Инженерный вестник Дона, №5 (2025)
ivdon.ru/ru/magazine/archive/n5y2025/10035

Разработка системы управления контентом сайта на базе фреймворков NestJS и NextJS

В.Е. Агапов

Волгоградский государственный технический университет, Волгоград

Аннотация: Рассматривается разработка системы автоматической генерации стартовых шаблонов сайтов для упрощения создания веб-приложений. Использование кодогенерации позволяет автоматизировать процесс написания повторяющегося кода, сокращая время разработки и повышая эффективность работы разработчиков. Система предоставляет удобный интерфейс для выбора и настройки шаблонов, исключая необходимость работы с консольными командами. Это позволяет ускорить процесс прототипирования и развертывания веб-приложений, что особенно актуально при создании проектов с множеством повторяющихся компонентов.
Ключевые слова: сайт, управление контентом, кодогенерация, система управление контентом, шаблон сайта, веб-приложение, фреймворк, серверная часть, клиентская часть, оптимизация.

Введение

Сайты играют ключевую роль в современном цифровом мире, предоставляя компаниям возможность представлять продукты и услуги, поддерживать связь с клиентами и укреплять бренды. Однако управление и поддержка сайта требуют регулярного обновления контента, что становится проблемой для владельцев без навыков программирования.

Существует несколько подходов для создания и управления сайтами, включая разработку с нуля и использование готовых систем управления контентом. Первый подход предоставляет больше свободы, но требует значительных усилий, а второй облегчает управление, но накладывает ограничения. Чтобы объединить преимущества обоих подходов, предлагается создать систему управления контентом на базе фреймворков NestJS и NextJS, позволяющую автоматизировать и оптимизировать процесс создания сайтов [1].

Характеристики проблемной области

Веб-приложение — это программное обеспечение, доступное через веб-браузер и работающее на удалённом сервере. Оно позволяет выполнять различные задачи через интернет (образование, соцсети, бронирования, игры и др.), не требуя установки на устройствах.

Веб-приложения состоят из двух частей:

Клиентская часть: отвечает за интерфейс и работу на устройстве пользователя. Включает HTML (структура), CSS (дизайн), JavaScript (интерактивность). Примеры технологий: React, Angular, Vue.js. [2]

Серверная часть: обрабатывает запросы, работает с базами данных и логикой. Используются языки Python, Java, Ruby, Node.js и фреймворки.

React, популярный клиентский фреймворк, ускоряет работу за счёт виртуального DOM, компонентности и однонаправленного потока данных. Однако он имеет ограничения, такие как влияние на SEO, отсутствие встроенного роутинга и работу с серверной логикой [3].

Для устранения недостатков React создан Next.js. Он улучшает производительность, SEO, упрощает маршрутизацию и работу с серверной логикой, но требует базового шаблонного кода для создания страниц и компонентов [4].

Способы решения задачи, их плюсы и минусы

Разработчики веб-приложений сталкиваются с необходимостью создания множества компонентов и страниц, а также написания шаблонного кода. Для упрощения процесса используются три подхода:

Копирование и редактирование кода: доступный, но времязатратный способ. Постоянное создание однотипных элементов неудобно.

Фреймворки: предоставляют инструменты для сокращения повторяющегося кода и упрощают процесс разработки. Однако они не решают проблему автоматизации на 100%.

Кодогенерация: самый актуальный метод. Утилиты позволяют автоматизировать создание компонентов и страниц, снижая вероятность ошибок. Недостаток: большинство инструментов работают через консоль, что не всегда удобно.

Применение кодогенерации с простым интерфейсом значительно ускоряет разработку [5, 6].

Анализ прототипов

Create React App: инструмент для быстрой настройки проектов React. Упрощает старт разработки, но не предоставляет встроенной генерации компонентов. Создавать и настраивать их приходится вручную.

Angular CLI: команда ng generate генерирует компоненты и настройки автоматически. Требует знания командной строки, а внутренние механизмы работы сложны для новичков. Работает только с Angular.

Vue CLI: интуитивный инструмент для Vue.js.. Позволяет генерировать компоненты, маршруты и состояние приложения. Однако избыточные настройки и работа через командную строку могут быть непривычны для новичков[7].

Поскольку разрабатываемый продукт не имеет прямых конкурентов на рынке из-за его особенной реализации, мы можем сравнить его с указанными выше прототипами по ключевым критериям. Это сравнение представлено в таблице № 1.

