МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка программных компонентов системы обработки документов

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.04.2024.308-334.ВКР

Научный руководитель,
ст. преподаватель кафедры СП
_____ П.Г. Верман

Автор работы,
студент группы КЭ-403
_____ Е.С. Боков

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-403
Бокову Егору Сергеевичу,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. № 764-13/12)
Разработка программных компонентов системы обработки документов.
2. **Срок сдачи студентом законченной работы:** 03.06.2024 г.
3. **Исходные данные к работе**
3.1. HTML5 Basics For Everyone Tired Of Reading About Deprecated Code. [Электронный ресурс] URL: https://html.com/html5/ (дата обращения: 06.02.2024 г.).
3.2. Современный учебник JavaScript. [Электронный ресурс] URL: https://learn.javascript.ru (дата обращения: 06.02.2024 г.).
4. **Перечень подлежащих разработке вопросов**
4.1. Провести обзор литературы.
4.2. Выполнить проектирование программных компонентов.
4.3. Выполнить реализацию программных компонентов.
4.4. Провести тестирование программных компонентов.
5. **Дата выдачи задания:** 29.01.2024 г.

Научный руководитель,
ст. преподаватель кафедры СП П.Г. Верман

Задание принял к исполнению Е.С. Боков

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

С каждым годом данные становятся все более ценным ресурсом для любой организации, будь то коммерческая компания или государственное учреждение. В центре процесса анализа данных часто находится Excel – мощный инструмент для работы с таблицами, который уже многие годы остается неотъемлемой частью офисной жизни. Несмотря на его универсальность, современные объемы информации и требования к скорости обработки данных заставляют искать новые подходы к использованию этого инструмента.

Ручная работа с большими массивами данных в Excel не только трудоемка, но и подвержена риску ошибок, которые могут стать причиной неверных управленческих решений. Кроме того, возникает потребность в более тесной интеграции Excel с другими IT-системами предприятия, что требует специализированных технических решений.

Разработка компонентов для автоматизации процессов работы с Excel документами и их интеграция в корпоративные системы открывает новые возможности: от ускорения обработки данных до повышения точности информации и улучшения взаимодействия между подразделениями. Это повышает эффективность работы отдельного сотрудника и способствует более глубокой аналитике текущих процессов в компании.

С помощью специализированных компонентов можно унифицировать методы работы с данными, что в свою очередь приведет к уменьшению ошибок и расхождений в данных. Это также способствует более эффективному обучению новых сотрудников и облегчает процесс внедрения новых технологий в рабочие процессы.

Постановка задачи

Целью выпускной квалификационной работы является разработка программных компонентов системы обработки документов.

Для достижения поставленной цели необходимо решить следующие задачи:

1) провести обзор литературы;
2) выполнить проектирование программных компонентов;
3) выполнить реализацию программных компонентов;
4) провести тестирование программных компонентов.

Структура и содержание работы

Работа состоит из введения, четырех глав, заключения и списка литературы. Объем работы составляет 74 страницы, объем списка литературы – 17 источников.

В первой главе описывается обзор предметной области и существующих решений для обработки и визуализации данных в формате excel. Основной предмет разработки – это создание программных компонентов для обработки excel-файлов, что требует учета особенностей данных форматов.

Вторая глава посвящена проектированию программных компонентов, выяснению требований к системе и самим компонентам, разбор вариантов использования системы и ее архитектуры.

В третьей главе описана реализация программных компонентов, а также разбор функциональных особенностей каждого компонента.

В четвертой главе проведено функциональное тестирование каждого программного компонента

В приложении находятся JavaScript код представления компонента обработки файлов с данными о событиях, сценарий взаимодействия компонента обработки файлов с данными об услугах с системой, код основного макета, код класса `EventExport` компонента обработки файлов с данными о событиях, а также код класса `AppealByMonths` компонента обработки файлов со статистикой данных по обращениям, диаграмма вариантов использования, архитектура системы, макет интерфейса, листинги определенных файлов.

1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Описание предметной области проекта

Была поставлена задача разработать программные компоненты для обработки документов формата xlsx и xls, которые составлены по определенным правилам. Наиболее популярное приложение для обработки файлов этих форматов это Microsoft Excel.

Excel-файл – это электронная таблица, созданная с помощью программного обеспечения Microsoft Excel, одного из наиболее популярных и мощных инструментов для работы с данными в мире. Excel-файлы широко используются в бизнесе, науке, образовании и многих других областях для хранения, анализа и визуализации данных. Они позволяют выполнять сложные математические, статистические расчеты, создавать различные графики и диаграммы.

Сейчас данные являются ценным активом для любой организации, и способность эффективно управлять этими данными может значительно повысить продуктивность работы и принятие обоснованных решений. Компоненты для обработки excel-файлов позволяют автоматизировать рутинные процедуры обработки данных, такие как сбор, анализ, обновление и распределение информации, что сокращает время на выполнение этих задач и минимизирует вероятность ошибок.

Интеграция таких компонентов в систему компании обеспечивает бесперебойное взаимодействие между различными информационными системами и базами данных. Это позволяет данным свободно перемещаться между системами, улучшая их актуальность и доступность для конечных пользователей.

Необходимо уделить внимание аспектам реализации инструментов обработки данных, их фильтрации, а также разработке методов для отображения большого объема информации.

1.2. Обзор существующих решений

Были рассмотрены примеры решений, связанные с аналогичными задачами обработки данных. Необходимо рассмотреть следующие направления: инструменты для обработки excel-файлов, веб-сайты с особенностями работы фильтров, а также сайты, которые предлагают возможности для просмотра большого количества данных, такими примерами являются tableconvert.com, wildberries.ru и dns-shop.ru соответственно.

TableConvert.com

TableConvert.com – это онлайн-сервис, который предоставляет возможность конвертировать табличные данные между разнообразными форматами, включая CSV, JSON, XML, HTML, SQL и Excel. Позволяет загружать, редактировать и скачивать данные в нужном формате. Сервис обеспечивает безопасность данных, не сохраняя их на сервере после завершения работы. На рисунке 1 представлен главная страница TableConvert.com.

Рисунок 1 – Главная страница TableConvert.com

Одним из ключевых преимуществ TableConvert является возможность редактировать данные прямо на сайте до их конвертации. При этом для использования сервиса не требуется устанавливать никакое дополнительное программное обеспечение, все операции производятся непосредственно в браузере.

Wildberries.ru

Wildberries.ru это один из крупнейших онлайн-ритейлеров в России, который продает широкий ассортимент товаров от одежды до бытовой техники. На сайте Wildberries пользователи могут сортировать список товаров по таким параметрам, как цена, бренд, популярность, и др. Основные фильтры сразу отображаются на странице с таблицей, если пользователю нужна более тонкая настройка, присутствует кнопка все фильтры, по нажатию на которую открывается еще множество разных фильтров. Все фильтры расположены в верхней части таблицы, позволяя сразу же отфильтровать список товаров по нужному параметру. Пример станицы с фильтрами на сайте Wildberries.ru представлен на рисунке 2.

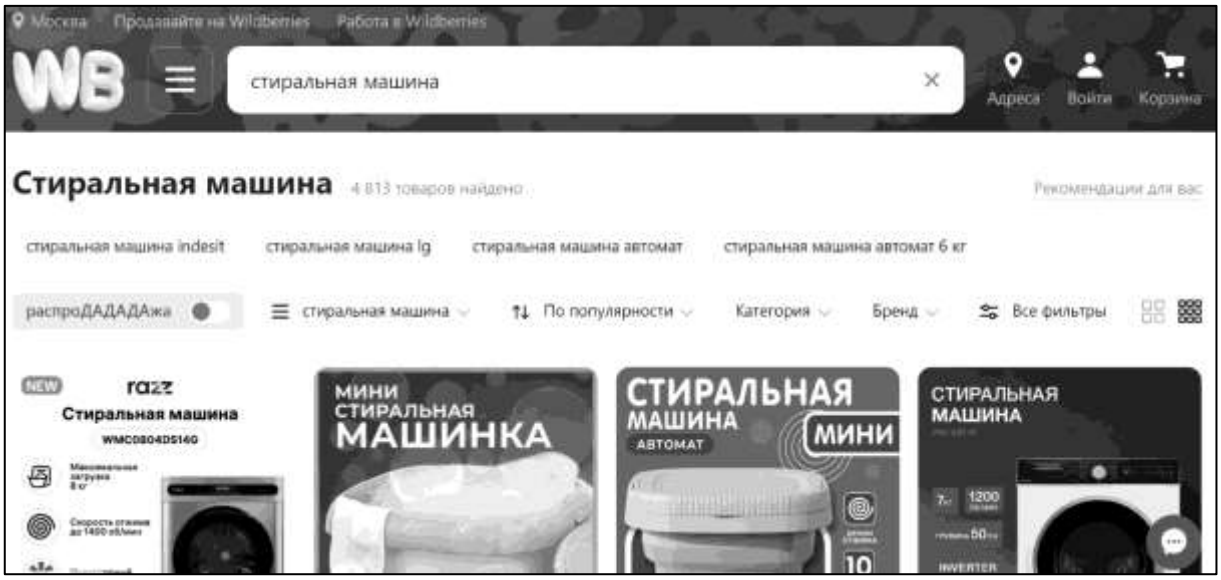

Рисунок 2 – Страница с фильтрами на сайте Wildberries.ru

Dns-shop.ru

Dns-shop.ru это крупная сеть магазинов электроники, предоставляющая широкий ассортимент товаров и услуг в сфере цифровой техники. На их сайте присутствует функция сравнения товаров. Она позволяет сравнивать характеристики различных товаров непосредственно на сайте магазина, опираясь на таблицу. В заголовке таблицы находятся товары, добавленные пользователем для сравнения, а ниже представлены параметры товаров. На

странице может отображаться только 4 товара, если же пользователь добавил больше товаров для сравнения, то для этого существуют специальные кнопки, при нажатии на которые происходит смена товаров для сравнения, таким образом решается проблема перегрузки информации на одном экране.

Пример таблицы со сравнением товара представлен на рисунке 3.

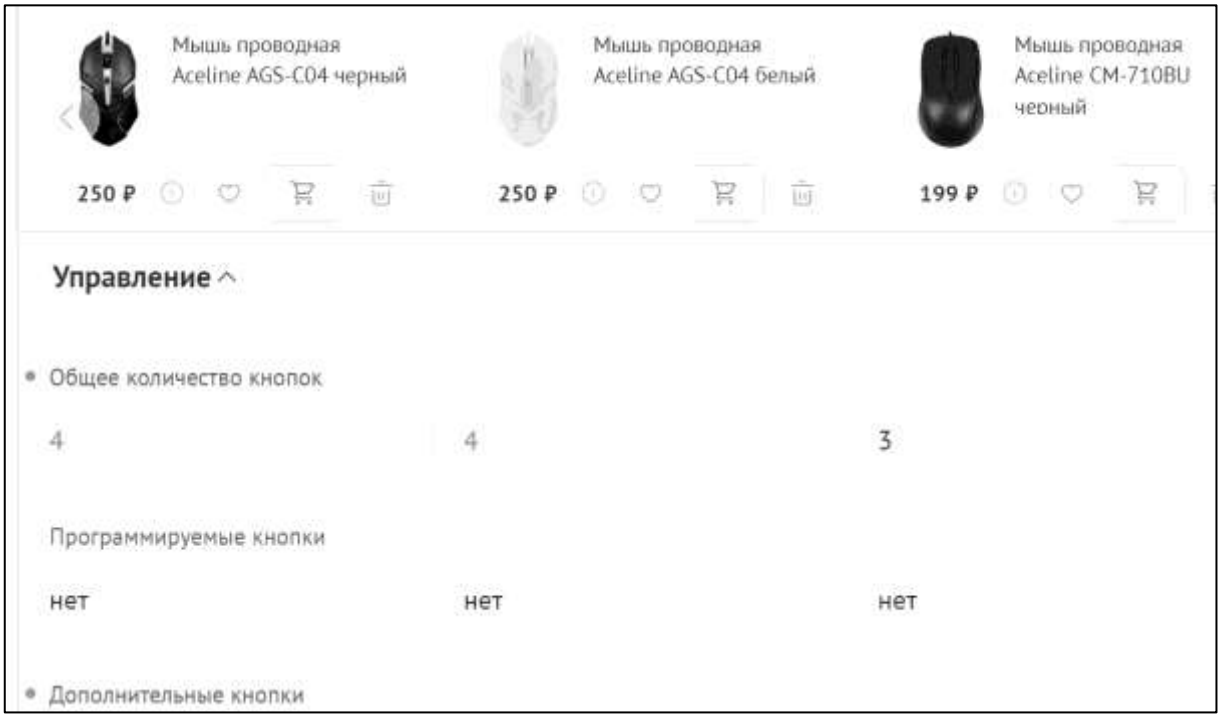

Рисунок 3 – Страница сравнения товаров на сайте dns-shop.ru

Вывод по первой главе

Был проведен подробный обзор предметной области и существующих решений для обработки и визуализации данных в формате Excel. Основной предмет разработки – это создание программных компонентов для обработки excel-файлов, что требует учета особенностей данных форматов и возможностей их интеграции с информационными системами компании.

