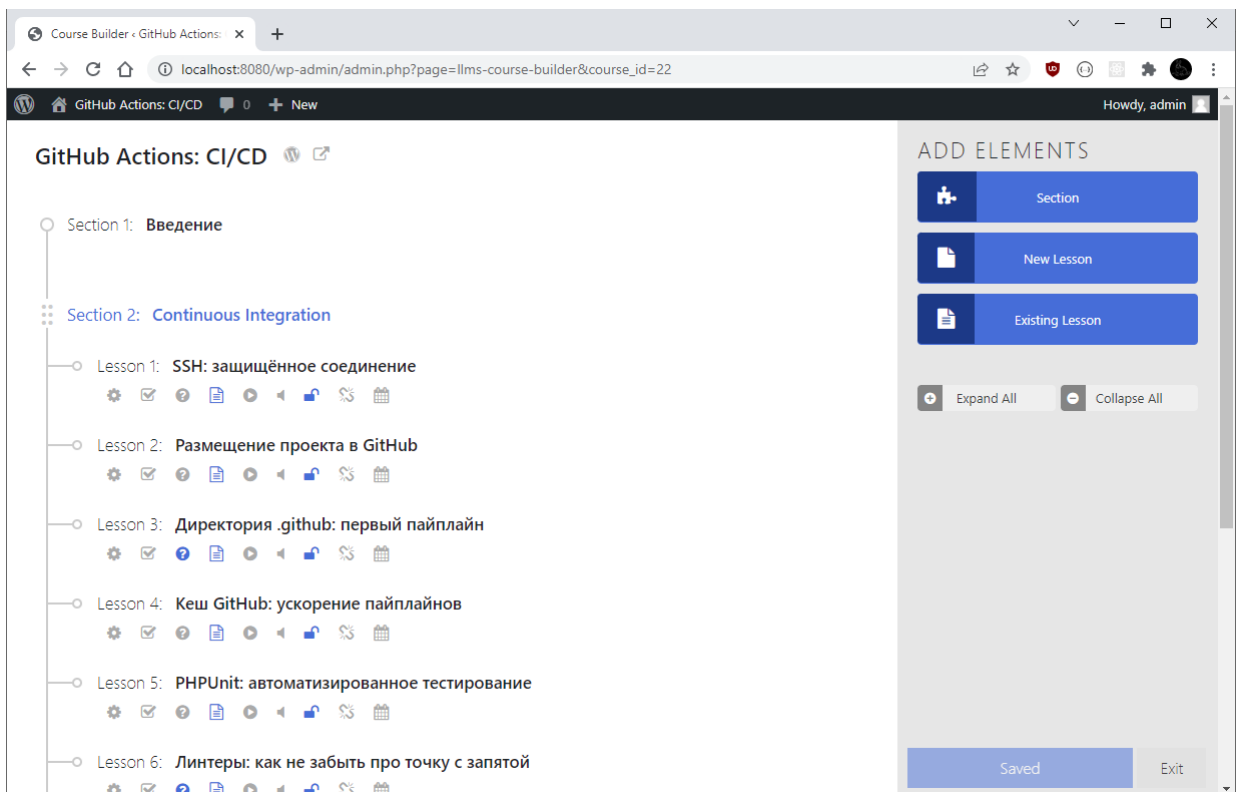


Рисунок 3 – Главы второго раздела

Каждую главу можно открыть в WYSIWYG-редакторе. Редактор позволяет удобно и быстро набирать текст, добавлять изображения, ссылки, видео и прочие материалы.

Во время написания статей автор руководствовался принципами, выработанными создателями серии книг и руководств Head First, а именно:

1. Разговорный стиль и персонификация. Исследования показывают [15], что студенты лучше усваивают материал и справляются с итоговыми тестами на 40% эффективнее, если учебные материалы, с которыми они работали, были написаны от первого лица с применением разговорного стиля вместо официального.

2. Попытка сделать так, чтобы читатель более глубоко вник в содержимое. Читатель должен быть мотивирован, заинтересован и вдохновлен, чтобы решать проблемы, делать выводы и усваивать новый материал. Для этого необходимо использовать упражнения и вопросы, вынуждающие студента задуматься и самостоятельно искать решение; при этом новые знания должны плавно вытекать из полученных ранее.