В рамках сравнения были выделены следующие критерии:

графический интерфейс: чтобы не вникать какие команды консоли нужны для создания компонента или стартового шаблон, важно иметь

графический интерфейс для простой настройки будущего стартового шаблона;

шаблоны сайтов: важно иметь готовые шаблоны сайтов для дальнейшего использования под определенные задачи;

добавление компонентов: необходимо иметь возможность добавления нужных компонентов для дальнейшей разработки;

редактирование компонентов: данная возможность позволяет разработчику быстрее создать минимальный шаблон проекта, который он хочет видеть для дальнейшей его модификации.

Таким образом, разрабатываемый продукт должен обладать всеми перечисленными характеристиками и решать проблемы, которые имеются у других имеющихся прототипов.

Таблица № 1

Сравнение прототипов и разрабатываемого продукта по ключевым критериям

	Разрабатываемый продукт	Create React App	Angular CLI	Vue CLI
Графический интерфейс	+	-	-	-
Шаблоны сайтов	+	-	-	-
Добавление компонентов	+	-	+	+
Редактирование компонентов	+	-	-	-

Способ разработки системы управления контентом сайта

Предлагается использовать фреймворк Next.js для клиентской части

приложения, так как он обеспечивает серверный рендеринг и статическую генерацию, что способствует высокой производительности интерфейса. Также Next.js имеет богатую экосистему плагинов и расширений.

Для серверной части приложения предлагается использовать фреймворк Nest.js, который является модульным и масштабируемым. Он предоставляет архитектурный подход, основанный на модулях, контроллерах и провайдерах, что способствует чистоте кода и повторному использованию.

Комбинация Next.js и Nest.js позволяет создать полноценное веб-приложение с тесной интеграцией клиентской и серверной частей. Благодаря сильному типированию и структуре кода в обоих фреймворках, разработка и поддержка приложения становятся более удобными и эффективными [8].

В предложенном решении выбран подход разделения на компоненты в фреймворке Next.js и подход MVC-архитектуры в фреймворке Nest.js. Это позволяет создать эффективное и легко поддерживаемое приложение. Подход разделения разделения на компоненты в Next.js позволяет организовать код приложения на уровне компонентов, что облегчает его понимание и расширение. В то же время, использование MVC-архитектуры в Nest.js обеспечивает разделение логики приложения на модели, представления и контроллеры, что способствует повторному использованию и чистоте кода [9].

В отношении формата хранения данных, выбран PostgreSQL как рекомендуемый формат. PostgreSQL обеспечивает надежность, мощные возможности запросов и высокую производительность. Для более удобного взаимодействия с базой данных и эффективного управления данными в приложении предлагается использовать ORM библиотеки.

Эти выборы обеспечивают гибкость, эффективность и легкость

поддержки разрабатываемого веб-приложения. В целом, предложенное решение обещает упростить процесс разработки веб-приложений, снизить сложность для разработчиков и повысить качество продукта. Оно комбинирует современные фреймворки и инструменты, обеспечивая мощную и гибкую основу для создания персонализированных веб-приложений [10].

Описание модели разрабатываемой системы

Разрабатываемая система – веб-приложение, которое генерирует исходный код выбранного и настроенного пользователем шаблона сайта. У данной системы достаточно простой и понятный интерфейс.

Для начала пользователя необходимо зарегистрироваться в системе на странице регистрации. Для этого ему необходимо ввести свой логин, электронную почту, пароль и нажать кнопку «Зарегистрироваться».

После успешной регистрации, на странице входа в систему пользователь, может ввести свою электронную почту и пароль от своей учетной записи и произвести вход в систему нажав кнопку «Войти».

Авторизованный пользователь попадает на главную страницу системы и имеет возможность воспользоваться разделами боковой панели: «Создать сайт», «Мои сайты», «Выход».

Выбрав раздел «Создать сайт», пользователь перенаправляется на страницу выбора шаблона создаваемого сайта.

Сделав выбор необходимого шаблона сайта, пользователю предоставляется выбор компонентов, которые будут использоваться в создаваемом сайте. Чтобы подтвердить свой выбор, пользователь должен нажать кнопку «Применить», после чего попадает на страницу настройки

компонентов создаваемого сайта.

