СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc186025163)

[1 ОБЩАЯ ЧАСТЬ 5](#_Toc186025164)

[1.1 Цель разработки 5](#_Toc186025165)

[1.2 Анализ использования разработки 6](#_Toc186025166)

[1.3 Анализ методов решения 7](#_Toc186025167)

[1.4 Обзор средств программирования 8](#_Toc186025168)

[1.5 Описание языка программирования 10](#_Toc186025169)

# ВВЕДЕНИЕ

Цифровые технологии получили широкое распространение и значительно изменили структуру образовательного процесса. Современные студенты все чаще осваивают учебный материал вне аудиторий, взаимодействуя с преподавателями в онлайн-режиме, выполняя задания дистанционно и проходя тестирование через интернет. Увеличение популярности удалённого обучения обусловило рост потребности в удобных и функциональных образовательных платформах.

Для обучающихся, осваивающих новые профессии, важна возможность обучения в удобное время и с индивидуальным темпом. В этом случае актуальным становится создание онлайн-ресурсов, которые могут эффективно дополнять или заменять традиционные формы обучения.

Разрабатываемая платформа обеспечивает эффективное взаимодействие между студентами и преподавателями посредством интуитивно понятного интерфейса и необходимого функционала. Пользователи получают возможность изучать материалы и проходить тесты без необходимости личного присутствия, что способствует экономии времени преподавателей и повышает гибкость образовательного процесса.

Итоговая версия проекта представляет собой универсальное решение для дистанционного обучения, пригодное как для самостоятельного изучения и проверки знаний студентами, так и для автоматизации учебного процесса преподавателями – от загрузки материалов до анализа успеваемости.

## 1 ОБЩАЯ ЧАСТЬ

### 1.1 Цель разработки

Целью данной дипломной работы является разработка образовательного ресурса для изучения дисциплины «Архитектура компьютерных систем», обеспечивающего доступ к лекциям, практическим заданиям и автоматизированному тестированию.

В рамках работы поставлены следующие задачи: создать удобный пользовательский интерфейс для студентов и преподавателей; подготовить и интегрировать учебный контент, включающий лекции, практические материалы и тесты; реализовать систему автоматической проверки тестовых заданий с сохранением результатов; обеспечить функционал управления контентом для преподавателей.

### 1.2 Анализ использования разработки

Разрабатываемый образовательный ресурс предназначен для эффективной поддержки изучения дисциплины «Архитектура компьютерных систем». Платформа предоставляет доступ к структурированным учебным материалам, интерактивным заданиям и проверочным тестам, что способствует повышению качества образования.

Ресурс оснащён инструментами для преподавателей, обеспечивающими проверку результатов тестирования и добавление дополнительных материалов. Преподаватели могут управлять содержанием сайта, редактировать и обновлять информационные блоки, поддерживая актуальность и полноту учебного контента.

Программный продукт ориентирован на две основные категории пользователей:

Студенты: имеют возможность изучать лекции, выполнять практические задания и проходить тестирование; просматривать оценки и отслеживать прогресс; использовать платформу для самоподготовки и повторения материала.

Преподаватели: загружают и обновляют учебные материалы, создают и назначают задания; получают доступ к статистическим данным успеваемости студентов для анализа эффективности обучения.

Разработанная онлайн-платформа обеспечивает самостоятельное изучение дисциплины «Архитектура компьютерных систем» студентами, упрощает организацию учебного процесса для преподавателей и повышает оперативность управления контентом.

### 1.3 Анализ методов решения

Для создания образовательного ресурса, соответствующего поставленным целям, необходимо подобрать оптимальные технологии и инструменты. Выбор средств разработки определяется требованиями к функциональности, масштабом проекта и доступными ресурсами.

Рассматриваются следующие методы разработки:

1. Конструкторы сайтов.

Конструкторы сайтов (website builders) – это онлайн-платформы, позволяющие создавать веб-ресурсы через визуальные редакторы и готовые шаблоны без необходимости программирования. Используя интерфейс drag-and-drop, пользователи быстро и интуитивно размещают и настраивают элементы страницы (тексты, изображения, кнопки и формы).

Популярные примеры: Tilda, Wix, Squarespace.

