Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

Пермский национальный исследовательский политехнический университет

(ПНИПУ)

Факультет: Электротехнический (ЭТФ)

Направление: 09.03.04 – Программная инженерия (ПИ)

Профиль: Разработка программно-информационных систем (РИС)

Кафедра информационных технологий и автоматизированных систем (ИТАС)

Допустить к защите

Зав. кафедрой ИТАС: д-р экон. наук, проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.А. Файзрахманов

«\_\_\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

на соискание академической степени бакалавра

на тему

**«**Проектирование архитектуры и разработка универсального программного обеспечения для формирования системы опросов на клиентских сайтах компании**»**

|  |  |  |
| --- | --- | --- |
| Студент: |  | Селиверстов Евгений Владимирович |
|  | (подпись, дата) |  |
| Группа: | РИС-17-1б | |

Состав ВКР:

1. Пояснительная записка на 81 стр.
2. Приложения на 15 стр.

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | к.т.н., доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Консультант по предметной области: |  | генеральный директор компании ООО «Марквиз» Г.А. Балачков |
| (подпись, дата) |
| Прохождение предзащиты: |  | к.т.н., доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Прохождение нормоконтроля: |  | заведующий лабораториями  Д.А. Карлов |
| (подпись, дата) |
| Прохождение проверки на объем заимствований: |  | инженер  Е.В. Ерискина |
|  | (подпись, дата) |  |

Пермь — 2021Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

Пермский национальный исследовательский политехнический университет

(ПНИПУ)

Факультет: Электротехнический (ЭТФ)

Направление: 09.03.04 – Программная инженерия (ПИ)

Профиль: Разработка программно-информационных систем (РИС)

Кафедра информационных технологий и автоматизированных систем (ИТАС)

УТВЕРЖДАЮ

Зав. кафедрой ИТАС: д-р экон. наук, проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.А. Файзрахманов

«\_\_\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**ЗАДАНИЕ**

на выполнение выпускной квалификационной работы бакалавра

Фамилия, имя, отчество: Селиверстов Евгений Владимирович

Группа: РИС-17-1б

Начало выполнения работы: 01.02.21

Контрольные сроки просмотра работы кафедрой:

1) 09.04.21, 2) 23.04.21, 3) 14.05.21

Защита работы на заседании ГЭК: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Наименование темы: «Проектирование архитектуры и разработка универсального программного обеспечения для формирования системы опросов на клиентских сайтах компании»

2. Исходные данные к работе: задание на разработку программного обеспечения для формирования системы опросов на клиентских сайтах компании «Marquiz»

3. Содержание пояснительной записки:

а) исследовательский раздел.

Описание сервиса Marquiz, исследование существующей системы формирования опросов и существующего алгоритма добавление опросов на сайт, описание требований для создания новой системы и нового представления опроса в виде widget

б) конструкторский раздел:

Проектирование архитектуры новой системы формирования опросов, алгоритма добавление опросов на сайт, клиентской части нового представления опроса в виде widget, описание средств разработки

в) технологический раздел:

Разработка спроектированной системы, клиентской части нового представления опроса в виде widget, настройка конфигурации Webpack, тестирование разработанной системы

г) социально-экономическая эффективность

Технико-экономическое обоснование проекта, определение капитальных вложений, расчет экономического эффекта

д) информационная безопасность и надежность

Описание используемых средств защиты информации

4. Дополнительные указания:

Дополнительных указаний нет

5. Основная литература:

а) Применение микросервисной архитектуры: плюсы, минусы, подводные камни. [Электронный ресурс] URL: <https://simpleone.ru/blog/primenenie-mikroservisnoj-arhitektury-plyusy-minusy-podvodnye-kamni/> (дата обращения 16.05.21)

б) Рендеринг веб-сайтов 101. [Электронный ресурс] URL: <https://habr.com/ru/post/484900/> (дата обращения 16.05.21)

в) Асинхронная загрузка javascript файлов. Ускорение и оптимизация процесса, увеличение производительности [Электронный ресурс] URL: <https://habr.com/ru/post/56974/> (дата обращения 17.05.21)

г) Concepts | webpack [Электронный ресурс] URL: <https://webpack.js.org/concepts/> (дата обращения 01.06.21)

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | к.т.н., доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Консультант по предметной области: |  | генеральный директор компании ООО «Марквиз» Г.А. Балачков |
| (подпись, дата) |
| Задание получил: |  | Селиверстов Евгений Владимирович |
| (подпись, дата) |

КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ

ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование этапа | Объем (в%) | Начало | Конец |
| 1. Сбор и анализ исходных данных, постановка задачи | 11 | 01.02.21 | 12.02.21 |
| 1. Изучение теоретического материала по предметной области; анализ и выбор методов и средств решения задачи | 24 | 15.02.21 | 12.03.21 |
| 1. Разработка теоретической части, методики решения; Выбор и разработка средств решения задачи | 22 | 15.03.21 | 09.04.21 |
| 1. Тестирование разработки, постановка экспериментов, формулировка выводов | 24 | 12.04.21 | 07.05.21 |
| 1. Оформление пояснительной записки | 19 | 10.05.21 | 31.05.21 |
| 1. Представление работы на проверку и отзыв руководителя квалификационной работы | - | 14.06.21 | 14.06.21 |
| 1. Предварительная защита работы | - | 18.06.21 | 18.06.21 |
| 1. Прохождение нормоконтроля | - |  |  |
| 1. Представление работы на кафедру | - |  |  |
| 1. Защита на заседании ГЭК | - |  |  |

|  |  |  |
| --- | --- | --- |
| Руководитель ВКР: |  | к.т.н., доцент каф. ИТАС  О.А. Полякова |
| (подпись, дата) |
| Задание получил: |  | Селиверстов Евгений Владимирович |
| (подпись, дата) |

# РЕФЕРАТ

Выпускная квалификационная работа содержит пояснительную записку на 81 странице формата А4, включающую 5 разделов, 31 рисунок, 2 таблицы, 9 источников, 3 приложения.

ОПРОС, ФОРМИРОВАНИЕ, СИСТЕМА, WIDGET, САЙТ, СТРАНИЦА, КОНСТРУКТОР.

Объектом исследования в рамках данной работы является система разработки формирования опросов на клиентских сайтах.

Целью данной работы является проектирование архитектуры иразработка универсального программного обеспечения для формирования опросов на сайтах клиентов компании.

В результате данной работы проведено исследование существующей системы формирования опросов, выявлены её достоинства и недостатки, разработана архитектура и программное обеспечение новой системы. Разработано новое представление опросов в виде widget, а также проведен расчет экономических показателей внедрения.

СОДЕРЖАНИЕ

[ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ 8](#_Toc74694318)

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ 9](#_Toc74694319)

[ВВЕДЕНИЕ 10](#_Toc74694320)

[1. Исследовательский раздел 12](#_Toc74694321)

[1.1. Описание микросервисной архитектуры приложений 12](#_Toc74694322)

[1.2. Описание структуры приложения системы создания опросов компании «Marquiz» 13](#_Toc74694323)

[1.2.1. Описание элементов конструктора опросов 14](#_Toc74694324)

[1.2.2. Описание содержания приложения Quiz 18](#_Toc74694325)

[1.2.3. Описание системы формирования опросов 21](#_Toc74694326)

[1.2.4. Описание серверной части приложения 21](#_Toc74694327)

[1.2.5. Описание серверной части модуля аналитики 22](#_Toc74694328)

[1.2.6. Описание сервиса загрузки файлов пользователей 22](#_Toc74694329)

[1.2.7. Вывод 22](#_Toc74694330)

[1.3. Исследование процесса отрисовки сайта 23](#_Toc74694331)

[1.4. Исследование существующей системы формирования опросов 24](#_Toc74694332)

[1.5. Исследование способов асинхронной загрузки приложений 27](#_Toc74694333)

[1.6. Разработка требований к новой системе формирования опросов 28](#_Toc74694334)

[1.7. Описание требований к новому представлению опроса 29](#_Toc74694335)

[1.7.1. Дизайн опроса в виде widget 30](#_Toc74694336)

[1.7.2. Дизайн интерфейса настроек нового представления в конструкторе опросов 32](#_Toc74694337)

[1.8. Вывод по разделу 33](#_Toc74694338)

[2. Конструкторский раздел 34](#_Toc74694339)

[2.1. Описание модели «AS-IS» 34](#_Toc74694340)

[2.2. Описание модели «TO-BE» 36](#_Toc74694341)

[2.3. Проектирование клиентской части нового представления опроса в виде widget 39](#_Toc74694342)

[2.4. Проектирование диаграммы классов 40](#_Toc74694343)

[2.5. Средства разработки 43](#_Toc74694344)

[2.6. Вывод по разделу 43](#_Toc74694345)

[3. Технологический раздел 45](#_Toc74694346)

[3.1. Разработка новой системы формирования опросов 45](#_Toc74694347)

[3.1.1. Реализация классов новой системы формирования опросов 45](#_Toc74694348)

[3.1.2. Новый код инициализации системы и добавления опроса 58](#_Toc74694349)

[3.2. Разработка нового способа формирования опросов в виде widget 60](#_Toc74694350)

[3.2.1. Реализация интерфейса в приложении Quiz 60](#_Toc74694351)

[3.2.2. Реализация интерфейса в конструкторе опросов 66](#_Toc74694352)

[3.2.3. Реализация класса Widget в системе формирования опросов 68](#_Toc74694353)

[3.3. Настройка Wepback для поддержки старой и новой версий системы 73](#_Toc74694354)

[3.4. Тестирование разработанной системы 74](#_Toc74694355)

[3.5. Вывод по разделу 78](#_Toc74694356)

[4. Социально-экономическая эффективность 79](#_Toc74694357)

[4.1. Технико-экономическое обоснование проекта 79](#_Toc74694358)

[4.2. Определение капитальных вложений 80](#_Toc74694359)

[4.3. Расчет экономического эффекта 81](#_Toc74694360)

[5. Информационная безопасность и надежность 83](#_Toc74694361)

[ЗАКЛЮЧЕНИЕ 84](#_Toc74694362)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 85](#_Toc74694363)

[ПРИЛОЖЕНИЕ А 86](#_Toc74694364)

[ПРИЛОЖЕНИЕ Б 94](#_Toc74694365)

[ПРИЛОЖЕНИЕ В 96](#_Toc74694366)

# ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

Quiz – это короткий online опрос, в конце которого показывается результат или форма заявки потенциального клиента.

Widget – встраиваемое стороннее приложение на сайте, представленное чаще всего в виде небольшого блока в углу экрана.

HTML-страница – документ или информационный интернет ресурс, доступ к которому осуществляется с помощью веб-браузера.

Tag – структурный элемент языка разметки гипертекста.

Атрибуты – это специальные слова, используемые внутри tag для управления поведением HTML-элемента.

Framework – программная платформа, которая упрощает создание и поддержку технически сложных или нагруженных проектов.

AJAX – набор методов веб-разработки для асинхронной обработки запросов к серверу в фоновом режиме.

SPA-приложение – одностраничное приложение, в котором используется единственный HTML-документ как оболочка для всех веб-страниц, и которое организует взаимодействие с пользователем через динамически подгружаемые файлы с помощью технологии AJAX.

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

JavaScript – язык программирования, который используется в браузерах как язык сценариев для обеспечения интерактивности веб-страниц.

Vue – JavaScript framework с открытым исходным кодом для создания пользовательских интерфейсов.

API – способы взаимодействия одной программы с другой (набор классов, процедур, функций, структур или констант).

Node.js – программная платформа для трансляции JavaScript в машинный код.

Loopback – это высоко расширяемый Node.js framework с открытым исходным кодом.

MongoDB – документоориентированная система управления базами данных.

Webpack – сборщик модулей JavaScript с открытым исходных кодом.

Laravel – бесплатный framework с открытым исходным кодом, разработанный на PHP.

DOM – программный интерфейс для получения доступа к HTML, XHTML и XML-документам.

# ВВЕДЕНИЕ

Одним из современных способов для привлечения клиентов к своему продукту или услуге в маркетинге является quiz. Данный способ представляет собой короткий online опрос, в конце которого формируется результат опроса пользователя, или специальная форма с контактными данными для обратной связи с ним.

Маркетинговые опросы используются интернет-маркетологами и предпринимателями для увеличения объема заказов, снижения затрат на рекламу для привлечения клиентов и повышения интереса потенциального клиента к товару или услуге.

Технологии проведения опросов используются в образовательных целях, как способ обновления базовых знаний и тестирования, для проведения социологических исследованиях.

В компании «Marquiz» разработан конструктор, где зарегистрированные пользователи за несколько минут могут создавать содержательный контент собственных опросов с использованием готовых шаблонов. Затем созданный опрос размещается в сети Интернет по прямой ссылке, или на собственном сайте пользователя, или в группе во ВКонтакте.

Статистические исследования компании показали, что самым эффективным способом является формирование специальной системой созданных опросов на собственном сайте пользователя, предоставленной компанией «Marquiz». Данная система является встраиваемым приложением, которое добавляется пользователем на собственный сайт.

Со временем, практически в любой системе появляется необходимость добавления нового функционала. Так, и в компании «Marquiz» появилась необходимость добавления нового способа представления опроса на сайтах интернет-магазинов, которые предпочитают сегодня более современную и компактную визуализацию, в виде widget.

В существующей системе формирования опросов не учтена возможность внедрения нового функционала, поэтому решение задачи внедрения нового формата отображения опроса в виде widget не представляется возможным в существующей системе.

Именно поэтому разработан проект, выполнены программная реализация и внедрение новой системы формирования опросов на сайтах пользователей, которая позволяет быстро, удобно и эффективно добавлять новый функционал.

Актуальностьпроектирования и разработки такой системы формирования опросов на сайтах подтверждена многочисленными заявками владельцев интернет-магазинов.

Целью данной работы является проектирование архитектуры иразработка универсального программного обеспечения для формирования опросов на сайтах клиентов компании «Marquiz».

Объектом исследованияи разработки является система формирования опросов на клиентских сайтах.

Для достижения цели необходимо выполнить следующие задачи:

* исследовать существующую систему формирования опросов, выявить её достоинства и недостатки;
* разработать проект архитектуры новой системы;
* разработать программное обеспечение спроектированной системы для формирования опросов;
* разработать новое представление опроса в виде widget;
* провести расчет экономических показателей внедренной системы.

# Исследовательский раздел

## Описание микросервисной архитектуры приложений

В настоящее время многие веб-приложения разрабатываются на основе микросервисной архитектуры, которая структурирует приложение, как набор слабосвязанных сервисов, взаимодействующих друг с другом. Данная архитектура является полной противоположностью монолитной архитектуры, которая воспринимает приложение как единое целое, где вся логика находится внутри одного модуля [1].

Достоинства микросервисной архитектуры:

* масштабируемость. Любой из сервисов можно дополнить и расширить, когда появится такая необходимость, не затрагивая остальные сервисы. Вся система, при этом, остается прежней;
* высокая отказоустойчивость. При появлении критических ошибок в одном из сервисов, все остальные продолжают работать. Таким образом, неполадки в отдельных сервисах не помешают всему рабочему процессу;
* гибкость. Появляется возможность внедрения новой технологии на отдельный микросервис, а не на всю систему сразу, что происходит значительно быстрее. В случае неудачи пострадает только один сервис, и восстановить систему будет проще;
* простота разработки. Чем меньше кода в приложении, которое разрабатывается несколькими программистами, тем им проще и быстрее разобраться в любом участке кода. Каждый отдельный сервис представляет собой целостную систему, именно поэтому нет необходимости разбираться в огромном количестве деталей, не касающихся конкретной поставленной задачи;
* возможность параллельного внедрения. Процесс внедрения нового функционала в работающую программу можно проводить параллельно для нескольких микросервисов [2].

Минусы микросервисной архитектуры:

* cложность построения коммуникации между сервисами. С увеличением количества микросервисов появляется сложность в построении их взаимодействия друг с другом;
* cложность тестирования. Необходимо разбираться отдельно с каждым сервисом, а затем тестировать корректное взаимодействие его с другими микросервисами [2].

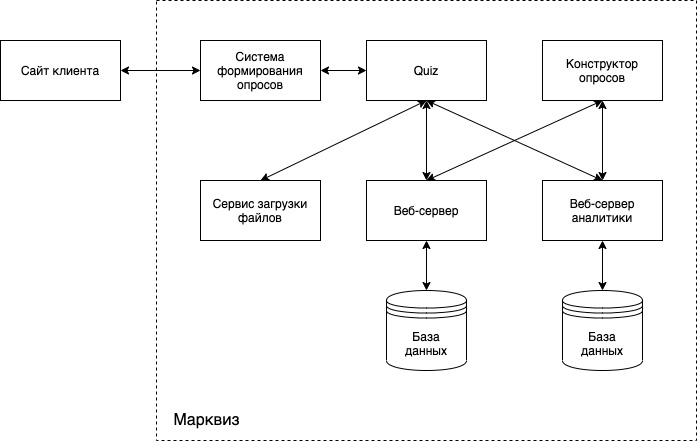
Компания «Marquiz» использует микросервисную архитектуру разработки всей системы создания опросов, поскольку она обеспечивает модульность, независимое развертывание, простое и быстрое добавление нового функционала.

## Описание структуры приложения системы создания опросов компании «Marquiz»

Структура системы создания опросов компании «Marquiz» состоит из 6 взаимосвязанный приложений:

* конструктор опросов, в котором происходит его редактирование;
* приложение Quiz, которое отображает созданный опрос;
* система формирования опросов, созданная для размещения опросов на сайты клиентов компании;
* веб-сервер. Серверная часть приложения, которая обрабатывает запросы с клиентских приложений компании;
* веб-сервер аналитики. Серверная часть модуля аналитики;
* сервис загрузки файлов пользователей в Quiz.

Структура системы создания опросов компании «Marquiz» представлена на рисунке 1.

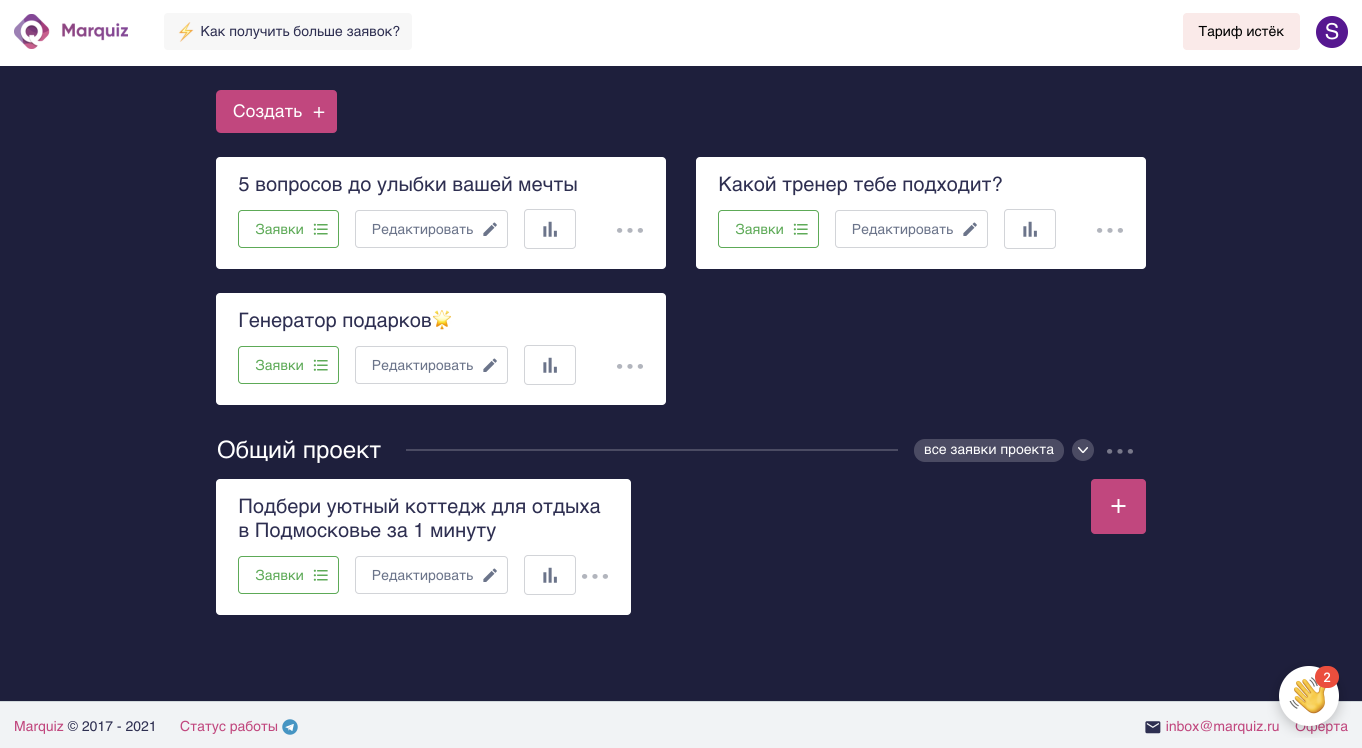
Рисунок 1 – Структура системы создания опросов

### Описание элементов конструктора опросов

Конструктор опросов представляет собой клиентское web-приложение, где любой зарегистрировавшийся пользователь может создавать свои собственные опросы, изменять их внешний вид, добавлять вопросы, бонусы за прохождение и другие различные улучшения.