3. Завоевание и удержания внимания читателя. Изучение нового технического сложного материала проходит намного легче, если читатель заинтересован, и ему не скучно.

4. Эмоции. Способность к запоминанию чего-либо зависит от эмоциональной составляющей. Читатель запоминает то, что для него важно; он должен чувствовать удовлетворение от проделанной работы или решённой головоломки.

5. Закон Парето (принцип 80/20). Предлагаемый материал не должен быть единственным, он не должен покрывать все возможные темы и связанные проблемы.

В редакторе курса также можно добавлять и редактировать викторины (см. рисунок 4).

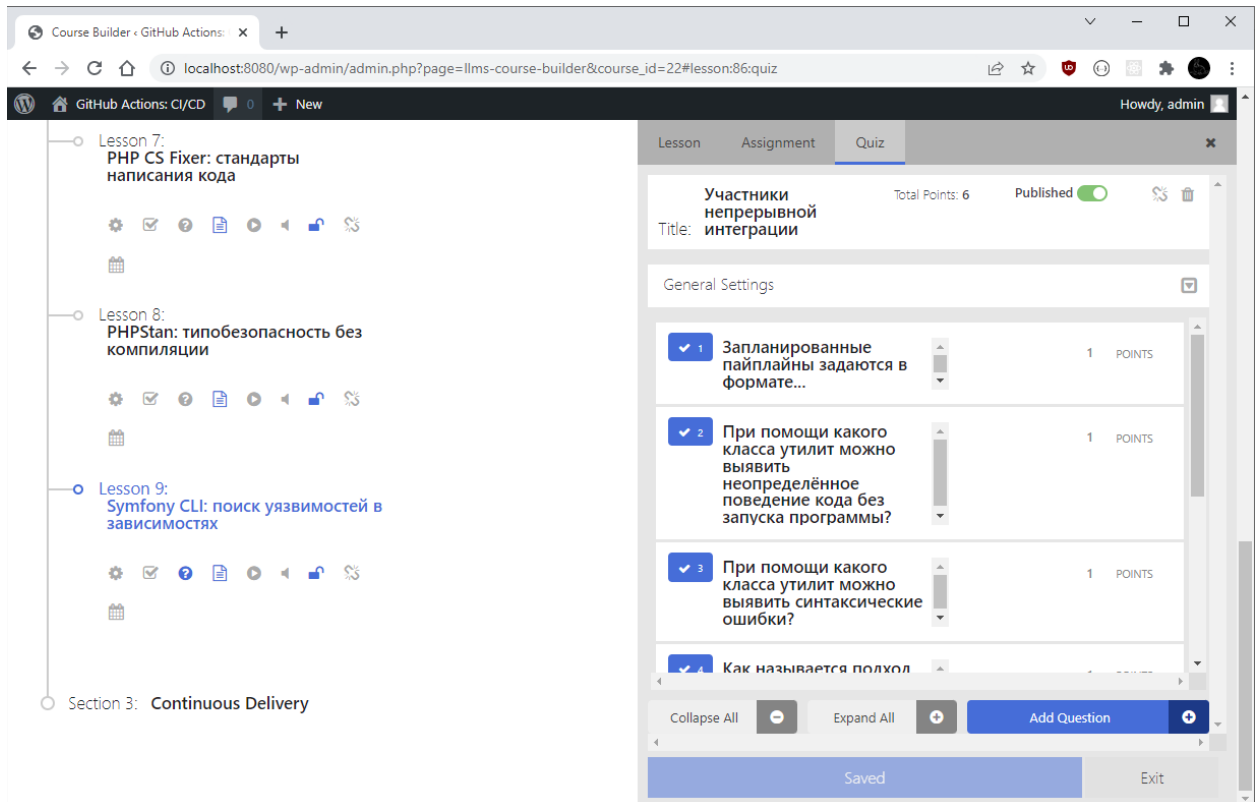


Рисунок 4 – Редактор викторин

В соответствии с разработанным планом были созданы викторины.