Преимущества конструкторов включают простоту использования, отсутствие необходимости знания кода, быструю разработку благодаря готовым шаблонам, наличие встроенного хостинга, домена, SSL-сертификатов и базовых SEO-инструментов. Платформы регулярно обновляют программное обеспечение и обеспечивают базовую защиту сайта.

Однако конструкторы ограничены функционально: сложно реализовать уникальные решения, глубоко изменить код или архитектуру сайта невозможно. Часто используется избыточный код, а шаблонность может затруднять масштабирование. При росте бизнеса или трафика может потребоваться переход на более гибкие технологии, так как перенос сайта на другую платформу зачастую невозможен без полной переработки.

Конструкторы широко применяются для быстрого создания лендингов, портфолио, презентационных сайтов и небольших интернет-магазинов. Например, стартапы и малые предприятия используют Wix или Tilda для запуска первых сайтов без привлечения специалистов. Творческие профессионалы – дизайнеры, фотографы, копирайтеры – создают простые и привлекательные сайты для демонстрации работ. В качестве примера: небольшой магазин одежды может запустить сайт на Wix с каталогом товаров и встроенной системой оплаты, что позволяет начать онлайн-продажи с минимальными затратами.

2. Системы управления контентом (CMS).

Системы управления контентом (CMS) – это программные платформы для создания, редактирования и организации содержимого веб-сайтов, предоставляющие удобный интерфейс для работы с текстами, изображениями, видео и структурой сайта без глубоких технических знаний. В отличие от конструкторов, CMS обеспечивают более широкие возможности кастомизации и расширения функционала благодаря плагинам, модулям и темам оформления.

Примеры CMS: WordPress, Joomla, Drupal.

CMS широко применяются для создания сложных и масштабируемых проектов: корпоративных сайтов, новостных порталов, образовательных платформ и интернет-магазинов. Например, WordPress популярен для ведения блогов и новостных ресурсов благодаря простоте использования и большому количеству дополнений. Joomla отличается более сложной архитектурой и расширенными возможностями управления пользовательскими правами и структурой сайта. Drupal используется для крупных и нагруженных проектов, таких как государственные порталы и университетские сайты, благодаря высокой безопасности и гибкости управления данными.

Преимущества CMS включают модульность, масштабируемость и возможность интеграции с внешними сервисами. Однако для эффективного использования требуется базовое понимание веб-технологий и навыки настройки серверного окружения.

Пример использования: университет может внедрить Drupal для создания сайта с разделами для студентов, преподавателей и сотрудников, интегрируя системы управления учебными материалами и расписанием, что обеспечивает удобство и функциональность образовательного процесса.

3. Ручная разработка (Custom Development).

Ручная разработка представляет собой создание веб-ресурсов с нуля с использованием языков программирования и фреймворков, что обеспечивает максимальную гибкость и позволяет реализовать уникальный дизайн и функциональность, полностью соответствующие требованиям проекта. Основные технологии включают HTML, CSS, JavaScript для фронтенда, а также серверные языки и фреймворки, такие как Python с Django, PHP с Laravel, Node.js. Для создания динамичного интерфейса часто применяются библиотеки React и Vue.js.

Данный подход подходит для проектов с нестандартной логикой, высокой нагрузкой и необходимостью интеграции с внешними системами – сложных веб-приложений, корпоративных порталов, социальных сетей и крупных интернет-магазинов. Ручная разработка требует высокой квалификации специалистов, знаний архитектуры веб-приложений, безопасности и оптимизации производительности.

Преимущества ручной разработки включают максимальную гибкость в реализации функций и дизайна, полный контроль над архитектурой, безопасностью и масштабируемостью, возможность оптимизации скорости работы и SEO, а также легкость расширения функционала по мере роста проекта. Однако этот метод отличается высокой трудоёмкостью, требует опытных разработчиков и значительных затрат на запуск и поддержку. Ответственность за обновления и безопасность полностью лежит на команде разработчиков.

Пример использования: крупная компания может заказать разработку корпоративного портала с интеграцией внутренней системы, системы управления проектами и аналитики, реализованной с помощью React и Django. Такой сайт будет соответствовать бизнес-целям и обеспечит надежную работу при высокой нагрузке.

