ВЫБОР ИНСТРУМЕНТОВ И ТЕХНОЛОГИЙ ДЛЯ СОЗДАНИЯ СОВРЕМЕННОГО ВЕБ-САЙТА

Исламова Г.Г., к.с.н., доцент,
Ахмедшин Ш.Р., студент,

Кашфутдинов Т.М., студент,
Башкирский ГАУ, г.Уфа, Россия

Аннотация. В статье рассматриваются основные подходы к созданию современного веб-сайта: самостоятельная разработка, использование готовых шаблонов и применение конструкторов. Особое внимание уделяется выбору языков программирования, программного обеспечения и серверной инфраструктуры. Показано, что выбор инструментов зависит от цели проекта, уровня подготовки разработчика, бюджета, требований к безопасности и дальнейшему развитию сайта.

Ключевые слова: сайт, веб-разработка, HTML, CSS, JavaScript, сервер, CMS, конструктор.

Создание сайта является одной из наиболее востребованных задач в сфере информационных технологий. Сайт может использоваться для представления компании, продажи товаров, публикации материалов, продвижения услуг, организации обратной связи с клиентами или автоматизации отдельных бизнес-процессов. При этом разработка сайта включает не только написание кода, но и выбор подходящих

инструментов, языков программирования, программного обеспечения, способа размещения и дальнейшего сопровождения.

В современной веб-разработке можно выделить несколько основных подходов к созданию сайта. Первый подход - самостоятельная разработка, когда структура, дизайн и функциональность создаются вручную с использованием языков программирования. Второй подход - использование готовых шаблонов, которые позволяют ускорить процесс создания сайта. Третий подход - применение конструкторов сайтов, где пользователь собирает страницу из готовых блоков. Также отдельно можно выделить использование CMS, то есть систем управления сайтом, например WordPress.

Самостоятельная разработка является наиболее гибким способом создания сайта. Она позволяет полностью контролировать внешний вид, структуру, безопасность и логику работы проекта. Такой подход подходит для сложных сайтов, веб-приложений, интернет-магазинов, личных кабинетов, информационных систем и сервисов с нестандартными функциями. Однако самостоятельная разработка требует знаний HTML, CSS, JavaScript, серверных языков, баз данных и принципов безопасности.

Готовые шаблоны позволяют ускорить создание сайта. Разработчик берет уже подготовленную структуру или дизайн и изменяет их под свои задачи. Такой вариант подходит для лендингов, портфолио, сайтов компаний и учебных проектов. Преимущество шаблонов заключается в экономии времени, но есть и недостатки: сайт может быть похож на другие проекты, а внутри шаблона может быть лишний или плохо организованный код.

Конструкторы сайтов удобны для быстрого запуска простых проектов. Они позволяют создать сайт без глубокого знания программирования. Пользователь выбирает готовые блоки, изменяет текст, изображения, цвета и публикует сайт. Такой подход подходит для сайтов-визиток, страниц мероприятий и небольших презентационных сайтов. Однако конструкторы ограничивают гибкость, усложняют перенос сайта на другую платформу и не всегда подходят для сложной логики.

CMS занимает промежуточное положение между самостоятельной разработкой и конструктором. Система управления контентом позволяет редактировать страницы, добавлять новости, изображения, товары и другие материалы без постоянного обращения к программисту. Наиболее известным примером является WordPress.

Таблица 1 - Сравнение основных подходов к созданию сайта

Подход

Сущность подхода

Преимущества

Недостатки

Когда лучше использовать

Самостоятельная разработка

Сайт создается вручную с использованием HTML, CSS, JavaScript, серверных языков и баз дан

Полный контроль над кодом, дизайном, безопасностью и логикой работы. Возможность создат

Требует знаний программирования, времени и опыта. Без навыков разработчика сопровождат

Для сложных сайтов, веб-приложений, личных кабинетов, интернет-магазинов и информацион

Готовые шаблоны

За основу берется уже созданный макет или структура сайта, которые изменяются под конкре

Экономия времени. Есть готовая визуальная основа. Подходит для быстрого запуска проекта

Сайт может быть похож на другие проекты. Иногда в шаблонах есть лишний код или ограниче

Для лендингов, портфолио, сайтов компаний, учебных проектов и небольших коммерческих ст

Конструкторы сайтов

Сайт создается из готовых блоков без полноценного написания кода

Простота использования. Быстрый запуск. Не требуется самостоятельная настройка сервера