2.3.3 Введение стимулирующих элементов

Игровые технологии, разработанные для компьютерных игр, частично могут быть использованы в учебном процессе с целью повышения активности учащихся, побуждения их к достижению более высоких результатов обучения [20].

Применения игровых технологий в процессах электронного обучения должно носить постоянный динамический характер. Общей идеей здесь является конструирование процесса освоения содержания курса с включением неформальных игровых элементов последовательно возрастающей сложности с увеличивающейся ценностью виртуальных наград [12].

Плагин «LifterLMS» предлагает элементы геймификации. Без установки дополнительных расширений доступна система бейджей, прогресса и сертификатов (см. рисунок 5). При этом возможности геймификации и

стимулирования можно расширить при помощи других плагинов, например, плагин «myCred – LifterLMS Integration» добавляет бальную систему и расширяет систему прогресса, а плагин «LifterLMS integration» от GamiPress расширяет возможности викторин.

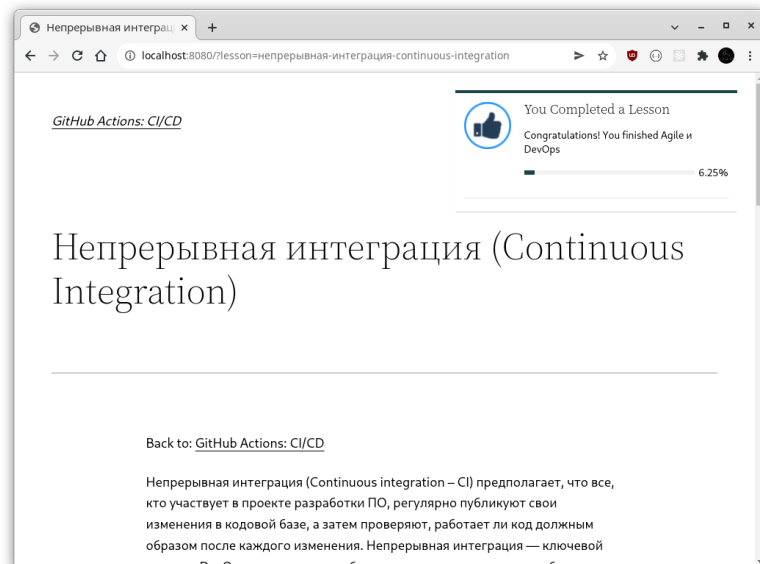


Рисунок 5 — Система отслеживания прогресса «LifterLMS»

2.4 Контейнеризация ресурса

Контейнеризация — это подход к разработке программного обеспечения, при котором приложение или служба, их зависимости и конфигурация (абстрактные файлы манифеста развёртывания) упаковываются вместе в образ контейнера. Контейнерное приложение может тестироваться как единое целое и развёртываться как экземпляр образа контейнера в операционной системе (ОС) узла. Контейнеризация программного обеспечения позволяет разработчикам и ИТ-специалистам развёртывать его в разных средах без каких-либо изменений или с минимальными изменениями. Каждый контейнер может вмещать целое веб-приложение или службу.

Контейнеры также изолируют приложения друг от друга в общей операционной системе. Контейнерные приложения выполняются на основе узла контейнеров, который в свою очередь работает в операционной системе (Linux

или Windows). Поэтому контейнеры требуют гораздо меньше ресурсов, чем образы виртуальных машин.

Иными словами, контейнеры предоставляют такие преимущества, как изоляция, переносимость, гибкость, масштабируемость и контроль, на протяжении всего жизненного цикла приложения. Самым важным преимуществом является изоляция среды разработки от рабочей среды.

Изначально стояла задача разработать электронный образовательный ресурс, который при необходимости можно копировать и перенести в другое рабочее окружение. При этом развёртывание должно происходить в полностью автоматическом режиме. Следовательно, необходимо контейнеризировать разрабатываемый ресурс.

Для функционирования WordPress-проекту необходим CGI-сервер и база данных, совместимая с MySQL (непосредственно MySQL или MariaDB).