В результате анализа доступных методов был выбран подход ручной разработки, так как он позволяет создать уникальный функционал и дизайн, невозможные при использовании конструкторов или стандартных CMS. Этот подход даёт полную свободу действий, позволяя реализовывать сложные сценарии взаимодействия пользователей и интегрироваться с внешними сервисами. Ручная разработка обеспечивает высокую гибкость и масштабируемость, облегчая расширение функционала и модернизацию архитектуры. Она также позволяет существенно повысить производительность и безопасность благодаря точной настройке кода и внедрению специализированных защитных механизмов. Несмотря на большие начальные затраты времени и ресурсов, этот метод наиболее эффективен для крупных проектов, требующих устойчивости, гибкости и свободы от ограничений типовых платформ.

### 1.4 Обзор средств программирования

При разработке образовательного ресурса важно не только чётко определить, какие функции должна выполнять система, но и подобрать оптимальные технологии для её реализации. После того как я решила реализовывать образовательный ресурс вручную, следующим шагом стало определение подходящих технологий и инструментов для разработки. От правильного выбора языков программирования и фреймворков зависит не только эффективность и качество работы, но и дальнейшая поддержка, масштабируемость и безопасность системы. Я рассмотрела основные варианты, которые могут быть использованы для создания такой системы:

1. Языки и технологии для клиентской части (Frontend):

HTML(HyperText Markup Language).

HTML (язык гипертекстовой разметки) – основной язык разметки для создания структуры веб-страниц. Он определяет расположение и иерархию элементов: заголовков, абзацев, изображений, таблиц, форм, ссылок и других компонентов.

Преимущества: Необходим для любой веб-страницы. Простая структура, легко читаемая браузерами. Поддерживается всеми современными браузерами. Легко изучается и используется новичками.

Недостатки: Не добавляет интерактивности и динамики без дополнительных технологий (CSS, JavaScript). Ограничен в возможностях визуального оформления и поведения элементов.

HTML (язык гипертекстовой разметки) — язык гипертекстовой разметки

Что такое HTML?

HTML — это основной стандартный язык разметки, используемый для создания и структурирования веб-страниц и веб-приложений. Он определяет, как различные элементы (заголовки, абзацы, изображения, таблицы, формы, ссылки и т. д.) будут располагаться на странице и взаимодействовать друг с другом. HTML не является языком программирования, поскольку не содержит логики или алгоритмов, а служит для описания структуры и содержания документа.

Основные элементы и возможности HTML

Теги — основа HTML. Каждый элемент страницы определяется тегом, заключённым в угловые скобки, например:

<h1>, <p>, <img>, <a>, <table>, <form>, <div>, <span>, <ul>, <li> и др.

Атрибуты — дополнительные параметры, задающие свойства элементов (например, src для <img>, href для <a>, alt для изображений, id и class для идентификации и стилизации).

Структурирование — HTML позволяет логически разбивать страницу на секции, статьи, навигацию, шапку, подвал и т.д. (теги <header>, <footer>, <nav>, <section>, <article>).

Встраивание мультимедиа — поддержка изображений, аудио, видео, SVG-графики и других медиаформатов.

Гиперссылки — основа гипертекста, связывающая страницы между собой и с внешними ресурсами.

Формы — интерактивные элементы для сбора и отправки пользовательских данных (теги <form>, <input>, <textarea>, <select>, <button>).

Преимущества HTML

Необходим для любой веб-страницы: любой сайт, независимо от сложности, начинается с HTML-разметки.

Простая структура: код легко читается и понимается как людьми, так и браузерами.

Совместимость: поддерживается всеми современными браузерами и устройствами.

Гибкость: позволяет создавать как простые статические страницы, так и сложные структуры для динамических веб-приложений.

Простота изучения: порог вхождения очень низкий, освоить основы HTML можно за несколько часов.

Стандартизация: язык развивается и поддерживается международным консорциумом W3C, что обеспечивает единые стандарты для всех разработчиков.

Семантика: современный HTML (HTML5) поддерживает семантические теги, которые помогают поисковым системам и вспомогательным технологиям лучше понимать структуру и смысл контента.

Недостатки HTML

Нет интерактивности: HTML не позволяет реализовать сложные сценарии взаимодействия с пользователем, динамическое обновление данных, анимацию и т. д. Для этого необходимы CSS и JavaScript.

