УДК 004.6

ИСПОЛЬЗОВАНИЕ LOW-CODE СИСТЕМ ДЛЯ РАЗРАБОТКИ ИНТЕРФЕЙСОВ ВЕБ-ПРИЛОЖЕНИЙ

Акимов А.Е.,

магистрант,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Кряжева Е. В.,

к.псих.н., доцент,

Калужский государственный университет им. К.Э. Циолковского,

Калуга, Россия

Аннотация.

В статье рассматривается проблема использования low-code системы для разработки интерфейса веб-приложения с минимальной необходимостью написания программного кода. Авторами описываются основные преимущества использования такого рода систем, их влияние на скорость разработки. Проводится сравнение данных систем с традиционным методом разработки путем написания программного кода. Также авторами предлагается пример использования такого метода при разработке интерфейса, показан результат работы. В конце статьи делаются выводы по проделанной работе.

Ключевые слова: веб-приложение, интерфейс, верстка, low-code система, tooljet.

USING LOW-CODE SYSTEMS TO DEVELOP WEB APPLICATION INTERFACES

Akimov A.E.,
Undergraduate,
Kaluga State University named after K.E. Tsiolkovsky,
Kaluga, Russia
Kryazheva E. V.,
Candidate of Psychological Sciences, Associate Professor,
Kaluga State University named after K.E. Tsiolkovsky,
Kaluga, Russia

Annotation.
The article discusses the problem of using a low-code system to develop a web application interface with minimal need to write program code. The authors describe the main advantages of using such systems, their impact on the speed of development. These systems are compared with the traditional development method by writing software code. The authors also offer an example of using this method when developing an interface, and show the result of the work. At the end of the article, conclusions are drawn on the work done.
Keywords: web application, interface, layout, low-code system, tooljet.

Устоявшимся на текущий момент процессом разработки интерфейса веб-приложения является метод написания на определенном наборе языков программирования программном коде, отвечающим за верстку, визуальный стиль, функционал и содержание приложения. Ранее для этого использовали исключительно стандартный набор библиотек и решения присутствующие в языке программирования по умолчанию. С развитием потребностей пользователей и технологий разработки, данный метод совершенствовался за счет создания специализированных библиотек, наборами универсальных стилей, позже появились фреймворки. Сам же процесс кардинально не менялся, основной всегда была работа внутри редактора кода с его «ручным

написанием» [3]. Ситуация начала изменяться с приходом новой идеи, что под разработкой можно понимать не только написание кода на языке программирования, но и любой процесс ответственный за создание «работающей» информационной системы, ведь главное в разработке это понимание и умение использовать алгоритмы, структуры и пути достижения определенных техническим заданием целей.

Для упрощения работы разработчиков, ответственных за создания интерфейсов, или же если разработчика с таким опытом нет, но интерфейс приложению необходим, на рынке образовались low-code системы – системы визуальной разработки, где процесс написания кода почти полностью заменен на расставление блоков, элементов через интерфейс среды разработки.

Таким образом целью данного работы будет рассмотрение возможности создания функционального веб-приложения с использованием low-code системы. Главным отличием low-code систем от традиционных является упрощения этапа разработки типизированных элементов присутствующих в подавляющем большинстве проектов, например: текстовых блоков, таблиц, кнопок, выпадающих списков и так далее. Надо отметить, что и процесс верстки через расставление блоков в области редактирования значительно проще воспринимается, чем создание HTML структуры, которые в основном мало чем отличаются от приложения к приложению.

Процесс традиционного написания кода зачастую затрагивает разработку сразу на как минимум трех языках программирования: HTML, CSS, JavaScript. Процесс занимает крайне большой промежуток времени, так как присутствует необходимость написания каждого элемента вручную, причем отдельно его блок, стиль и функционал. Даже начать данный процесс уже является время затратной операцией, поскольку необходимо прописать нужные импорты, объявить библиотеки, распределить системные файлы в директории проекта, настроить взаимосвязи между ними. Для каждого проекта этот процесс почти не отличается. В low-code системах, большинство

этих действий не требуют необходимости, так как процесс заранее автоматизирован, а настройки выполняются с помощью интерфейса. Для анализа такого рода систем, возьмем в качестве объекта систему «ToolJet».