После авторизации пользователь попадает на страницу со всеми созданными опросами. На данной странице он может создать или удалить опрос, перейти к его редактированию или к просмотру заявок и аналитики. Также пользователь имеет возможность создать проект, которому можно предоставить доступ для других пользователей сервиса.

Интерфейс страницы с опросами представлен на рисунке 2.

Рисунок 2 – Интерфейс страницы с опросами

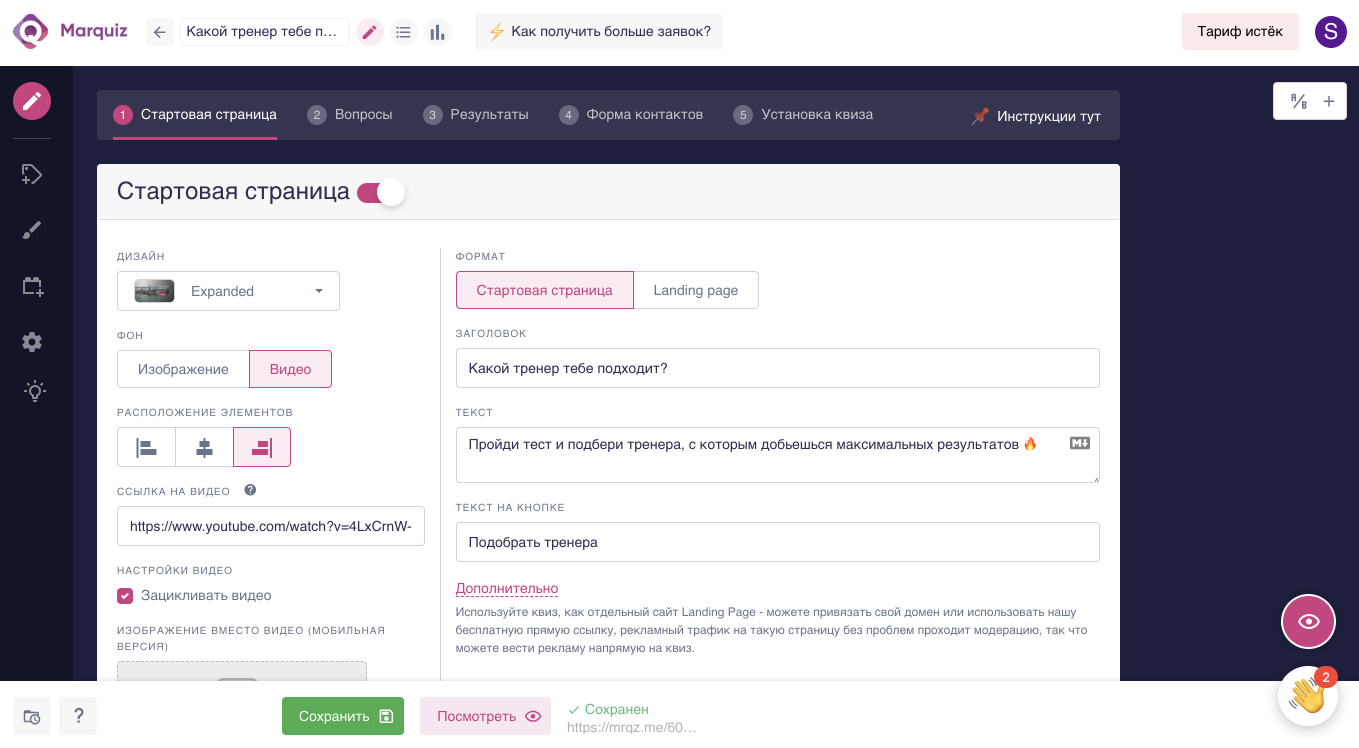
Редактирование опроса начинается с настройки стартовой страницы, которую видит пользователь, проходящий опрос. В данном разделе есть возможность выбрать три различных вида отображения стартовой страницы: расширенная, по центру, стандартная. Также можно установить картинку или видео на фон, изменить заголовок и текст, добавить контактные данные и логотип компании, если они имеются. Интерфейс редактирования стартовой страницы представлен на рисунке 3.

Рисунок 3 – Интерфейс редактирования стартовой страницы

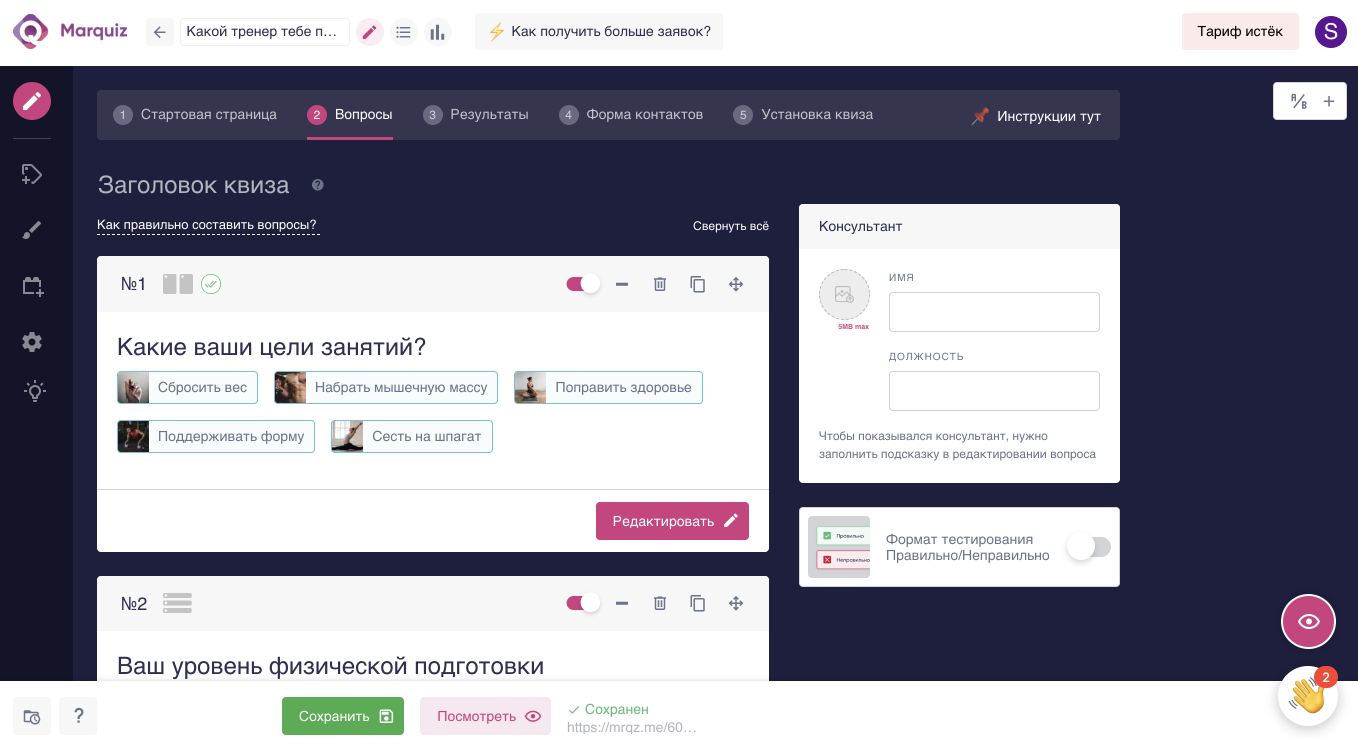
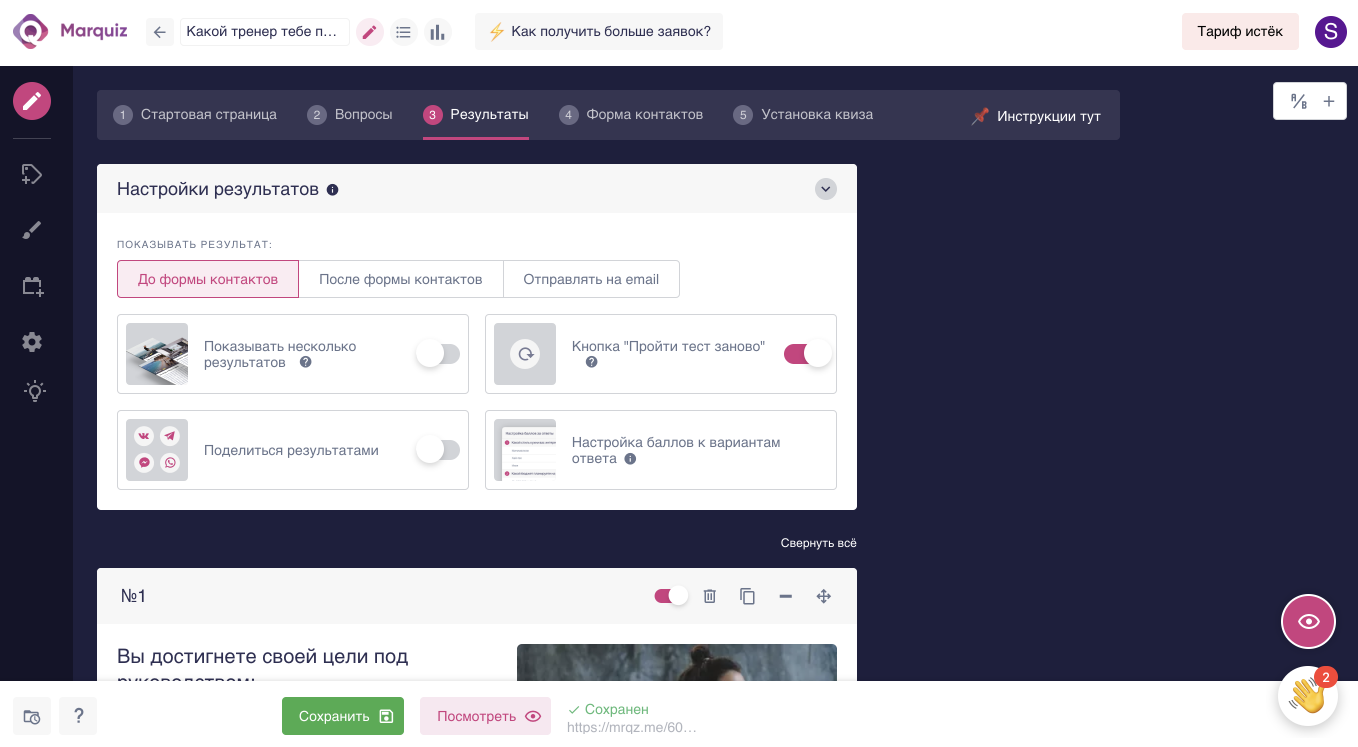
Вторым шагом идет настройка вопросов, где пользователь может их добавлять, удалять и редактировать. Кроме этого, можно добавить условия отображения отдельных вопросов, добавить заголовок опроса, консультанта, и настроить формат тестирования, когда в опросе проверяется правильность ответов. Интерфейс настройки вопросов представлен на рисунке 4.

Рисунок 4 – Интерфейс настройки вопросов

Третьим шагом идет настройка результатов опроса. Результат является предпоследним или последним шагом в опросе и представляет собой небольшую страницу с описанием товара или услуги, картинкой или видео и ценой, если товар или услуга её имеют. На данном шаге можно добавить несколько результатов и настроить их отображение в зависимости от ответов на опрос. Можно выбрать, когда добавленные результаты будут показаны пользователю: перед формой контактов, после формы контактов или будут отправлены на указанную почту.

Интерфейс настройки результатов представлен на рисунке 5.

Рисунок 5 – Интерфейс настройки результатов

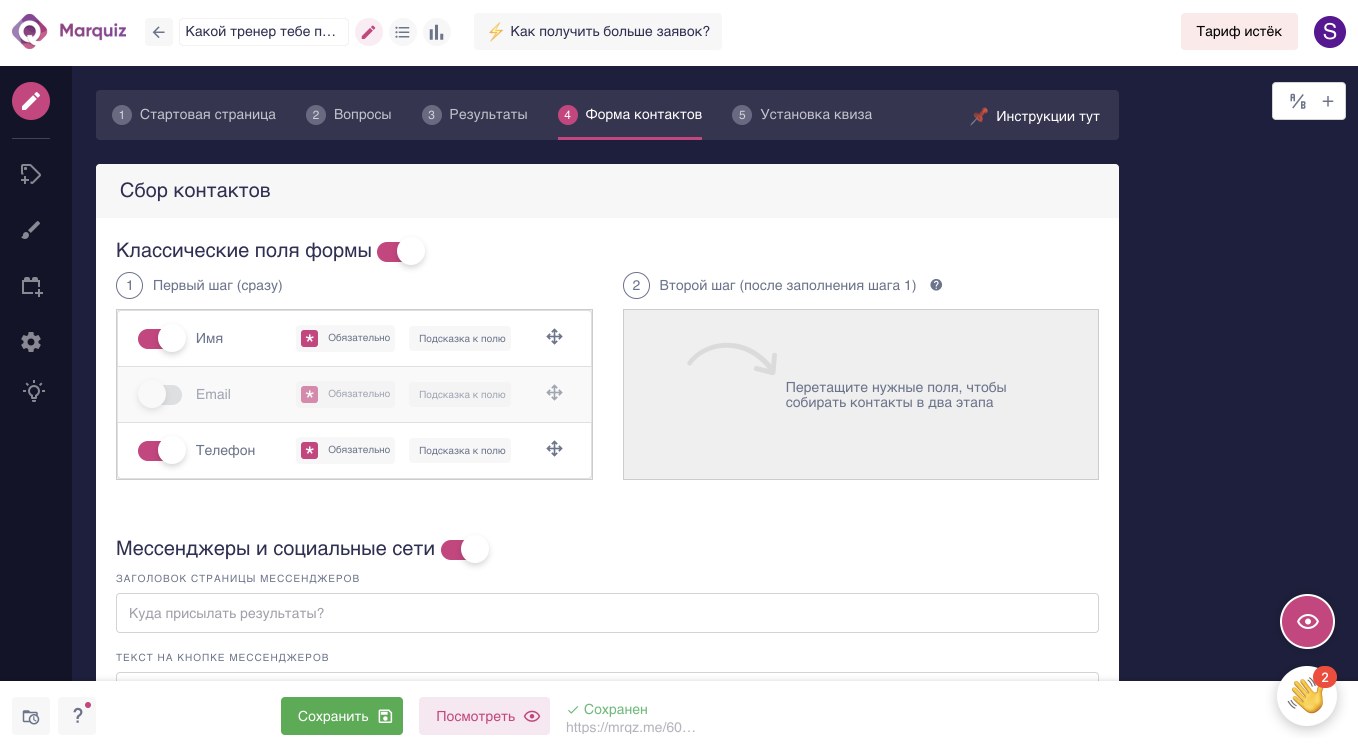
Следующим шагом идет настройка формы контактов. Здесь пользователь сервиса может настроить, какие данные необходимо будет оставить пользователю, проходящему опрос. Также можно добавить мессенджеры и социальные сети, настроить форму в два шага, и добавить редирект на другую страницу после того, как пользователь введет свои контактные данные. Интерфейс настройки формы контактов представлен на рисунке 6.

Рисунок 6 – Интерфейс настройки формы контактов

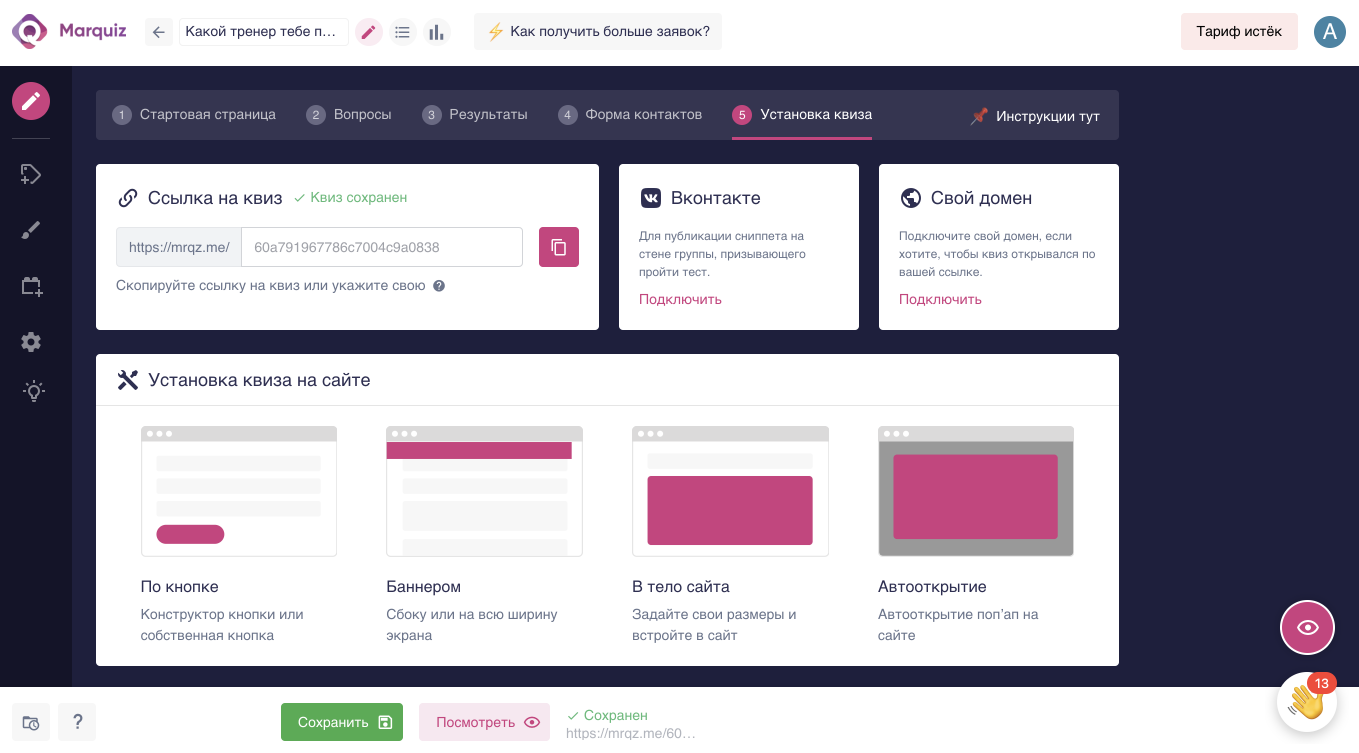
Последним шагом настройки опроса является его размещение в сети Интернет. Созданный опрос можно использовать по автоматически созданной ссылке, создать свою уникальную ссылку, привязать к уже имеющемуся домену, добавить в группу во Вконтакте, или установить на свой сайт. Интерфейс выбора установки опроса представлен на рисунке 7.

Рисунок 7 – Интерфейс выбора установки опроса в сети Интернет

Кроме указанных пяти пунктов настройки опроса, в конструкторе также можно добавить бонусы и скиду, выбрать другую тему опроса, изменить дизайн некоторых его элементов, и установить интеграцию с популярными сервисами рассылок, аналитики и CRM-систем.

Данное приложение разработано на Vue. Результатом всех манипуляций с опросом в конструкторе является JavaScript объект, который отправляется на сервер. Взаимодействие с сервером происходит с помощью технологии AJAX.