На странице настройки компонентов создаваемого сайта, пользователю будет предложено дать название для создаваемого шаблона и задать значения для выбранных ранее компонентов. Настроив компоненты, пользователь должен нажать кнопку «Применить», после чего сайт с выбранными и настроенными компонентами будет создан.

Выбрав раздел «Мои сайты», пользователь попадает на страницу с созданными им сайтами, если же сайты не были созданы, то страница будет пустой. В блоке созданного сайта будет указан его тип (например Landing), название шаблона (которое было задано на странице настройки компонентов) и три кнопки: «Скачать шаблон созданного сайта», «Редактировать шаблон созданного сайта», «Удалить шаблон созданного сайта».

При нажатии на «Удалить шаблон созданного сайта» наш шаблон будет удален.

При нажатии на «Редактировать шаблон созданного сайта», мы будем перенаправлены на страницу используемых компонентов и настройку этих компонентов.

При нажатии на «Скачать шаблон созданного сайта», будет произведено скачивание сгенерированного проекта в виде архива.

Выбрав раздел «Выход», пользователь выходит из системы и попадает на страницу входа в систему.

Структура классов серверной части

При разработке серверной части нашего приложения мы применили современную архитектуру, основанную на паттерне MVC (Model-View-Controller), которая позволяет нам легко управлять и поддерживать код.

Данный подход можно разделить на три основные части.

Контроллеры (controllers) - классы, отвечающие за обработку входящих запросов и управление потоком данных. Они служат точкой входа в приложение и обеспечивают обработку запросов от клиента. У нас есть контроллеры для различных функциональных областей, таких как аутентификация, лендинг, портфолио и компоненты лендинга и портфолио.

Сервисы (services) - классы, реализующие бизнес-логику нашего приложения. Они отвечают за обработку запросов, взаимодействие с базой данных и другими сервисами. Сервисы служат прослойкой между контроллерами и репозиториями, обеспечивая выполнение необходимых операций и обработку данных. Мы создали сервисы для аутентификации, работы с лендингом, портфолио и компонентами лендинга и портфолио.

Репозитории (repositories) - классы, отвечающие за взаимодействие с базой данных. Они предоставляют методы для выполнения операций чтения, записи, обновления и удаления данных. Репозитории используются сервисами для получения и сохранения данных в базе данных. У нас есть репозитории для работы с аутентификацией, лендингом, портфолио и компонентами лендинга и портфолио (рис.1).

