МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

РЕАЛИЗАЦИЯ ПОЛЬЗОВАТЕСЛЬСОКОГО ЧАТА В КАЧЕСТВЕ ПОВТОРОНО ИСПОЛЬЗУЕМОГО ЭЛЕМЕНТА ПАКЕТА NPM

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 273 группы
направления 02.03.04 Математическое обеспечение и администрирование информационных систем
факультета компьютерных наук и информационных технологий
Шкабуры Никиты Дмитриевича

Научный руководитель:
старший преподаватель _____ Е. Е. Лапшева
подпись, дата

Зав. кафедрой:
к. ф.-м. н., доцент _____ М. В. Огнева
подпись, дата

Саратов 2025

ВВЕДЕНИЕ

Актуальность темы.

Современная веб-разработка требует создания модульных, переиспользуемых и масштабируемых компонентов пользовательского интерфейса. Одним из таких компонентов является чат, который используется в широком спектре веб-приложений: от корпоративных платформ и CRM-систем до образовательных порталов и сервисов клиентской поддержки. При этом значительная часть усилий разработчиков уходит на повторную реализацию одного и того же функционала, что снижает производительность и увеличивает технический долг.

Одним из решений этой проблемы является разработка NPM-библиотек с переиспользуемыми UI-компонентами. Такой подход позволяет создать стандартизированный компонент, легко интегрируемый в разные проекты с минимальной настройкой. Это особенно актуально в условиях работы над несколькими проектами параллельно или при необходимости быстрого масштабирования решений.

Цель магистерской работы заключается в разработке и предоставлении инструмента, который позволит упростить и оптимизировать процесс разработки WEB приложений, на примере реализации чата в качестве переиспользуемого элемента npm библиотеки.

Поставленная цель определила **следующие задачи**:

1. Изучение существующих npm пакетов и инструментов для автоматизации разработки Front-end проектов.
2. Анализ потребностей и требований WEB разработчиков. выяснить основные задачи и проблемы, с которыми они сталкиваются в процессе работы.
3. Проектирование и разработка npm пакета.
4. Реализация чата в качестве преиспользуемого компонента
5. Тестирование и отладка чата и пакета в целом.
6. Документирование и демонстрация пакета.

7. Оценка эффективности и применимости пакета.

Методологические основы.

Работа опирается на современные подходы к проектированию архитектуры фронтенд-приложений, включая Domain-Driven Design, Micro Frontends и Feature Slice Design, а также на исследования в области разработки NPM-библиотек, Vue.js и технологий управления состоянием.

Практическая значимость бакалаврской работы.

Разработанный компонент чата может быть использован в различных веб-приложениях, обеспечивая разработчикам возможность быстрой и стандартизированной интеграции функционала общения, что существенно снижает временные и трудозатраты при реализации данного интерфейса.

Структура и объём работы.

Магистерская работа состоит из введения, 4 разделов, заключения, списка использованных источников и 4 приложений. Общий объем работы – 74 страницы, из них 50 страниц – основное содержание, включая 7 рисунков, список использованных источников информации – 27 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор существующих инструментов для автоматизации разработки Front-end проектов» посвящен обзору существующих решений для упрощения и автоматизации процесса разработки WEB-приложений. Такие инструменты позволяют сместить фокус с рутинной и монотонной работы на более сложные задачи. Их использование особенно актуально в рамках бурно растущих проектов либо при наличии нескольких параллельно развивающихся продуктов, в которых важная такая вещь как их единообразие.

Проведен обзор популярных UI-фреймворков и библиотек компонентов: Element Plus, Bootstrap, Buefy и других. Рассмотрены преимущества и недостатки каждого из решений с точки зрения гибкости, адаптивности и возможности модификации.

Основные преимущества таких инструментов – легкость в использовании, которая заключается в возможности использования готовых компонентов без необходимости писать код с нуля, что позволяет быстро создавать красивые интерфейсы. А также расширяемость, которая заключается в возможности создавать собственные компоненты на основе уже представленных.