### Описание содержания приложения Quiz

Quiz отображает созданный опрос в сети Интернет. Quiz – это опрос, который является самой важной частью данной системы, поскольку именно он приносит заявки клиентам компании «Marquiz». Quiz представляет собой клиентское веб-приложение, разработанное на Vue.

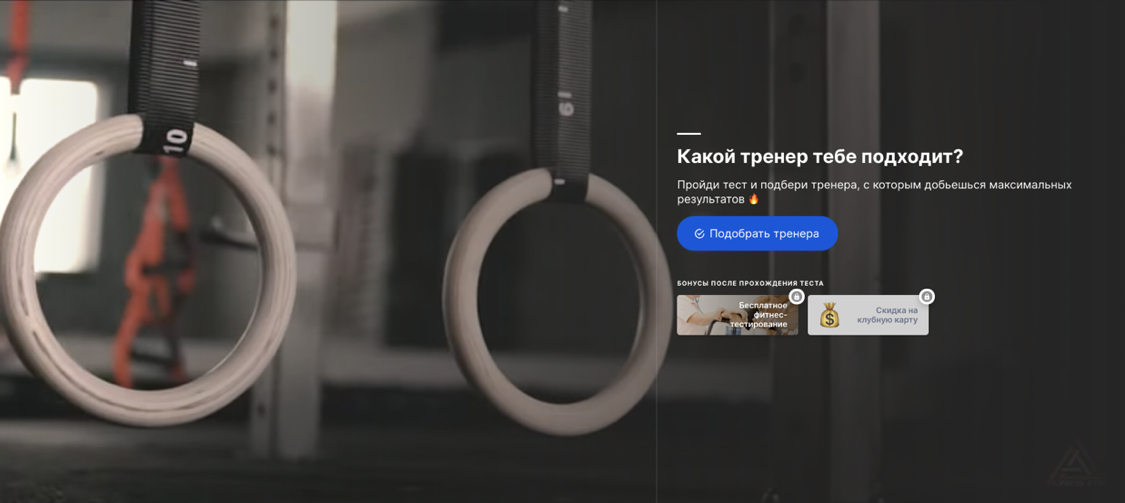
Сначала пользователь попадает на стартовую страницу, где указаны заголовок и небольшое описание опроса, а также бонусы, которые пользователь может получить после прохождения опроса. Пример стартовой страницы представлен на рисунке 8.

Рисунок 8 – Пример стартовой страницы

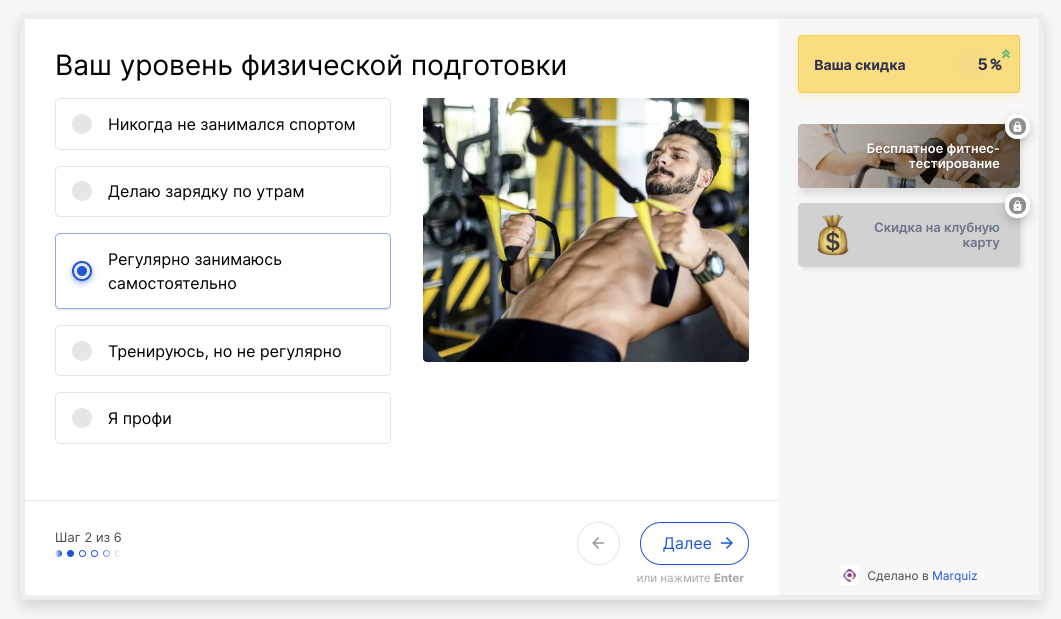
Далее пользователь попадает на опрос, где по очереди отвечает на все вопросы. Справа блока расположены мотивирующие дополнения: скидка и бонусы, а также консультант для пояснений. Внизу формы опроса указано, на сколько вопросов осталось ответить, а также размещены управляющие кнопки для перемещения между вопросами. Пример опроса представлен на рисунке 9.

Рисунок 9 – Пример опроса

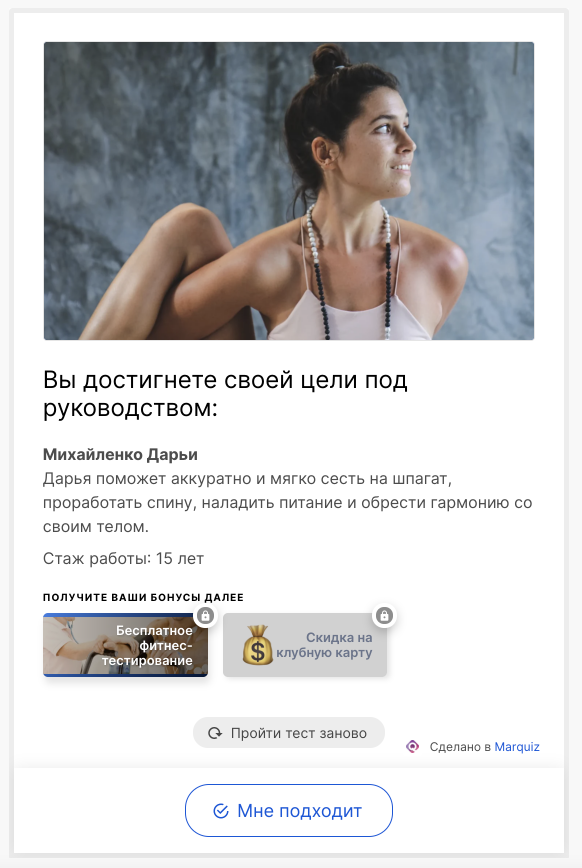
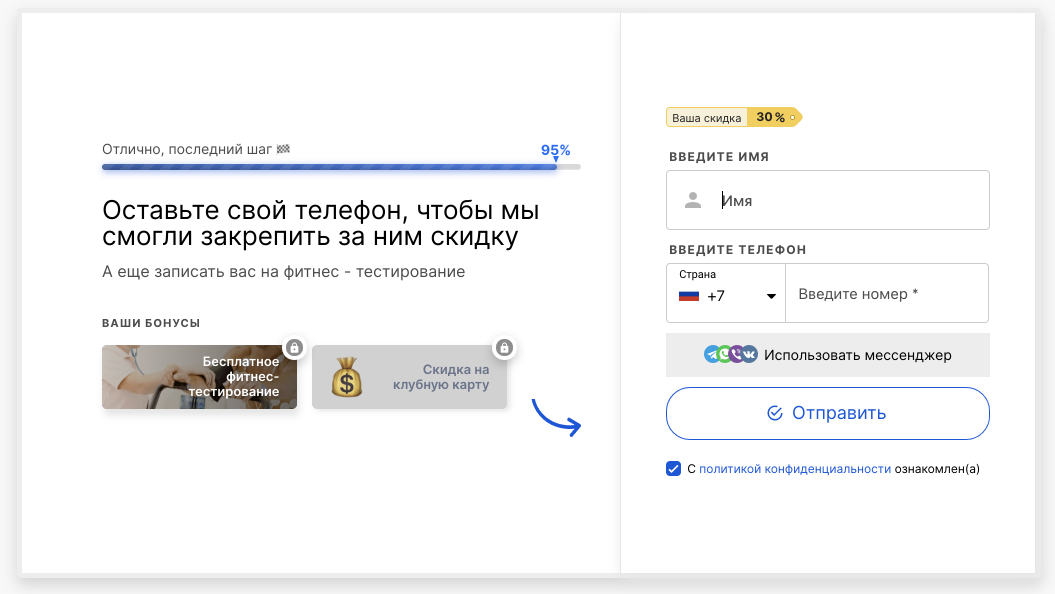
После того, как пользователь ответит на все поставленные ему вопросы, он попадает на страницу с полученным результатом. Пример страницы с результатом представлен на рисунке 10.

Рисунок 10 – Пример страницы с результатом

Последним шагом прохождения опроса является форма, где пользователь оставляет свои контактные данные. Здесь ещё раз указаны бонусы и скидка, которые он получит после заполнения формы.

Пример формы контактов представлен на рисунке 11.

Рисунок 11 – Пример формы контактов

### Описание системы формирования опросов

Система формирования опросов представляет собой JavaScript приложение, которое собирается в один файл с помощью Webpack, устанавливается на сайт клиентов компании и создает опросы на их странице в виде модального окна, которое открывается через кнопку или баннер, или добавляет опрос сразу на сайт. В рамках данной выпускной квалификационной работы была спроектирована архитектура и разработана новая система формирования опросов, которая включает в себя новый способ формирования опроса в виде widget.

### Описание серверной части приложения

Веб-сервер предоставляет API для клиентских приложений на получение, изменение, удаление и создание опросов, а также регистрацию и авторизацию пользователей, настройку интеграций с другими сервисами, прием оплат, и множество других функций.

Веб-сервер разработан с помощью Loopback на платформе Node.js. В качестве СУБД используется MongoDB.

### Описание серверной части модуля аналитики

Модуль аналитики имеет отдельный веб-сервер, который предоставляет API, с помощью которого некоторые ключевые действия пользователей при прохождении опросов обрабатываются и сохраняются в базе данных. Затем клиент, создавший опрос, для которого подключена аналитика, может просматривать собранные данные в конструкторе.

Веб-сервер аналитики разработан на Laravel. В качестве СУБД используется PostgreSQL.

### Описание сервиса загрузки файлов пользователей

Сервис загрузки файлов представляет собой небольшое JavaScript приложение, которое отвечает за загрузку файлов пользователей с приложения Quiz на сервер, и предоставление ссылки на последующее скачивание файла в конструкторе опросов. Загрузка файлов специально выделена в отдельное приложение, поскольку требует большего количества времени, чтобы ответить на запросы, чем на других веб-серверах.

### Вывод

Таким образом, приложение создания опросов компании «Marquiz» имеет микросервисную архитектуру, которая разделяется на 3 клиентских и 3 серверных приложения, которые взаимодействуют друг с другом.

## Исследование процесса отрисовки сайта

Для полного понимания алгоритма формирования опросов на клиентских сайтах необходимо понимать, как браузер осуществляет отрисовку веб-страниц.

Первым шагом браузер получает от сервера HTML-документ, после чего начинает обрабатывать его и формировать DOM-дерево. Процесс обработки происходит синхронно, то есть браузер встречает на своем пути HTML-теги и поочередно добавляет их в DOM. По мере того, как браузер обрабатывает файл, он встречает теги, которые содержат ссылки на сторонние ресурсы, такие как link, script, img и другие – по мере их обнаружения происходит запрос за этими ресурсами. После загрузки сторонних скриптов начинается обработка и исполнение загруженных файлов, и только после их завершения продолжается обработка начального HTML файла, то есть в этот момент отрисовка страницы блокируется. При загрузке стилей браузер формирует CSSOM. CSSOM представляет собой набор API-интерфейсов для манипуляции CSS стилями с помощью JavaScript. После создания DOM-дерева, CSSOM и загрузки всех сторонних ресурсов начинается постепенное отображение сайта в браузере [3].

Система формирования опросов является сторонним ресурсом, который при загрузке блокирует дальнейшую отрисовку сайта, и добавляется с помощью тега script, который клиент может получить в конструкторе опросов.

В настоящее время довольно популярными стали SPA-приложения, которые имеют немного другой процесс отображения сайта в браузере. Ключевым различием является то, что с сервера приходит небольшой HTML-документ, в котором практически не указана полезная информация для посетителя сайта. В данном файле расположено один или несколько тегов script на скачивание JavaScript файлов, в которых и указана основная информация, необходимая для посетителей. Данная информация начнет отображаться на странице только после того, как обработка полностью закончит свою работу.

## Исследование существующей системы формирования опросов

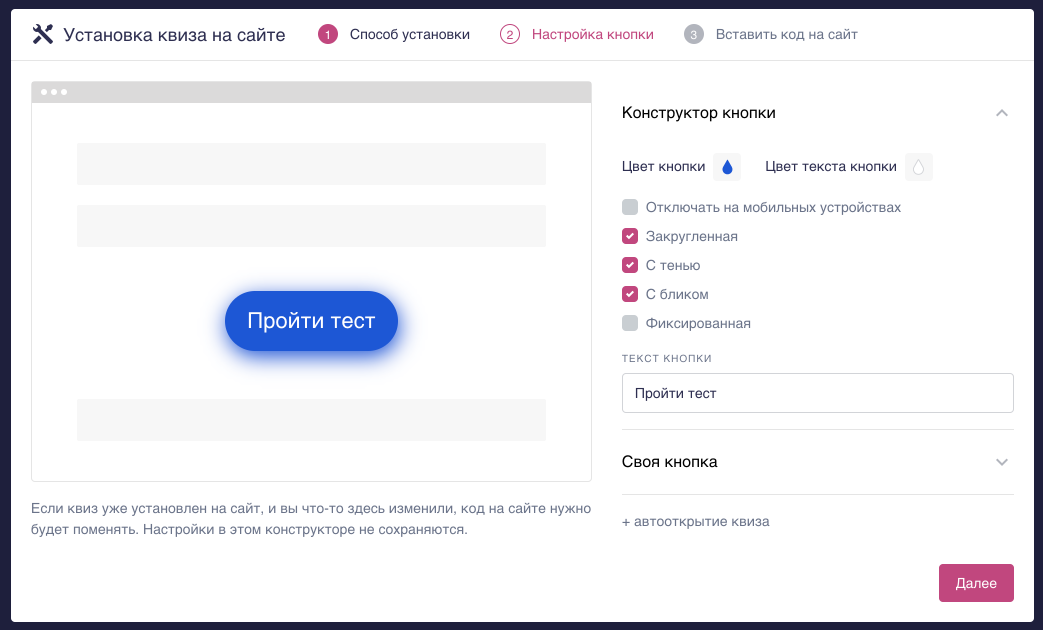
Чтобы пользователю сервиса разместить созданный опрос на свой сайт, ему необходимо выбрать один из указанных способов установки в разделе «Установка квиза» в конструкторе опросов, представленном на рисунке 7. После выбора способа установки пользователю будет предложена настройка внешнего вида кнопки, баннера, опроса, в случае прямой вставки опроса на сайт, или настройка автооткрытия, в зависимости от начального выбора. Интерфейс настройки кнопки представлен на рисунке 12.

Рисунок 12 – Интерфейс настройки кнопки

После настроек пользователю будет предоставлено два кода: код инициализации и код добавления опроса.

Страница с кодом инициализации и кодом добавления опроса представлены на рисунке 13.

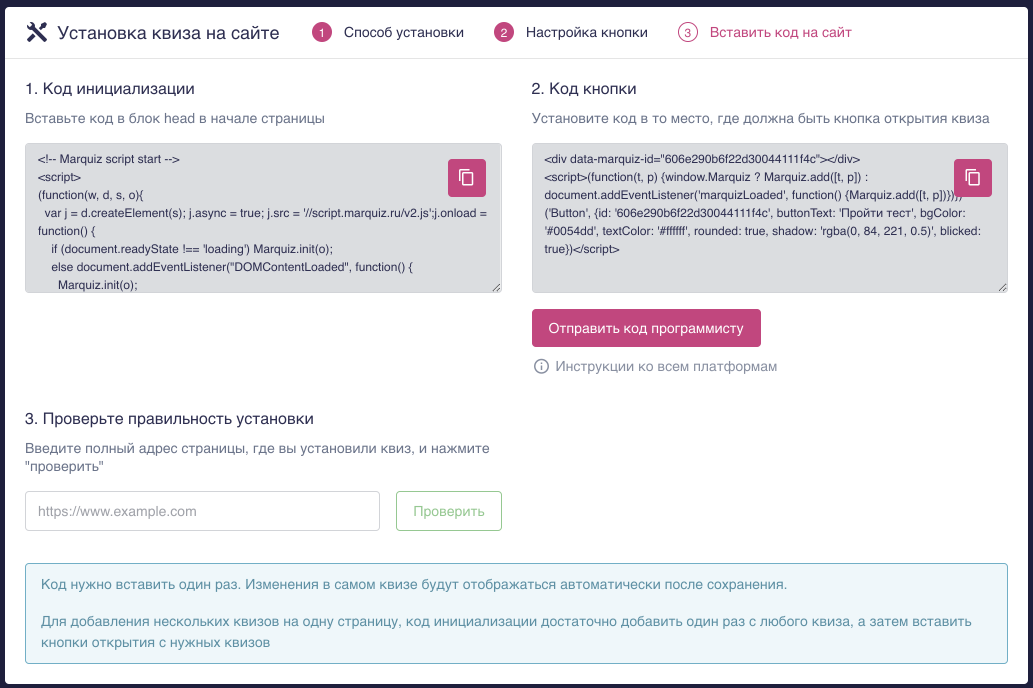


Рисунок 13 – Страница с кодом инициализации и кодом добавления опроса

Пример кода инициализации указан в листинге 1.

Листинг 1 – Пример кода инициализации

<!-- Marquiz script start -->

<script src="//script.marquiz.ru/v1.js" type="application/javascript"></script>

<script>

document.addEventListener("DOMContentLoaded", function() {

Marquiz.init({

host: '//quiz.marquiz.ru',

id: '608bea22282a70001acdf094',

autoOpen: false,

autoOpenFreq: 'once',

openOnExit: false

});

});

</script>

<!-- Marquiz script end -->

Данный код размещается в начале HTML файла в теге head и отвечает за загрузку JavaScript файла, который является системой формирования опросов, а также за инициализацию данной системы и автооткрытие опроса в случае, если клиент указал это в настройках при выборе способа установки.

Пользователю также необходимо добавить HTML-код нужного ему способа вставки в любое место на сайте, где он хочет разместить кнопку или опрос. Пример кода добавления кнопки, открывающей опрос в модальном окне, на сайт указан в листинге 2.

Листинг 2 – Пример кода добавления кнопки

<div class="marquiz\_\_container">

<a class="marquiz\_\_button marquiz\_\_button\_blicked marquiz\_\_button\_rounded marquiz\_\_button\_shadow" href="#popup:marquiz\_608bea22282a70001acdf094" data-fixed-side="" data-alpha-color="rgba(211, 64, 133, 0.5)" data-color="#d34085" data-text-color="#ffffff">Пройти тест</a>

</div>

В существующей системе имеется 4 способа формирования опроса на сайте:

* кнопка. На сайт устанавливается кнопка, при нажатии на которую открывается модальное окно с опросом;
* баннер. На сайте появляется баннер с установленной в настройках задержкой, при нажатии на который открывается модальное окно с опросом;
* установка опроса сразу на сайт. Данный способ создает блок на сайте, в котором опрос сразу виден посетителю сайта. Для мобильной версии сайта автоматически формируется кнопка для открытия опроса в модальном окне;
* автооткрытие. Через установленное пользователем время в настройках опрос автоматически откроется в модальном окне после полной загрузки сайта.

Система формирует опросы на сайте, используя iframe окна с прямыми ссылками на созданные опросы в приложение Quiz.

Текущая система формирования опросов не позволяет установить опросы в SPA-приложения, поскольку компонентный подход к разработке в таких приложениях не позволяет разместить код добавления кнопок, баннеров или самих опросов до момента инициализации системы. Установленные кнопки и баннеры не будут работать, а опросы, установленные сразу на сайт, просто не отобразятся на странице.