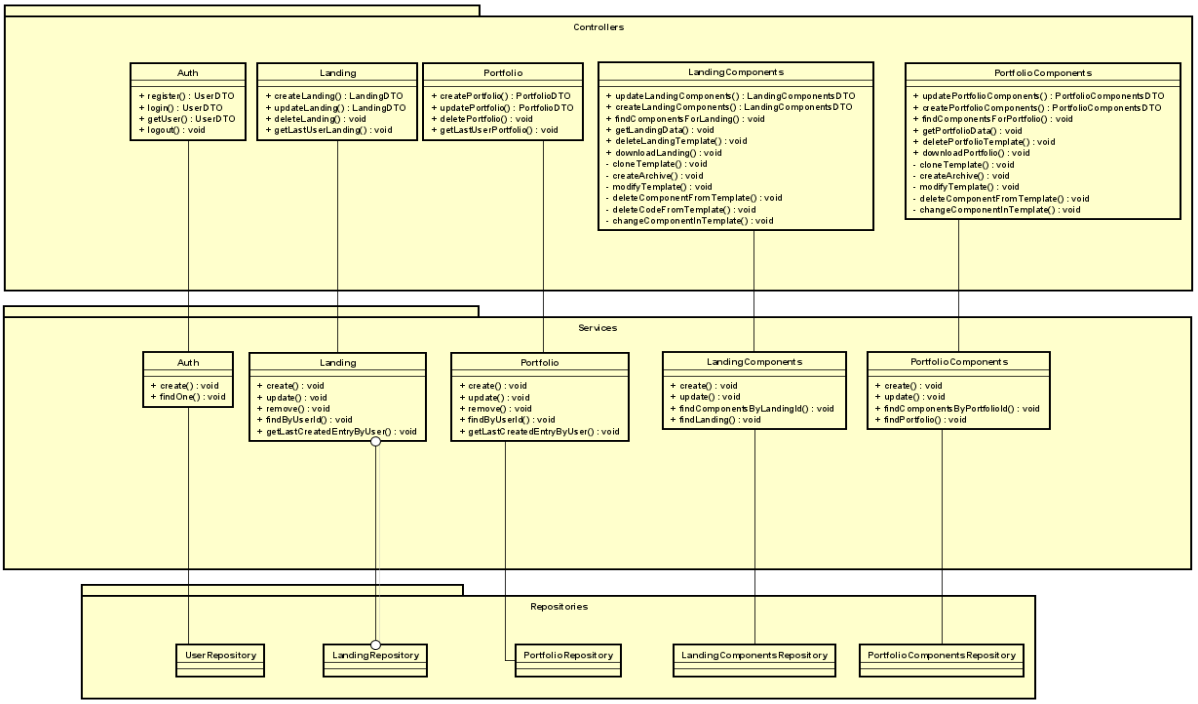

Рис. 1. – Структура программы на уровне классов

Структура клиентской части части

Структура клиентской части максимально проста. В папке "api" находятся файлы, отвечающие за взаимодействие с сервером и содержащие логику для работы с различными типами данных (например, landing, portfolio, users).

В папке "components" располагаются переиспользуемые компоненты, такие как Sidebar и Layout, а также компоненты, отображающие элементы списка для созданных сайтов (Portfolio и Landing).

В папке "pages" находятся основные страницы проекта, такие как главная страница (index), страницы входа (login), регистрации (register), админ-панель (admin), страница со списком сайтов (sites) и директории "new-site" и "site", содержащие страницы для создания и редактирования контента соответственно (landing и portfolio).

В папке "styles" находятся файлы с CSS-стилями, которые

применяются к компонентам и страницам проекта.

Такая структура помогает организовать код проекта и упрощает его разработку и поддержку, разделяя функциональность на логические блоки.

Пример работы программы

Для начала пользования разработанным решением пользователю надо зарегистрироваться в системе. В форме регистрации обязательно нужно заполнить следующие поля: логин, электронная почта и пароль (рис.2).

После чего пользователь будет перенаправлен на страницу входа, где требуется ввести данные для входа в систему (рис.3).

После успешного входа в систему, пользователь может увидеть информацию о текущем авторизованном пользователе и три раздела на боковой панели: «Создать сайт», «Мои сайты», «Выход» (рис.4).

Перейдем к созданию сайта, для этого на боковой панели выберем раздел «Создать сайт», где предлагается сделать выбор создаваемого шаблона, в нашем случае это «Сайт портфолио» и «Сайт лендинг» (рис.5).