«ToolJet» — это low-code система визуального программирования, основанная на популярной библиотеке для разработки интерфейсов веб-приложений React [2]. Система создана по принципам открытого исходного кода, и активно дорабатывается сообществом. Среди основных преимуществ системы можно выделить следующее:

– Возможность развертывания на собственных серверах, используя контейнеры Docker;

– поддержка работы с использованием как собственной системы хранения данных, так и подключения внешних источников и API;

– поддержка написания скриптов и сложного функционала с использованием популярных языков программирования JavaScript и Phyton;

– гибкая настройка стилей оформления за счет возможности редактирования CSS каждого элемента как в отдельности, так и группами классов, подклассов;

– поддержка современными браузерами;

– возможность использования с системами контроля версий git.

Таким образом ToolJet может подойти для разработки как простейших информационных проектов, так и для веб-приложений для решения определённых задач. Рассмотрим создание интерфейса с использованием системы «ToolJet».

Процесс работы в ToolJet начитается с экрана редактора после создания проекта. Открывается рабочая область, продемонстрированная на изображении ниже.

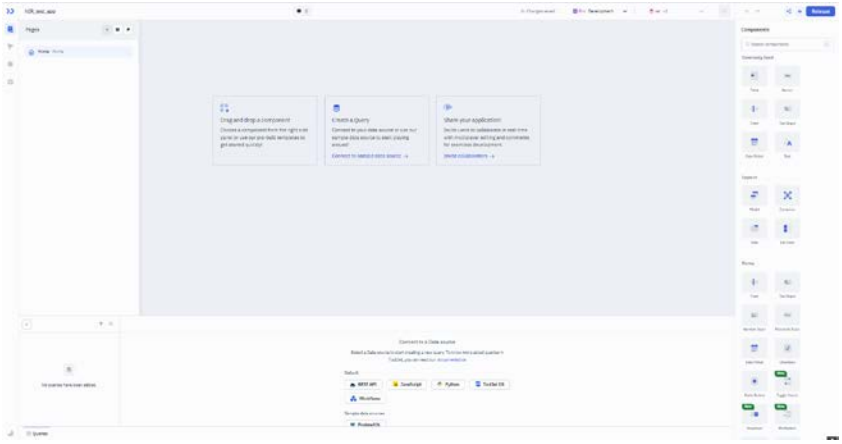

Рис. 1 - Интерфейс системы "ToolJet" (составлено авторами)

Интерфейс предлагает следующие возможности:

– Управление страницами веб-приложения (справа);
– Рабочая область для размещения элементов (по центру);
– Наборы элементов для использования в рабочей области (справа);
– Область работы с данными и скриптами снизу).

Для рассмотрения процесса разработки на Toollet рассмотрим процесс создания приложения, выводящего данные некоторой таблицы клиентов, полученной от созданного ранее API с возможностью добавления новых строк.

Для начала создадим контейнер для размещения элементов. Выполняется данное действие путем перетаскивая компонента «Container» из набора элементов рабочую область. Таким образом выполним и добавления оставшихся элементов. Для реализации описанного функционала потребуется следующие элементы:

– Таблица – для отображения таблицы данных клиентов.
– Кнопка вызова модального окна.
– Модальное окно, содержащее элементы для ввода данных.

Таким образом, построенный интерфейс будет иметь следующий вид:

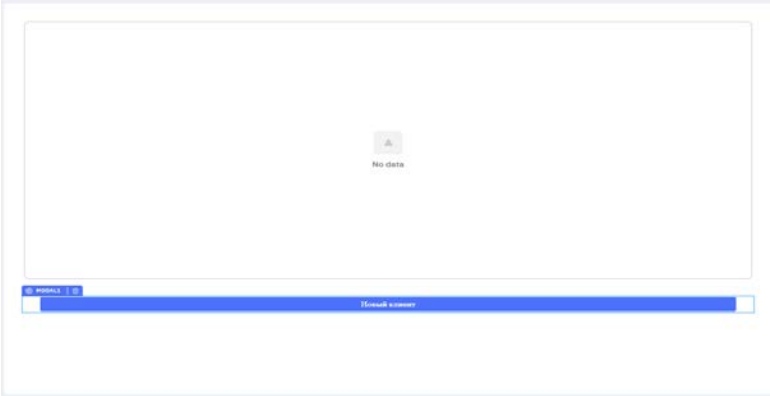

Рис. 2 - Результат построения интерфейса (составлено авторами)

Рис. 3 - Результат построения модального окна (составлено авторами)

На текущем этапе, данная страница не обладает должным функционалом, так как не подключены ресурсы получения данных, но уже обладает кнопками с настроенными слушателями событий по их нажатию, кнопка вызова модального окна отображает на странице соответствующее модальное окно, а кнопка сохранения внутри него, закрывает его. При необходимости каждый элемент можно гибко настроить, например, изменить «плейсхолдеры» и «тултипы». Настройки элемента «кнопка» выглядят следующим образом (рис. 4-5).