Вся система описана в одном классе Marquiz, где происходит формирование всех видов опросов: кнопок, баннеров и прямых вставок на сайт. Добавление ещё одного способа формирования опроса вызывает сложности, поскольку необходимо разбираться в одном большом файле и изменять его в нескольких местах.

## Исследование способов асинхронной загрузки приложений

Поскольку загрузка JavaScript файлов сторонних приложений блокирует дальнейшую отрисовку сайта, пока полностью не скачает файл и не исполнит его, то это может сильно влиять на скорость загрузки сайта для его посетителей. Такая загрузка называется синхронной, то есть все выполняется поочередно. Сторонние JavaScript файлы также можно загружать асинхронно. Асинхронная загрузка файла подразумевает загрузку сторонних скриптов в фоновом режиме, не блокируя дальнейшую отрисовку страницы, что в итоге может значительно повысить скорость загрузки сайта.

Существует 2 способа асинхронной загрузки файлов:

* добавление атрибута defer для тега script;
* добавление атрибута async для тега script.

При первом способе браузер при обработке HTML файла при встрече данного тега начнет загрузку файла в фоне и продолжит обработку и отрисовку страницы дальше до конца файла. Загруженный скрипт начнет исполняться только после завершения отрисовки страницы [4].

Данный способ позволит ускорить загрузку страницы, но так как файл исполнится в самую последнюю очередь, то опросы также будут сформированы и добавлены на сайт только после его полной загрузки. Для небольших сайтов этот способ может подойти, но если сайт крупный и опрос необходимо загрузить в самом начале, чтобы пользователь мог с ним начать взаимодействовать до полной отрисовки страницы, то данный способ уже не подходит.

При втором способе браузер при обработке HTML файла при встрече данного тега также начнет загрузку файла в фоне и продолжит обработку и отрисовку страницы дальше до конца файла, но загруженный скрипт начнет исполняться сразу после того, как загрузится. Используя данный способ, опросы будут формироваться на сайте настолько быстро, насколько это возможно для посетителя сайта, при этом не блокируя отрисовку страницы в момент загрузки файла [4].

## Разработка требований к новой системе формирования опросов

Формирование требований к новой разрабатываемой системе основано на исследовании существующей системы формировании опросов, а также на основании пользовательских требований к проекту. Ключевой проблемой являлась сложность добавления нового способа формирования опросов, в виде widget, поэтому разрабатываемая система должна содержать данный способ, а также не вызывать проблем при добавлении нового функционала.

Кроме этого, новая проектируемая система должна иметь другой алгоритм добавления опросов на страницы, который учитывает асинхронную загрузку файлов и особенность отрисовки SPA-приложений.

Стоит отметить, что разрабатываемая система также должна поддерживать старые способы добавления опросов на сайты для удобства пользователей.

Требования к системе:

* система не должна вызывать проблем при добавлении нового функционала;
* система должна содержать новый способ формирования опросов;
* система должна поддерживать SPA-приложения;
* система должна устанавливаться на сайт клиентов асинхронно;
* система должна поддерживать старый способ добавления опросов.

## Описание требований к новому представлению опроса

Компанией Marquiz были указаны следующие требования для нового представления опросов в виде widget:

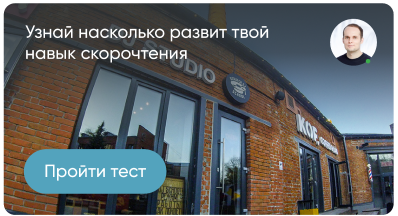
* опрос должен иметь возможность появления с задержкой;
* опрос может быть расположен внизу экрана справа или слева;
* опрос не должен закрываться до полного его прохождения;
* в случае не полного прохождения опроса, он должен иметь возможность сворачиваться в компактную форму;
* опрос должен иметь возможность автооткрытия с задержкой;
* опрос должен иметь возможность быть скрытым с сайта на мобильной версии.

Дизайн нового представления опроса на сайте и дизайн интерфейса настроек в конструкторе опросов разработан специалистами компании.

### Дизайн опроса в виде widget

Опрос в виде widget имеет два состояния: открыт и свернут. В открытом состоянии опрос необходимо отображать таким же, как и на мобильных устройствах, а для свернутого состояния компания разработала новый дизайн.

Дизайн стартовой страницы в свернутом опросе в виде widget представлен на рисунках 14, 15 и 16.

Рисунок 14 – Стартовая страница опроса в виде widget

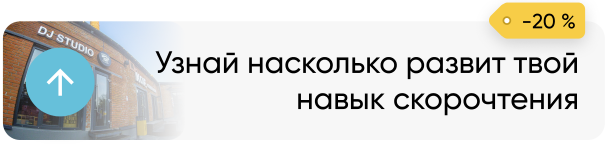
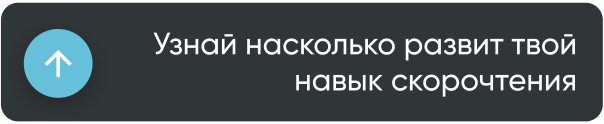
Рисунок 15 – Стартовая страница опроса в виде widget для мобильных устройств

Рисунок 16 – Стартовая страница опроса в виде widget без картинки или видео

Стартовая страница имеет три вида. Первый вид, представленный на рисунке 14, используется, если на фон стартовой страницы установлены картинка или видео. Также в данном виде указывается консультант и скидка, если они добавлены в настройках в конструкторе опросов. Второй вид, представленный на рисунке 15, используется на мобильных устройствах, и имеет те же свойства, что и первый вид. В случае, если пользователь сервиса не добавил фото или видео на фон стартовой страницы, то используется её третий вид. Цвет фона, кнопки и текста должен соответствовать цвету, который пользователь выбрал в конструкторе опросов в разделе «Дизайн».

Дизайн вопросов в свернутом опросе в виде widget представлен на рисунках 17 и 18.

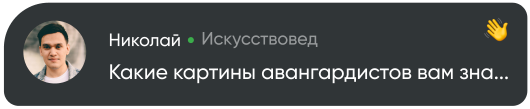
Рисунок 17 – Этап вопросов в опросе в виде widget с консультантом

Рисунок 18 – Этап вопросов в опросе в виде widget

Этап вопросов имеет 2 вида. Первый вид, представленный на рисунке 17, используется, когда в конструкторе в разделе «Вопросы» пользователь добавил консультанта, а второй, когда консультант не добавлен.

Дизайн результатов в свернутом опросе в виде widget представлен на рисунках 19, 20 и 21.

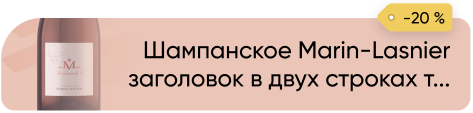
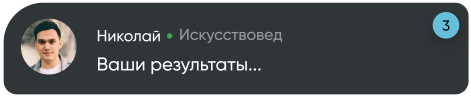
Рисунок 19 – Результат в опросе в виде widget 

Рисунок 20 – Результат в опросе в виде widget для мобильных устройств

Рисунок 21 – Страница с несколькими результатами в опросе в виде widget

В случае, если посетителю опроса будет показано несколько результатов одновременно, используется вид, указанный на рисунке 21.

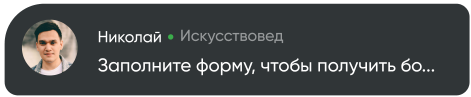
Дизайн формы контактов в свернутом опросе в виде widget представлен на рисунке 22.

Рисунок 22 – Форма контактов в опросе в виде widget

### Дизайн интерфейса настроек нового представления в конструкторе опросов

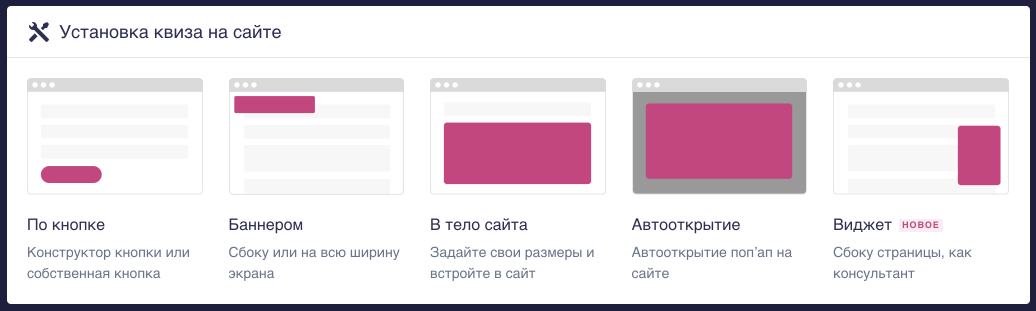
В конструкторе опросов добавится новый способ установки опроса на сайт. Дизайн интерфейса выбора установки опроса с новым представлением в виде widget представлен на рисунке 23.

Рисунок 23 – Интерфейс выбора установки опроса с новым представлением в виде widget

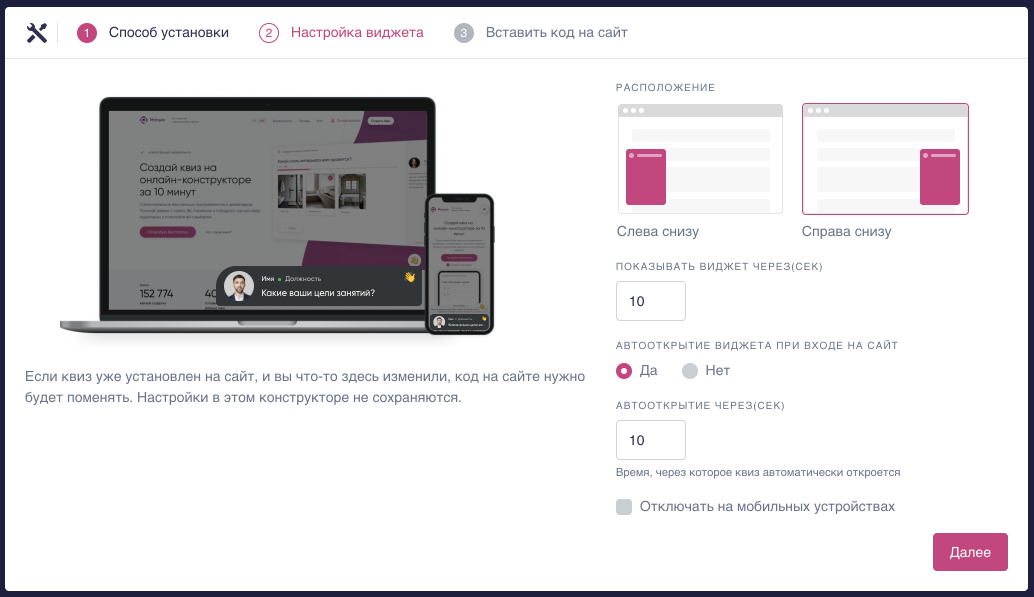
Дизайн интерфейса настроек установки опроса в виде widget представлены на рисунке 24.

Рисунок 24 – Интерфейс настроек установки опроса в виде widget

Здесь можно настроить размещение опроса на странице, задержку перед первым появлением опроса, задержку для автооткрытия опроса, а также имеется возможность отключения опроса на мобильных устройствах.

## Вывод по разделу

В данном разделе описана структура системы компании «Marquiz», проведено исследование текущей системы формирования опросов, отрисовки сайтов и SPA-приложений браузером, асинхронной загрузки сторонних файлов, разработаны требования для проектирования и разработки новой системы формирования, а также описаны требования и дизайн для нового представления опроса в виде widget.

# Конструкторский раздел

## Описание модели «AS-IS»

Для описания существующего алгоритма формирования опросов на сайте клиента сервиса «Marquiz» используется модель «AS-IS», проиллюстрированная с помощью UML диаграммы деятельности. Данная модель позволяет систематизировать и визуализировать протекающие в данный момент процессы. На основе этого выявляются узкие места в организации и взаимодействии процессов, определяется необходимость тех или иных изменений в существующем алгоритме [5]. Диаграмма деятельности существующего алгоритма формирования опросов на сайт представлена на рисунке 25.

При обработке HTML-файла браузером сначала ищется код инициализации системы. Если данный код не был найден, то браузер продолжит обработку файла до конца и отрисует сайт без опросов. Если код инициализации встретился, то начинается загрузка стороннего файла, в котором реализована система формирования опросов. На этом этапе обработка HTML-файла блокируется, и браузер ждет полной загрузки стороннего файла. После получения и исполнения скачанного файла создается обработчик события, в котором вызывается инициализация системы, на полную загрузку сайта. То есть, после полной обработки HTML-файла и отрисовки сайта, вызовется функция инициализации системы.

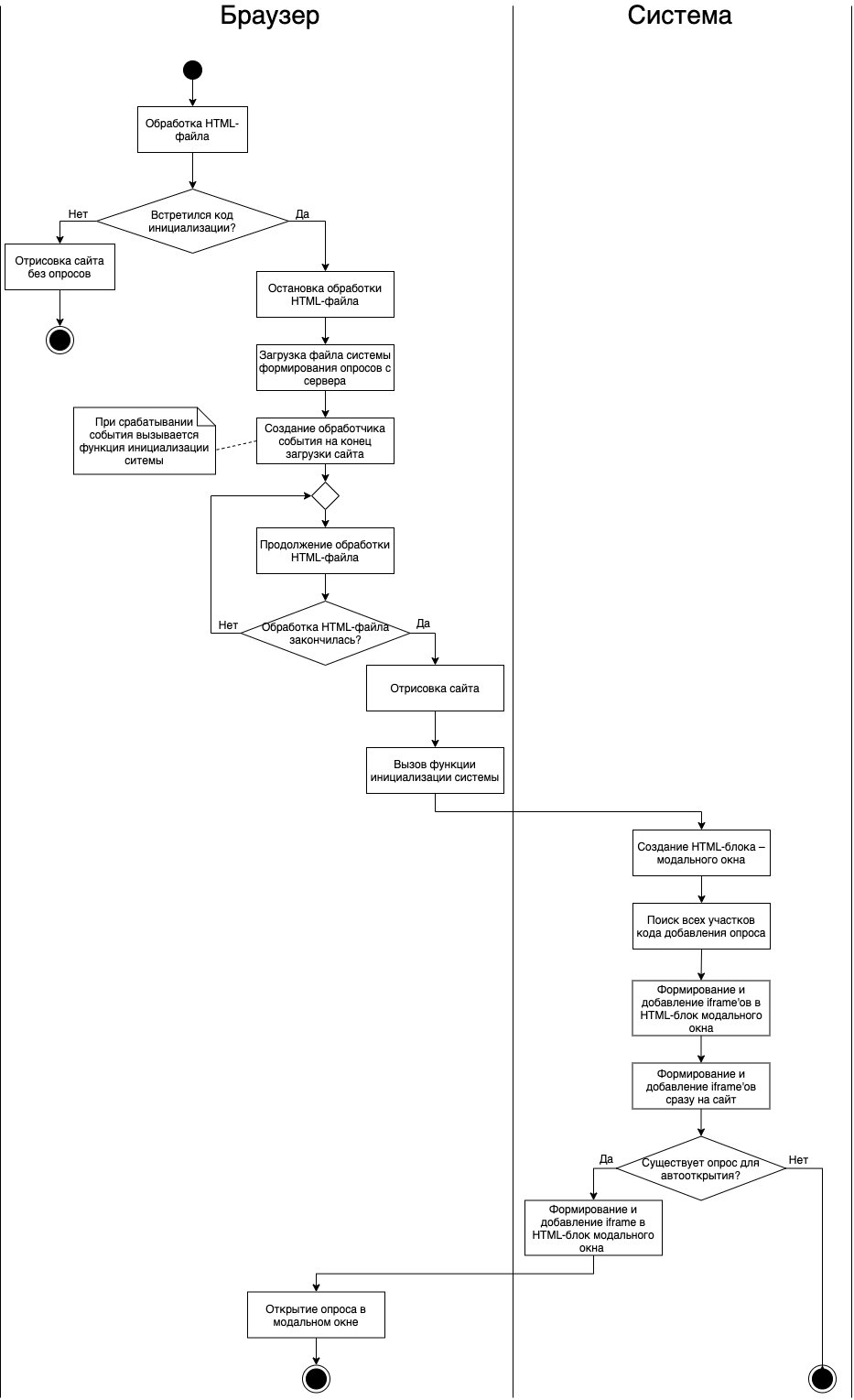
Инициализация системы начинается с поиска кодов добавления всех существующих видов опросов: кнопки, баннеры, прямой вставки на сайт. Создается и добавляется на сайт HTML-код модального окна, в котором затем размещаются опросы, открывающиеся через кнопки и баннеры. После формирования и добавления опросов в модальном окне, происходит формирование опросов, установленных прямиком на сайт. В самом конце проверяется наличие опроса на автооткрытие. Если такой имеется, то происходит добавление данного опроса на сайт и автоматическое открытие модального окна с ним.

Рисунок 25 – Диаграмма деятельности существующего алгоритма формирования опросов на сайт

Основными недостатками алгоритма можно выделить: блокировку обработки HTML-файла в момент загрузки стороннего файла, что может замедлить общее время загрузки страницы, а также формирование опросов только после полной отрисовки страницы, что не позволяет добавлять опросы в SPA-приложения, поскольку поиск кодов добавления опросов начинается до их фактического добавления в DOM-дерево.

## Описание модели «TO-BE»

На этапе анализа модели «AS-IS» были выделены следующие недостатки:

* блокировка обработки HTML-файла в момент загрузки стороннего файла;
* формирование опросов только после полной отрисовки сайта.

Для устранения этих недостатков разрабон новый алгоритм формирования опросов на сайт. Для иллюстрации предлагаемого решения разработана функциональная модель «TO-BE», проиллюстрированная с помощью UML диаграммы деятельности представленной на рисунках 26 и 27.

Одним из основных отличий данного алгоритма заключается в асинхронной загрузке стороннего файла, что не блокирует дальнейшую обработку HTML-файла. Вторым основным отличием является создание собственного события «marquizLoaded», которое создается после загрузки JavaScript файла и создания экземпляра глобального класса Marquiz, через которое и происходит дальнейшее добавление опросов.

Если браузер при обработке HTML-файла во время загрузки стороннего файла встречает код добавления опроса, то создается обработчик события «marquizLoaded», в котором происходит обращение к экземпляру класса Marquiz для добавления опроса на сайт. После полной загрузки стороннего JavaScript файла с системой создается экземпляр класса Marquiz и событие «marquizLoaded». Все ранее созданные функции добавления опроса вызываются, и опросы формируются и добавляются на сайт. При дальнейшей обработке HTML-файла функция добавления будет вызываться сразу, поскольку экземпляр класса Marquiz уже существует.

Когда сайт будет полностью отрисован, вызовется функция инициализации системы, которая сформирует опросы, добавленные старым способом. Также при вызове данной функции будет добавлен опрос для автооткрытия, если он был настроен.