Ограниченные возможности визуального оформления: без CSS страница выглядит очень просто и не может быть адаптирована под разные устройства.

Нет программной логики: HTML не может выполнять вычисления, условия, циклы и другие логические элементы.

Статичность: сам по себе HTML создаёт только статические страницы, которые не реагируют на действия пользователя без дополнительных технологий.

Безопасность: HTML не содержит встроенных средств защиты от атак, таких как XSS, — этим занимаются серверные языки и JavaScript.

Примеры использования HTML

xml

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<title>Пример страницы</title>

</head>

<body>

<header>

<h1>Добро пожаловать на сайт!</h1>

<nav>

<a href="#about">О нас</a>

<a href="#contact">Контакты</a>

</nav>

</header>

<section id="about">

<h2>О компании</h2>

<p>Мы занимаемся разработкой образовательных ресурсов.</p>

<img src="logo.png" alt="Логотип компании">

</section>

<footer>

<p>&copy; 2025 Все права защищены.</p>

</footer>

</body>

</html>

Современные возможности HTML5

Встроенные аудио- и видеофайлы: <audio>, <video>

Поддержка Canvas и SVG: для создания графики и анимации

Веб-хранилище: локальное и сессионное хранилище данных в браузере

Геолокация: определение местоположения пользователя

API перетаскивания: поддержка перетаскивания элементов

Формы с проверкой: новые типы полей, встроенная проверка данных

CSS (Cascading Style Sheets).

CSS (каскадные таблицы стилей) используется для стилизации и оформления HTML-страниц. С помощью CSS задаются цвета, шрифты, отступы, выравнивание, анимация и адаптивность для разных устройств.

Преимущества: Гибкое управление внешним видом сайта. Поддержка медиазапросов для адаптивного дизайна. Позволяет создавать сложные макеты (Flexbox, Grid). Универсальная поддержка браузерами.

Недостатки: CSS зависит от HTML — без разметки не работает. Трудности с управлением большими стилями без препроцессоров. Нет встроенных инструментов для отладки ошибок.

JavaScript

JavaScript — основной язык для добавления интерактивности на веб-страницы.

Преимущества: Позволяет изменять структуру и содержимое страницы в реальном времени (манипуляции с DOM). Обрабатывает пользовательские события (клики, ввод, прокрутку). Позволяет работать с сервером асинхронно (AJAX, Fetch API). Единственный язык, выполняемый непосредственно в браузере.

Недостатки: Может замедлять работу страницы при плохой оптимизации. Требует тестирования в разных браузерах из-за различий в реализации.

React:

React — популярная JavaScript-библиотека для создания пользовательских интерфейсов, созданная Facebook.

Преимущества: Компонентная архитектура: интерфейс разбивается на независимые элементы, что облегчает поддержку и повторное использование кода. Виртуальный DOM: ускоряет обновление интерфейса и повышает производительность. Поддержка одностороннего потока данных и реактивности. Отлично подходит для сложных динамических приложений.

Недостатки: Требуется знание JSX и глубокое понимание JavaScript. Для крупных проектов требуются дополнительные библиотеки (роутинг, управление состоянием).

Vue.js:

Vue.js — легковесный и гибкий JavaScript-фреймворк для создания интерфейсов.

Преимущества: Простой синтаксис, понятная документация, низкий порог вхождения. Легко внедряется в существующие проекты. Реактивные данные и виртуальный DOM для высокой производительности. Подходит для малых и средних проектов, а также для быстрого создания прототипов.

Недостатки: Меньше готовых решений и инструментов для крупных проектов по сравнению с React. Меньшее сообщество, чем у React.

2. Языки и фреймворки для серверной части (Backend)

Python:

Python — современный язык программирования для серверной логики, работы с базами данных, API, обработки файлов и бизнес-логики.

Преимущества: Простой и понятный синтаксис, быстрое освоение.

Большое количество библиотек и фреймворков.

Отлично подходит для работы с данными, машинного обучения и автоматизации.

Кроссплатформенность.

Недостатки:

Более низкая производительность по сравнению с компилируемыми языками.

Не лучший выбор для ресурсоёмких задач.