2. ПРОЕКТИРОВАНИЕ КОМПОНЕНТОВ

2.1. Требования к проекту

При проектировании компонентов необходимо учитывать функциональные и нефункциональные требования к системе, в которую эти компоненты будут интегрироваться. Важно также учитывать эти требования при работе над реализацией компонентов.

Функциональные требования

Рассмотрим функциональные требования к системе.

1. Система должна обеспечивать возможность загрузки файлов пользователем.
2. Система должна обеспечивать возможность отправки файла пользователю для скачивания.
3. Система должна предоставлять возможность авторизации пользователя.
4. Система должна обеспечивать свободное переключение между компонентами обработки файлов для авторизированного пользователя.

Нефункциональные требования

Рассмотрим нефункциональные требования к системе.

1. Система должна быть реализована на языке программирования PHP версии 8.2 и фреймворке Laravel версии не меньше 10.
2. Система должна использовать СУБД MySQL.
3. Система должна запускаться и корректно отображаться в любом из популярных браузеров: Google Chrome, Opera, Microsoft Edge, Яндекс.Браузер, Mozilla Firefox.

2.2. Варианты использования

При разработке компонентов необходимо учитывать различные сценарии их использования. В данном случае, пользователь является единственным актером, взаимодействующим с системой обработки документов.

Этот пользователь имеет доступ ко всем функциям системы. Диаграмма вариантов использования будет отражать различные сценарии, в рамках которых пользователь взаимодействует с компонентами. Диаграмма вариантов использования представлена на рисунке 1 приложения З.

Представлены следующие варианты использования.

1. Работа с данными событий. Используя компонент обработки файлов с данными событий, пользователь может просматривать и анализировать эти данные.

2. Работа с данными обращений. Используя компонент обработки файлов с данными обращений, пользователь может просматривать и анализировать эти данные.

3. Импортировать данные. Пользователь может загрузить в систему файл с новыми данными, которые импортируются в таблицу базы данных.

4. Скачать отчет. Пользователь может скачать полный отчет по всем данным или только по отфильтрованным.

5. Отфильтровать данные. Пользователь может найти необходимые данные в таблице с помощью фильтров.

6. Посмотреть справку по фильтрам. Пользователь может посмотреть справку по фильтрам таблицы.

7. Экспортировать данные. Пользователь может экспортировать все или только отфильтрованные данные из таблицы.

8. Загрузка данных об услугах. Используя компонент обработки файлов с данными об услугах, пользователь может загрузить данные в систему.

9. Загрузка статистики по доле обращений. Используя компонент обработки файлов со статистикой по доле обращений, пользователь может загрузить данные по доле обращений в систему.

Данные прецеденты предполагают, что загружаются конкретные данные и конкретные типы файлов.

2.3. Архитектура системы

На рисунке 2 приложения И представлена архитектура всей системы, толстым пунктиром обозначены разрабатываемые компоненты.

Архитектура данной системы строится на классической трехуровневой архитектуре веб-приложения, которая включает в себя три основных слоя: представление, бизнес-логику и хранение данных. Каждый уровень взаимодействует с другими, обеспечивая работоспособность приложения. Трехуровневая архитектура облегчает масштабирование, поддержку и разработку приложений, поскольку разделяет функциональность на логические блоки.

1. Слой представления (клиентская часть). Данный уровень отвечает за отображение пользовательского интерфейса и обработку взаимодействий с пользователем.

2. Слой бизнес-логики (Apache сервер). На этом уровне происходит обработка запросов от клиентской части, выполнение логики приложения и взаимодействие с базой данных. Бизнес-логика реализуется с помощью серверных языков программирования. В данном примере используется язык PHP с фреймворком Laravel..

3. Слой хранения данных (СУБД MySQL): Этот компонент отвечает за сохранение всех данных, необходимых для работы приложения. Слой хранения данных обеспечивает надежное и эффективное управление информацией, с которой работает приложение.

Компонент обработки файлов с данными событий и компонент обработки файлов с данными обращений имеют одинаковые функциональные требования.

Функциональные требования

Рассмотрим функциональные требования этих для компонентов.

1. Компонент должен предоставлять возможность загрузки данных из excel-файла в базу данных.

2. Компонент должен предоставлять визуализацию данных из базы данных в виде таблицы.

3. Компонент должен предоставлять возможность фильтрации данных в таблице.

4. Компонент должен предоставлять справку по видам фильтров таблицы.

5. Компонент должен предоставлять возможность скачать отчет в виде excel-файла, составленного по определенным правилам, как только по отфильтрованным данным, так и по всем.

6. Компонент должен предоставлять возможность экспортировать как все данные из базы данных, так и только отфильтрованные.

2.4. Компонент обработки файлов с данными о событиях

Данный компонент отвечает за обработку excel-файлов с данными о плановых и внеплановых событиях, связанных с разными сферами ЖКХ. Данные файлы имеют определенную неизменную структуру в виде таблицы на 63 столбца.

При импорте содержимого файла в базу данных, необходимо проверять наличие дублирующихся значений по столбику «Идентификатор события», в базе данных это поле event_id. Данные записываются в таблицу for_omas_control. В данной таблице запись о событии разделена на множество полей, таких как, дата и время возникновения события, федеральный округ, субъект и район, в котором произошло событие, также краткое описание события и множество его характеристик, например сфера ЖКХ, тип происшествия, наименование объекта, адрес объекта, , дата и время устранения. Поля содержат разные типы данных, такие как текст, числа, даты, это необходимо учитывать при реализации фильтров для таблицы.

На рисунке 6 представлен фрагмент структуры таблицы базы данных с полями, хранящими данные о том, где произошло событие, для компонента обработки файлов с данными о событиях.

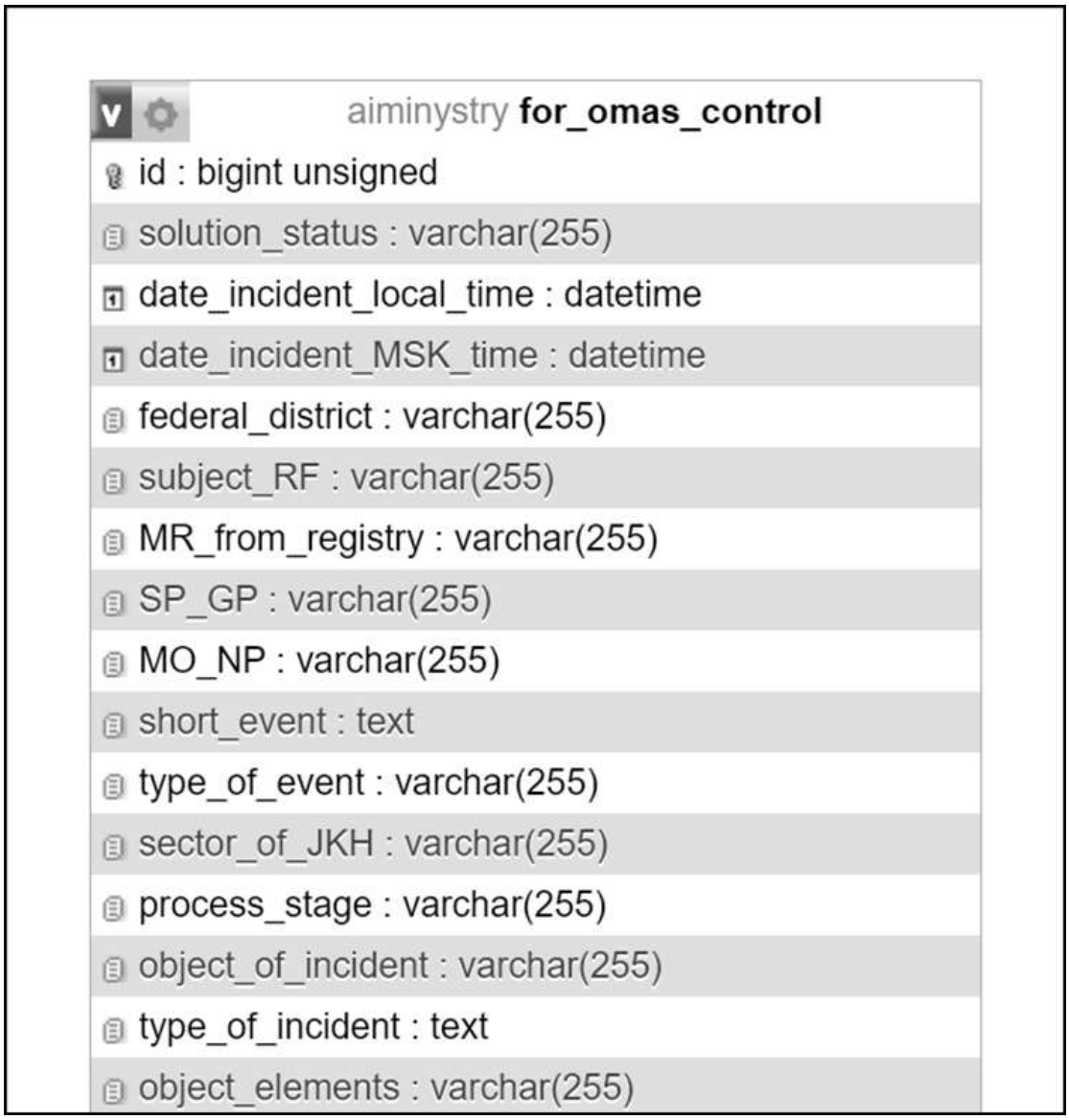

Рисунок 6 – Фрагмент структуры таблицы базы данных for_omas_control

2.5. Компонент обработки файлов с данными об обращениях

Данный компонент отвечает за обработку excel-файлов с данными об обращениях или же инцидентах, произошедших в разных частях Челябинской области. Данные файлы имеют определенную неизменную структуру в виде таблицы на 41 столбец. Данные для этой таблицы находятся в социальных сетях в виде записей или комментариев пользователей, люди, занимающиеся мониторингом социальных сетей, формируют эту таблицу.

При импорте содержимого файла в базу данных, необходимо проверять наличие дублирующихся значений по столбику «Номер инцидента», в базе данных это поле local_id. Данные записываются в таблицу tsur_detaling. В данной таблице запись о событии разделена на множество полей, таких как, регион, исполнитель, общее время первого ответа, время с даты создания обращения до времени первого ответа, тип инцидента, наличие медиафайла в сообщении, URL автора. Поля содержат разные типы данных, такие как текст, числа, даты, это необходимо учитывать при реализации фильтров для таблицы. На рисунке 7 представлен фрагмент структуры таблицы базы данных с полями, содержащими информацию о исполнителе, который реагирует на обращение и некоторых его действиях, для компонента обработки файлов с данными об обращениях.