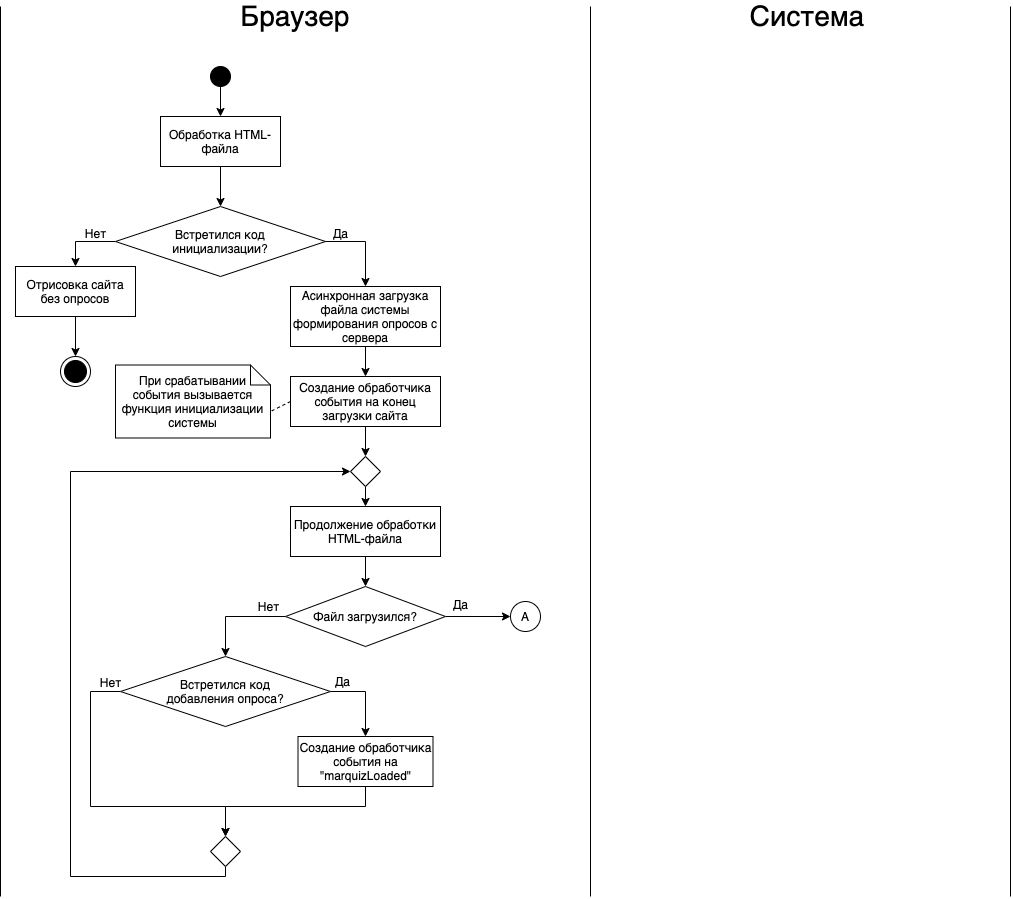
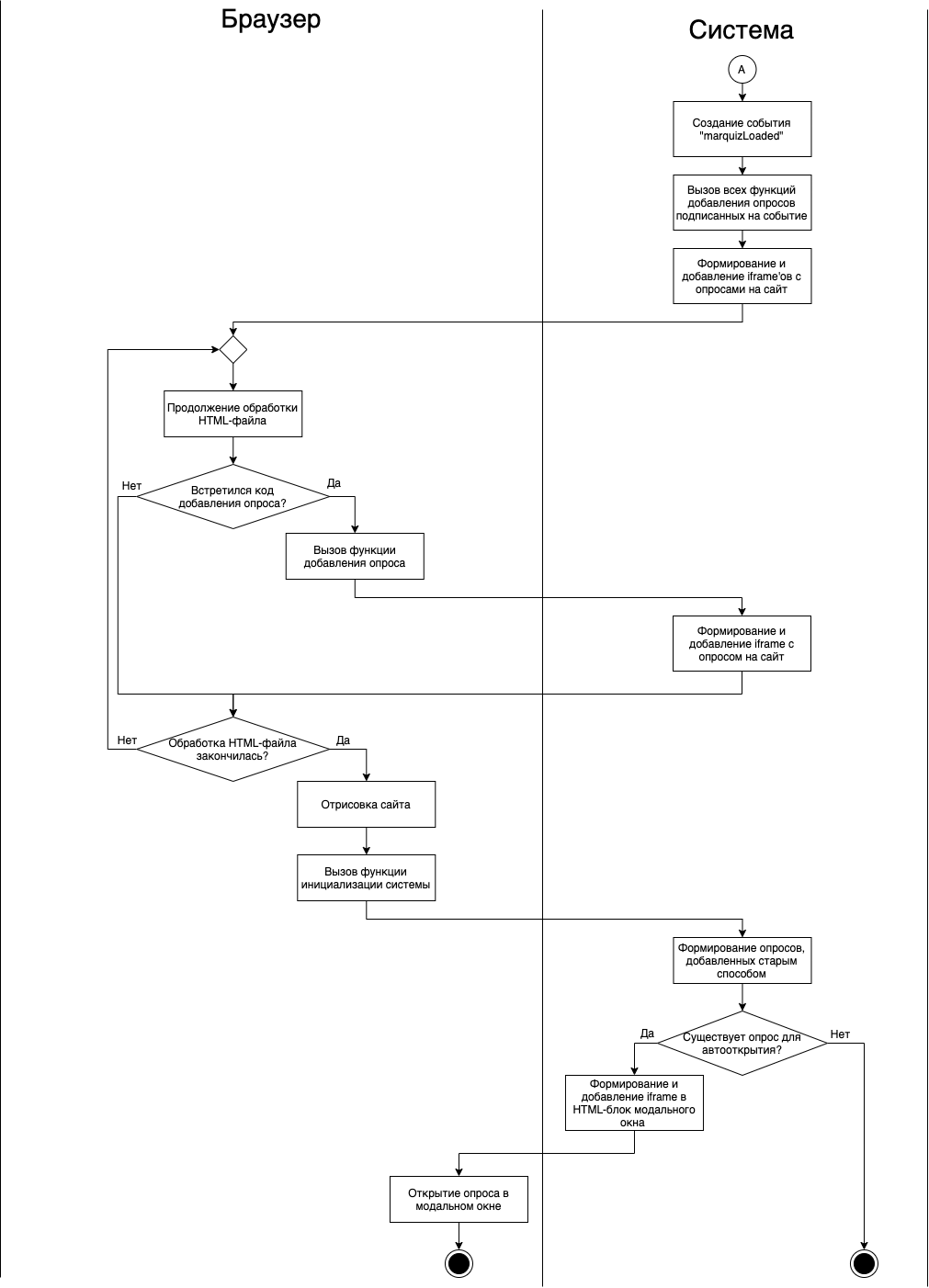
Данные улучшения позволяют избавиться от недостатков прошлого алгоритма. Асинхронная загрузка файла позволяет не блокировать обработку HTML-файла, а создание события позволяет формировать опросы во время обработки HTML-файла браузером.

Рисунок 26 – Диаграмма деятельности алгоритма формирования опроса на сайте

Рисунок 27 – Продолжение диаграммы деятельности алгоритма формирования опроса на сайте

## Проектирование клиентской части нового представления опроса в виде widget

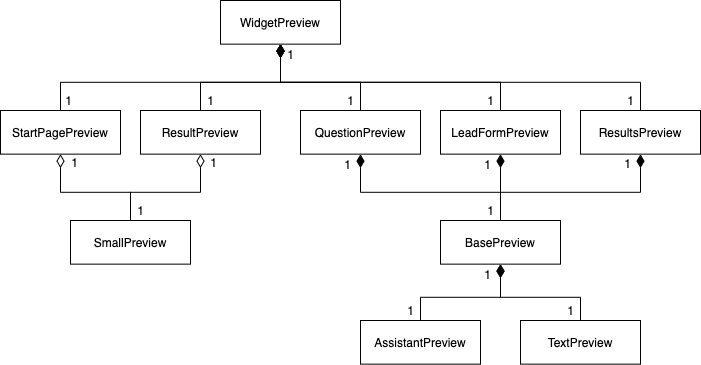
По дизайну клиентской части нового представления опроса в виде widget, описанному в разделе 1.7.1 была составлена диаграмма vue-компонентов, которая представлена в виде UML диаграммы классов на рисунке 28.

Рисунок 28 – Диаграмма vue-компонентов клиентской части нового представления в виде widget

Для отображения опроса в свернутом положении используется основной компонент WidgetPreview, где прописана логика отображения необходимого представления опроса на каждый его этап, то есть по мере прохождения опроса пользователем он будет поочередно отображать StartPagePreview, QuesionPreview, ResultsPreview или ResultPreview и в конце LeadFormPreview.

Компоненты StartPagePreview и ResultPreview используются для отображения стартовой страницы и этапа с результатом. Поскольку оба этих компонента имеют похожий дизайн для мобильных устройств, то они оба содержат компонент SmallPreview.

Компоненты QuestionPreview, LeadFormPreview и ResultsPreview используются для отображения этапа вопросов, формы контактов и страницы с несколькими результатами. Поскольку все эти компоненты имеют похожий дизайн, то они все используют компонент BasePreview, который в свою очередь разделяется на 2 вида: с консультантом, без консультанта. Для вида с консультантом используется компонент AssistantPreview, для вида без консультанта – TextPreview.

## Проектирование диаграммы классов

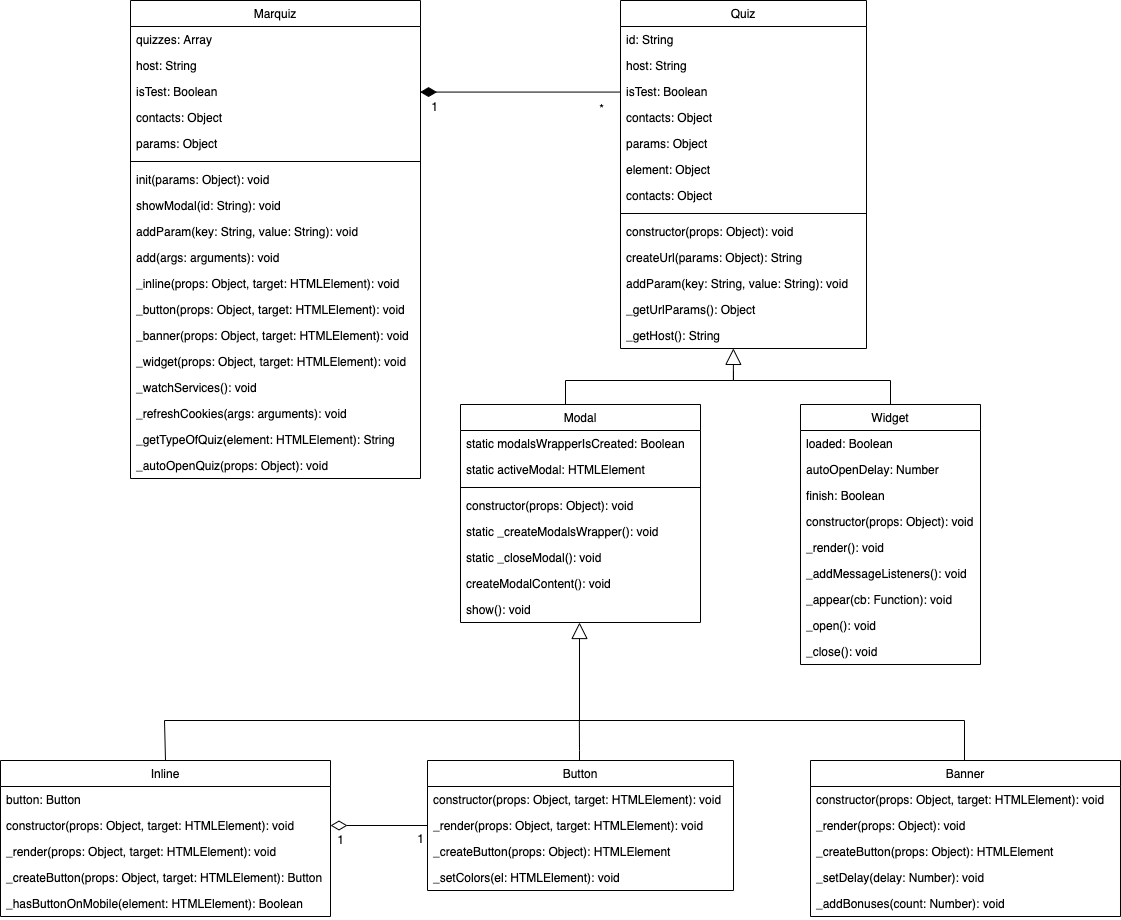
Существующая система формирования опросов целиком находится в одном файле, что неудобно для дальнейшего расширения функционала. Для упрощения понимания и добавления нового представления в виде widget была составлена диаграмма классов, представленная на рисунке 29.

Рисунок 29 – Диаграмма классов

Основной класс Marquizиспользуется для инициализации всего приложения. Экземпляр данного класса создается сразу же после загрузки JavaScript файла на сайт. Поле quizzes содержит в себе массив всех установленных опросов на сайт. Поле host содержит домен актуальной ссылки на приложение Quiz. Поля isTest и contacts используются для автоматической подстановки тестовых данных в форму контактов в конструкторе опросов, чтобы клиент сразу мог проверить, приходят ли ему ответы. Поле params содержит параметры, установленные самим клиентом сервиса при добавлении кода инициализации на свой сайт. Обычно данное поле используют для передачи различных UTM-меток или cookies. Значения данных полей будут передаваться во все добавленные опросы автоматически, если при создании экземпляра класса самого опроса они не будут переопределены.

Для добавления опроса используется метод add, которому параметрами передается способ вставки и его настройки. Внутри данного метода уже вызывается необходимая функция для добавления опроса нужного вида. Для добавления отдельных видов опросов используются методы: \_inline, \_button, \_banner, \_widget. При их вызове происходит создание экземпляров соответствующих классов и их добавление в массив quizzes.

Также класс Marquiz содержит метод init, который вызывается при полной загрузке сайта и формирует опросы, добавленные старым способом. Также в данном методе вызывается функция \_autoOpenQuiz, которая создает и добавляет опрос для автооткрытия. Класс Marquiz содержит метод showModal, который позволяет пользователю самому программно вызывать открытие модального окна с нужным ему опросом. У пользователя есть возможность передавать собственные данные прямиком в опрос в приложение Quiz с помощью метода addParam. Методы \_watchServices и \_refreshCookies позволяют передавать данные, хранимые в cookies браузера, популярных сервисов автоматически при инициализации системы.

У каждого из вида опросов есть одинаковые поля, такие как id опроса, домен для создания ссылки для iframe, контакты и различные параметры, которые пользователь может передать в опрос при необходимости. Все вышеперечисленные свойства определяются и наследуются в производные классы из класса Quiz*.* Кроме упомянутых полей в данном классе есть поле element, которое содержит в себе HTML-элемент с iframe.

В классе Quiz с помощью метода createUrl происходит формирование ссылки в приложение Quiz, которое потом размещается в iframe окне.

Опросы, которые добавлены прямиком на сайт, кнопки или баннера могут открываться на сайте в модальном окне, поэтому все они наследуются от одного класса Modal*.* Данный класс содержит статические поля и методы, поскольку блок модального окна один, в котором содержится несколько iframe окон, если пользователь добавил больше одного опроса на сайт. Поле modalsWrapperIsCreated является логическим и является истинным, когда HTML-код модального окна был добавлен на сайт. Поле activeModal содержит HTML-код текущего открытого опроса. Методы close и show служат для закрытия и открытия модального окна соответственно. В случае добавления нескольких одинаковых опросов на сайт, класс Modal не будет дублировать создание одинаковых iframe окон.

Опрос для автооткрытия создается с помощью класса Modal, поскольку не нуждается ни в кнопке, ни в баннере.

Классы Inline, Button и Banner отвечают за создание опросов в виде прямой вставки на сайт, кнопки и баннера соответственно. Класс Inline, кроме этого, содержит в себе экземпляр класса Button, поскольку прямая вставка на сайт может содержать кнопку на мобильной версии сайта, если пользователь указал это в настройках установки опроса.

Новый способ представления опроса widget добавляется с помощью класса Widget, который наследуется напрямую от класса Quiz, потому что не нуждается в открытии в модальном окне. Поле loaded становится истинным после полного формирования опроса в iframe окне. Изначально опрос появляется на сайте в свернутом виде, после истечении времени, указанном в поле autoOpenDelay, опрос автоматически открывается в полном виде. Метод \_appear отображает опрос на экране в первый раз. Затем опрос в виде widget может открываться и закрываться множество раз с помощью методов \_open и \_close соответственно. После полного прохождения опроса переменная finish становится истинной и опрос закрывается до следующего открытия сайта. С помощью метода \_addMessageListeners реализовано общение между системой формирования и приложением Quiz для отслеживания закрытие и открытия виджета, а также изменение высоты приложения Quiz, поскольку в свернутом положении опрос в виде widget имеет разную высоту на разных этапах прохождения опроса.

Общение между системой и приложением Quiz реализовано через postMessage – метод, который позволяет безопасно отправлять кроссдоменные запросы.

## Средства разработки

Для разработки системы формирования опросов на сайт будет использоваться язык JavaScript, поскольку именно данный язык используется браузерами для программного доступа к объектам страницы.

Поскольку новая система формирования опросов разрабатывается для компании «Marquiz», то использованы те средства разработки, которые применяются в компании. В качестве сборщика модулей приложения используется Webpack. Для создания нового представления опроса в виде widget, кроме написания новой системы формирования опросов, необходимо также внести изменения в существующее приложение Quiz, написанное на Vue.

## Вывод по разделу

В данном разделе разработан новый алгоритм формирования опросов на клиентских сайтах, который не содержит недочетов, полученных при анализе существующей системы, подробно описан новый способ формирования опросов в виде widget и спроектирована диаграмма классов новой системы формирования.

# Технологический раздел

## Разработка новой системы формирования опросов

### Реализация классов новой системы формирования опросов

Конструктор основного класса Marquiz, который используется для инициализации всей системы указан в листинге 3.

Листинг 3 – Конструктор класса Marquiz

constructor() {

this.quizzes = []

this.host = '//quiz.marquiz.ru'

this.isTest = false

this.contacts = null

this.params = {}

}

В конструкторе происходит объявление всех переменных и присвоение им значений по умолчанию. Все поля были описаны при проектировании системы в разделе 2.4.

Добавление опросов на сайт осуществляется при вызове метода add, который в качестве параметров принимает тип представление опроса и его настрйоки. Метод add представлена в листинге 4.

Листинг 4 – Реализация метода добавления

add(args) {

const [type, props] = args

const target = document.querySelector(`div[data-marquiz-id='${props.id}']`)

switch (type) {

case 'Button':

if (target) this.\_button(props, target)

else console.error('marquiz\_\_error', 'Target was not found')

break

case 'Inline':

if (target) this.\_inline(props, target)

else console.error('marquiz\_\_error', 'Target was not found')

break

case 'Banner':

this.\_banner(props, target)

break

case 'Widget':

this.\_widget(props)

break

default:

break

}

}

В зависимости от типа представления: кнопка, прямая вставка опроса на сайт, баннер или виджет, вызывается соответствующая функция добавления. Для кнопки и прямой вставки также необходимо указать место добавления. Если место указано не будет, то кнопка и опрос не добавятся, и выведется ошибка в консоль.

Функции добавления определенных типов опросов представлен в листингах 5, 6, 7 и 8.

Листинг 5 – Функция добавления опроса в виде кнопки

\_button(props, target) {

this.quizzes.push(new Button(props, target))

}

Листинг 6 – Функция добавления опроса прямиком на сайт

\_inline(props, target) {

this.quizzes.push(new Inline(props, target))

}

Листинг 7 – Функция добавления опроса в виде баннера

\_banner(props, target) {

this.quizzes.push(new Banner(props, target))

}

Листинг 8 – Функция добавления опроса в виде widget

\_widget(props) {

this.quizzes.push(new Widget(props))

}

Каждая из функций добавляет новый экземпляр необходимого класса в массив quizzes.

Добавленные опросы, пользователь сервиса может самостоятельно программно открывать с помощью метода showModal, представленной в листинге 9.

Листинг 9 – Метод открытия модального окна опроса

showModal(id) {

if (!id) return

if (id === this.id) this.wasOpened = true

let modal = null

if (this.quizzes[id]) {

this.quizzes[id].show()

} else {

modal = new Modal({ id })

this.quizzes.push(modal)

modal.show()

}

}

Если при вызове данного метода не указать id опроса, то модальное окно не откроется. В случае попытки открыть не добавленный опрос, создастся экземпляр класса Modal, и опрос откроется.

Для автоматической передачи данных в опрос в приложение Quiz используется поле params, но если пользователь не может передать их при инициализации приложения, то он может воспользоваться методом addParam, указанной в листинге 10.

Листинг 9 – Метод передачи данных в опрос в приложение Quiz

addParam(key, value) {

this.params[key] = value

for (const id in this.quizzes) {

this.quizzes[id].addParam(key, value)

}

}

Передача данных осуществляется во все добавленные опросы.

Для автоматической передачи данных популярных сервисов используется метод \_watchServices, которая каждые 8 секунд обновляет данные и отправляет их повторно. Данный метод представлена в листинге 10.