Перенос кодовой базы проекта не вызывает трудностей: всё содержимое директории с проектом можно хранить в системе контроля версий; также WordPress-проект можно установить и настроить автоматически при помощи инструмента «wp-cli». Но электронный образовательный ресурс – это, в первую очередь, данные. Всё текстовое содержимое статей и некоторые настройки проекта хранятся в базе данных; поэтому содержимое базы данных также должно быть доступно для переноса. В этом случае механизм дампов является единственным возможным способом перенести всё содержимое из одной базы данных в другую. Поэтому дамп базы данных также должен храниться в системе контроля версий.

Таким образом для функционирования разрабатываемому ресурсу требуются два внешних инструмента:

1. CGI-сервер;
2. База данных, совместимая с MySQL.

При первом развёртывании также необходимо выполнить серию скриптов, загружающих дампы базы данных.

На сегодняшний день Docker является наиболее популярным инструментом для контейнеризации. Docker основан на системе виртуализации Linux (LXC) и, по сути, является более удобным интерфейсом к низкоуровневому сервису.

Всегда возможны два варианта контейнеризации:

1. Упаковка нескольких инструментов в один контейнер;
2. Оркестрация нескольких контейнеров, каждый из которых отвечает за одну составляющую системы.

Руководствуясь принципом единственной ответственности (каждая сущность должна иметь одну ответственность и эта ответственность должна быть полностью инкапсулирована в ней), можно утверждать, что предпочтение следует отдать второму варианту.

Конфигурация Docker-контейнеров разрабатываемого ресурса представлена в приложении №1. Переменные окружения, используемые в файле «docker-compose.yml» располагаются в файле «.env» (см. приложение №2).

В качестве CGI-сервера используется веб-сервер Apache (контейнер «WordPress»).

В качестве базы данных используется СУБД MySQL (контейнер «db»).

Дистрибутив WordPress, и в том числе, загруженные файлы, загружаются из директории «app». Дамп базы данных автоматически загружается при первом запуске контейнеров. Содержимое базы данных контролируется Docker и хранится в Docker-томе «db_data». Для развёртывания копии ресурса достаточно воспользоваться командой «docker-compose up».

ЗАКЛЮЧЕНИЕ

В данной работе было проведено исследование возможностей применения компьютерных и цифровых технологий в обучении, в частности, - возможность разработки электронного образовательного ресурса практической направленности, предназначенного для изучения практик CI/CD в среде GitHub.

Цель данной работы заключалась в разработке электронного образовательного ресурса «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта», готового к развёртыванию в виде Docker-приложения.

Для достижения поставленной цели было сделано следующее:

1. Изучена сущность электронного образовательного ресурса, классификация электронных образовательных ресурсов и особенности их разработки.
2. Проведён анализ существующих инструментов, предназначенных для разработки электронных образовательных ресурсов.
3. Адаптированы требования к электронному образовательному ресурсу с учётом особенностей изучения практик CI/CD.
4. Спроектирован электронный образовательный ресурс «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта».
5. В соответствии с планом создан электронный образовательный ресурс «Использование GitHub как системы для непрерывной интеграции и развёртывания современного веб-проекта» с применением технологий WordPress, MySQL, Docker (Compose), Git.

Таким образом, все поставленные задачи были реализованы.

Выдвинутая гипотеза (возможно разработать электронный образовательный ресурс практической направленности «Использование GitHub как системы для

непрерывной интеграции и развёртывания современного веб-проекта») подтвердилась.

Разработанный ресурс может быть использован учащимися для освоения практик CI/CD или применён при разработке других учебных материалов, следовательно, цель выпускной квалификационной работы была достигнута.