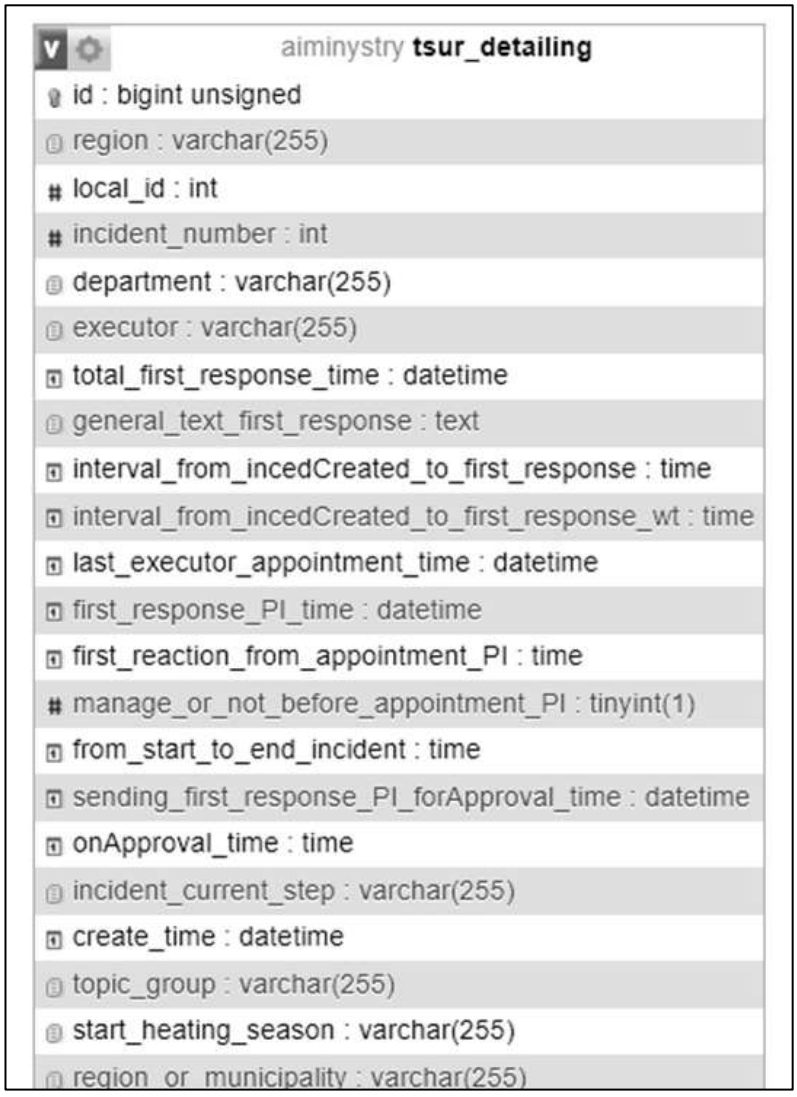

Рисунок 7 – Фрагмент структуры таблицы базы данных tsur_detaling

2.6. Компонент обработки файлов с данными об услугах

Данный компонент отвечает за обработку excel-файлов с данными об услугах, которые оказывает Министерство строительства и жилищно-коммунального хозяйства Российской Федерации. Данные файлы имеют определенную неизменную структуру в виде нескольких листов: «общий», «МСЗУ ЧО», «оценка качества по месяцам», «поданные заявления по месяцам», «отказы по заявлениям по месяцам», «положительные по месяцам». На каждом листе находится определенная таблица.

На листе «общий» находится весь список услуг со ссылками на эти услуги, а также общая информация со всех остальных листов: общее количество поданных заявлений по годам, общее количество отказов по годам, общее количество черновиков по годам, общее количество ошибок по годам, общая оценка, итоговые значения.

На листе «оценка качества по месяцам» находится весь список услуг со ссылками на эти услуги, а также все оценки, расписанные по годам и месяцам.

На листе «поданные заявления по месяцам» находится весь список услуг со ссылками на эти услуги, а также общее количество заявлений на услуги, разбитое по месяцам и годам.

На листе «отказы по заявлениям по месяцам» находится весь список услуг со ссылками на эти услуги, а также общее количество отказов по заявлениям на услуги, разбитое по месяцам и годам.

На листе «положительные по месяцам» находится весь список услуг со ссылками на эти услуги, а также общее количество положительных заявлений на услуги, разбитое по месяцам и годам.

Данный компонент должен импортировать данные из файла, в определенные таблицы базы данных.

1. Таблица service. Данная таблица содержит полный список услуг и имеет поля id, название, ссылка на госуслуги, в нее импортируется информация с листа «общий».

2. Таблица quality_control_by_month. В данную таблицу импортируется информация с листа «оценка качества по месяцам», то есть записывается оценка качества за каждый месяц каждого года по каждой услуге, также эта таблица имеет поля id, service_id для связи с таблицей service, date и grade.

3. Таблица statements_by_month. Данная таблица имеет поля id, service_id для связи с таблицей service, date, data_quantity, number_of_failures, number_of_positive, в нее импортируются данные из листов «поданные заявления по месяцам», «отказы по заявлениям по месяцам», «положительные по месяцам», то есть для каждой услуги и каждого месяца записывается положительное и общее количество заявлений, а также количество отказов.

На рисунке 8 представлена структура таблиц для компонента обработки файлов с данными об услугах.

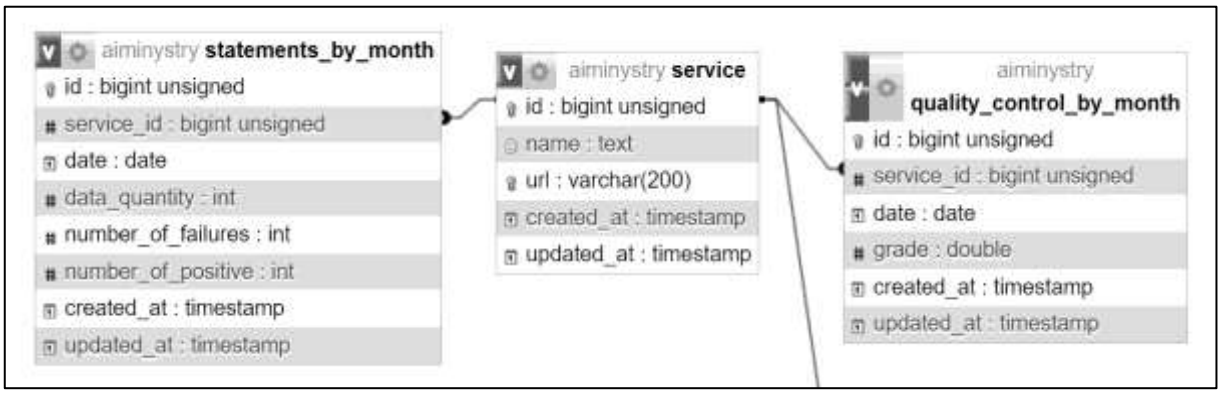

Рисунок 8 – Структура таблиц с данными об услугах

Последовательность взаимодействий компонента с базой данных

Диаграмма последовательности показывает очередность обработки excel-файла компонентом. При реализации компонента необходимо опираться именно на эту последовательность иначе будет нарушена целостность данных в базе, так как таблица service имеет ключевую роль, исходя из данных в этой таблице происходит запись данных в другие таблицы, так как без записи услуги невозможно будет записать статистику по этой услуге. На рисунке 3 приложения К представлена диаграмма последовательности

взаимодействия компонента и базы данных после загрузки файла пользователем.

2.7. Компонент обработки файлов со статистикой по доле обращений

Данный компонент отвечает за обработку excel-файла со статистическими данными по доле обращений по оказанию услуг. Данный файл имеет определенную неизменную структуру в виде таблицы на 299 столбцов. Из необходимых полей, для обработки данных из файла в данной таблице указаны наименование услуги, сфера оказания услуги, а также количество заявлений, поданное в электронном виде, общее количество заявлений и процент электронных заявлений от общего количества. Один файл содержит информацию за один месяц. Необходимо обратить внимание на то, что при импорте данных нужно учитывать услуги только из сфер «имущество, стройка» и «стройка», а также данные в файлах собираются накопительным итогом, то есть если в файле за январь обращений в каком-либо районе было 10, а в файле за февраль 35, то в базу данных за февраль должно отправляться 25 обращений, аналогично и для других месяцев. В файлах января любого года, количество обращений считается с нуля. Данному компоненту представлены следующие функциональные требования.

1. Компонент должен предоставлять возможность импортировать данные из файла, предназначенного для этого компонента, в определенные таблицы базы данных.
2. Компонент должен предоставлять выбор месяца, за который грузится файл.
3. Компонент должен предоставлять выбор года, за который грузится файл.

Данный компонент использует таблицу service из компонента обработки файлов с данными об услугах, а также таблицу area и appeals_by_months.

Таблица area имеет полный список районов и округов, импортированных из загруженного файла, имеет поля id и name.

В таблице appeals_by_months находятся следующие данные: список всех обращений по каждому округу или району за каждый год и месяц, количество электронных заявлений, общее количество заявлений и доля электронных заявлений от общих. Таблица имеет поля id, area_id для связи с таблицей area, service_id для связи с таблицей service, month, year, number_applications_in_electronic_form, number_applications, %_of_electronic_forms_of_total. Уникальность записи проверяется по полям area_id, service_id, month и year. Необходимо учесть эту информацию при реализации компонента. На рисунке 9 представлена структура таблиц для компонента обработки файлов со статистикой по доле обращений, а также отображена связь между таблицами.

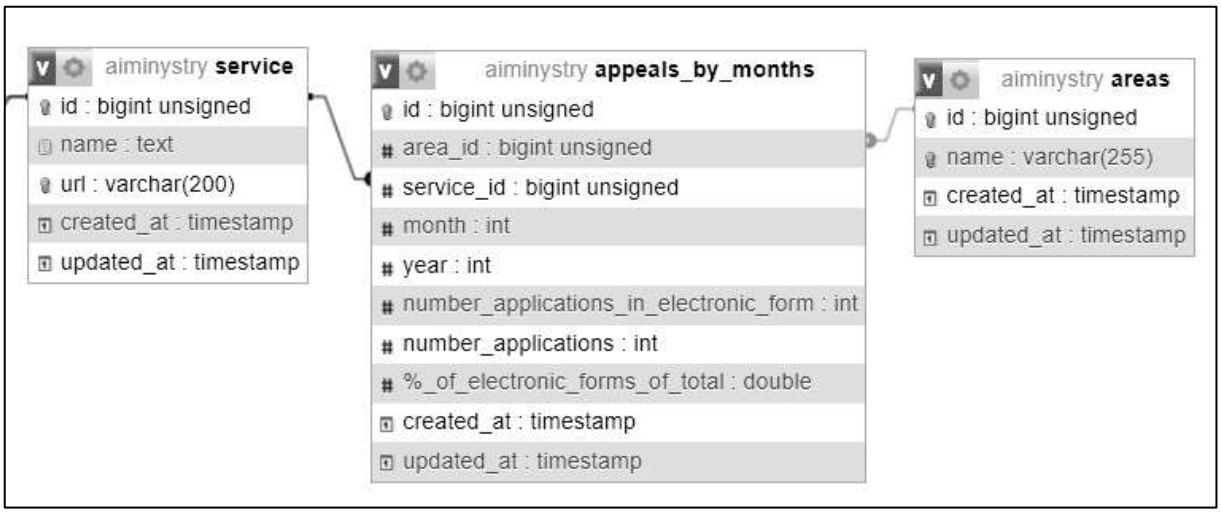

Рисунок 9 – Структура таблиц со статистикой по доле обращений

Последовательность взаимодействий компонента с базой данных

На первоначальном этапе пользователь загружает файл, указывая желаемый месяц и год. Затем система импортирует данные о всех регионах, содержащихся в файле, в таблицу под названием area. Также система получает из файла список услуг по сферам «имущество, стройка» и «стройка» и из базы данных информацию об услугах, хранящихся в таблице service. Далее происходит проверка указанного месяца и, если это январь, данные

напрямую сохраняются в базу данных по каждой услуге, району и указанному году для января, иначе, система извлекает данные из базы данных для соответствующего месяца и года, после чего вычитает их из импортированных значений. Полученные результатом вычитания данные затем записываются обратно в базу данных.

Диаграмма последовательности иллюстрирует очередность операций, необходимых для обработки excel-файла компонентом. В процессе реализации этого компонента важно придерживаться заданной последовательности действий, чтобы гарантировать корректность выполнения каждой операции. На рисунке 4 приложения Л представлена диаграмма последовательности взаимодействия компонента и базы данных после выбора месяца, года и загрузки файла пользователем.

2.8. Макеты клиентской части компонентов

На веб-странице каждого компонента присутствует заголовок. В этом заголовке содержится логотип компании, а также ссылки для перехода на страницы компонентов.

Макет для компонента обработки файлов с данными о событиях и компонента обработки файлов с данными об обращениях

У данных компонентов одинаковая визуальная составляющая, их различие только в серверной части, в структурах обрабатываемых файлов. Это обусловливает использование одинакового макета интерфейса для обеих компонентов. На рисунке 5 приложения М отображен макет интерфейса для этих компонентов, также предложенный макет удовлетворяет всем функциональным требованиям, которые были установлены для этих компонентов.