Рис. 2. – Страница регистрации

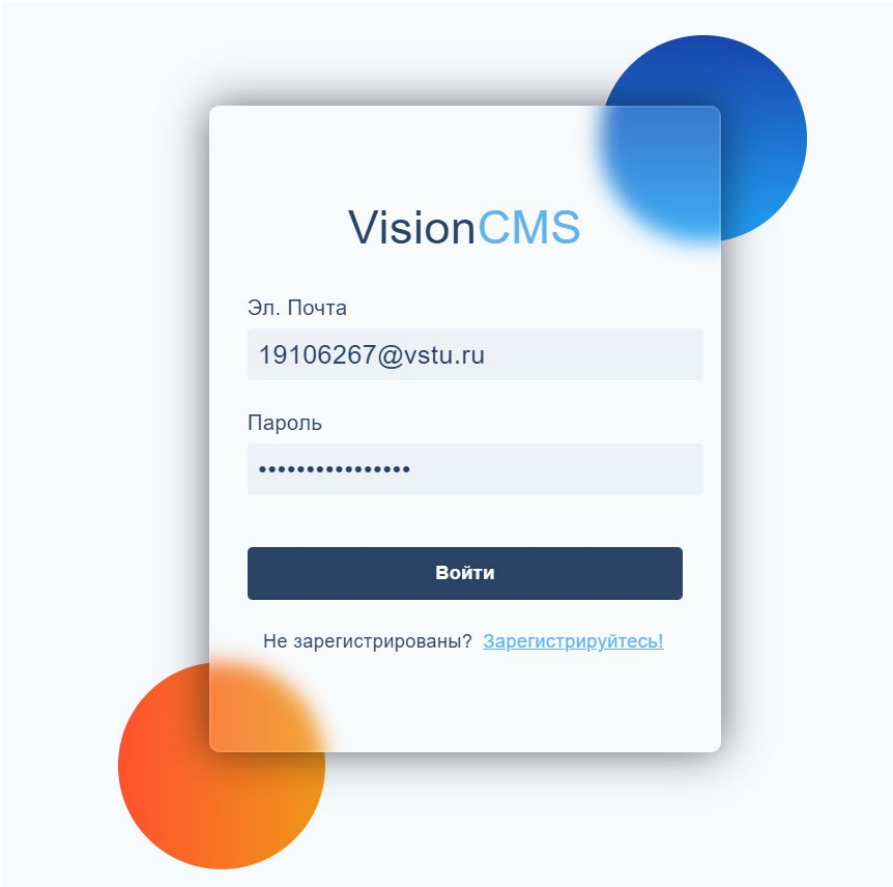

Рис. 3. – Страница входа в систему

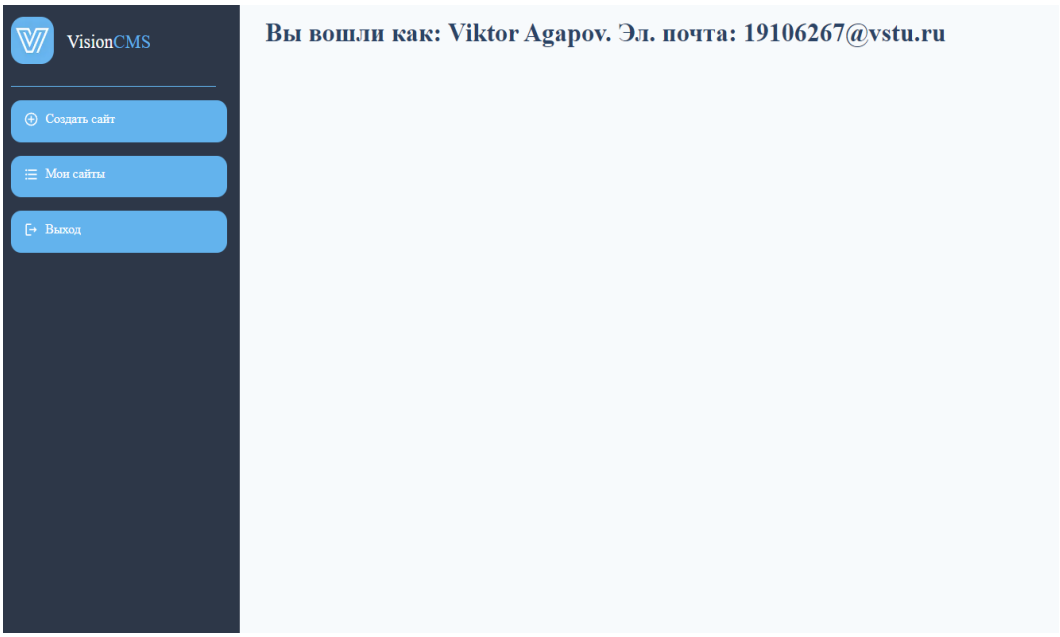

Рис. 4. – Главная страница админ панели

Рис. 5. – Страница выбора шаблона сайта

Рассмотрим пример создания сайта на примере шаблона «Сайт лендинг». Выбрав шаблон пользователь попадает на страницу выбора компонентов, которые будут использоваться на создаваемом сайте. Выбираем необходимые компоненты и нажимаем кнопку «Применить» (рис.6).

Рис. 6. – Страница выбора компонентов создаваемого сайта

После выбора компонентов, пользователю будет предложено дать название созданному шаблону, а также задать значения для выбранных ранее компонентов. Выбираем необходимые компоненты и нажимаем кнопку

«Применить» (рис.7).