Листинг 10 – Метод передачи данных популярных сервисов

\_watchServices() {

// Roistat

let previousOnRoistatModuleLoaded = () => {}

if (window.onRoistatModuleLoaded !== undefined) {

previousOnRoistatModuleLoaded = window.onRoistatModuleLoaded

}

window.onRoistatModuleLoaded = () => {

previousOnRoistatModuleLoaded()

window.roistatVisitCallback = () => {

this.\_refreshCookies()

}

}

// Comagic

if (window.\_\_cs) {

window.\_\_cs.push(['waitForTrackViewAndRun', () => {

const creds = Comagic.getCredentials() || {};

this.addParam('comagic\_id', creds.comagic\_id)

}])

}

// Other services

setTimeout(() => this.\_refreshCookies(), 8000)

}

Для передачи cookies используется метод \_refreshCookies, представленная в листинге 11.

Листинг 11 – Метод передачи cookies

\_refreshCookies() {

this.addParam('cookies', getCookies())

}

Для инициализации приложения используется метод init, которая вызывается после полной загрузки сайта. Данный метод представлен в листинге 12.

Листинг 12 – Метод инициализации системы формирования опросов

init({ host, id, autoload, autoOpen = false, openOnExit, autoOpenFreq = 'once', test = false, contacts = {}, disableOnMobile, query = {}, region }) {

// Override global variables for v1 quizzes

if (host) this.host = host

this.isTest = test

this.contacts = contacts

this.\_query = query

// Autoload & open on exit props

this.id = id

this.wasOpened = false

this.loaded = false

// Search v1 quizzes

const elements = document.querySelectorAll('a[href^="#popup:marquiz"],a[href^="http://#popup:marquiz"],a[href^="https://#popup:marquiz"]')

for (let i = 0; i < elements.length; i++) {

const element = elements[i]

const id = this.\_getIdFromLink(element.href)

const type = this.\_getTypeOfQuiz(element)

switch (type) {

case 'button':

this.\_button({ id, autoload }, element)

break

case 'inline':

this.\_inline({ id }, element)

break

case 'banner':

this.\_banner({ id, autoload }, element)

break

default:

break

}

}

this.\_autoOpenQuiz({

autoOpenFreq, autoOpen, openOnExit, disableOnMobile,

})

this.\_watchServices()

}

В данном методе сначала происходит процесс перезаписи полей, объявленных в конструкторе класса для опросов, установленных старым способом. Затем идет поиск кодов добавления опросов старым способом, и происходит их добавление на сайт. В конце вызывается метод автооткрытия квиза и ранее описанный метод автоматической передачи данных популярных сервисов.

Метод автооткрытия опроса \_autoOpenQuiz представлена в листинге 13.

Листинг 13 – Метод автооткрытия опроса

\_autoOpenQuiz({ autoOpenFreq, autoOpen, openOnExit, disableOnMobile }) {

const close = disableOnMobile && document.body.clientWidth < 768

if (!this.id || close) return

// Disable autoload if it was happened

if (autoOpenFreq !== 'always' && this.\_getOpenCount(this.id) > 0) {

return

}

// Preload quiz

if (autoOpen !== false || openOnExit) {

setTimeout(() => {

const quiz = new Modal({ id: this.id })

quiz.createModalContent()

this.quizzes.push(quiz)

}, 500)

}

// Open on exit

if (openOnExit) {

this.\_beforeLeaveListener()

}

// Autoload

if (autoOpen !== false) {

setTimeout(() => {

if (!this.wasOpened) this.\_showModalWhenMounted(this.id, 200)

}, autoOpen \* 1000)

}

}

Основным классом для опроса является Quiz. В конструкторе данного класса происходит объявление полей и присваивание значений по умолчанию. Конструктор представлен в листинге 14.

Листинг 14 – Конструктор класса Quiz

constructor({ host, isTest, contacts })

this.id = id

this.host = host || window.Marquiz.host

this.isTest = isTest || window.Marquiz.isTest

this.contacts = contacts || window.Marquiz.contacts

this.params = window.Marquiz.params

this.element = null

this.frame = null // iframe DOM element

}

Значения по умолчанию могут передаваться при создании экземпляра класса или присваиваться от значений в классе Marquiz.

В данном классе происходит создание ссылки, которая ведет на опрос в приложении Quiz и после используется для добавления в iframe. Данный метод представлен в листинге 15.

Листинг 15 – Метод создания ссылки на опрос

createUrl(params = {}) {

let urlParams = {}

urlParams.href = window.location.href

urlParams.mode = 'modal'

if (this.isTest) {

urlParams.test = 1

urlParams = Object.assign(urlParams, this.contacts)

}

urlParams = Object.assign(urlParams, params)

// Cookies

const cookies = getCookies()

if (JSON.stringify(cookies) !== '{}') urlParams.cookies = JSON.stringify(cookies)

// GA

const ga = this.\_getGAClientId()

if (ga) urlParams.\_ga = ga

// Ads markers & utm tags

urlParams = Object.assign(urlParams, this.\_getUrlParams())

// Custom params

if (JSON.stringify(this.params) !== '{}') {

urlParams.params = JSON.stringify(this.params)

}

return `${this.\_getHost()}/${this.id}?${queryString.stringify(urlParams)}`

}

В данном методе происходит сбор всех добавленных параметров в поле params, значение cookies, Google аналитики и UTM тегов. Метод возвращает сформированную ссылку, которая после добавляется в iframe окно.

Также в данном классе реализован метод addParam для передачи данных в опрос в приложение Quiz. Данный метод представлен в листинге 16.

Листинг 16 – Метод передачи данных в опрос в приложение Quiz

addParam(key, value) {

this.params[key] = value

if (this.frame) {

this.frame.contentWindow.postMessage({

'addParam', this.params,

}, '\*')

}

}

Все виды опросов, кроме widget, могут открываться в модальном окне, и поэтому наследуются от класса Modal. Конструктор данного класса представлен в листинге 17.

Листинг 17 – Конструктор класса Modal

constructor(props) {

super(props)

this.frame = null

if (!Modal.modalsWrapperIsCreated) {

Modal.\_createModalsWrapper()

}

}

В данном методе происходит создание HTML-кода модального окна, куда после будут добавлять iframe окна добавляемых опросов.

Создание iframe окон осуществляется с помощью метода createModalContent, который представлен в листинге 18.

Листинг 18 – Метод создания iframe окон

createModalContent() {

if (this.frame) return

const alreadyCreatedModal = document.getElementById(`marquiz\_\_frame\_${this.id}`)

if (!alreadyCreatedModal) {

const frame = document.createElement('iframe')

frame.id = `marquiz\_\_frame\_${this.id}`

frame.className = 'marquiz\_\_frame'

frame.src = this.createUrl()

frame.scrolling = 'auto'

frame.frameBorder = 0

frame.allow = 'autoplay'

this.frame = frame

Modal.modal.appendChild(frame)

} else {

this.frame = alreadyCreatedModal

}

}

В данном методе в случае попытки повторного добавления iframe окна, ведущего на тот же опрос, она будет проигнорирована.

Для открытия опроса в модальном окне используется метод show, представленный в листинге 19.

Листинг 19 – Метод открытия модального окна

show() {

if (Modal.activeModal) return

if (!this.frame) {

this.createModalContent()

}

Modal.bg.classList.add('marquiz\_\_bg\_open')

Modal.activeModal = this

this.frame.classList.add('marquiz\_\_frame\_open')

}

Если данный метод будет вызван, когда какой-то опрос уже открыт в модальном окне, то новый опрос не откроется. В случае, если будет произведена попытка открыть опрос, но iframe ещё не был сформирован, то сначала произойдет формирование iframe окна, а затем опрос откроется.

Закрытие модального окна осуществляется с помощью метода \_closeModal, представленном в листинге 20.

Листинг 20 – Метод закрытия модального окна

static \_closeModal() {

Modal.bg.classList.remove('marquiz\_\_bg\_open')

Modal.modal.querySelectorAll('iframe').forEach((frame) => {

frame.classList.remove('marquiz\_\_frame\_open')

})

Modal.activeModal = null

}

В конструкторе класса Button происходит вызов метода \_render, который представлен в листинге 21.

Листинг 21 – Метод создания кнопки

\_render(props, target) {

const link = this.\_createButton(props)

this.element = link

if (target) {

target.removeAttribute('data-marquiz-id')

target.classList.add('marquiz\_\_container')

if (props.disableOnMobile) target.classList.add('marquiz\_\_container\_hide-on-mobile')

target.append(link)

}

}

В данном методе происходит создание HTML кода с помощью метода \_createButton, который затем добавляется в нужное место на сайт.

В конструкторе класса Banner также происходит вызов метода \_render, который представлен в листинге 22, но после вызывается метод \_setDelay, которые добавляет баннеру задержку перед появлением на сайте. Данный метод представлен в листинге 23.

Листинг 22 – Метод создания баннера

\_render(props) {

const { rounded, shadow, pulse, blicked, position, svgColor, closeColor, bgColor, textColor, disableOnMobile } = props

const container = document.createElement('div')

container.classList.add('marquiz-pops')

if (disableOnMobile) container.classList.add('marquiz-pops\_hide-on-mobile')

if (blicked) container.classList.add('marquiz-pops\_blicked')

if (rounded) container.classList.add('marquiz-pops\_rounded')

if (shadow) container.classList.add('marquiz-pops\_shadow')

if (pulse) container.classList.add('marquiz-pops\_pulse')

container.classList.add(`marquiz-pops\_${position}`)

if (position !== 'position\_top' && position !== 'position\_bottom') container.classList.add('marquiz-pops\_position')

container.setAttribute('style', `--marquiz-pop-text-color:${textColor}; --marquiz-pop-background-color:${bgColor}; --marquiz-pop-svg-color:${svgColor}; --marquiz-pop-close-color:${closeColor}; --marquiz-pop-color-pulse:${pulse}; --marquiz-pop-color-pulse-alpha:${shadow};`)

const link = this.\_createLink(props)

container.append(link)

this.element = link

document.body.append(container)

}

Листинг 23 – Метод добавления задержки баннеру перед появлением

\_setDelay(delay) {

let time

if (delay !== undefined) time = delay // for v2 quizzes

else time = this.element.dataset.marquizPopDelay ? parseInt(this.element.dataset.marquizPopDelay) : 0 // for v1 quizzes

setTimeout(() => {

this.element.parentNode.style.display = 'flex'

}, time \* 1000)

}

В конструкторе класса Inline сначала происходит создание кнопки, путем создания экземпляра класса Button, а затем происходит вызов метода \_render, который представлен в листинге 23.

Листинг 23 – Метод добавления опроса прямиком на сайт

render(props, target) {

const isV1 = target.classList.contains('marquiz\_\_button') // target for v1 quizzes is button, for v2 quizzes is container

const element = isV1 ? target.parentNode : target

const { height, width, radius, indent, } = isV1 ? element.dataset : props, frame = document.createElement('iframe')

frame.id = `marquiz\_\_inline-frame\_${props.id}`

frame.className = 'marquiz\_\_inline-frame'

frame.src = this.createUrl({ mode: 'inline' })

frame.scrolling = 'auto'

frame.frameBorder = '0'

frame.style.width = width ? `${width}px` : '100%'

frame.style.height = height ? `${height}px` : '600px'

frame.allow = 'autoplay'

const buttonOnMobile = isV1 ? this.\_hasButtonOnMobile(element) : !!props.buttonOnMobile

if (buttonOnMobile) element.classList.add('marquiz\_\_container\_has\_button')

if (indent) {

element.style.marginTop = `${indent}px`

element.style.marginBottom = `${indent}px`

}

if (!this.\_isMobile()) {

element.style.width = width ? `${width}px` : '100%'

element.style.height = height ? `${height}px` : '600px'

}

if (radius) {

frame.style.borderRadius = `${radius}px`

element.style.borderRadius = `${radius}px`

}

element.appendChild(frame)

element.style.height = 'auto'

this.element = element

if (!isV1) element.appendChild(this.button.element)

}

В данном методе происходит создание HTML кода, который затем добавляется на сайт.

### Новый код инициализации системы и добавления опроса

Для новой системы формирования опросов были разработаны новый код инициализации и коды добавления опросов. Код инициализации представлен в листинге 24.

Листинг 24 – Код инициализации системы

<script>

(function(w, d, s, o){

var j = d.createElement(s); j.async = true; j.src = '//script.marquiz.ru/v2.js';j.onload = function() {

if (document.readyState !== 'loading') Marquiz.init(o);

else document.addEventListener("DOMContentLoaded", function() {

Marquiz.init(o);

});

};

d.head.insertBefore(j, d.head.firstElementChild);

})(window, document, 'script', {

host: '//quiz.marquiz.ru',

id: '606e290b6f22d30044111f4c',

autoOpen: false,

autoOpenFreq: 'once',

openOnExit: false,

disableOnMobile: false

}

);

</script>

Данный код реализован в виде самовызывающейся функции, в которой происходит создание тега script на скачивание стороннего файла с системой формирования опросов, а также вызов метода init после полной загрузки сайта.

Код добавления кнопки представлен в листинге 25.

Листинг 25 – Код добавления кнопки

<div data-marquiz-id="606e290b6f22d30044111f4c"></div>

<script>(function(t, p) {window.Marquiz ? Marquiz.add([t, p]) : document.addEventListener('marquizLoaded', function() {Marquiz.add([t, p])})})('Button', {id: '606e290b6f22d30044111f4c', buttonText: 'Пройти тест', bgColor: '#0054dd', textColor: '#ffffff', rounded: true, shadow: 'rgba(0, 84, 221, 0.5)', blicked: true})</script>

При добавлении кнопки сначала указывается div с id нужного опроса. Данный div служит указателем, куда после будет добавлена кнопка системой формирования опросов. Затем происходит вызов функции add или подписка на событие marquizLoaded, в случае если файл ещё не был загружен. Опрос добавляемый в виде прямой вставки на сайт имеет такой же код, только при вызове функции add первым параметром будет «Inline».

Для добавления баннера используется другой код, представленный в листинге 26.

Листинг 26 – Код добавления баннера

<script>(function(t, p) {window.Marquiz ? Marquiz.add([t, p]) : document.addEventListener('marquizLoaded', function() {Marquiz.add([t, p])})})('Banner', {id: '606e290b6f22d30044111f4c', title: 'Пройти тест', text: 'Заголовок квиза', delay: 20, textColor: '#ffffff', bgColor: '#0054dd', svgColor: '#ffffff', closeColor: '#ffffff', bonusCount: 2, bonusText: 'Вам доступны бонусы и скидка', type: 'side', position: 'position\_top-left', rounded: true, shadow: 'rgba(0, 84, 221, 0)', blicked: true})</script>

В данном коде отсутствует блок div, потому что баннер закрепляется на экране и не нуждается в указателе, на который будет установлен.

Если пользователь хочет добавить опрос только для автооткрытия, то ему будет достаточно использовать код инициализации.

## Разработка нового способа формирования опросов в виде widget

### Реализация интерфейса в приложении Quiz

Основным компонентом, который отображает опрос в виде widget является WidgetPreview, который представлен в листинге 27.

Листинг 27 – Компонент WidgetPreview

<template lang="pug">

.widget-preview(

v-resize.throttle="onResize"

@click="openWidget"

:class="[`widget-preview\_${type}`, { 'widget-preview\_open': isWidgetOpen, 'widget-preview\_fixed': isFixed }]"

)

template(v-if="isPreviewVisible")

.widget-preview\_\_start-page(v-if="currentType === 'start-page'")

StartPagePreview(:isMobile="isMobile")

.widget-preview\_\_question(v-if="currentType === 'question'")

QuestionPreview

.widget-preview\_\_lead-form(v-if="currentType === 'lead-form'")

LeadFormPreview

.widget-preview\_\_results(v-if="currentType === 'results'")

ResultsPreview

.widget-preview\_\_result(v-if="currentType === 'result'")

ResultPreview(:isMobile="isMobile")

.widget-preview\_\_close(v-show="isWidgetOpen" @click.stop="closeWidget")

</template>

Здесь находится логика отображения нужных компонентов на каждом из этапов прохождения опроса. Также с помощью библиотеки vue-resize-directive реализована обработка изменения высоты. При изменении высоты опроса посылается postMessage в iframe на сайте, где высота iframe окна подстраивается под высоту опроса. Кроме этого, тут есть функции openWidget и closeWidget, которые отвечают за открытие и закрытие опроса на сайте клиента. Данные функции также посылают postMessage, на который система формирования опросов реагирует, открывает или закрывает опрос в виде widget.

Компонент StartPagePreview, который отображает свернутую версию стартовой страницы опроса, наследуется от компонента StartPage, то есть наследует все его свойства и методы. Компонент представлен в листинге 28.

Листинг 28 – Компонент StartPagePreview

<template lang="pug">

.start-page-preview(:class="{ 'start-page-preview\_large': !isMobile && isBg }")

template(v-if="!isMobile && isBg")

.start-page-preview\_\_bg(

v-if="bgType === 'image' && !!bgUrl || isVideoMobileBg"

:style="{ backgroundImage: `url(${this.bgUrl})` }"

)

VideoBg.start-page-preview\_\_video-bg(

v-if="isVideoBg && !isVideoMobileBg"

:loop="videoLoop"

:videoUrl="videoUrl"

:muted="isVideoMuted"

)

.start-page-preview\_\_content(

:style="{ backgroundImage: bgGradient }"

)

.start-page-preview\_\_header

.start-page-preview\_\_title

| {{ title }}

.start-page-preview\_\_assistant(v-if="avatarUrl")

img(:src="avatarUrl")

button.start-page-preview\_\_button.button.is-primary.is-blicked.has-light-shadow(

v-if="!isMobile"

:class="{'is-outline': buttonStyle.outline && this.spTheme !== 'expanded', 'is-square': buttonStyle.square }"

)

| {{ buttonText || $t('Пройти тест') }}

SmallPreview(v-else :hasButton="true" :imageUrl="bgUrl" :title="title")

</template>

В данном компоненте, если установлена на фон картинка или видео, то они отрисовываются, затем добавляется заголовок, консультант, если он был добавлен, и кнопка. В случае, если картинка или видео не были установлены, или опрос открыт на мобильном устройстве, то используется компонент SmallPreview.

Компонент ResultPreview, который отображает свернутую версию страницы с результатом опроса, наследуется от компонента Result, то есть наследует все его свойства и методы. Компонент представлен в листинге 29.

Листинг 29 – Компонент ResultPreview

<template lang="pug">

.result-preview(:class="{ 'result-preview\_large': !isMobile && (isVideoBg || isImageBg) }")

template(v-if="!isMobile && (isVideoBg || isImageBg)")

.result-preview\_\_bg(

v-if="isImageBg"

:style="{ backgroundImage: `url(${this.bgUrl})` }"

)

VideoBg.result-preview\_\_video-bg(

v-if="isVideoBg"

:loop="true"

:videoUrl="videoUrl"

:muted="mutedVideo"

)

.result-preview\_\_content(

:style="{ backgroundImage: bgGradient }"

)

.result-preview\_\_title

| {{ result.title }}

.result-preview\_\_discount

DiscountLabel -{{ $format.number(fixedDiscount) }} {{ discountCurrencySymbol }}

SmallPreview(v-else :imageUrl="bgUrl" :title="result.title" :bigRadius="true")

</template>

Здесь также, как в компоненте StartPagePreview сначала отрисовывается картинка или видео, затем заголовок результата, но в случае, если картинка или видео не были добавлены, или опрос открыт на мобильном устройстве, то используется компонент SmallPreview, который представлен в листинге 30.

Листинг 30 – Компонент SmallPreview

<template lang="pug">

.small-preview(:class="{ 'small-preview\_has-discount': discount, 'small-preview\_radius': bigRadius }")

.small-preview\_\_content

.small-preview\_\_bg(

v-if="imageUrl"

:style="{ backgroundImage: bgImage }"

)

button.small-preview\_\_button.button.is-primary.is-blicked.has-light-shadow(v-if="hasButton")

icon(:icon="['mdi', `mdi-arrow-up`]")

.small-preview\_\_title {{ title }}

.small-preview\_\_discount(v-if="discount")

DiscountLabel -{{ $format.number(discount) }} {{ discountCurrencySymbol }}

</template>

В данный компонент в качестве параметров передается картинка и заголовок, которые необходимо отобразить. Также из хранилища передается скидка, если она была установлена.

Компоненты LeadFormPreview, QuestionPreview и ResultsPreview отображают компонент BasePreview, в который просто передают разные параметры. Компонент BasePreview представлен в листинге 31.