На данном макете, вверху страницы, под заголовком, расположена форма для загрузки файла в систему, состоящую из кнопки выбрать файл, поля в котором будет отображаться название выбранного файла, а также кнопка загрузить, для отправки файла в систему. Под полем, который отоб-

ражает имя файла находятся две кнопки для скачивания отчета, одна отвечает за скачивание отчета по всем данным из базы данных, другая за скачивание отчета только по отфильтрованным данным, также там находится кнопка со справкой по фильтрам, при нажатии по которой пользователь может прочитать как пользоваться фильтрами таблицы, самая правая кнопка отвечает за экспорт данных из базы в excel-файл. Ниже находится селектор для выбора активных и неактивных столбцов таблицы, рядом с ним кнопки добавить фильтр, и очистить фильтр. При нажатии на кнопку добавить фильтр создаются еще 3 поля для фильтрации данных таблицы, а также кнопка, которая очищает конкретно фильтр, и кнопка удалить. Таких форм с фильтрами можно добавить несколько.

Макет для компонента обработки файлов с данными об услугах

На рисунке 12 представлен макет интерфейса для компонента обработки файлов с данными об услугах.

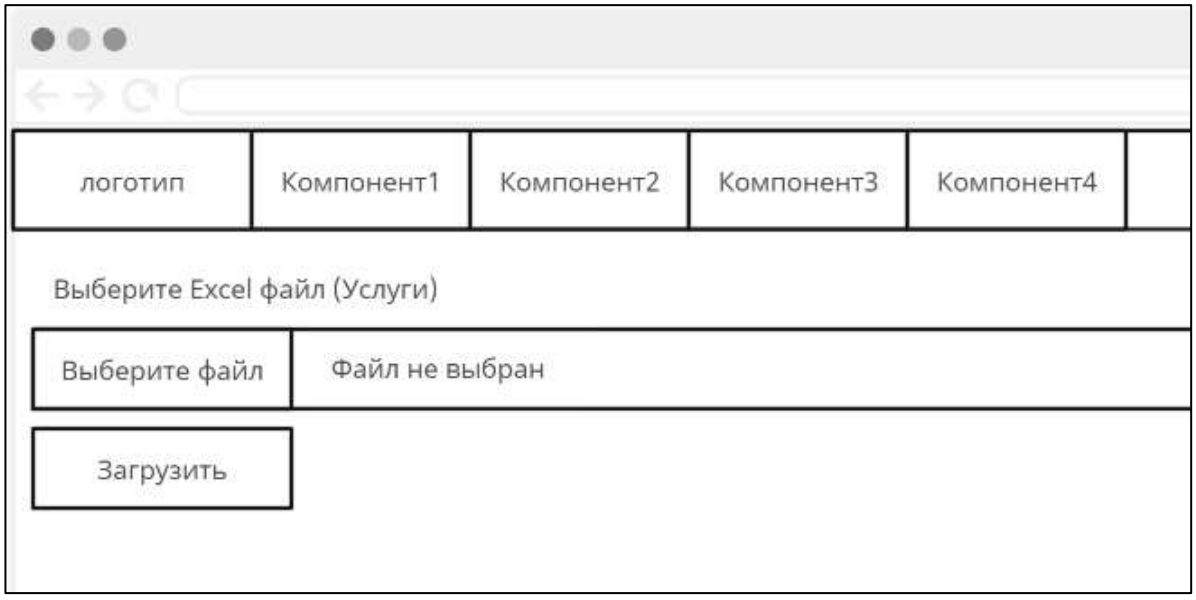

Рисунок 12 – Макет интерфейса

На данном макете, вверху страницы, под заголовком, расположена форма для загрузки файла в систему, состоящую из кнопки выбрать файл, поля в котором будет отображаться название выбранного файла, а также кнопка загрузить, для отправки файла в систему.

Макет для компонента обработки файлов со статистикой по доле обращений

На странице данного компонента под заголовком помимо формы для загрузки файла в систему, ниже расположены селектор элементов, при нажатии на который открывается список для выбора месяца, а также поле для ввода года. Кнопка загрузить отправляет в систему не только файл, но и выбранный в селекторе месяц, а также прописанный в последнем поле год. На рисунке 13 представлен макет интерфейса для компонента обработки файлов со статистикой по доле обращений.

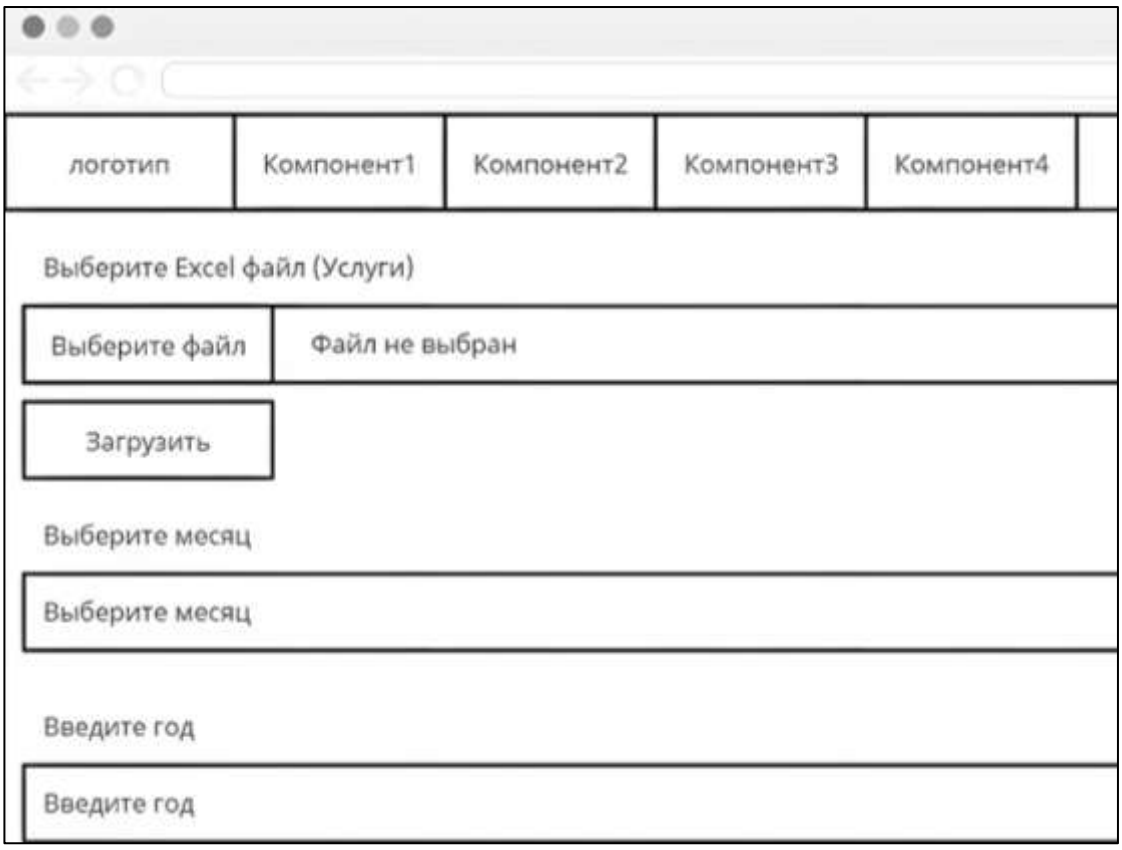

Рисунок 13 – Макет интерфейса

Вывод по второй главе

В ходе работы были определены все необходимые требования к системе и спроектированы основные компоненты. Пользователь может импортировать, экспортировать и фильтровать данные, а также работать с различными типами файлов. Система построена на трехуровневой архитектуре веб-приложения.

3. РЕАЛИЗАЦИЯ КОМПОНЕНТОВ

3.1. Средства реализации

Реализация проводилась с использованием языка программирования PHP [1]. Это скриптовый язык программирования общего назначения, который часто используется для разработки веб-приложений и веб-сайтов. PHP обычно интегрируется напрямую в HTML [4] и может быть встроен в HTML-код, или использоваться в сочетании с различными веб-фреймворками, контент-менеджмент системами и другими инструментами для создания динамических веб-приложений. PHP также поддерживает большое количество баз данных.

Также Laravel [3], популярного фреймворка для создания веб-приложений на языке программирования PHP. Он предоставляет разработчикам множество готовых инструментов и библиотек для упрощения процесса создания веб-приложений, включая удобную маршрутизацию, работу с базами данных, аутентификацию пользователей и многое другое. Laravel также обеспечивает поддержку современных практик разработки, таких как MVC (Model-View-Controller).

Системы управления баз данных MySQL [6] - одна из самых популярных систем управления реляционными базами данных (СУБД) в мире. MySQL поддерживает множество функций, включая возможность хранения больших объемов информации, а также обладает высокой производительностью и надежностью.

Веб-приложения phpMyAdmin [7] - бесплатный инструмент с открытым исходным кодом, предназначенный для управления администрирования баз данных MySQL через веб-интерфейс. Он предоставляет удобные средства для создания, удаления и изменения баз данных, таблиц, полей, а также для выполнения SQL-запросов, импорта и экспорта данных и многих других операций.

3.2. Структура фреймворка laravel

В фреймворке Laravel реализован паттерн MVC (Model-View-Controller) для структурирования приложений.

1. Модель (Model). Модели в Laravel представляют собой слой данных, отвечающий за работу с базой данных. Каждая модель обычно связана с определенной таблицей в базе данных и предоставляет способ взаимодействия с этими данными. Модели также содержат бизнес-логику приложения, валидацию данных и другие методы, осуществляющие манипуляции с данными.

2. Представление (View). Представления в Laravel представляют собой шаблоны, которые отвечают за отображение данных пользователю. Они могут быть в виде HTML-страниц, JSON-ответов или других типов данных, возвращаемых пользователю. Laravel предоставляет мощные инструменты для работы с представлениями, включая шаблонизацию, работу с макетами и возможность передачи данных из контроллера в представление.

3. Контроллер (Controller). Контроллеры в Laravel обрабатывают входящие HTTP-запросы и взаимодействуют как с моделями, так и с представлениями. Они принимают запросы от клиента, обращаются к соответствующей модели для получения необходимых данных, а затем передают эти данные в представление для отображения. Контроллеры также могут обрабатывать входные данные от пользователя, выполнять необходимые операции и возвращать результат пользователю.

Одним из главных элементов фреймворка Laravel является маршрутизация [14]. В Laravel маршрутизация представляет собой механизм определения, как приложение обрабатывает входящие HTTP-запросы. Маршруты обеспечивают централизованную систему маршрутизации запросов в приложении, позволяя определить, какие действия должны быть выполнены в зависимости от URL-адреса, с которого приходит запрос. Механизм маршрутизации в Laravel обеспечивает структурированное управление потоком

HTTP-запросов и повышает эффективность обработки запросов веб-приложением. Код файла маршрутизации представлен в листинге 5 приложения Д.

Макеты в Laravel используются для создания общей структуры HTML-страницы, что облегчает управление и обслуживание кода. Макет позволяет определить основной шаблон страницы, включая элементы, которые повторяются на всех страницах, также в основном макете подключаются JavaScript библиотеки, шрифты, css стили. Фрагмент кода основного макета представлен в листинге 2 приложения Б.

3.3. Компонент обработки файлов с данными о событиях

При реализации компонента, в первую очередь был создан класс модели `Event` и миграция для создания таблицы for_omas_control в базе данных. Фрагмент кода файла миграции представлен в листинге 6 приложения Е.

Также был создан класс контроллер `EventController`. Данный класс имеет несколько ключевых методов.

Метод `import(Request $request)` импортирует данные из файла Excel, обновляет базу данных. Код данного метода представлен в листинге 1.

Листинг 1 – Код метода `import()` класса `EventController`

```
public function import(Request $request)
    {
        $request->validate([
            'file' => 'required|file|mimes:xlsx,xls'
        ], [
            'file.required' => 'Выберите файл для загрузки',
            'file.mimes' => 'Выбранный файл должен быть формата Excel
(XLSX, XLS)'
        ]);

        $file = $request->file("file");

        if($file->isValid())
        {
            Excel::import(new EventImport, $file);
            return redirect()->back()->with('success', 'all good');
        } else {
            return redirect()->back()->withErrors(['file' => 'Выбранный
файл некорректен']);
        }
    }
```

Метод `export()` позволяет скачать, составленный по определенным правилам, отчет используя все данные из таблицы базы данных.