Django:

Django — мощный Python-фреймворк для быстрой разработки веб-приложений.

Преимущества:

Встроенные инструменты: ORM, аутентификация, админ-панель, работа с формами.

Высокая безопасность, регулярные обновления.

Подходит для крупных проектов с большими объёмами данных.

Быстрая разработка и масштабируемость.

Недостатки:

«Тяжёлый» фреймворк — не всегда рационален для простых сайтов.

Требует глубокого понимания архитектуры.

PHP:

PHP — один из самых популярных языков для серверной разработки, особенно для динамических сайтов и работы с базами данных.

Преимущества:

Простота освоения, большая экосистема, множество готовых решений.

Поддерживается большинством хостингов.

Хорошая интеграция с СУБД (MySQL, PostgreSQL и др.).

Недостатки:

Исторически сложившаяся нестрогая типизация, разная поддержка стандартов в разных версиях.

Требует продуманной архитектуры для крупных проектов.

Laravel:

Laravel — современный PHP-фреймворк для быстрой и качественной разработки веб-приложений.

Преимущества:

Удобная маршрутизация, встроенная поддержка авторизации, очередей, рассылок.

Чистый и понятный синтаксис, хорошая документация.

Подходит для разработки сложных и масштабируемых проектов.

Недостатки:

Медленнее, чем Node.js, при высокой нагрузке.

Требует технических навыков для тонкой настройки.

Node.js:

Node.js — среда выполнения JavaScript на сервере, позволяющая создавать быстрые и масштабируемые веб-приложения.

Преимущества:

Асинхронная, событийно-ориентированная архитектура — высокая производительность при большом количестве одновременных запросов.

Возможность использовать JavaScript на клиенте и сервере (fullstack).

Быстрая обработка запросов, поддержка большого количества подключений.

Недостатки:

Не лучший выбор для задач, требующих интенсивных вычислений на процессоре.

Требует аккуратной организации кода и знаний в области асинхронного программирования.

После анализа различных языков программирования и фреймворков для разработки образовательного ресурса был выбран стек технологий, включающий HTML, CSS, JavaScript для клиентской части и PHP для серверной стороны. Такой выбор обусловлен несколькими ключевыми факторами: HTML и CSS являются базовыми инструментами создания веб-страниц и отвечают за структуру и внешний вид интерфейса. Использование современных возможностей CSS (например, Flexbox) позволяет создать адаптивный и удобный дизайн, который корректно отображается на разных устройствах. JavaScript добавляет интерактивность к статичным страницам: обработка форм, динамическое обновление контента без перезагрузки страницы, проверка тестов в реальном времени. Он обеспечивает необходимый уровень взаимодействия между пользователем и системой, делая обучение более эффективным и увлекательным. PHP выбран в качестве серверного языка благодаря его широкому распространению, простоте внедрения и хорошей совместимости с базами данных. Он отлично подходит для реализации таких функций, как авторизация пользователей, работа с тестами, хранение результатов и управление контентом со стороны преподавателей. PHP не требует сложной настройки сервера и легко развертывается на большинстве хостингов, что делает его практичным решением для проектов средней сложности.

Выбранный стек технологий полностью соответствует поставленным задачам и позволяет реализовать все ключевые функции образовательного ресурса: от предоставления учебного материала до автоматизации проверки знаний и управления пользователями.

### 1.5 Описание языка программирования

HTML (HyperText Markup Language) – это стандартный язык разметки, используемый для создания структуры веб-страниц. Основной функцией HTML является определение структуры и содержания страницы: заголовков, параграфов, изображений, таблиц, списков и других элементов.

Основные функции:

* Создание структуры страницы: HTML позволяет задавать такие элементы, как текст, изображения, видео, формы и другие компоненты, которые составляют содержимое веб-страницы;
* Теги и атрибуты, Гиперссылки;
* Встраивание мультимедиа: HTML поддерживает встраивание аудио, видео, а также поддержку различных форматов мультимедийных данных;

Формы для ввода данных: HTML предоставляет теги для создания форм, которые позволяют пользователям вводить данные, такие как текст, числа, даты и т.д.