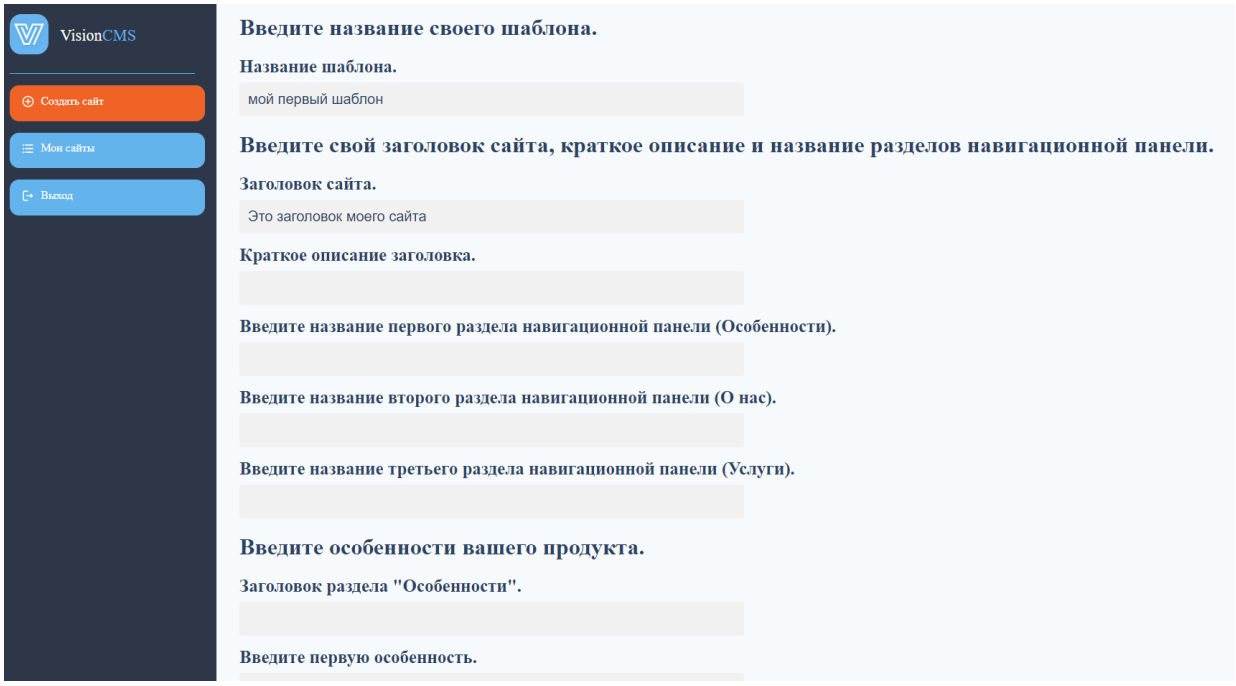

Рис. 7. – Страница настройки компонентов создаваемого сайта

После настройки компонентов сайта пользователь может перейти в раздел «Мои сайты», где отображается созданный сайт типа «Landing» с заданным названием (рис.8).

На странице доступны три кнопки:

«Скачать шаблон созданного сайта» — скачивание архива с проектом.

«Редактировать шаблон созданного сайта» — переход к настройке компонентов.

«Удалить шаблон созданного сайта» — удаление шаблона

Распаковав архив мы открываем сгенерированный проект и видим готовый проект с выбранными нами компонентами и их заданными значениями. Запустим сгенерированный проект сайта. И можем наблюдать хороший стартовый шаблон с помощью которого можно легко продолжить

разработку нужного проекта (рис.9).