Метод `exportFilteredData(Request $request)` позволяет скачать, составленный по определенным правилам, отчет из отфильтрованных данных, на основе переданных фильтров. Код данного метода представлен в листинге 2.

Листинг 2 – Код метода `exportFilteredData()` класса `EventController`

```
public function exportFilteredData(Request  $request)
    {
        $filters = $request->input('filters', []);
        $query = Event::select('MR_from_registry',
                               'MO_NP',
                               'address_of_object',
                               'sector_of_JKH',
                               'type_of_event',
                               'short_event',
                               'name_of_expl_org',
                               'date_incident_local_time',
                               'date_incident_elimination_local_time',
                               'solution_status',);
        $filters = json_decode($filters, true);
        $filters = is_array($filters) ? $filters : [];
        foreach ($filters as $filter) {
            $field = $filter['field'];
            $type = $filter['type'];
            $value = $filter['value'];
            switch ($type) {
                case 'like':

                    $query->where($field, 'like', '%' . $value . '%');
                    break;
                case '=':
                case '<':
                case '<=':
                case '>':
                case '>=':
                case '!=':
                    $query->where($field, $type, $value);
                    break;
                    default:
                    $query->where($field, $value);
                    break;
            }
        }
        $filteredData = $query->get();
        return Excel::download(new EventExport($filteredData), 'Таблица
контроль OMAC.xlsx');
    }
```

Метод `viewDataTable()` отображает таблицу событий на веб-странице, загружая данные из базы данных. Код данного метода представлен в листинге 3.

Листинг 3 – Код метода `viewDataTable()` класса `EventController`

```
public function viewDataTable()
    {
        $events = Event::all();
        $json = file_get_contents(public_path('data/events.json'));
        $columns = json_decode($json, true);
        return view("/events/events", ['events'=>$events, 'columns' => $col-
umns]);
    }
```

Метод `import()` использует класс `EventImport`, который импортирует информацию из файла в таблицу базы данных. Фрагмент класса импорта `EventImport` представлен в листинге 4.

Листинг 4 – Фрагмент кода класса `EventImport`

```
class EventImport implements ToModel, WithValidation, WithHeadingRow {
    public function model(array $row){
        $row = $this->transformValues($row);
        return new Event([
            'solution_status' => $row['Статус решения инцидента/аварии'],
            'date_incident_local_time' => $row['Дата и местное время возн-я
соб'],
            'date_incident_MSK_time' => $row['Дата и время возн-я соб
МСК'],
            'federal_district' => $row['Федеральный округ'],
            'subject_RF' => $row['Субъект РФ'],
            'MR_from_registry' => $row['МР из реестра'],
            'SP_GP' => $row['СП/ГП'],
            'MO_NP' => $row['МО/НП'],
            'short_event' => $row['Краткое событие'],
            'type_of_event' => $row['Тип события'],
            'sector_of_JKH' => $row['Сфера ЖКХ'],
            'process_stage' => $row['Этап технологического процесса'],
            'object_of_incident' => $row['Объект/вид происшеств'],
            'type_of_incident' => $row['Тип объекта/происшеств'],
            'object_elements' => $row['Элементы объекта'],
            //Остальные поля базы данных
            'date_post_update' => $row['Дата обновления записи'],
            'user_login' => $row['Логин пользователя'],
            'user_name' => $row['Имя пользователя'],
            'organization_name' => $row['Организация'],

        ]);}
    public function rules(): array {
        return [
            'event_id'=> 'unique:for_omas_control'];}
    public function transformValues(array $row){
        if (!empty($row['Дата и местное время возн-я соб'])){
            $row['Дата и местное время возн-я соб'] = DateTime::createFrom-
Format('d.m.Y H:i', $row['Дата и местное время возн-я соб']);}
        if (!empty($row['Дата и время возн-я соб МСК'])){
            $row['Дата и время возн-я соб МСК'] = DateTime::createFrom-
Format('d.m.Y H:i', $row['Дата и время возн-я соб МСК']);}
        if (!empty($row['Дата и местное время устр.соб.'])){
            $row['Дата и местное время устр.соб.'] = DateTime::createFrom-
Format('d.m.Y H:i', $row['Дата и местное время устр.соб.']);}
        if (!empty($row['Дата и время устр.соб. МСК'])){
            $row['Дата и время устр.соб. МСК'] = DateTime::createFrom-
Format('d.m.Y H:i', $row['Дата и время устр.соб. МСК']);)}
```

```
        $row['Дата создания записи'] = DateTime::createFromFormat('d.m.Y
H:i', $row['Дата создания записи']);
        $row['Дата обновления записи'] = DateTime::createFromFormat('d.m.Y
H:i', $row['Дата обновления записи']);
        return $row;
    }}
```

Методы `export()` и `exportFilteredData(Request $request)` используют класс `EventExport` для формирования и отправки отчета пользователю. Отчет формируется с правилами, описанными при проектировании компонента. Фрагмент кода класса экспорта `EventExport` представлен в листинге 3 приложения В.

Часть функций компонента, таких как, фильтрация данных в таблице, отображение справки по фильтрации, генерация таблицы, реализована с помощью JavaScript [5] кода и прописана в файле представления. Были использованы такие библиотеки, как Tabulator.js [12] для отображения таблицы и реализации фильтров в заголовках таблицы, DataRangePicker.js [13] для корректной записи даты в фильтры, которые находятся в заголовках таблицы. JavaScript код представления представлен в листинге 1 приложения А.

Интерфейс страницы компонента изображен на рисунке 14.

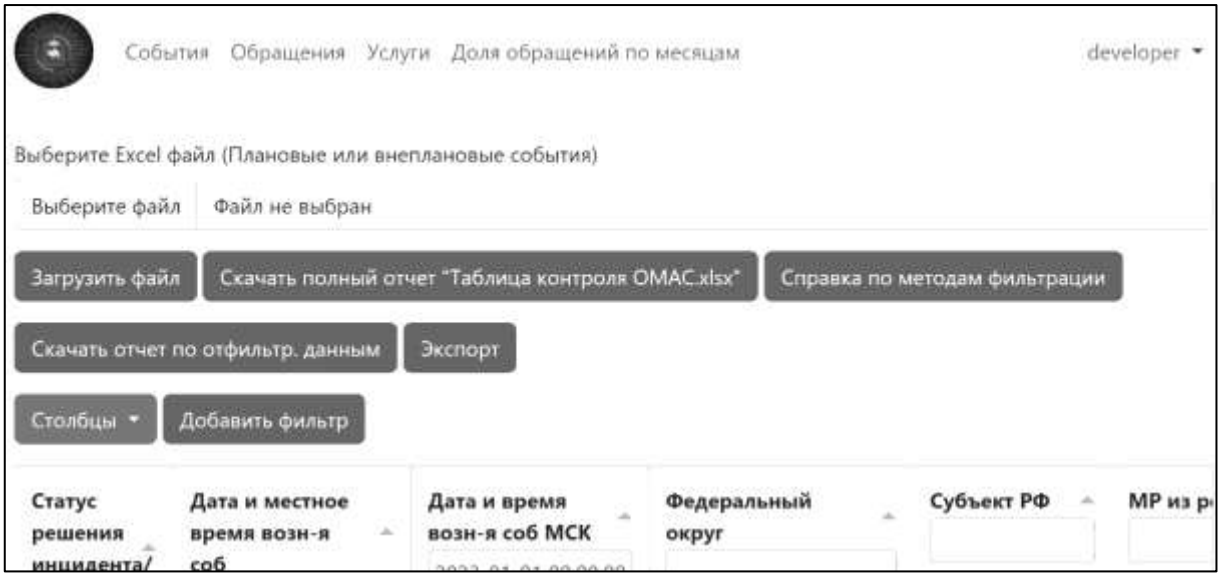

Рисунок 14 – Интерфейс страницы компонента обработки файлов с данными о событиях

3.4. Компонент обработки файлов с данными об обращениях

Данный компонент похож на предыдущий, миграция, модель, представление, контроллер, файл с классом импорта имеют схожую структуру. Отличия присутствуют при формировании отчета и отправки его пользователю. Логика обработки данных и формирования отчета представлена в `AppealExport`. Основная разница в структуре загружаемого файла и в структуре формируемого отчета. Код класса `AppealExport` представлен в листинге 7 приложения Ж.

Этот класс реализует несколько интерфейсов, он предоставляет специальные функции для форматирования и структурирования данных для экспорта. Класс содержит методы для формирования заголовков, определения стилей, ширины столбцов и маппинга данных для отображения в таблице.

3.5. Компонент обработки файлов с данными об услугах

Так как данному компоненту необходимо 3 таблицы в базе данных, то были созданы соответствующие модели и миграции для создания таблиц в базе данных. Модель Service привязана к таблице service, модель `QualityControlByMonth` к таблице quality_control_by_month, а модель `StatementsByMonth` к таблице statements_by_month. Помимо связи модели с таблицей необходимо прописать отношения модели Service к другим моделям, чтобы через определенную услугу можно было обращаться к ее оценке качества по месяцам и к количеству поданных, отрицательных и положительных обращений. Код класса модели `Service` представлен в листинге 5.

Листинг 5 – Код класса модели `Service`

```
class Service extends Model
{
    use HasFactory;

    protected $table = 'service';
    protected $guarded = ['id'];

    public function statement(): HasMany
    {
        return $this->hasMany(StatementsByMonth::class);
```

```
    }

    public function quality(): HasMany
    {
        return $this->hasMany(QualityControlByMonth::class);
    }

    public function appealByMonths(): HasMany
    {
        return $this->hasMany(AppealByMonths::class);
    }
}
```

В данном коде видно, что модель `Service` имеет отношения [15] один ко многим с моделью `StatementsByMonth` и моделью `QualityControl-ByMonth`. Так как значения у определенной услуги в таблицах quality_control_by_month и statements_by_month хранятся по месяцам, то у одной услуги может быть больше одной связанной записи в этих таблицах. Соответственно в моделях, связанных с другими таблицами прописана обратная зависимость через конструкцию `BelongsTo`.

Также был создан контроллер `ServiceController` и прописан метод `import(Request $request)`, в котором прописана логика импорта данных из загружаемого файла. Код метода `import()` класса `ServiceController` представлен в листинге 6.

Листинг 6 – Код метода `import()` класса `ServiceController`

```
public function import(Request $request)
    {
        $request->validate([
            'file' => 'required|file|mimes:xlsx,xls'
        ], [
            'file.required' => 'Выберите файл для загрузки',
            'file.mimes' => 'Выбранный файл должен быть формата Excel
(XLSX, XLS)'
        ]);
        $file = $request->file("file");
        if($file->isValid())
        {
            Excel::import(new ServiceImport, $file);
            Excel::import(new QualityControlByMonthImport, $file);
            Excel::import(new StatementsByMonthImport, $file);
            return redirect()->back()->with('success', 'all good');
        } else {
            return redirect()->back()->withErrors(['file' => 'Выбранный
файл некорректен']);
        }
    }
```

Данный метод использует 3 импорт класса `ServiceImport`, `QualtyControlByMonthImport` и `StatementsByMonthImport`. Они используются в определенной очередности, согласно диаграмме последовательности.

В первую очередь используется класс `ServiceImport`, который импортирует данные об услугах из загруженного excel-файла с листа «общий» в таблицу service в базе данных, записывая имя услуги и ссылку на нее на сайт госуслуги, уникальность проверяется по ссылке. Код класса `ServiceImport` представлен в листинге 7.

Листинг 7 – Код класса `ServiceImport`

```
class ServiceImport implements ToModel, WithHeadingRow, WithMultipleSheets,
WithValidation
{
    use Importable;

    private $rowCount = 0;
    public static $fieldsIndexes =
    [
        'LEVEL' => 'Уровень',
        'URL' => 'Форма'
    ];
    public function model(array $row)
    {
        $existingRecord = Service::where('url', $row[self::$fieldsIn-
dexes['URL']])->first();

        if(!empty($row[self::$fieldsIndexes['URL']]) && !$existingRecord){
        return new Service([
            'name' => $row[self::$fieldsIndexes['LEVEL']],
            'url' =>  $row[self::$fieldsIndexes['URL']]
        ]);}
    }
    public function rules(): array{return ['url' =>'unique:service'];}
    public function sheets(): array{return ['общий' => $this];}
    public function headingRow(): int{return 8;}
}
```

Вторым используется класс импорта `QualityControlByMonthImport`, который из загруженного excel-файла с листа оценка качества по месяцам считывает услугу, проверяет по ее ссылке есть ли такая услуга в таблице service и, если такая есть записывает оценку качества по каждому месяцу в таблицу quality_control_by_month.