На рисунке настроек также продемонстрирован способ взаимодействия компонентов между собой путем их вызова с помощью обращения к нему через код – {{components.textinput1.value}}. Здесь components – это вызов метода обращения к компонентам, textinput1 – идентификатор компонента, value – метод возвращения значения, введенного в компонент. Такая

настройка дает для кнопки «тултип» с динамически изменяемым значением в зависимости от введенного в полях ввода.

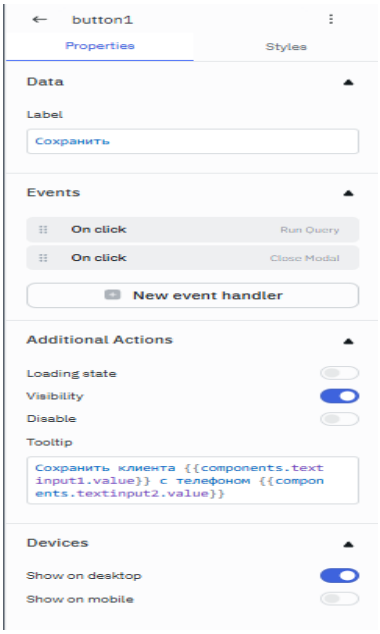

Рис. 4 - Пример окна настройки свойств (составлено авторами)

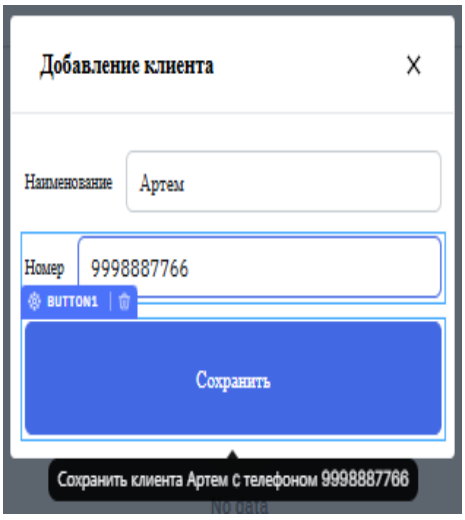

Рис. 5 - Результат настройки свойств (составлено авторами)

Следующим этапом будет добавление ресурсов данных. Как и было описано ранее, необходимая база данных создана на сервере и обратиться к ней можно путем запроса через API. Для реализации описанного функционала

потребуется два запроса, один на получение данных, второй для внесения. Эти запросы имеют следующий URL: http://25.9.255.255:8000/test/customer/get_all_custs и http://25.9.255.255:8000/test/customer/create_customer соответственно. Для реализации их использования создадим файлы запросов типа REST API. Настройка запроса на добавления нового клиента выглядит таким образом

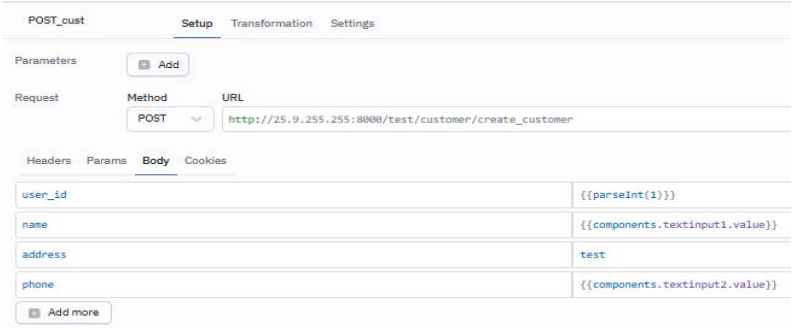

Рис. 6 - Параметры запроса для отправки данных (составлено авторами)

Для отправки в запрос данных используется тот же способ указания источника данных, как и в примере с «тултипом».