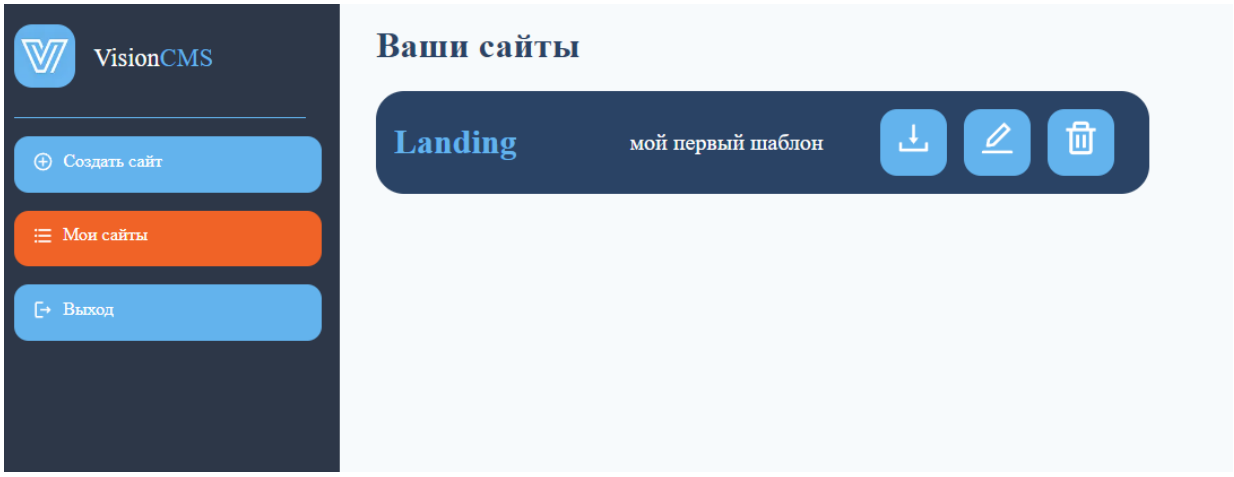

Рис. 8. – Страница созданных пользователем сайтов

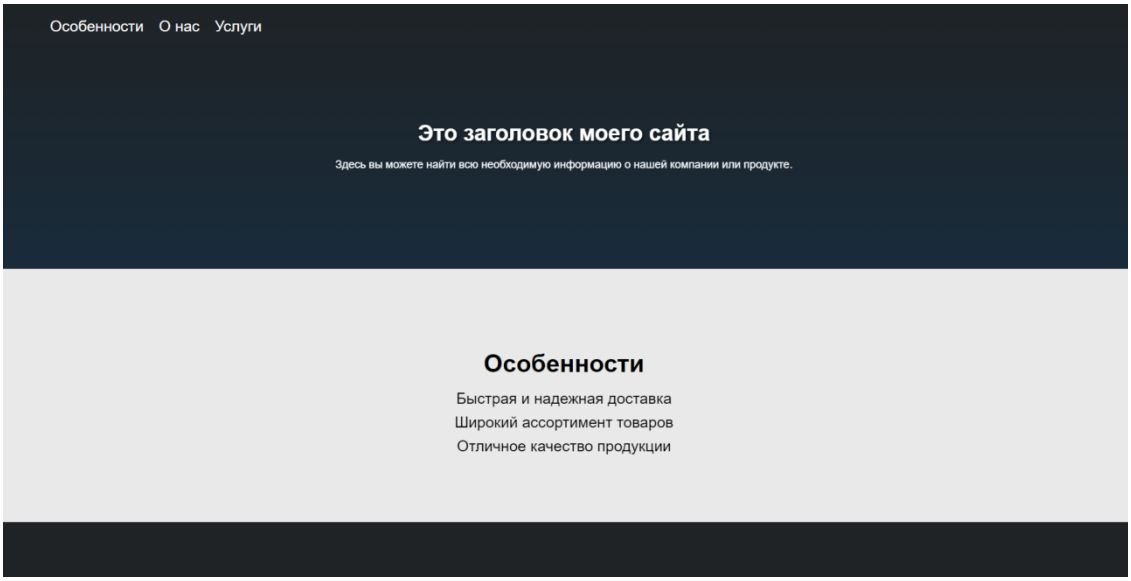

Рис. 9. – Страница сгенерированного сайта

Заключение

В ходе работы проведен анализ существующих решений, выявлены их достоинства и недостатки, что стало основой для разработки более эффективного продукта. Использование фреймворков Next.js и Nest.js позволило создать инновационное веб-приложение с высокой степенью интеграции клиентской и серверной частей, обеспечивая гибкость, производительность и легкость поддержки.

Разработанный программный продукт использует современные технологии и архитектуру MVC, что подтверждает его структурированность и возможность повторного использования кода. Функциональное тестирование подтвердило корректность работы системы генерации шаблонов сайтов, ее готовность к внедрению и соответствие требованиям.

Решение предоставляет надежные инструменты для создания и настройки веб-сайтов, ускоряет разработку и повышает качество веб-приложений. Все задачи выполнены, продукт готов к использованию и обладает преимуществами перед существующими аналогами, способствуя улучшению процессов в предметной области.