Код класса QualityControlByMonthImport представлен в листинге 8.

Листинг 8 – Код метода класса QualityControlByMonthImport

```
class QualityControlByMonthImport implements ToCollection, WithMultiple-
Sheets
{
    public function collection(Collection $rows)
    {
        $startReadingRow = 8;
        $startReadingItem = 6;
        $headingRowNumber = 7;
        $serviceIdQualifier = 5;

        for ($row = $startReadingRow; $row <= $rows->count()-2; $row++) //
-2 для того, чтобы исключить строку с общим итогом
        {
            for ($item = $startReadingItem; $item <= $rows[$row]->count()-
1; $item++)
            {
                $service = Service::where('url', $rows[$row][$serviceId-
Qualifier])->first();
                if($rows[$headingRowNumber][$item] != null){
                    $date = DateTime::createFromFormat('Y-m-d',
$rows[$headingRowNumber][$item]);
                    QualityControlByMonth::updateOrCreate(
                        ['service_id' => $service->id,
                        'date' => $date->format('Y-m-d')],
                        ['grade' => $rows[$row][$item]]
                    );
                }
            }
        }}
    public function sheets(): array{return ['оценка качества по месяцам' =>
$this];}}
```

Третьим используется импорт класс StatementsByMonthImport. Код импорт класса StatementsByMonthImport представлен в листинге 9.

Листинг 9 – Код импорт класса StatementsByMonthImport

```
class StatementsByMonthImport implements WithMultipleSheets
{
    public function sheets(): array
    {
        return [
            'поданные заявления по месяцам' => new StatementsSubmittedBy-
MonthImport(),
            'отказы по заявлениям по месяцам' => new StatementsNegativeBy-
MonthImport(),
            'положительные по месяцам' => new StatementsPositiveByMonthIm-
port(),
        ];
    }
}
```

Данный класс используется для импорта данных из нескольких листов загруженной excel-таблицы. В методе sheets() определены соответствия

между названиями листов в таблице и классами, отвечающими за импорт данных с этих листов. Так как в таблицу statements_by_month необходимо записывать информацию с нескольких листов загруженной excel-таблицы, значит на каждый из этих листов необходимо назначить импорт класс. Данные классы имеют схожий функционал, так как данные в листах загруженного файла имеют один и тот же вид. Рассмотрим класс `StatementsSubmittedByMonthImport`. Код класса `StatementsSubmittedByMonthImport` представлен в листинге 10.

Листинг 10 – Код класса `StatementsSubmittedByMonthImport`

```
class StatementsSubmittedByMonthImport implements ToCollection
{
    public function collection(Collection $rows)
    {
        $startReadingRow = 8;
        $startReadingItem = 6;
        $headingRowNumber = 7;
        $serviceIdQualifier = 5;
        for ($row = $startReadingRow; $row <= $rows->count()-2; $row++) //
-2 для того, чтобы исключить строку с общим итогом{
            for ($item = $startReadingItem; $item <= $rows[$row]->count()-
1; $item++){
                $service = Service::where('url', $rows[$row][$serviceId-
Qualifier])->first();
                if($rows[$headingRowNumber][$item] != null){
                    $date = DateTime::createFromFormat('Y-m-d',
$rows[$headingRowNumber][$item]);
                    StatementsByMonth::updateOrCreate(
                        ['service_id' => $service->id,
                        'date' => $date->format('Y-m-d')],
                        ['data_quantity' => $rows[$row][$item]]
                    );}}}}}
```

Данный класс считывает из загруженного excel-файла с листа оценка качества по месяцам услугу, проверяет по ее ссылке есть ли такая услуга в таблице service и, если такая есть записывает количество поданных заявлений по месяцам в таблицу statements_by_month. Классы `StatementsNegativeByMonthImport` и `StatementsPositiveByMonth-Import` работают аналогичным образом, только записывают отказы по заявлениям по месяцам и положительные заявления по месяцам соответственно.

Интерфейс этого компонента выполнен в соответствии с макетом. Интерфейс компонента обработки файлов с данными об услугах представлен на рисунке 15.

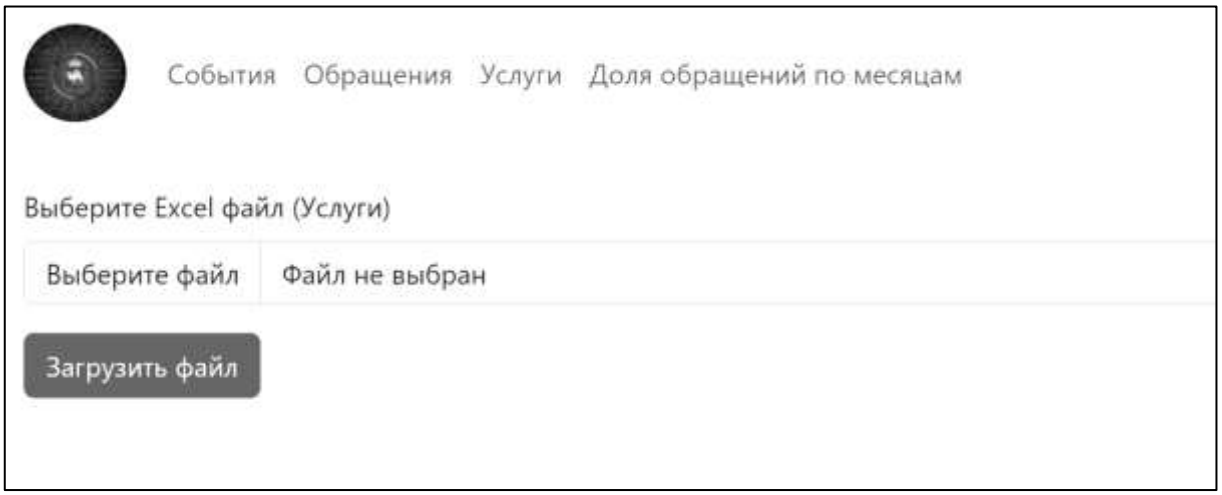

Рисунок 15 – Интерфейс страницы компонента обработки файлов с данными об услугах

Данный интерфейс включает в себя заголовок, под ним расположена форма для загрузки файла в систему, состоящую из кнопки выбрать файл, поля в котором будет отображаться название выбранного файла, а также кнопка загрузить, для отправки файла в систему.

3.6. Компонент обработки файлов со статистикой по доле обращений

Данному компоненту необходимо 3 таблицы в базе данных, модель Service и таблица service были описаны в реализации компонента обработки файлов с данными об услугах. Были созданы еще 2 модели и миграции для создания таблицы в базе данных. Модель Area привязана к таблице areas, а модель `AppealByMonths` к таблице appeals_by_months. Модель Area, также как и модель Service связаны с моделью `AppealByMonths` связью один ко многим, так как у одной конкретной услуги, в одном конкретном районе может быть разное количество обращений в разные месяца и разные года.

Код класса модели AppealByMonths представлен в листинге 11.

Листинг 11 – Код класса модели AppealByMonths

```
class AppealByMonths extends Model
{
    use HasFactory;

    protected $table = 'appeals_by_months';

    protected $guarded = ['id'];

    public function service(): BelongsTo
    {
        return $this->belongsTo(Service::class);
    }

    public function area(): BelongsTo
    {
        return $this->belongsTo(Area::class);
    }
}
```

У конкретной услуги может быть множественное количество обращений в разные месяцы, года, в разных районах. Также в определенном районе может быть много обращений по разным услугам, месяцам, годам. Данные отношения построены для того, чтобы, обращаясь к определенной услуги или району можно было обратиться и к статистике за определенный месяц и год.

Был создан класс контроллер AppealByMonthsController и реализован метод import(Request $request), в котором прописана логика импорта данных из загружаемого файла. Код метода import() класса AppealByMonthsController представлен в листинге 12.

Листинг 12 – Код метода import() класса AppealByMonthsController

```
public function import(Request $request)
    {
        $request->validate([
            'file' => 'required|file|mimes:xlsx,xls',
            'month' => 'required|integer|between:1,12',
            'year' => 'required|integer|between:2000,3000'
        ], [
            'file.required' => 'Выберите файл для загрузки',
            'file.mimes' => 'Выбранный файл должен быть формата Excel
(XLSX, XLS)',
            'month.required' => 'Выберите месяц',
            'month.integer' => 'Значение месяца должно быть числом',
            'month.between' => 'Значение месяца должно быть от 1 до 12',
            'year.integer' => 'Значение года должно быть числом',
            'year.between' => 'Значение года должно быть от 2000 до 3000'
```

```
        ]);
        $file = $request->file("file");
        $month = intval($request->input('month'));
        $year = intval($request->input('year'));
        if($file->isValid())
        {
            Excel::import(new AreaImport, $file);
            Excel::import(new AppealByMonthsImport($month, $year), $file);
            return redirect()->back()->with('success', 'all good');
        } else {
            return redirect()->back()->withErrors(['file' => 'Выбранный
файл некорректен']);
        }
    }
```

В данном методе добавлена валидация данных, приходящих из клиентской части. Помимо валидации расширения и наличия файла, которая есть во всех компонентах, также добавлена валидация на корректность месяца и года. Метод `import()` использует два импорт класса `AreaImport` и `AppealByMonthsImport`. Они используются в определенной очередности, согласно диаграмме последовательности. Считается, что при использовании этого компонента база данных уже имеет список услуг в таблице service.

Первым используется импорт класс `AreaImport`, который проходится по ячейкам загруженного excel-файла до ячейки со значением «Итоговое значение», считывает все районы и добавляет их в таблицу areas в базе данных. Код импорт класса `AreaImport` представлен в листинге 13.

Листинг 13 – Код импорт класса `AreaImport`

```
class AreaImport implements ToCollection
{
    public function collection(Collection $rows)
    {
        $areasRow = 0;
        $areaStepInRow = 5;
        $startReadingItem = 5;
        $flagForStopLoop = 'Итоговое значение';

        for($area = $startReadingItem; $area<$rows[$areasRow]->count()-1;
$area+=$areaStepInRow)
        {
            if(mb_strtolower($rows[$areasRow][$area]) !=
mb_strtolower($flagForStopLoop)) Area::createOrFirst(['name'=>
$rows[$areasRow][$area]]); else break;
        }
    }}
```

Вторым используется импорт класс `AppealByMonthsImport`, который заполняет таблицу appeals_by_months данными из загруженного excel-файла. В экземпляр данного класса поступает месяц и год, которые ввел пользователь при заполнении формы, дальше скрипт начинает искать услуги из сфер «имущество, стройка» и «стройка», которые есть в таблице service, следом в цикле в одной итерации из таблицы areas скрипт получает район, далее происходит проверка указанного месяца и, если это январь, данные напрямую сохраняются таблицу appeals_by_months, то есть, по данной услуге, всем районам, январю и указанному году создаются записи в таблицу, если месяц не январь, то система с помощью SQL запроса извлекает данные из таблицы за прошлые месяца относительного введенного пользователем месяца для соответствующего года, который ввел пользователь, после чего каждый раз вычитает их из импортированных значений. Таким образом скрип выполняется с помощью двойного цикла, основной перебирает услуги, а вложенный перебирает районы и рассчитывает значения обращений если выбран месяц не январь, а далее записывает все в таблицу appeals_by_months смотря уникальность записи по полям: услуга, район, месяц, год. Полученные результатом вычитания данные затем записываются в базу данных. Код импорт класса `AppealByMonthsImport` представлен в листинге 4 приложения Г.

Скрипт обеспечивает динамическое обновление данных в зависимости от введенных пользователем параметров, корректно учитывая предыдущие периоды и обеспечивая уникальность записей в таблице appeals_by_months

Интерфейс этого компонента выполнен в соответствии с макетом. На странице данного компонента находится форма для загрузки файла в систему, ниже расположены селектор, при нажатии на который открывается список для выбора месяца, а также поле для ввода года.