CSS (Cascading Style Sheets) используется для описания внешнего вида HTML-элементов. С помощью CSS можно задавать стили, такие как шрифты, цвета, отступы, выравнивание и анимацию.

Основные функции:

* Стилизация элементов: CSS позволяет изменять внешний вид элементов на странице, например, задавать цвет текста, фона, размер шрифта, рамки и другие параметры.
* Адаптивный дизайн: CSS предоставляет средства для создания адаптивного дизайна, который изменяет внешний вид страницы в зависимости от размера экрана устройства (мобильных телефонов, планшетов, компьютеров).
* Макетирование: CSS предлагает различные способы расположения элементов на странице, включая flexbox и grid-системы, которые позволяют создавать сложные макеты.
* Анимации и переходы: CSS поддерживает создание плавных анимаций и переходов, что помогает улучшить пользовательский опыт и визуальную привлекательность.
* Использование фреймворков: CSS-фреймворки, такие как Bootstrap и Tailwind CSS, помогают ускорить разработку и обеспечивают готовые компоненты для UI (покрытие всех аспектов дизайна, включая кнопки, формы, модальные окна и т.д.).

JavaScript (с фреймворком React) – это язык программирования, который используется для создания интерактивности на веб-страницах. JavaScript позволяет манипулировать элементами HTML-документа и взаимодействовать с сервером, что делает веб-приложение более динамичным.

Основные функции:

* Манипуляция DOM (Document Object Model): JavaScript позволяет изменять структуру и содержание веб-страницы в реальном времени. Это позволяет обновлять данные, элементы интерфейса и выполнять различные действия без перезагрузки страницы.
* Обработка событий: JavaScript позволяет реагировать на действия пользователя (например, клики, ввод данных, прокрутка и т.д.).
* Работа с асинхронными запросами: JavaScript использует технологии, такие как AJAX или Fetch API, для асинхронных запросов к серверу, что позволяет обновлять данные на странице без её перезагрузки.
* Фреймворк React:
  + Компонентная архитектура: React помогает строить приложения, разделяя интерфейс на повторно используемые компоненты. Это упрощает управление состоянием и обновления интерфейса.
  + Виртуальный DOM: React использует виртуальный DOM для оптимизации изменений в реальном DOM, что повышает производительность при рендеринге больших и сложных интерфейсов.
  + Интерактивность: React идеально подходит для создания динамичных и интерактивных интерфейсов с минимальными усилиями, что важно для образовательных ресурсов с тестами и заданиями.

PHP с фреймворком Laravel – это серверный язык программирования, который широко используется для создания динамичных веб-сайтов. Он взаимодействует с сервером и базой данных для обработки запросов, отправки данных и генерации веб-страниц.

Основные функции PHP:

* Работа с базами данных: PHP позволяет взаимодействовать с различными СУБД, включая MySQL, PostgreSQL и SQLite. Это важно для образовательного ресурса, так как он требует хранения и обработки большого объема данных, таких как результаты тестов.
* Обработка форм: PHP используется для обработки данных, отправляемых через формы, например, для регистрации пользователей, отправки тестов или работы с результатами.
* Сеансовое управление: PHP предоставляет функции для управления сессиями и куки, что важно для реализации авторизации пользователей.
* Безопасность: PHP включает в себя механизмы защиты от атак, таких как SQL-инъекции, XSS и CSRF.

Фреймворк Laravel:

* Структурированность кода: Laravel использует архитектуру MVC (Model-View-Controller), что помогает организовать код, обеспечивая его поддержку и масштабируемость.
* ORM (Eloquent): Laravel предоставляет удобный инструмент для работы с базами данных через ORM Eloquent, который позволяет легко взаимодействовать с базой данных без необходимости писать сложные SQL-запросы.
* Миграции и сиды: Laravel поддерживает миграции базы данных, что позволяет легко управлять изменениями в структуре данных, а также сиды, которые позволяют наполнить базу данных тестовыми данными.
* Безопасность: Laravel включает встроенные механизмы для защиты от основных уязвимостей, таких как CSRF и XSS, а также функции для хэширования паролей и безопасной аутентификации пользователей.
* Шаблонизация: В Laravel используется шаблонизатор Blade, который позволяет легко создавать динамичные HTML-шаблоны и улучшать разделение логики и представления.