Основные недостатки большинства решений – сильная зависимость от всей экосистемы или высокая стоимость внедрения и слишком большой размер файлов, что может замедлить загрузку сайта. Это подтверждает необходимость создания собственного переиспользуемого решения в рамках независимого пакета.

Второй раздел «Основные инструменты, используемые во Front-end разработке» посвящен изучению вопроса выбора существующих технологий и инструментов, используемых для создания современных WEB-приложений. В нем проводится сравнительный анализ фреймворков, языков программирования, инструментов, применяемых для стилизации проекта. Для каждого инструмента разбираются его преимущества и недостатки.

Основным фреймворком выбран Vue.js, благодаря его легковесности, простоте интеграции и активному сообществу. В качестве языка разработки использован Type Script, что позволило повысить безопасность и читаемость кода.

Для стилизации интерфейса выбран SCSS, обеспечивающий гибкость и поддержку переменных, наследованию, вложенности и миксинов. Управление состоянием внутри фреймворка реализовано с помощью Pinia – современной легковесной альтернативы базового стейт-менеджера Vuex.

Также использована технология SignalR для реализации связи с сервером в режиме реального времени, что обеспечило мгновенный обмен сообщениями и файлами между пользователями.

Третий раздел «Методология разработки архитектуры приложения» посвящен разбору архитектурных подходов разработки WEB-приложений, так как это играет ключевую роль на всех этапах жизненного цикла программного обеспечения. В данном разделе разбираются такие методологии Domain-Driven Design, Micro Frontends и Feature Slice Design, а именно рассматриваются их подходы и недостатки.

Для обеспечения модульности и масштабируемости проекта выбрана архитектура Feature Slice Design (FSD), предусматривающая деление приложения на независимые "срезы", каждый из которых включает компоненты UI, состояния, логики и API. Это позволило создать изолированные области функционала, облегчило тестирование и скоординировало командную работу, а также повысило уровень понимания переиспользуемости компонентов.

Освещается ряд проблем, с которыми может столкнуться команда разработки проекта, при переходе на какой-либо новый архитектурный подход. Как решение этих проблем предлагается реализовать атематическую проверку архитектуры проекта в рамках работы pre-commit хуков. Для автоматической валидации соблюдения архитектурных границ реализованного проекта использован инструмент dependency-cruiser. С его

помощью реализована система в виде пользовательских правил, предотвращающая некорректные связи между слоями и фичами, что значительно снизило вероятность архитектурных ошибок.

Четвертый раздел «Создание пользовательского чата в качестве переиспользуемого элемента npm библиотеки» посвящен реализации WEB-приложения с функционалом чата, вынесенного в пакет NPM для его дальнейшей повторной используемости в разных проектах.

Базовые функции:

- Возможность выбора диалога из предоставленного списка;
- Возможность отправки и получения текстовых сообщений и файлов;
- Наличие индикатора новых сообщений на кнопке чата;
- Наличие каунтера с количеством непрочитанных сообщений у каждого диалога в списке диалогов;
- Возможность скрыть чат при необходимости;

Дополнительные функции:

- Возможность просмотра всех полученных и отправленных файлов в переписке;
- Возможность поиска сообщений в диалоге;
- Возможность моментально переместиться в конец диалога;
- Возможность «перетаскивания» чата в любое удобное место на странице.

Практическое взаимодействие клиент – сервер в данном продукте, реализовано при помощи технологии SignalR. Данный инструмент используется для поддержки прямой связи с сервером. Поэтому вся информация в чате обновляется в режиме реального времени.

Таким образом, при отображении списка диалогов, с сервера приходит список в виде массива объектов со всей необходимой информацией, такой как превью последнего сообщения, заголовок диалога, время и дата последнего сообщения. Данный массив заносится в сетйт внутри Pinia, после чего посредством перебора отображается в верстке приложения.

Отображение сообщений и диалогов реализовано на основе реактивных данных из Pinia, а визуальные элементы адаптированы под различные размеры экранов. Асинхронные запросы реализованы с помощью Axios.

В активном диалоге, у пользователя реализована возможность отправки и получения текстовых сообщений и файлов. Обмен данными с собеседником происходит в режиме реального времени благодаря технологии SignalR.