Для получения данных, поскольку запрос имеет простую конструкцию GET особых настроек не требуется, достаточно указание URL. Так как данный запрос возвращает данные, их надо отобразить в таблице, в таком случае в настройках компонента таблицы, укажем источник данных в виде этого запроса, также аналогичным образом, но с использованием метода обращения к файлу запроса – queries. Таким образом результатом проделанной работы, занявший значительно более короткий срок, чем создание приложения традиционным методом, будет простое тестовое приложение умеющее выводить данные, полученные по API с возможностью добавления новых данных. Полученный интерфейс продемонстрирован ниже.

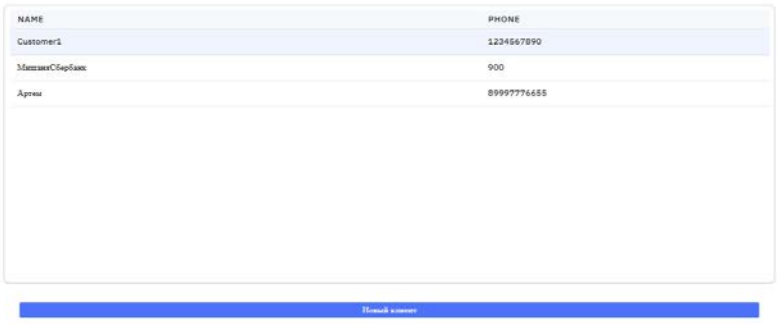

Рис. 7 - Готовый интерфейс приложения (составлено авторами)

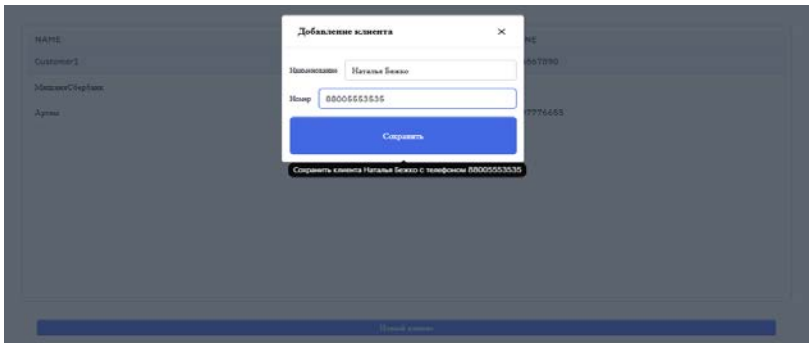

Рис. 8 - Готовый интерфейс модального окна (составлено авторами)

Рис. 9 - Готовый интерфейс приложения после внесения новых данных (составлено авторами)

Подводя итоги, стоит отметить, что использование систем визуального low-code программирования для создания интерфейсов веб-приложения крайне значительно снижает время разработки, сводя его к минимальным затратам времени разработчика, которые он сможет использовать для более качественной и продуманной работы над созданием макетов и стиля, построению карт путей пользователя для получения более приятного и

удобного конечному пользователю интерфейса. Рассмотренное в статье приложение получилось полностью работоспособным и функционал позволяет реализовать поставленную к разработке цель. Таким образом, можно констатировать что использование системы «ToolJet» было полностью оправдано.

Библиографический список:

1. Rokis K., Kirikova M. Challenges of low-code/no-code software development: a literature review // International Conference on Business Informatics Research. — Cham: Springer International Publishing, 2022. — С. 3–17.

2. База знаний ToolJet. Режим доступа: https://docs.tooljet.ai/docs (Дата обращения: 03.02.2025).

3. Вагин Д.В., Петров Р.В. Современные технологии разработки веб-приложений. — 2019.

4. Майкл Хагенбауэр. The No-Code Handbook: How to Build Web Apps without Coding. — Packt Publishing Ltd, 2022.

5. Рэми Шарма. No-Code Websites: Design & Develop Beautiful Sites with No Programming Skills. — O’Reilly Media Inc, 2023.

6. Тельо-Родригес М. и др. Путеводитель по проектированию удобных Web-API // Труды Института системного программирования РАН. — 2021. — Т. 33. — № 1. — С. 173–188.

7. Шредер П. Low Code Development Platforms: A Practical Guide to Building Applications Without Writing Code. — APress, 2021.

Оригинальность 77%