СПИСОК ЛИТЕРАТУРЫ

1. Азевич Алексей Иванович WordPress как обучающая интерактивная платформа // Вестник РУДН. Серия: Информатизация образования. 2013. №3. URL: <https://cyberleninka.ru/article/n/wordpress-kak-obuchayuschaya-interaktivnaya-platforma> (дата обращения: 16.03.2022).
2. Арланов И. В. Основные методы и компоненты систем непрерывной интеграции в ИТ-проектах // Новое слово в науке и практике: гипотезы и апробация результатов исследований. 2013. №8. URL: <https://cyberleninka.ru/article/n/osnovnye-metody-i-komponenty-sistem-nepreryvnoy-integratsii-v-it-proektah> (дата обращения: 16.03.2022).
3. Бодрова Екатерина Григорьевна, Дегтеренко Людмила Николаевна Цифровые инструменты и сервисы в профессиональной деятельности современного педагога // Современная высшая школа: инновационный аспект. 2021. №2 (52). URL: <https://cyberleninka.ru/article/n/tsifrovye-instrumenty-i-servisy-v-professionalnoy-deyatelnosti-sovremennogo-pedagoga> (дата обращения: 20.03.2022).
4. ГОСТ Р 52653-2006. Информационно-коммуникационные технологии в образовании. Термины и определения. – Введ. с 01.07.2008. – с. 1 — 4.
5. ГОСТ Р 53620-2009. Информационно-коммуникационные технологии в образовании. Электронные образовательные ресурсы. – Введ. с 01.01.2011. – с. 2 — 4.
6. Джун Ким, Джонг Хан Ким, Бер К., Спаффорд Д. Проект «Феникс». Роман о том, как DevOps меняет бизнес к лучшему. - М.: Бомбора, 2022. - 384 с.
7. Единые требования к электронным образовательным ресурсам. – М., 2011.
8. Козюкова Т.П., Кийкова Е.В. Выбор инструментария для разработки электронных образовательных ресурсов // Современные научные исследования и инновации. 2015. № 7. Ч. 2

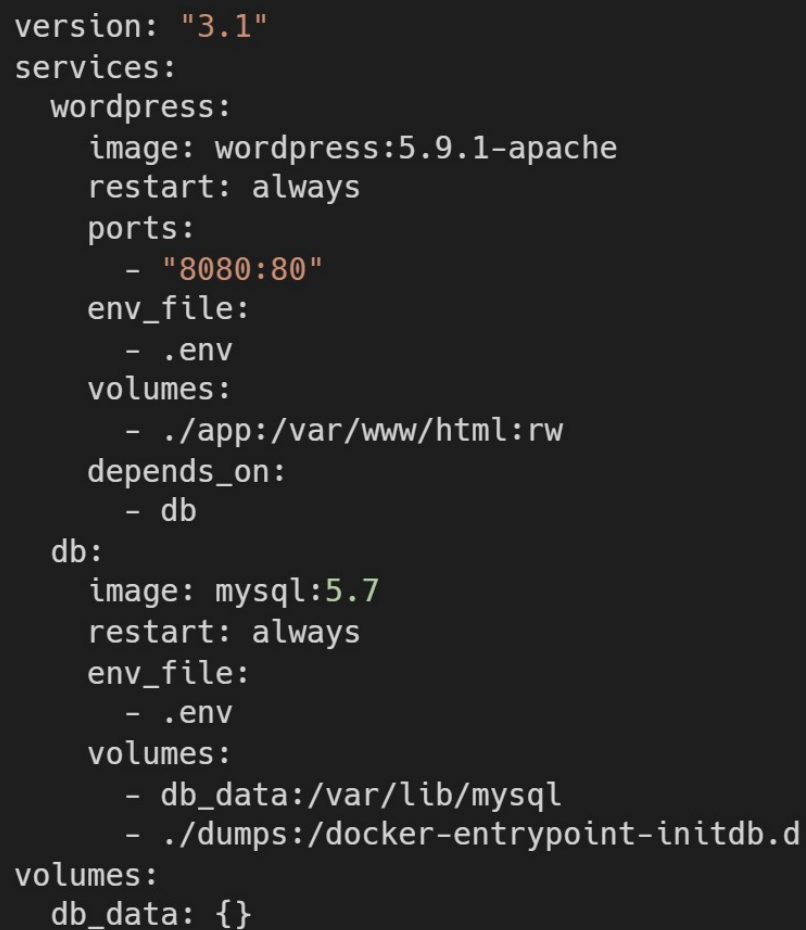
9. Котеров, Д. В. РНР 7 / Д. В. Котеров, И. В. Симдянов. — СПб.: БХВ-Петербург, 2016. — 1088 с.: ил. — (В подлиннике)
10. Норенков И. П. Электронные образовательные ресурсы // Машиностроение и компьютерные технологии. 2009. №12. URL: <https://cyberleninka.ru/article/n/elektronnye-obrazovatelnye-resursy> (дата обращения: 16.03.2022).
11. Ракчеев Алексей Юрьевич Работа протокола ssh на практике // Научный журнал. 2020. №3 (48). URL: <https://cyberleninka.ru/article/n/rabota-protokola-ssh-na-praktike> (дата обращения: 16.03.2022).
12. Стародубцев В. А. Элементы геймификации в LMS Moodle / В. А. Стародубцев, И. В. Ряшенцев // Международный научно-исследовательский журнал. — 2017. — № 07 (61) Часть 1. — С. 98—102.
13. Федеральная целевая программа «Научные и научно-педагогические кадры инновационной России» на 2009-2013 годы // Портал Министерства образования и науки РФ. - Режим доступа: <http://www.fasi.gov.ru/fcp/nпки/>
14. Федеральный государственный образовательный стандарт Высшего образования – Бакалавриат по направлению подготовки 09.03.01 Информатика и вычислительная техника от 19 сентября 2017 г. // Национальная ассоциация развития образования и науки. - Режим доступа: <https://fgos.ru/>
15. Фримен Э., Робсон Э., Сьерра К., Бейтс Б. Head First. Паттерны проектирования. Обновленное юбилейное издание. — СПб.: Питер, 2018. — 656 с.: ил. — (Серия «Head First O'Reilly»).
16. Хориков Владимир. Принципы юнит-тестирования. — СПб.: Питер, 2021 — 320 с.: ил. — (Серия «Для профессионалов»).
17. Электронные учебники: рекомендации по разработке. – М.: Федеральный институт развития образования, 2012. – 24 с.
18. Dias, S.B. and Diniz, J.A. and Hadjileontiadis, L.J. Towards an Intelligent Learning Management System Under Blended Learning: Trends, Profiles and Modeling Perspectives. — Springer International Publishing, 2013. — 235 p.