Листинг 31 – Компонент BasePreview

<template lang="pug">

.base-preview

AssistantPreview(

v-if="info.assistant.name"

:assistant="info.assistant"

:text="title"

:isOpen="isWidgetOpen"

)

slot

TextPreview(v-else-if="!isWidgetOpen" :title="title")

slot

</template>

В данном компоненте прописана логика отображения опроса в свернутом виде с консультантом или без него. Если пользователь сервиса добавил консультанта в конструкторе опросов, то отображается компонент AssistantPreview, в который передается в качестве параметров данные консультанта и выводимый текст. Если консультанта не добавили, то используется компонент TextPreview, который просто выводит текст.

Компонент AssistantPreview представлен в листинге 32.

Листинг 32 – Компонент AssistantPreview

<template lang="pug">

.assistant-mobile.assistant-mobile\_preview(:class="{ 'assistant-mobile\_open': isOpen }")

.assistant-mobile\_\_block

.assistant-mobile\_\_person

.assistant-mobile\_\_avatar

img(:src="imageUrl" v-if="imageUrl")

.assistant-mobile\_\_text-container

.assistant-mobile\_\_text

.assistant-mobile\_\_name

.assistant-mobile\_\_title

| {{ getFirstName(assistant.name) }}

.assistant-mobile\_\_subtitle(v-if="assistant.title")

.assistant-mobile\_\_status

| {{ assistant.title }}

.assistant-mobile\_\_text-html {{ text }}

.assistant-mobile\_\_slot

slot

</template>

Компонент TextPreview представлен в листинге 33.

Листинг 33 – Компонент TextPreview

<template lang="pug">

.text-preview

.text-preview\_\_title {{ title }}

.text-preview\_\_slot

slot

</template>

### Реализация интерфейса в конструкторе опросов

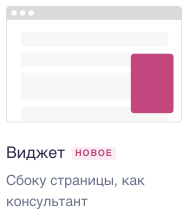
В конструкторе опросов был добавлен новый способ установки опросов в виде widget в уже имеющийся интерфейс. Новое представление в виде widget в конструкторе опросов представлен на рисунке 30.

Рисунок 30 – Новое представление в виде widget в конструкторе опросов

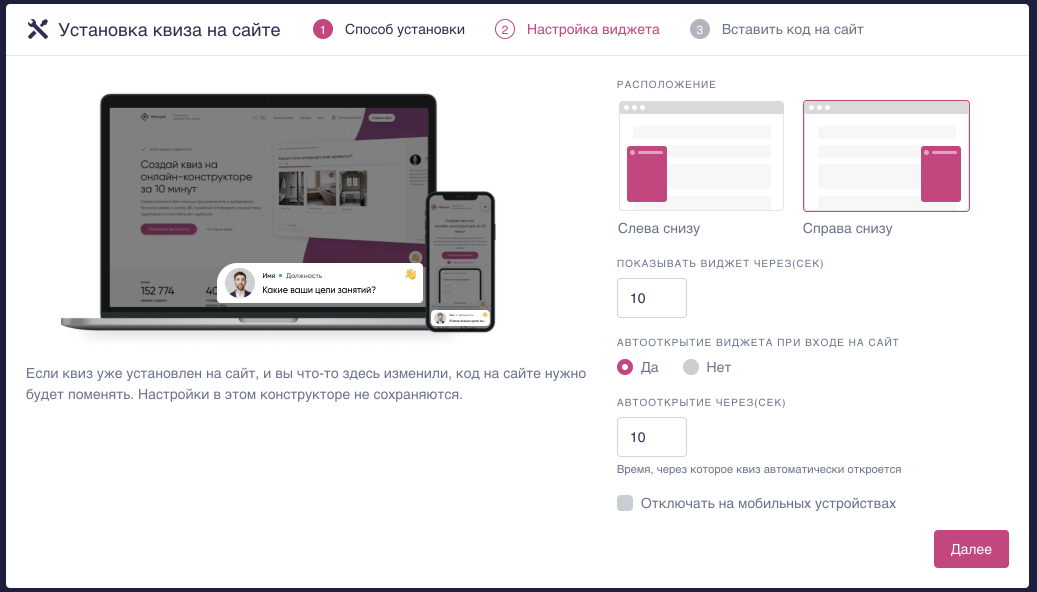
Интерфейс настроек нового способа формирования опроса представлены на рисунке 31.

Рисунок 31 – Интерфейс настроек нового представления

В разделе настроек можно установить расположение опроса справа снизу или слева снизу экрана, время задержки перед первым появлением опроса, задержку перед автооткрытием или полностью отключить на мобильных устройствах. Также было добавлено предварительное отображение свернутой версии опроса слева в блоке настроек.

Код добавления опроса в виде widget похож на код добавления баннера, поскольку также не указателя места установки, так как фиксируется на экране. Код добавления опроса в виде widget указан в листинге 34.

Листинг 34 – Код добавления опроса в виде widget

<script>(function(t, p) {window.Marquiz ? Marquiz.add([t, p]) : document.addEventListener('marquizLoaded', function() {Marquiz.add([t, p])})})('Widget', {id: '606e290b6f22d30044111f4c', position: 'right', delay: 10, autoOpen: 10})</script>

### Реализация класса Widget в системе формирования опросов

Для отображения нового представления опроса в виде widget был разработан класс Widget. Данный класс наследуется напрямую от класса Quiz, поскольку не нуждается в открытии в модальном окне. Конструктор данного класса представлен в листинге 35.

Листинг 35 – Конструктор класса Widget

constructor(props) {

super(props)

this.loaded = false

this.autoOpenDelay = props.autoOpen

this.appearDelayPassed = false

this.finish = false

this.\_setAppearDelay(props.delay)

this.\_addMessageListeners()

this.\_render(props)

}

В конструкторе происходит объявление полей и присваивание им значений по умолчанию. Также происходит вызов метода \_render, который создает HTML-код и добавляет опрос на сайт, вызов метода \_addMessegaListeners, в котором реализовано общение между системой формирования опросов и приложением Quiz, и также вызов метода \_setAppearDelay, который устанавливает задержку перед первым появлением опроса.

Метод \_render представлен в листинге 36.

Листинг 36 – Метод \_render

\_render({ position, disableOnMobile }) {

const widget = document.createElement('div')

widget.classList.add('marquiz-widget')

if (disableOnMobile) widget.classList.add('marquiz-widget\_hide-on-mobile')

this.element = document.createElement('div')

this.element.classList.add('marquiz-widget\_\_container')

this.element.classList.add(`marquiz-widget\_\_container\_${position}`)

this.element.classList.add(`marquiz-widget\_\_container\_${this.\_getWidgetType()}`)

widget.append(this.element)

const stick = document.createElement('div')

stick.classList.add('marquiz-widget\_\_stick')

this.element.append(stick)

const swiper = new SwipeToClose(this.element, stick, {

tapAdditionalSizes: {

top: '30px', bottom: '30px', left: '400%', right: '400%',

},

distance: '50%',

cbFinish: () => {

this.element.parentNode.removeAttribute('style')

this.\_close(widget)

},

...this.\_setBackgroundChanging(),

})

const iframe = document.createElement('iframe')

iframe.id = `marquiz-widget\_${this.id}`

iframe.className = 'marquiz-widget\_\_iframe'

iframe.src = this.createUrl({ mode: 'widget', 'widget-type': this.\_getWidgetType() })

iframe.scrolling = 'auto'

iframe.frameBorder = '0'

iframe.allow = 'autoplay'

this.frame = iframe

this.element.append(iframe)

const closeButton = document.createElement('span')

closeButton.classList.add('marquiz-widget\_\_close')

this.element.append(closeButton)

widget.addEventListener('click', (e) => {

if (e.target.classList.contains('marquiz-widget\_open')) {

swiper.close()

}

})

document.querySelector('body').append(widget)

}

В данном методе происходит создание HTML кода с последующим добавлением его на сайт.

Метод \_addMessageListeners представлен в листинге 37.

Листинг 37 – Метод \_addMessageListeners

\_addMessageListeners() {

window.addEventListener('message', ({ data }) => {

try {

data = JSON.parse(data)

if (data.id === this.id) {

switch (data.action) {

case 'loaded':

this.\_loadingHandler()

break

case 'openWidget':

this.\_open()

break

case 'closeWidget':

this.\_close()

break

case 'setPreviewHeight':

this.\_setHeight(data.height)

break

case 'finished':

this.finish = true

break

default:

break

}

}

} catch (e) {}

})

}

В данном методе происходит подписка на событие message, которое генерируется в момент отправки postMessage. При получении из приложения Quiz сообщения «loaded», происходит вызов метода \_loadedHanlder, который в свою очередь вызывает метод \_appear. При получении сообщения «openWidget» вызывается метод \_open для открытия опроса. При получении сообщения «closeWidget» вызывается метод \_close для сворачивания опроса. При получении сообщения «setPreviewHeight» происходит изменение высоты iframe окна, а при получении «finished», которое обозначает, что пользователь прошел опрос до конца, полю finished присваивается истинное значение.

Для первого появление опроса в виде widget на сайте используется метод \_appear, представленный в листинге 38.

Листинг 38 – Метод \_appear

\_appear(cb = () => {}) {

if (this.appearDelayPassed && this.loaded) {

this.element.style.display = 'flex'

this.element.classList.add('marquiz-widget\_\_container\_fade')

this.element.onanimationend = () => {

this.element.classList.remove('marquiz-widget\_\_container\_fade')

cb()

}

}

}

Для открытия опроса из свернутого вида используется метод \_open, представленный в листинге 39.

Листинг 39 – Метод \_open

\_open() {

this.element.parentNode.classList.add('marquiz-widget\_open')

this.element.classList.add('marquiz-widget\_\_container\_open-fade')

this.element.onanimationend = () => {

this.element.classList.remove('marquiz-widget\_\_container\_open-fade')

}

}

Для сворачивания опроса используется метод \_close, представленный в листинге 40.

Листинг 40 – Метод \_close

\_close() {

const appearDelay = 750

const elClasses = this.element.classList

const elParentClasses = this.element.parentNode.classList

if (this.\_getWidgetType() !== 'mobile') {

elClasses.add('marquiz-widget\_\_container\_closing')

this.element.onanimationend = () => {

elClasses.remove('marquiz-widget\_\_container\_closing')

elParentClasses.remove('marquiz-widget\_open')

if (this.finish) this.\_finish()

else this.\_appearAfterClosing(appearDelay)

}

} else {

this.sendDataToFrame('closeWidget')

elParentClasses.remove('marquiz-widget\_open')

if (this.finish) this.\_finish()

else this.\_appearAfterClosing(appearDelay)

}

}

Если опрос был пройден, то после сворачивания опроса, он больше не появится на экране. Для повторного прохождения необходимо обновить страницу.

## Настройка Wepback для поддержки старой и новой версий системы

При внедрении новой системы формирования опросов, не все пользователи сразу смогут перейти на неё. Для этого была реализована возможность одновременного использования как старого способа добавления, так и нового. Кроме этого, необходимо реализовать одновременную поддержку старой и новой версии системы формирования, поскольку способы добавления опросов сильно отличаются.

Структура приложения была разделена на 2 директории: v1 и v2. В директории v1 расположены файлы существующей системы, в директории v2 – разработанной системы. Для настройки одновременной сборки для двух директорий в конфигурации Webpack необходимо изменить свойство entry для настройки входа и свойство output для настройки выхода [6] Под входом подразумевается начальный файл, который используется как основной при разработке, и в который импортируются все остальные файлы. Под выходом подразумевается конечный один файл, где будет собрано все приложение целиком.

В обеих директориях содержится файл index.js, который служит входом для сборки файла, который будет устанавливаться на сайт клиентов. Настройка входа для сборки файла указаны в листинге 41.

Листинг 41 – Настройка входа для сборки файла

entry: {

v1: './src/v1/index.js',

v2: './src/v2/index.js'

},

Также была изменена настройка выхода для сборки файлов. Необходимо, чтобы было сформировано два файла, со старой и новой системой. Настройка выхода из сборки файла указана в листинге 42.

Листинг 42 – Настройка выхода для сборки файла

output: {

filename: '[name].js',

path: path.resolve(\_\_dirname, 'dist'),

publicPath: publicPath || `http://localhost:${port}/`

},

С данной настройкой после процесса сборки будет сформировано 2 файла: v1.js и v2.js.

Полная конфигурация Webpack указана в приложении Б.

## Тестирование разработанной системы

Цель тестирования системы заключается в проверке соответствия ожидаемого и реального её поведения. По итогам тестирования, если присутствуют ошибки, производится отладка системы – выявление причин ошибок и их устранение.

Для тестирования новой разработанной системы формирования опросов был создан сайт, указанный в приложении В. Для демонстрации тестирования и его результатов была создана таблица программы и методики испытаний, указанные в таблице 1.

Таблица 1 – Программа и методика испытаний

|  |  |  |  |
| --- | --- | --- | --- |
| Цель тестирования | Сценарий тестирования | Результат при успешном прохождении | Результат испытания |
| Протестировать корректность формирования опроса в виде кнопки, добавленной старым способом | 1) Установить на тестируемый сайт опрос в виде кнопки старым способом  2) Открыть сайт | Кнопка сформирована. При нажатии на кнопку открывается модальное окно с опросом | Успешно |
| Протестировать корректность формирования опроса в виде баннера, добавленным старым способом | 1) Установить на тестируемый сайт опрос в виде баннера старым способом  2) Открыть сайт | Баннер сформирован. При нажатии на баннер открывается модальное окно с опросом | Успешно |
| Протестировать корректность формирования опроса в виде прямой вставки на сайт, добавленным старым способом | 1) Установить на тестируемый сайт опрос в виде прямой вставки на сайт старым способом  2) Открыть сайт | Опрос сформирован и добавлен на сайт | Успешно |
| Протестировать корректность формирования опроса в виде кнопки, добавленной новым способом | 1) Установить на тестируемый сайт опрос в виде кнопки новым способом  2) Открыть сайт | Кнопка сформирована. При нажатии на кнопку открывается модальное окно с опросом | Успешно |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| Протестировать корректность формирования опроса в виде баннера, добавленным новым способом | 1) Установить на тестируемый сайт опрос в виде баннера новым способом  2) Открыть сайт | Баннер сформирован. При нажатии на баннер открывается модальное окно с опросом | Успешно |
| Протестировать корректность формирования опроса в виде прямой вставки на сайт, добавленным новым способом | 1) Установить на тестируемый сайт опрос в виде прямой вставки на сайт новым способом  2) Открыть сайт | Опрос сформирован и добавлен на сайт | Успешно |
| Протестировать корректность формирования опроса в виде widget | 1) Установить на тестируемый сайт опрос в виде widget  2) Открыть сайт | Опрос в виде widget сформирован и появился на сайте. При нажатии на опрос, он открывается | Успешно |
| Протестировать корректность задержки перед первым появлением опроса в виде widget | 1) Настроить в конструкторе опросов задержку в 5 секунд перед появлением  2) Установить на тестируемый сайт опрос  3) Открыть сайт | Опрос сформирован и появился на странице через 5 секунд после загрузки сайта | Успешно |
| Протестировать корректность задержки перед автооткрытием опроса в виде widget | 1) Настроить в конструкторе опросов задержку в 5 секунд для автооткрытия опроса  2) Установить на тестируемый сайт опрос  3) Открыть сайт | Опрос сформирован и появился на странице. Через 5 секунд после появления опрос автоматически открылся | Успешно |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| Протестировать корректность настройки места появления опроса в виде widget | 1) Настроить в конструкторе место появления опроса слева снизу  2) Установить на тестируемый сайт опрос  3) Открыть сайт | Опрос сформирован и появился на странице слева снизу экрана | Успешно |
| Протестировать корректность отключения опроса в виде widget на мобильных устройствах | 1) Настроить в конструкторе отключение опроса на мобильных устройствах  2) Установить на тестируемый сайт опрос  3) Открыть сайт на мобильном устройстве | Опрос не появился на сайте | Успешно |
| Протестировать корректность отображение всех этапов опроса в виде widget в свернутом виде | 1) Установить опрос на сайт  2) Открыть сайт  3) Открыть опрос и перейти на этап вопросов  5) Свернуть опрос  6) Открыть опрос и перейти на этап с результатом  7) Свернуть опрос  8) Открыть опрос и перейти на форму контактов  9) Свернуть опрос  10) Открыть опрос и заполнить форму контактов  11) Свернуть опрос | Опрос сформирован и появился на странице. Свернутый опрос на каждом из этапов имеет соответствующий дизайну внешний вид. После полного прохождения опроса, он больше не появляется на странице. | Успешно |

Продолжение таблицы 1

|  |  |  |  |
| --- | --- | --- | --- |
| Протестировать корректность установки опроса во Vue приложение | 1) Создать базовое Vue-приложение  2) Добавить код инициализации в index.html  3) Добавить код добавления в виде отдельного компонента  4) Запустить приложение | Опрос сформирован и появился на странице. | Успешно |
| Протестировать корректность установки опроса в React приложение | 1) Создать базовое React приложение  2) Добавить код инициализации в index.html  3) Добавить код добавления в виде отдельного компонента  4) Запустить приложение | Опрос сформирован и появился на странице. | Успешно |

## Вывод по разделу

В данном разделе описан процесс разработки спроектированной системы и нового представления опроса в виде widget, а также процесс тестирования разработанной системы.

# Социально-экономическая эффективность

## Технико-экономическое обоснование проекта

Для оценки эффективности разработанной системы необходимо принять во внимание затраты на ее создание и последующую эксплуатацию.

Целью экономической части является расчет эффективности от внедрения новой системы формирования опросов на сайтах клиентов сервиса «Marquiz».

Основной задачей данного раздела является определение целесообразности разработки новой системы формирования опросов и ее экономической и технической эффективности, а также экономического эффекта от ее внедрения в компании «Marquiz».

Для определения целесообразности разработки и внедрения, а также их экономических и технических эффективностей, рассчитаны технико-экономические показатели.

Для проведения сравнительного анализа создан сайт с минимальным количеством визуальных элементов. На сайт поочередно установлены опросы существующей и разработанной систем. Анализ проведен с помощью сервиса Google Page Speed и представлен в таблице 2.

Таблица 2 – Основные показатели сравнительного анализа вариантов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Показатель | Ед.изм. | Варианты | | Результат сравнения |
| Существующий | Разработанный |
| Затраты времени на первую отрисовку контента | Сек. | 1.9 | 0.8 | сокращение времени в 2.375 раза |
| Затраты времени на полную загрузку сайта | Сек. | 2.1 | 1.0 | сокращение времени в 2.1 раз |
| Затраты времени на загрузку сайта, готового к взаимодействию | Сек. | 2.2 | 2.0 | сокращение времени на 0.2 секунды |

## Определение капитальных вложений

К капитальным вложениям на разработку и внедрение программного обеспечения относятся затраты на приобретение технических средств, программных средств, транспортировку, разработку и пуско-наладочные мероприятия.

Расчет капитальных вложений производим по формуле:

, (1)

где ЦТС – цена приобретения технических средств, руб.;

ЦПС – цена приобретения программных средств, руб.;

Н – накладные расходы, руб.;

СР – стоимость разработки программного обеспечения, руб.;

СПН – стоимость пуско-наладочных мероприятий, руб.

Цена приобретения технических средств и цена приобретения программных средств равны нулю, так как не требуется приобретение дополнительных технических и программных средств, поскольку разработка новой системы формирования опросов выполняется уже на имеющихся средствах компании.

Накладные расходы также равны нулю, поскольку не требуется дополнительных приобретений.

Стоимость разработки новой системы формирования опросов на сайты складывается из затрат на оплату труда работника, равную 80000 руб./мес. На разработку системы потребовалось 2 месяца, следовательно стоимость разработки составляет 160000 руб.

Расчет капитальных затрат составляет:

руб.

## Расчет экономического эффекта

Экономический эффект рассмотрен как увеличение выручки компании «Marquiz» за счет привлечения новых пользователей и возвращения текущих пользователей, которые отказались от сервиса «Marquiz» из-за низкой скорости загрузки сайта.