Интерфейс компонента обработки файлов со статистикой по доле обращений представлен на рисунке 16.

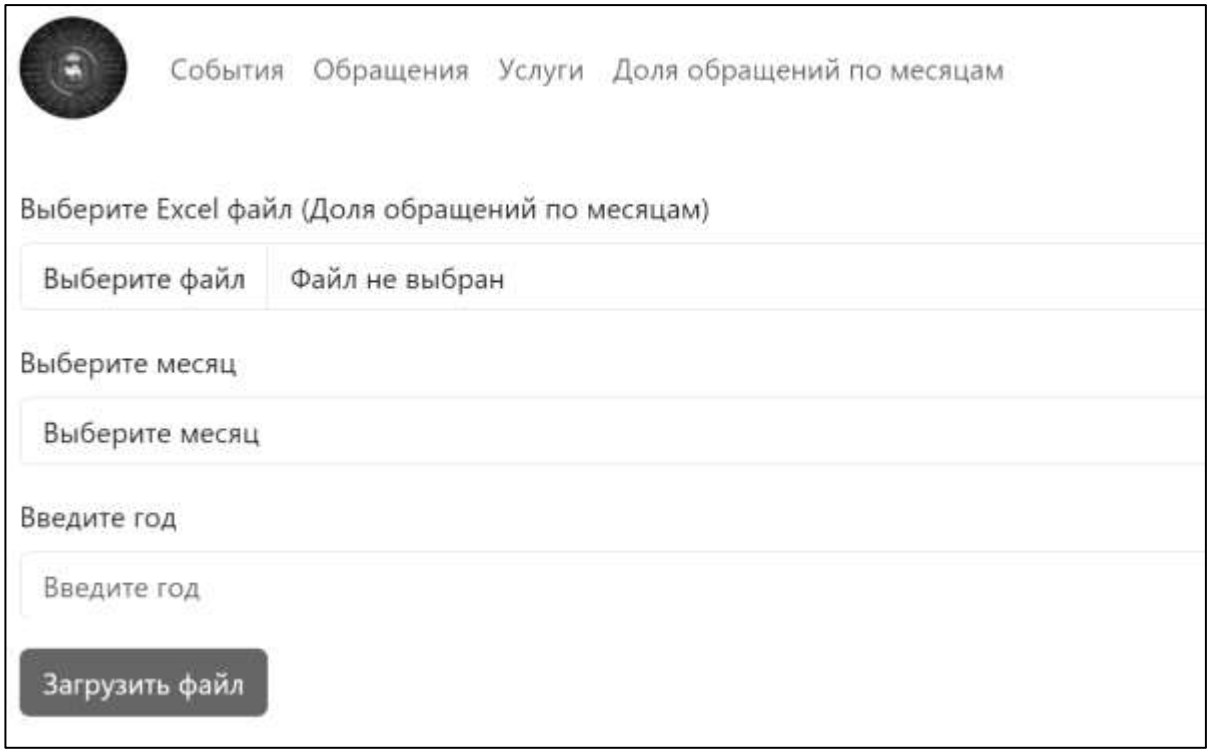

Рисунок 16 – Интерфейс страницы компонента обработки файлов со статистикой по доле обращений

Вывод по третьей главе

В данной главе описана реализация спроектированных компонентов. Все компоненты были реализованы в соответствии с их проектированием, функциональные требования каждого компонента выполнены. Интерфейс для каждого компонента выполнен в едином стиле и создан в соответствии со спроектированными макетами.

4. ТЕСТИРОВАНИЕ КОМПОНЕНТОВ

Для тестирования компонентов использовалось функциональное тестирование. Функциональное тестирование – это тестирование программного обеспечения в целях проверки реализуемости функциональных требований, то есть способности программного обеспечения в определенных условиях решать задачи, нужные пользователям [17]. Функциональные требования определяют, что именно делает программное обеспечение, какие задачи оно решает.

Компонент обработки файлов с данными о событиях

Набор тестов на функциональность компонента обработки файлов с данными о событиях представлен в таблице 1.

Таблица 1 – Тестирование компонента обработки файлов с данными о событиях

№	Аспект работы	Действие	Результат	Тест пройден?
1	Загрузка xlsx файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла загрузятся в базу данных и добавятся в таблицу	Да
2	Загрузка xls файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xls файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла загрузятся в базу данных и добавятся в таблицу	Да
3	Загрузка «ничего»	Нажать кнопку «Загрузить».	Получено уведомление о том, что нужно загрузить файл	Да
4	Загрузка не xlsx файл	1. Нажать кнопку «Выбрать файл». 2. Выбрать не xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено сообщение «Выбранный файл должен быть формата Excel (XLSX, XLS)»	Да

Продолжение таблицы 1

№	Аспект работы	Действие	Результат	Тест пройден?
5	Скачать отчет по всем данным из таблицы	1. Нажать кнопку «Скачать полный отчет». 2. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
6	Просмотр справки по методам фильтрации	Нажать кнопку «Справка по методам фильтрации».	Откроется модальное окно с текстом по фильтрам	Да
7	Убрать столбец Федеральный округ	1. Открыть селектор «Столбцы» 2. Убрать галочку на позиции Федеральный округ	Столбец Федеральный округ пропадет, станет неактивным	Да
8	Добавить фильтр данных	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Орловское»	В таблице останутся только записи, у которых в поле «СП/ГП» прописано «Орловское»	Да
9	Скачать отчет только по записям, где в поле «СП/ГП» прописано «Орловское»	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Орловское» 5. Нажать кнопку «Скачать отчет» 6. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
10	Экспортировать все данные	1. Нажать кнопку «Экспорт» 2. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
11	Экспортировать только отфильтрованные данные	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Орловское» 5. Нажать кнопку «Экспорт» 6. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да

Окончание таблицы 1

№	Аспект работы	Действие	Результат	Тест пройден?
12	Использование фильтр в заголовке	1. В фильтре столбца «МО/НП» выбрать «Симское»	В таблице останутся только записи, у которых в поле «МО/НП» прописано «Симское»	Да
13	Добавить два фильтра данных	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Миасс» 5. Нажать кнопку «Добавить фильтр» 6. Выбрать колонку «Сфера ЖКХ» 7. Выбрать метод фильтрации «Точное значение» 8. Прописать значение для фильтрации «Газоснабжение»	В таблице останутся только записи, у которых в поле «СП/ГП» прописано «Миасс» и в поле «Сфера ЖКХ» прописано «Газоснабжение»	Да
14	Очистить фильтры	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Миасс» 5. Увидеть, что в таблице остались только записи, у которых в поле «СП/ГП» прописано «Миасс» 6. Нажать кнопку «Очистить»	Получить таблицу без фильтров, со всеми данными	Да

Компонент обработки файлов с данными об обращениях

Набор тестов на функциональность компонента обработки файлов с данными об обращениях представлен в таблице 2.

Таблица 2 – Тестирование компонента обработки файлов с данными об обращениях

№	Аспект работы	Действие	Результат	Тест пройден?
1	Загрузка xlsx файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла загрузятся в базу данных и добавятся в таблицу	Да
2	Загрузка xls файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xls файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла загрузятся в базу данных и добавятся в таблицу	Да
3	Загрузка «ничего»	Нажать кнопку «Загрузить».	Получено уведомление о том, что нужно загрузить файл	Да
4	Загрузка не xlsx файл	1. Нажать кнопку «Выбрать файл». 2. Выбрать не xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено сообщение «Выбранный файл должен быть формата Excel (XLSX, XLS)»	Да
5	Скачать отчет по всем данным из таблицы	1. Нажать кнопку «Скачать полный отчет». 2. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
6	Просмотр справки по методам фильтрации	Нажать кнопку «Справка по методам фильтрации».	Откроется модальное окно с текстом по фильтрам	Да
7	Убрать столбец «Регион»	1. Открыть селектор «Столбцы» 2. Убрать галочку на позиции «Регион»	Столбец «Регион» пропадет, станет неактивным	Да
8	Добавить фильтр данных	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «Группа тем» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «ЖКХ»	В таблице останутся только записи, у которых в поле «Группа тем» прописано «ЖКХ»	Да

Продолжение таблицы 2

№	Аспект работы	Действие	Результат	Тест пройден?
9	Скачать отчет только по записям, где в поле «Группа тем» прописано «ЖКХ»	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «Группа тем» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «ЖКХ» 5. Нажать кнопку «Скачать отчет» 6. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
10	Экспортировать все данные	1. Нажать кнопку «Экспорт» 2. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
11	Экспортировать только отфильтрованные данные	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «СП/ГП» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «Орловское» 5. Нажать кнопку «Экспорт» 6. Скачать xlsx файл	Файл скачан, данные экспортированы корректно	Да
12	Использование фильтр в заголовке	В фильтре столбца «Итог» выбрать «Разъяснено»	В таблице останутся только записи, у которых в поле «Итог» прописано «Разъяснено»	Да
13	Добавить два фильтра данных	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «Группа тем» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «ЖКХ» 5. Нажать кнопку «Добавить фильтр» 6. Выбрать колонку «Аудитория» 7. Выбрать метод фильтрации «Точное значение» 8. Прописать значение для фильтрации «322452»	В таблице останутся только записи, у которых в поле «Группа тем» прописано «ЖКХ» и в поле «Аудитория» прописано «322452»	Да

Окончание таблицы 2

№	Аспект работы	Действие	Результат	Тест пройден?
14	Очистить фильтры	1. Нажать кнопку «Добавить фильтр» 2. Выбрать колонку «Группа тем» 3. Выбрать метод фильтрации «Точное значение» 4. Прописать значение для фильтрации «ЖКХ» 5. Увидеть, что в таблице остались только записи, у которых в поле «Группа тем» прописано «ЖКХ» 6. Нажать кнопку «Очистить»	Получить таблицу без фильтров, со всеми данными	Да

Компонент обработки файлов с данными об услугах

Набор тестов на функциональность компонента обработки файлов с данными об услугах представлен в таблице 3.

Таблица 3 – Тестирование компонента обработки файлов с данными об услугах

№	Аспект работы	Действие	Результат	Тест пройден?
1	Загрузка xlsx файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно загрузятся в базу данных	Да
2	Загрузка xls файла для компонента	1. Нажать кнопку «Выбрать файл». 2. Выбрать xls файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла загрузятся в базу данных и добавятся в таблицу	Да
3	Загрузка «ничего»	Нажать кнопку «Загрузить».	Получено уведомление о том, что нужно загрузить файл	Да

Окончание таблицы 3

№	Аспект работы	Действие	Результат	Тест пройден?
4	Загрузка не xlsx файл	1. Нажать кнопку «Выбрать файл». 2. Выбрать не xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено сообщение «Выбранный файл должен быть формата Excel (XLSX, XLS)»	Да

Компонент обработки файлов со статистикой по доле обращений

Набор тестов на функциональность компонента обработки файлов со статистикой по доле обращений представлен в таблице 4.

Таблица 4 – Тестирование компонента обработки файлов со статистикой по доле обращений

№	Аспект работы	Действие	Результат	Тест пройден?
1	Загрузка xlsx файла для компонента, месяц январь, год 2024	1 Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно загрузятся в базу данных	Да
2	Загрузка xlsx файла для компонента, месяц февраль, год 2024	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно преобразуются загрузятся в базу данных	Да
3	Загрузка xlsx файла для компонента, месяц март, год 2024	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно преобразуются загрузятся в базу данных	Да
4	Загрузка xlsx файла для компонента, месяц январь, год 2025	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно загрузятся в базу данных без преобразований	Да

Окончание таблицы 4

№	Аспект работы	Действие	Результат	Тест пройден?
5	Загрузка xlsx файла для компонента, месяц февраль, год 2025	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Данные из файла корректно преобразуются загрузятся в базу данных	Да
6	Загрузка «ничего»	1. Нажать кнопку «Загрузить».	Получено уведомление о том, что нужно загрузить файл	Да
7	Загрузка не xlsx файл	1. Нажать кнопку «Выбрать файл». 2. Выбрать не xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено сообщение «Выбранный файл должен быть формата Excel (XLSX, XLS)»	Да
8	Загрузка xlsx файла для компонента, месяц «отсутствует, год 2024	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено уведомление «Выберите один из пунктов списка»	Да
9	Загрузка xlsx файла для компонента, месяц январь, год «отсутствует»	1. Нажать кнопку «Выбрать файл». 2. Выбрать xlsx файл для компонента. 3. Нажать кнопку «Загрузить».	Получено уведомление «Заполните это поле»	Да

Вывод по четвертой главе

Было проведено функциональное тестирование всех компонентов. Результаты тестирования подтверждают выполнение всех функциональных требований к компонентам, а также корректную работоспособность системы в рамках поставленной задачи.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы были разработаны программные компоненты системы обработки документов, которые будут использоваться заказчиком для улучшения внутренних рабочих процессов. На этапе исследования были изучены современные подходы и технологии в области обработки документов. Были рассмотрены существующие решения и их недостатки, что позволило определить направления для улучшения. На основе проведенного анализа были спроектированы архитектура и спецификации программных компонентов. На этапе разработки были применены необходимые методы и инструменты программирования, также была проведена серия тестов, включающая функциональное тестирование.