19. Kats, Y. Learning Management Systems and Instructional Design: Best Practices in Online Education. — IGI Global, 2013.
20. Werbach К. Геймификация, [Электронный ресурс] / Werbach К. // Coursera. – 2017. – URL: <https://ru.coursera.org/learn/gamification> (дата обращения: 25.08.2016).

ПРИЛОЖЕНИЯ

Приложение 1

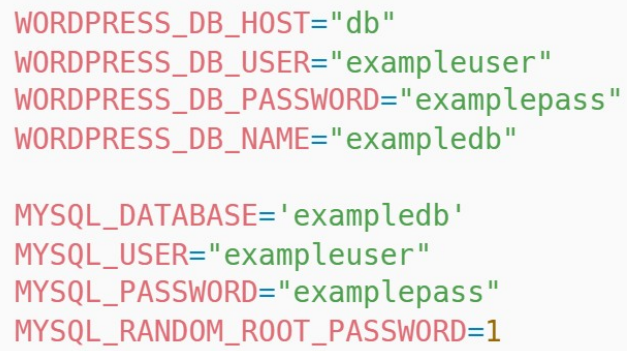
Файл конфигурации Docker Compose – docker-compose.yaml



```
version: "3.1"
services:
  wordpress:
    image: wordpress:5.9.1-apache
    restart: always
    ports:
      - "8080:80"
    env_file:
      - .env
    volumes:
      - ./app:/var/www/html:rw
    depends_on:
      - db
  db:
    image: mysql:5.7
    restart: always
    env_file:
      - .env
    volumes:
      - db_data:/var/lib/mysql
      - ./dumps:/docker-entrypoint-initdb.d
volumes:
  db_data: {}
```

Приложение 2

Файл конфигурации переменных окружения - .env



```
WORDPRESS_DB_HOST="db"
WORDPRESS_DB_USER="exampleuser"
WORDPRESS_DB_PASSWORD="examplepass"
WORDPRESS_DB_NAME="exampledb"

MYSQL_DATABASE='exampledb'
MYSQL_USER="exampleuser"
MYSQL_PASSWORD="examplepass"
MYSQL_RANDOM_ROOT_PASSWORD=1
```