Литература

1. Короленко И.А. Анализ проблем последовательности загрузки клиентских веб-приложений // Инженерный вестник Дона. 2023. №11. URL: ivdon.ru/ru/magazine/archive/n11y2023/8777.
2. Короленко И.А. О фреймворках Vue, Svelte, Solid и Lit для разработки клиентских веб-приложений // Инженерный вестник Дона. 2023. №11. URL: wmv.ivdon.ru/ru/magazine/archive/n11y2023/8804.
3. Евстратов О.Д. Сравнительный анализ фреймворков для frontend-разработки // Молодой ученый. 2024. URL: moluch.ru/archive/521/114864/.
4. Потовиченко М.А., Шатилов Ю.Ю. Разработка клиентской части одностраничного веб-приложения с использованием библиотеки React // Научный журнал. 2020. URL: science-engineering.ru/ru/article/view?id=1278.
5. Чернецкий И.И. Создание инновационного метода адаптивной веб-разработки для повышения производительности и удобства использования // Молодой ученый. 2024. URL: moluch.ru/archive/528/116769/.

6. Иванов Ф.Ф. Вестник кибернетики. 2016. URL: cyberleninka.ru/article/n/avtomaticheskaya-kodogeneratsiya-i-reinzhiniring-programmnogo-obespecheniya-pri-sozdanii-avtomatizirovannyh-i-avtomaticheskih.
7. Матвеев В.С. Современные фреймворки для разработки информационных систем на примере платформы Vue // CyberLeninka. URL: cyberleninka.ru/article/n/sovremennye-freymvorki-dlya-razrabotki-informatsionnyh-sistem-na-primere-platformy-vue.
8. Сергачева М.А., Михалевская К.А. Журнал Кронос: естественные и технические науки, 2020. URL: cyberleninka.ru/article/n/analiz-freymvorkov-dlya-razrabotki-sovremennyh-veb-prilozheniy.
9. Dragos-Paul Pop. Procedia Computer Science, 2014. URL: sciencedirect.com/science/article/pii/S187770581400352X.
10. Азимова Д.Ю. Современный подход к разработке web-ресурсов // Молодой ученый. 2016. URL: moluch.ru/archive/125/34299/.

References

1. Korolenko I.A. Inzhenernyj vestnik Dona, 2023, №11. URL: ivdon.ru/ru/magazine/archive/n11y2023/8777
2. Korolenko I.A. Inzhenernyj vestnik Dona, 2023, №11. URL: ivdon.ru/ru/magazine/archive/n11y2023/8804
3. Evstratov O.D. Sravnitelnyj analiz freymvorkov dlya frontend-razrabotki [Comparative analysis of frameworks for frontend development]. Molodoj uchenyj. 2024. URL: moluch.ru/archive/521/114864
4. Potovichenko M.A., Shatilov Yu.Yu. Razrabotka klientckoy chasti odnostranichnogo veb-prilozheniya s ispolzovaniem biblioteki React [Development of the client-side of a single-page web application using the

React library]. Nauchnyj zhurnal. 2020. URL: science-engineering.ru/ru/article/view?id=1278

5. Chernetskij I.I. Sozdanie innovatsionnogo metoda adaptivnoj veb-razrabotki dlya povysheniya proizvoditelnosti i udobstva ispolzovaniya [Creation of an innovative method of adaptive web development to improve performance and usability]. Molodoj uchenyj. 2024. URL: moluch.ru/archive/528/116769
6. Ivanov F.F. Vestnik kibernetiki. 2016. URL: cyberleninka.ru/article/n/avtomaticheskaya-kodogeneratsiya-i-reinzhiniring-programmnogo-obespecheniya-pri-sozdanii-avtomatizirovannyh-i-avtomaticheskih
7. Matveev V.S. Sovremennye freymvorki dlya razrabotki informatsionnykh sistem na primere platformy Vue [Modern frameworks for the development of information systems using the Vue platform]. URL: cyberleninka.ru/article/n/sovremennye-freymvorki-dlya-razrabotki-informatsionnyh-sistem-na-primere-platformy-vue
8. Sergacheva M.A., Mikhalevskaya K.A. Zhurnal Kronos: estestvennye i tekhnicheskie nauki, 2020. URL: cyberleninka.ru/article/n/analiz-freymvorkov-dlya-razrabotki-sovremennyh-veb-prilozheniy
9. Dragos-Paul Pop. Procedia Computer Science, 2014. URL: sciencedirect.com/science/article/pii/S187770581400352X
10. Azimova D.Yu. Sovremennyj podkhod k razrabotke web-resursov [Modern approach to web resource development] Molodoj uchenyj. 2016. URL: moluch.ru/archive/125/34299

Дата поступления: 18.02.2025

Дата публикации: 25.04.2025