Ограниченная гибкость. Зависимость от платформы. Сложнее перенести сайт на другой серви

Для сайтов-визиток, страниц мероприятий, презентационных сайтов и простых лендингов

CMS

Сайт создается на системе управления контентом, например WordPress

Удобное редактирование материалов. Много готовых тем и плагинов. Подходит для регулярно

Требует обновлений и контроля безопасности. При большом количестве плагинов сайт может

Для блогов, корпоративных сайтов, новостных ресурсов и небольших интернет-магазинов

Базовыми технологиями для создания сайта являются HTML, CSS и JavaScript. HTML отвечает за структуру веб-страницы: заголовки, абзацы, изображения, ссылки, формы и таблицы. CSS используется для оформления внешнего вида сайта: цветов, шрифтов, отступов, расположения блоков и адаптивности. JavaScript добавляет интерактивность: обработку кнопок, проверку форм, выпадающие меню, динамическое обновление данных и другие функции.

Для более сложных проектов часто используется TypeScript. Он основан на JavaScript, но добавляет строгую типизацию. Это помогает раньше находить ошибки, упрощает сопровождение крупных проектов и делает код более понятным для команды. TypeScript особенно полезен при разработке веб-приложений, где есть много компонентов, состояний и взаимодействий между частями системы.

Для frontend-разработки применяются React, Vue, Angular и другие фреймворки. React удобен для создания интерфейсов из компонентов. Vue проще для изучения и подходит для небольших и средних проектов. Angular чаще используется в крупных корпоративных системах, где важны строгая структура и масштабируемость. Выбор фреймворка зависит от сложности проекта, опыта команды и требований к дальнейшему развитию.

Серверная часть сайта может быть написана на Node.js, Python, PHP, Java, C# и других языках. Node.js позволяет использовать JavaScript на сервере. Python удобен благодаря простому синтаксису и фреймворкам Django и Flask. PHP широко применяется в классической веб-разработке и CMS. Java и C# чаще используются в крупных корпоративных информационных системах, где важны надежность, безопасность и

интеграция с другими сервисами.

Для хранения данных применяются базы данных. Наиболее распространенными являются PostgreSQL, MySQL, SQLite и MongoDB. PostgreSQL подходит для надежных проектов со сложными связями между данными. MySQL часто используется в веб-разработке и CMS. SQLite удобен для небольших проектов и учебных задач. MongoDB применяется там, где нужны гибкие документы и нет жесткой табличной структуры.

Таблица 2 - Основные языки и технологии веб-разработки

Технология	**Назначение**	**Преимущества**	**Ограничения**
HTML			

Создание структуры веб-страницы

Прост в изучении. Является обязательной основой любого сайта

Не отвечает за оформление и интерактивность

CSS

Оформление внешнего вида сайта

Позволяет задавать цвета, шрифты, отступы, адаптивность и анимации

В крупных проектах требует правильной организации стилей

JavaScript

Добавление интерактивности на сайт

Работает в браузере. Может использоваться и на сервере через Node.js

При плохой структуре код быстро становится сложным для сопровождения

TypeScript

Расширение JavaScript со строгой типизацией

Помогает находить ошибки раньше. Удобен для крупных проектов

Требует дополнительного изучения и настройки проекта

Python

Разработка серверной части, обработка данных и автоматизация

Имеет простой синтаксис. Подходит для backend-разработки через Django и Flask

В некоторых высоконагруженных проектах может уступать по скорости другим языкам

PHP

Серверная разработка сайтов и работа с CMS

Широко используется в веб-разработке. Особенно часто применяется в WordPress

В старых проектах может встречаться устаревший и плохо структурированный код

SQL

Работа с базами данных

Позволяет хранить, выбирать, изменять и анализировать данные

Требует понимания структуры базы данных и связей между таблицами

Для написания сайта необходимо выбрать подходящее программное обеспечение. Одним из самых удобных редакторов является Visual Studio Code. Он бесплатный, поддерживает большое количество расширений и подходит для работы с HTML, CSS, JavaScript, TypeScript, Python, PHP и другими языками. Для профессиональной разработки можно использовать WebStorm, который удобен для JavaScript, TypeScript, React и Node.js. Для Python-проектов подойдет PyCharm, для PHP - PhpStorm, для Java - IntelliJ IDEA.