По предоставленным данным компании «Marquiz» известно, что 0.84% от текущего количества пользователей отказались от сервиса из-за низкой скорости загрузки их сайта, что составляет примерно 584 пользователя. Около 30% пользователей привлечено в первый месяц, и около 50% за второй месяц. Таким образом, если 175 пользователей в первый месяц оплатят минимальный тариф в 600 рублей, то компания заработает 105000 рублей. За 2 месяца – 280200 рублей. Поскольку тарифы оплачиваются ежемесячно, то выручка за месяц увеличится на 175200 рублей.

Кроме увеличения скорости загрузки в новой системе формирования опросов также разработано новое представление опроса в виде widget, которое компания собирается использовать для развития на рынке интернет-магазинов. Целью компании является привлечь 50 различных интернет-магазинов в первый месяц. Если учитывать, что интернет-магазины оплатят минимальный тариф в 600 рублей, то компания заработает в первый месяц 30000 рублей.

Поскольку доход от новых пользователей на рынке интернет-магазинов рассчитать сложно, он не будет включен в расчет экономического эффекта. В первый месяц выручка компании увеличится на 105000 рублей, в последующие месяцы на 175200 рублей. Общий экономический эффект, рассчитанный за 2 месяца, будет равен 280200 рублей.

Срок окупаемости капитальных вложений – отрезок времени, за который средства, затраченные на разработку и внедрение системы, будут возвращены через экономический эффект от внедрения.

Расчет срока окупаемости производится по формуле:

, (2)

где – капитальные вложения на разработку системы, руб.;

Э – экономический эффект от внедрения системы, руб.

Срок окупаемости равен:

# Информационная безопасность и надежность

В компании Marquiz клиентские приложения хранятся в Amazon Web Services, серверные – в Heroku. Защиту приложений предоставляют указанные сервисы.

Amazon Web Services (AWS) предоставляет надежные, масштабируемые и недорогие вычислительные ресурсы для размещения приложений. AWS позволяет контролировать, где хранятся данные, кто может получить к ним доступ, а также позволяет свободно шифровать и перемещать данные. Помимо этого, используются и дополнительные уровни шифрования, например межрегиональный пиринговый трафик между всеми VPC и подключения TLS между сервисом и клиентом и между несколькими сервисами [7].

В AWS дополнительно подключен сервис защиты от DDoS атак – AWS Shield. Он защищает приложения, работающие на AWS, обеспечивает непрерывное обнаружение и автоматическую линейную нейтрализацию атак, сокращая время простоя и задержку приложений и избавляя пользователя от необходимости обращаться в службу поддержки AWS в случае DDoS-атак [8].

Heroku – платформа облачных приложений, которая фокусируется на управлении инфраструктурой, масштабировании и безопасности. Физическая инфраструктура Heroku размещается и управляется в AWS, но, кроме этого, в Heroku используется процессор Braintree для шифрования и обработки платежей, предоставляется защита от DDoS атак, спуфинга и снифинга. Помимо этого, платформа автоматически создает резервные копии приложений и баз данных и восстанавливает приложения в случае сбоя [9].

# ЗАКЛЮЧЕНИЕ

В результате выпускной квалификационной работы достигнута поставленная цель, спроектирована архитектура и разработано универсальное программное обеспечение для формирования опросов на сайтах клиентов компании «Marquiz».

Для достижения цели выполнены следующие задачи:

* исследована существующая система формирования опросов, выявлены достоинства и недостатки;
* разработан новый алгоритм формирования опросов на сайте;
* разработан проект архитектуры новой системы;
* разработано программное обеспечение спроектированной системы формирования опросов;
* разработано новое представление опроса в виде widget;
* проведен расчет экономических показателей внедрения.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Крис Ричардсон, Микросервисы. Паттерны разработки и рефакторинга. Санкт Петербург, 2020 г. – 544 с.
  2. Применение микросервисной архитектуры: плюсы, минусы, подводные камни. [Электронный ресурс] URL: <https://simpleone.ru/blog/primenenie-mikroservisnoj-arhitektury-plyusy-minusy-podvodnye-kamni/> (дата обращения 16.05.21)
  3. Рендеринг веб-сайтов 101. [Электронный ресурс] URL: <https://habr.com/ru/post/484900/> (дата обращения 16.05.21)
  4. Асинхронная загрузка javascript файлов. Ускорение и оптимизация процесса, увеличение производительности [Электронный ресурс] URL: <https://habr.com/ru/post/56974/> (дата обращения 17.05.21)
  5. AS-IS модель | Глоссарий ПитерСофт [Электронный ресурс] URL: <https://piter-soft.ru/knowledge/glossary/process/as-is-model.html> (дата обращения 26.05.21)
  6. Concepts | webpack [Электронный ресурс] URL: <https://webpack.js.org/concepts/> (дата обращения 01.06.21)
  7. Безопасность облака – Amazon Web Services (AWS) [Электронный ресурс] URL: <https://aws.amazon.com/ru/security/> (дата обращения 06.06.21)
  8. AWS Shield – Amazon Web Services (AWS) [Электронный ресурс] URL: <https://aws.amazon.com/ru/shield/> (дата обращения 06.06.21)
  9. Heroku Security | Heroku [Электронный ресурс] URL: <https://www.heroku.com/policy/security> (дата обращения 06.06.21)

# ПРИЛОЖЕНИЕ А

Техническое задание

СОГЛАСОВАНО

Генеральный директор компании ООО «Марквиз»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.А. Балачков

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021

УТВЕРЖДАЮ

Зав. кафедрой: д-р экон. наук, проф.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р. А. Файзрахманов

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Проектирование архитектуры и разработка универсального программного обеспечения для формирования системы опросов на клиентских сайтах компании

Техническое задание на 8 листах

Разработал студент гр. РИС-17-1б

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Селиверстов Е. В.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Действует с 4 февраля 2021 г.

**Содержание**

[1. Общие сведения 88](#_Toc74594008)

[1.1. Полное наименование системы и ее условное обозначение 88](#_Toc74594009)

[1.2. Шифр темы 88](#_Toc74594010)

[1.3. Наименование предприятия разработчика и заказчика 88](#_Toc74594011)

[1.4. Основание для разработки 88](#_Toc74594012)

[1.5. Источники финансирования 88](#_Toc74594013)

[1.6. Плановые сроки начала и окончания работ 89](#_Toc74594014)

[1.7. Порядок оформления и предъявления результатов работ 89](#_Toc74594015)

[2. Назначение и цели создания системы 89](#_Toc74594016)

[2.1. Назначение системы 89](#_Toc74594017)

[2.2. Цели и задачи создания системы 89](#_Toc74594018)

[3. Требования к системе 90](#_Toc74594019)

[3.1. Требования к системе в целом 90](#_Toc74594020)

[3.1.1. Требования к надежности системы 90](#_Toc74594021)

[3.1.2. Требования к безопасности 90](#_Toc74594022)

[3.1.3. Требования по эргономике 90](#_Toc74594023)

[3.1.4. Дополнительные требования 90](#_Toc74594024)

[3.2. Требования к функциям 90](#_Toc74594025)

[3.3. Требования к видам обеспечения 91](#_Toc74594026)

[3.3.1. Требования к информационному обеспечению 91](#_Toc74594027)

[3.3.2. Требования к техническому обеспечению 91](#_Toc74594028)

[3.3.3. Требования к программному обеспечению 91](#_Toc74594029)

[4. Состав и содержание работ по созданию системы 91](#_Toc74594030)

[4.1. Этапы создания системы 92](#_Toc74594031)

[5. Порядок контроля и приемки 92](#_Toc74594032)

[6. Требования к документированию 92](#_Toc74594033)

[7. Источники разработки 93](#_Toc74594034)

# Общие сведения

## Полное наименование системы и ее условное обозначение

Настоящее Техническое задание определяет требования и порядок проектирование архитектуры и разработку универсального программного обеспечения для формирования опросов на клиентских сайтах.

## Шифр темы

Настоящее Техническое задание разработано в рамках выполнения выпускной квалификационной работы «Проектирование архитектуры и разработка универсального программного обеспечения для формирования системы опросов на клиентских сайтах». Кафедра ИТАС, специальность 09.03.04 – Программная инженерия (ПИ).

## Наименование предприятия разработчика и заказчика

Заказчик: ООО «Марквиз».

Исполнитель: студент группы РИС-16-1б Селиверстов Евгений Владимирович.

## Основание для разработки

Основанием для проектирования новой архитектуры и разработки системы является желание компании внедрить новой способ представления опроса в виде widget на сайтах клиентов и потребности клиентов компании.

## Источники финансирования

Финансирование разработки осуществляется из средств компании ООО «Марквиз».

## Плановые сроки начала и окончания работ

Плановые сроки начала работ – 20 марта 2021 года. Окончания работ – 20 мая 2021 года.

## Порядок оформления и предъявления результатов работ

Работы по созданию системы должны осуществляться в порядке, установленном в разделе 4 настоящего Технического задания.

Приемка результатов работ должна проводиться в порядке, установленном в разделе 5 настоящего Технического задания.

# Назначение и цели создания системы

## Назначение системы

Система предназначена для формирования опросов на странице клиентов компании «Марквиз».

## Цели и задачи создания системы

Целью данной работы является проектирование архитектуры иразработка универсального программного обеспечения для формирования опросов на сайтах клиентов компании.

Для достижения цели необходимо выполнить следующие задачи:

* исследовать существующую систему формирования опросов, выявить её достоинства и недостатки;
* разработать проект архитектуры новой системы;
* разработать программное обеспечение спроектированной системы для формирования опросов;
* разработать новое представление опроса в виде widget;

# Требования к системе

## Требования к системе в целом

Общие требования к системе:

* выполнение всех функций, предусмотренных настоящим документом в пункте 3.2;
* выполнение требований (численности, квалификации персонала; надежности; безопасности, эргономике и дополнительных требований при наличии) к системе, предусмотренных настоящим документом в пунктах 3.1.1., 3.1.2., 3.1.3., 3.1.4., 3.1.5. соответственно.

### Требования к надежности системы

Система должна безошибочно выполнять все требуемые функции, и быть пригодной для эксплуатации.

### Требования к безопасности

Система не должна собирать и передавать личные данные в сервисы сторонние от компании «Марквиз».

### Требования по эргономике

Для использования системы должен быть установлен любой браузер и создан сайт или SPA-приложение, на который можно добавить опрос.

### Дополнительные требования

Дополнительных требований к системе не предусмотрено.

## Требования к функциям

Основными функциями системы должны быть:

* формирование кнопки на сайте для открытия опроса в модальном окне;
* формирование баннера на сайте для открытия опроса в модальном окне;
* формирование опроса прямо на сайте в виде отдельного блока;
* формирование опроса на сайте в виде widget;
* автооткрытие опроса в случае, если пользователь сервиса указал это в настройках;
* открытие и закрытие модального окна с опросом;
* формирование опросов, добавленных старым способом.

## Требования к видам обеспечения

### Требования к информационному обеспечению

Для взаимодействия системы формирования опросов с приложением Quiz требуется использовать метод postMessage, генерирующий событие «message».

### Требования к техническому обеспечению

Для работы системы формирования опросов требуется наличие операционной системы, поддерживающей выход в интернет через браузеры: Google Chrome, Mozilla Firefox, Opera, Safari, Яндекс Браузер или популярные мобильные браузеры.

## Требования к программному обеспечению

Для работы системы к программному обеспечению требуется использовать браузер, который поддерживает JavaScript версией выше ES6.

# Состав и содержание работ по созданию системы

## Этапы создания системы

Этапы создания системы приведены в таблице 3.

Таблица А.1 – Этапы создания системы

|  |  |  |
| --- | --- | --- |
| Наименование этапа | Начало | Конец |
| Исследование предметной области | 20.03.2021 | 25.03.2021 |
| Проектирование системы | 26.03.2021 | 05.04.2021 |
| Разработка системы | 06.04.2021 | 15.05.2021 |
| Представление работы на проверку | 16.05.2021 | 20.05.2021 |

# Порядок контроля и приемки

Сдача-приемка работ производится поэтапно, в соответствии с Календарным графиком представленном в разделе 4, и осуществляется разработчиками приложения.

Приемка этапа заключается в рассмотрении и оценке проведенного объема работ и предъявленной технической документации в соответствии с требованиями настоящего технического задания.

Ответственность за организацию и проведение приемки системы должен нести заказчик. Приемка системы должна производиться по завершению приемки всех задач системы. При этом необходимо предоставить обеспечение материальной частью (технические средства), проектной документацией и специально выделенным персоналом.

Заказчик должен предъявлять систему ведомственной приемочной комиссии, при этом он обязан обеспечить нормальные условия работы данной комиссии в соответствии с принятой программой приемки.

Завершающим этапом при приемке системы должно быть составление акта приемки.

# Требования к документированию

Требований к программной документации не предъявляется.

# Источники разработки

В качестве источников по разработке данного технического задания были использованы:

* ГОСТ 34.602-89 «Техническое задание на создание АС»;
* ГОСТ 34.003-90 «Автоматизированные системы. Термины и определения»;
* документация и материалы на аналогичные системы;
* документация из курсовых и дипломных работ студентов ПНИПУ на аналогичную тему.

# ПРИЛОЖЕНИЕ Б

Полная конфигурация Webpack

webpack.config.js

const path = require('path');

const webpack = require('webpack');

const HtmlWebpackPlugin = require('html-webpack-plugin');

const port = process.env.PORT || 82

let publicPath = process.env.PUBLIC\_PATH;

if (!publicPath && process.env.DEPLOY\_URL) publicPath = `${process.env.DEPLOY\_URL}/`; // Variable from Netlify

module.exports = {

mode: 'development',

entry: {

v1: './src/v1/index.js',

v2: './src/v2/index.js'

},

output: {

filename: '[name].js',

path: path.resolve(\_\_dirname, 'dist'),

publicPath: publicPath || `http://localhost:${port}/`

},

devServer: {

contentBase: path.join(\_\_dirname, 'dist'),

hot: true,

port: port,

hotOnly: true,

host: '0.0.0.0',

headers: {

"Access-Control-Allow-Origin": "\*"

}

},

plugins: [

new HtmlWebpackPlugin({

title: 'Hot Module Replacement'

}),

new webpack.HotModuleReplacementPlugin()

],

module: {

rules: [

{

test: /\.js$/,

exclude: /(node\_modules|bower\_components)/,

use: {

loader: 'babel-loader',

options: {

presets: ['@babel/preset-env']

}

}

},

{

test: /\.css$/,

use: [

'style-loader',

'css-loader'

]

},

{

test: /\.styl$/,

use: [

'style-loader',

'css-loader',

'postcss-loader',

'stylus-loader',

]

},

{

test: /\.(woff2?|eot|ttf|otf)(\?.\*)?$/,

loader: 'url-loader',

options: {

limit: 10000,

name: 'fonts/[name].[hash:7].[ext]'

}

},

{

test: /\.(png|jpe?g|gif|svg)(\?.\*)?$/,

loader: 'url-loader',

options: {

limit: 10000,

name: 'img/[name].[hash:7].[ext]'

}

},

]

}

};

# ПРИЛОЖЕНИЕ В

Сайт для тестирования

index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=1200px, initial-scale=1.0">

<link rel="stylesheet" href="./styles/style.css">

<title>Магазин игровых видеокарт HYPERGC</title>

<!-- Marquiz script start -->

<script src="//script.marquiz.ru/v1.js" type="application/javascript"></script>

<script>

document.addEventListener("DOMContentLoaded", function() {

Marquiz.init({

host: '//quiz.marquiz.ru',

id: '609a3dd14d4756001aed0e19',

autoOpen: false,

autoOpenFreq: 'once',

openOnExit: false

});

});

</script>

<!-- Marquiz script end -->

</head>

<body>

<div id="root">

<header>

<div class="wrapper">

<h1><a href="/">HYPERGC</a></h1>

<ul>

<li><a href="" class="scroll-to scroll-to-welcome" data-scroll-to="welcome">Главная</a></li>

<li><a href="" class="scroll-to scroll-to-buyers" data-scroll-to="buyers">Наши преимущества</a></li>

<li><a href="" class="scroll-to scroll-to-graphic-cards" data-scroll-to="graphic-cards">Видеокарты</a></li>

<li><a href="" class="scroll-to scroll-to-delivery-and-payment" data-scroll-to="delivery-and-payment">Доставка и оплата</a></li>

<li><a href="" class="scroll-to scroll-to-reviews" data-scroll-to="reviews">Отзывы</a></li>

<li><a href="" class="scroll-to scroll-to-questions" data-scroll-to="questions">Задать вопрос</a></li>

<li><a href="" class="scroll-to scroll-to-checkout" data-scroll-to="checkout">Оформить заказ</a></li>

<li><a href="" class="scroll-to scroll-to-contacts" data-scroll-to="contacts">Контакты</a></li>

</ul>

</div>

</header>

<div class="pages">

<section class="welcome">

<div class="wrapper">

</div>

</section>

<section class="buyers">

<div class="wrapper">

<h2>Наши преимущества</h2>

<div class="marquiz\_\_container">

<a class="marquiz\_\_button marquiz\_\_button\_blicked marquiz\_\_button\_rounded marquiz\_\_button\_shadow" href="#popup:marquiz\_609a3dd14d4756001aed0e19" data-fixed-side="" data-alpha-color="rgba(211, 64, 133, 0.5)" data-color="#d34085" data-text-color="#ffffff">Пройти тест</a>

</div>

<div class="sections">

<div class="info">

<h2>+МАЙНИНГ</h2>

<div class="text">

<p>Абсолютно все продаваемые в нашем магазине видеокарты <span>подходят для майнинга</span>.</p>

<p>В характеристиках каждой видеокарты указано сколько она cпособна выдавать <span>мегахешей в секунду</span>.</p>

</div>

</div>

<div class="info">

<h2>+КАЧЕСТВО</h2>

<div class="text">

<p>Все продаваемые нами видеокарты проходят строжайший <span>контроль качества</span>.</p>

<p>Для всех продаваемых видеокарт <span>предоставляется гарантия</span> не только от производителя, но и от нашего магазина.</p>

</div>

</div>

<div class="info">

<h2>+ОПТ</h2>

<div class="text">

<p>Все представленные в каталоге видеокарты мы продаем не только в розницу, но и <span>оптом (от 5 штук)</span>.</p>

<p>Клиентам, которые осуществляют оптовую покупку в нашем магазине, предоставляются <span>уникальные скидки</span>.</p>

</div>

</div>

</div>

</div>

</section>

<section class="delivery-and-payment">

<div class="wrapper">

<h2>Доставка и оплата</h2>

<div class="content">

<div class="delivery">

<img src="assets/delivery-truck.svg" alt="wallet">

<div class="description">

<p>Мы осуществляем доставку по всем странам СНГ с помощью следующих транспортных компаний: CDEK, Почта Росии, Boxberry, DPD, Деловые линии.</p>

<p>Срок доставки товара зависит от выбранной транспортной компании и места назначения.</p>

</div>

</div>

<div class="payment">

<img src="assets/wallet.svg" alt="wallet">

<div class="description">

<p>Полная оплата возможна как безналичным платежом после оформления заказа, так и при получении товара в пункте выдаче.</p>

<p>Для окончательного оформления заказа необходимо внести предоплату в размере 30% от суммы заказа.</p>

</div>

</div>

</div>

</div>

</section>

<section class="questions">

<div class="wrapper">

<h2>Остались вопросы?</h2>

<p>Если у вас остались вопросы, вы можете задать их с помощью формы ниже, и мы в ближайшие сроки ответим на ваш вопрос, прислав ответ на указанный вами <span>email</span>.</p>

<div class="form">

<form class="form-question">

<div class="row-1">

<div class="input-box">

<p>Имя</p>

<input type="text" placeholder="Ваше имя" pattern="{3,}" required>

</div>

<div class="input-box">

<p>Почта</p>

<input type="email" placeholder="Ваш email" pattern="{3,}" required>

</div>

</div>

<div class="row-2">

<div class="input-box">

<p>Вопрос</p>

<textarea placeholder="Задайте ваш вопрос" pattern="{3,}" required></textarea>

</div>

</div>

<div class="row-3">

<button class="btn-primary">Отправить вопрос</button>

</div>

</form>

</div>

</div>

</section>

<footer>

<div class="wrapper">

<p>Copyright © 2021 <span>HYPERGC</span>. Все права защищены.</p>

</div>

</footer>

</div>

</div>

</body>

</html>