Текстовые сообщения вводятся в специальное поле, затем для его отправки необходимо нажать на специальную кнопку справа от поля или нажав на клавишу Enter. Файлы можно прикрепить, нажав на кнопку в виде скрепки слева от поля ввода и выбрать нужный файл в проводнике, или же его можно перетащить на диалог при помощи технологии DrugAndDrop. Файлы из диалога можно скачать на компьютер, нажав на кнопку, которая появляется при наведении на нужный файл.

В активном диалоге, у пользователя помимо базового функционала, по типу отправки и получения сообщений, был реализован ряд дополнительных возможностей. В шапке диалога присутствуют три интерактивные кнопки. Благодаря ним пользователь имеет возможность вернуться к списку диалогов, скрыть чат, и открыть меню с кнопками по поиску сообщений в чате и просмотру всех файлов в диалоге. Если пользователь пролистает чат достаточно высоко, то у него появится кнопка для возвращения в конец диалога, а также если в этот момент в чате появятся новые непрочитанные сообщения, то на этой кнопке отобразится их количество.

Создана документация с использованием VitePress. В ней представлены:

- Структура пакета;
- Инструкции по установке и использованию;
- Список свойств и методов компонента;
- Примеры кода с различными вариантами интеграции.

Компонент опубликован в публичном реестре npm и доступен для установки через команду `npm install`. Также проведена интеграция с CI/CD для автоматической проверки и публикации новых версий пакета.

Разработанный компонент позволяет:

- Сократить сроки внедрения функционала чата в новые проекты;
- Обеспечить единый стандарт качества и архитектуры интерфейсов;
- Упростить сопровождение и развитие UI-решений в крупных системах.

Компонент уже внедрен в реальные проекты, где используется в веб-приложениях, и показал себя как стабильное и удобное решение.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы, были изучены и применены на практике принципы и методики создания npm пакета для применения его в процессе разработки WEB-приложения. Были подробно изучены возможности языков программирования JavaScript и Type Script, его фреймворки и библиотеки, также рассмотрены их преимущества и недостатки. Выбран фреймворк Vue.js основным инструментом создания WEB-приложения, по причине удобства, производительности и его легковесности. Были изучены такие технологии как HTML и CSS, SASS, SCSS и Styled Components. А также технологии Element-Plus, Bootstrap и Buefy.

Была выбрана FSD архитектура, и реализована автоматическая проверка корректности ее построения в проекте при помощи пользовательских сценариев, прописанных в рамках библиотеки dependency-cruiser.

На основе полученных знаний был реализован пользовательский чат, в качестве переиспользуемого компонента внутри библиотеки npm. Результаты проделанной работы были представлены на студенческой научной конференции СГУ. По итогам факультетского этапа, работа заняла 3 место.

Основные источники информации:

1. Халиманенков А.С. Автоматизация и оптимизация веб-сайтов с помощью системы сборки задач gulp // Постулат, Т. 8, 2021. С. 1-18.
2. Колесников Л.А., Пименов А.А. Повышение скорости разработки веб-сайтов с использованием фреймворка bootstrap // Инженерные и информационные технологии, экономика и менеджмент в промышленности. 2020. pp. 229-231.
3. Flanagan D. JavaScript: The Definitive Guide. Sebastopol: O'Reilly Media, 2020.

4. Ribeiro H. Vue.js 3 Cookbook: Practical recipes to help you build modern frontend web apps with the latest Vue.js and TypeScript. Birmingham: Packt Publishing Ltd, 2020.
5. Официальная документация Pinia [Электронный ресурс] URL: https://pinia.vuejs.org/ (дата обращения: 17.09.2024).
6. Weinman C., Ogoartua J. Front-End Web Development: The Big Nerd Ranch Guide. Atlanta; Georgia: Big Nerd Ranch, 2017.
7. Grant K.J. CSS in Depth. New York: Manning Publications, 2018.
8. Shivam S. NPM: The Node Package Manager - A Comprehensive Guide. [Электронный ресурс] [2023]. URL: https://dev.to/martinpersson/create-and-publish-your-first-npm-package-a-comprehensive-guide-3l0a (дата обращения: 03.12.2023).