Помимо редактора кода, разработчику нужны дополнительные инструменты. Git используется для контроля версий и позволяет сохранять историю изменений. GitHub или GitLab применяются для хранения кода и командной работы. Figma используется

для проектирования интерфейса и предварительного создания дизайна сайта. Postman помогает тестировать серверные запросы. Docker позволяет запускать проект в одинаковой среде на разных устройствах и снижает проблемы при переносе сайта.

Отдельное значение имеет выбор сервера. Сайт можно разместить на обычном хостинге, арендованном VPS, облачной платформе или собственном сервере. Обычный хостинг подходит для простых сайтов и CMS. VPS дает больше свободы: можно устанавливать нужное программное обеспечение, настраивать базу данных, серверную часть, SSL-сертификаты и резервное копирование. Облачные платформы удобны для масштабируемых проектов, но могут быть сложнее и дороже.

Собственный сервер дает полный контроль над оборудованием, операционной системой, настройками безопасности и хранением данных. Это может быть преимуществом для организаций, которым важна независимость и контроль над инфраструктурой. Однако собственный сервер требует технических знаний, постоянного обслуживания, защиты от атак, настройки сети, резервного питания и резервного копирования. Если у организации нет специалиста, такой вариант может стать не преимуществом, а проблемой.

Арендованный сервер часто является более рациональным решением для небольших и средних проектов. Провайдер отвечает за размещение оборудования, питание, интернет-канал и базовую доступность. Разработчику остается настроить программную часть проекта. Поэтому нельзя сказать, что собственный сервер всегда лучше арендованного. Собственный сервер лучше при наличии ресурсов и специалистов, а аренда удобнее для быстрого запуска и снижения технической нагрузки.

При выборе инструментов необходимо учитывать назначение сайта. Для простого сайта-визитки достаточно HTML, CSS и JavaScript или конструктора. Для блога и корпоративного сайта можно использовать CMS. Для интернет-магазина потребуется система управления товарами, корзина, оплата и база данных. Для сложной информационной системы лучше использовать frontend-фреймворк, серверную часть и надежную базу данных.

Также важно учитывать безопасность. Современный сайт должен использовать HTTPS, защищенную авторизацию, проверку вводимых данных, резервное копирование и регулярное обновление зависимостей. Особенно это важно для сайтов, где есть личные данные пользователей, платежи, заказы или административная панель. Даже красивый сайт нельзя считать качественным, если он уязвим и может потерять данные.

Таким образом, выбор инструментов и технологий для создания сайта зависит от цели проекта, уровня сложности, бюджета и требований к развитию. Самостоятельная разработка дает максимальную гибкость и контроль, но требует знаний и времени. Шаблоны позволяют ускорить работу, но ограничивают уникальность. Конструкторы подходят для простых сайтов, но не всегда подходят для сложных задач. CMS удобна для регулярного обновления контента, но требует контроля безопасности. Наиболее правильный подход заключается в том, чтобы выбирать технологии не по популярности, а по задачам конкретного проекта. В этом случае сайт будет не только внешне привлекательным, но и надежным, безопасным и удобным для дальнейшего развития.

Литература

1. MDN Web Docs. Learn web development [Электронный ресурс]. — URL: https://developer.mozilla.org/en-US/docs/Learn_web_development — Дата обращения: 28.05.2026.

2. GitHub. Octoverse 2025: A new developer joins GitHub every second as AI leads TypeScript to #1 [Электронный ресурс]. — URL: https://github.blog/news-insights/octoverse/octoverse-a-new-developer-joins-github-every-second-as-ai-leads-typescript-to-1/ — Дата обращения: 28.05.2026.

3. Stack Overflow. 2025 Developer Survey. Technology [Электронный ресурс]. — URL: https://survey.stackoverflow.co/2025/technology — Дата обращения: 28.05.2026.

4. JetBrains. The State of Developer Ecosystem Report 2025 [Электронный ресурс]. — URL: https://devecosystem-2025.jetbrains.com/ — Дата обращения: 28.05.2026.

5. Фримен Э., Робсон Э. Изучаем HTML, XHTML и CSS. — Санкт-Петербург: Питер, 2014. — 656 с.

6. Дакетт Дж. HTML и CSS. Разработка и дизайн веб-сайтов. — Москва: Эксмо, 2018. — 480 с.

7. Дакетт Дж. JavaScript и jQuery. Интерактивная веб-разработка. — Москва:

Эксмо, 2017. — 640 с.

8. Макграт М. Программирование на JavaScript для начинающих. — Москва: Эксмо, 2020. — 192 с.