Основные результаты

При выполнении поставленной задачи были достигнуты следующие основные результаты.

1. Проведен обзор литературы.
2. Выполнен проектирование программных компонентов.
3. Выполнена реализация программных компонентов.
4. Проведено тестирование программных компонентов.

Данные компоненты были внедрены в опытную эксплуатацию и используются в коммерческих целях, это подтверждает акт о внедрении научно-технической продукции, подписанный управляющим компании ООО «Что За Софт».

Направления дальнейших исследований

В рамках дальнейших исследований планируется поддержка компонентов, их оптимизация, улучшение системы, а также написание новых компонентов системы обработки документов.

ЛИТЕРАТУРА

1. Руководство по PHP. [Электронный ресурс] URL: https://www.php.net/manual/ru/ (дата обращения: 06.02.2024 г.).

2. HTTP аутентификация-HTTP. [Электронный ресурс] URL: https://developer.mozilla.org/ru/docs/Web/HTTP/Authentication (дата обращения: 06.02.2024 г.).

3. Фреймворк Laravel | Документация. [Электронный ресурс] URL: https://laravel.com/docs/10.x/documentation (дата обращения: 06.02.2024 г.).

4. HTML5 Basics For Everyone Tired Of Reading About Deprecated Code. [Электронный ресурс]. URL: https://html.com/html5/ (дата обращения: 06.02.2024 г.).

5. Современный учебник JavaScript. [Электронный ресурс] URL: https://learn.javascript.ru (дата обращения: 06.02.2024 г.).

6. MySQL | Documentation. [Электронный ресурс]. URL: https://dev.mysql.com/doc/ (дата обращения: 09.02.2024 г.).

7. phpMyAdmin | Documentation. [Электронный ресурс]. URL: https://www.phpmyadmin.net/docs/ (дата обращения: 09.02.2024 г.).

8. Laravel Excel | Documentation. [Электронный ресурс]. URL: https://docs.laravel-excel.com/3.1/getting-started/ (дата обращения: 10.02.2024 г.).

9. Дакетт Д. HTML и CSS. Разработка и дизайн веб-сайтов. / Джон Дакетт; [пер. с англ. М.А. Райтмана]. – М.: Эксмо, 2013. – 480 с.: ил. (дата обращения: 15.02.2024 г.).

10. Начало работы с вебом. [Электронный ресурс] URL: https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web (дата обращения: 15.02.2024 г.).

11. Основной учебник JavaScript. [Электронный ресурс] URL: https://code.mu/ru/javascript/book/prime/ (дата обращения: 02.03.2024 г.).

12. Tabulator.js | Documentation. [Электронный ресурс] URL: https://tabulator.info/docs/6.2 (дата обращения: 10.03.2024 г.).

13. Data Range Picker. [Электронный ресурс] URL: https://www.daterangepicker.com/ (дата обращения: 29.03.2024 г.).

14. Laravel | Маршрутизация. [Электронный ресурс] URL: https://laravel.su/docs/10.x/routing (дата обращения: 30.03.2024 г.).

15. Laravel | Eloquent. [Электронный ресурс] URL: https://laravel.su/docs/10.x/eloquent-relationships (дата обращения: 15.04.2024 г.).

16. Apache HTTP Server Version 2.4 Documentation. [Электронный ресурс] URL: https://httpd.apache.org/docs/2.4/en/ (дата обращения: 02.03.2024 г.).

17. Функциональное тестирование ПО: задачи, виды, методы проведения. [Электронный ресурс] URL: https://gb.ru/blog/funktsionalnoe-testirovanie-po/ (дата обращения: 10.05.2024 г.).

ПРИЛОЖЕНИЕ

Приложение А. Компонент обработки файлов с данными о событиях

Листинг 1 – JavaScript код представления компонента обработки файлов с данными о событиях

```
<script type="text/javascript">
            $(document).ready(function () {
                let tableData = {!! json_encode($events) !!};
                let columns = {!! json_encode($columns) !!};
                function minMaxFilterFunction(headerValue, rowValue, row-
Data, filterParams) {
                    if(headerValue.start == "" && headerValue.end == "") {
                        return true;
                    }
                    if(!isNaN(parseInt(rowValue))){
                        if (headerValue.start != "") {
                            if (headerValue.end != "") {
                                return rowValue >= headerValue.start &&
rowValue <= headerValue.end;
                            } else {
                                return rowValue >= headerValue.start;
                            }
                        } else {
                            if (headerValue.end != "") {
                                return rowValue <= headerValue.end;
                            }
                        }
                    }
                    return false;
                }
                 var minMaxFilterEditor = function(cell, onRendered, suc-
cess, cancel, editorParams){
                    var end;
                    var container = document.createElement("span");
                    //create and style inputs
                    var start = document.createElement("input");
                    start.setAttribute("type", "number");
                    start.setAttribute("placeholder", "Минимум");
                    start.setAttribute("min", 0);
                    start.setAttribute("max", 10000);
                    start.style.padding = "4px";
                    start.style.width = "50%";
                    start.style.boxSizing = "border-box";
                    start.value = cell.getValue();
                    function buildValues(){
                        success({
                            start:start.value,
                            end:end.value,
                        });
                    }
                    function keypress(e){
                        if(e.keyCode == 13){
                            buildValues();
                        }
                        if(e.keyCode == 27){
                            cancel();
                        }
                    }
```

```
                    end = start.cloneNode();
                    end.setAttribute("placeholder", "Максимум");
                    start.addEventListener("change", buildValues);
                    start.addEventListener("blur", buildValues);
                    start.addEventListener("keydown", keypress);
                    end.addEventListener("change", buildValues);
                    end.addEventListener("blur", buildValues);
                    end.addEventListener("keydown", keypress);
                    container.appendChild(start);
                    container.appendChild(end);
                    return container;
                    }
                    function rangeDateFilterFunction(headerValue, rowValue,
rowData, filterParams) {
                    if(!headerValue.dateField){
                       return true;
                   }
                   if(rowValue){
                       let dates = headerValue.dateField.split(' - ');
                       let pickerStart = new Date(dates[0]);
                       let pickerEnd = new Date(dates[1]);
                       let rowDate = new Date(rowValue);
                       if(rowDate.getTime() >= pickerStart.getTime() &&
rowDate.getTime() <= pickerEnd.getTime()){
                            return true;
                       }
                   }
                   return false;
               }
               let rangeDateFilterEditor = function (cell, onRendered, suc-
cess, cancel, editorParams) {
                   let dateField = document.createElement("input");
                   dateField.setAttribute("type", "text");
                   dateField.setAttribute("name", "dateFilter");
                   dateField.setAttribute("placeholder", "Промежуток");
                   dateField.style.padding = "4px";
                   dateField.style.width = "100%";
                   dateField.style.boxSizing = "border-box";
                   dateField.value = cell.getValue();
                   $(dateField).daterangepicker({
                       timePicker: true,
                       timePicker24Hour: true,
                       timePickerSeconds: true,
                       showDropdowns: true,
                       autoApply: true,
                       autoUpdateInput: true,
                       minDate: "2010-01-01 00:00:00",
                       maxDate: "2040-01-01 00:00:00",
                       startDate: "2023-01-01 00:00:00",
                       endDate: "2024-01-01 00:00:00",
                       ranges: {
                           'Сегодня': [moment(), moment()],
                           'Вчера': [moment().subtract(1, 'days'), mo-
ment().subtract(1, 'days')],
                           'Последние 7 дней': [moment().subtract(6,
'days'), moment()],
                           'Последние 30 дней': [moment().subtract(29,
'days'), moment()],
                           'Этот месяц': [moment().startOf('month'), mo-
ment().endOf('month')],
                           'Прошлый месяц': [moment().subtract(1,
'month').startOf('month'), moment().subtract(1, 'month').endOf('month')],
```

```
                        'Прошлые полгода': [moment().subtract(6,
'month'), moment()]
                    },
                    locale: {
                        "format": "YYYY-MM-DD HH:mm:ss",
                        "separator": " - ",
                        "applyLabel": "Применить",
                        "cancelLabel": "Отменить",
                        "fromLabel": "С",
                        "toLabel": "По",
                        "customRangeLabel": "Другое",
                        "weekLabel": "Н",
                        "daysOfWeek": [
                            "Вс",
                            "Пн",
                            "Вт",
                            "Ср",
                            "Чт",
                            "Пт",
                            "Сб"
                        ],
                        "monthNames": [
                            "Январь",
                            "Февраль",
                            "Март",
                            "Апрель",
                            "Май",
                            "Июнь",
                            "Июль",
                            "Август",
                            "Сентябрь",
                            "Октябрь",
                            "Ноябрь",
                            "Декабрь"
                        ],
                        "firstDay": 1
                    }
                }, 
                $(dateField).on("apply.daterangepicker", function(ev,
picker){
                    success({dateField : $(dateField).val()});
                });
                $(dateField).on('cancel.daterangepicker', function(ev,
picker) {
                    success({dateField : ''});
                });
                return dateField;
            }
             // Initialize Tabulator
             let table = new Tabulator("#data-table", {
                 height: "1000",
                 data: tableData,
                 layout: "fitData",
                 movableColumns: true,
                 columns: columns,
                 pagination: "local",
                 paginationSize: 10,
                 paginationSizeSelector: [10, 25, 50, 100],
                 stickyHeader: true, // Закрепить заголовок таблицы при
прокрутке данных
             });
             table.on("tableBuilt", function (){
```

```
                    for (let i = 0; i < columns.length; i++) {
                        switch (columns[i]['cssClass']){
                            case 'table-list-filter':
                                columns[i]['headerFilter'] = "list";
                                columns[i]['headerFilterParams'] = {
                                    "valuesLookup": true,
                                    "multiselect": true
                                };
                                columns[i]['editor'] = "list";
                                columns[i]['editorParams'] = {"val-
uesLookup": "active"};
                            break;
                            case 'table-minmax-filter':
                                columns[i]['headerFilter'] = minMaxFil-
terEditor;
                                columns[i]['headerFilterFunc'] = minMaxFil-
terFunction;
                                columns[i]['headerFilterLiveFilter'] =
false;
                            break;
                            case 'table-datetime-filter':
                                columns[i]['headerFilter'] = rangeDateFil-
terEditor;
                                columns[i]['headerFilterFunc'] = rangeDate-
FilterFunction;
                                columns[i]['headerFilterLiveFilter'] =
false;
                            break;
                        }
                    }
                    table.setColumns(columns);
                    var endTime = performance.now();
                    console.log(`Call to doSomething took ${endTime -
startTime} milliseconds`);
                });
                 document.getElementById("download-xlsx").addEventLis-
tener("click", function () {
                    table.download("xlsx", "export_events_" + new
Date().toJSON().slice(0, 10).replace(/-/g, '-') + ".xlsx", {sheetName:
"Events" + new Date().toJSON().slice(0, 10).replace(/-/g, '-')});
                 });
                 // Add/Delete columns from dropdown
                 $('.dropdown-column').on('click', function () {
                    let columnId = $(this).attr('id');
                    let checkVisible = $(this).is(':checked')
                    table.updateColumnDefinition(columnId, {
                        visible: checkVisible
                    });
                    table.getColumn(columnId).setHeaderFilterValue("");
                 });
                 // Filter handling
                 function updateFilters() {
                    var filters = [];
                    $(".filter-row:visible").each(function () {
                        var field = $(this).find(".filter-field").val();
                        var type = $(this).find(".filter-type").val();
                        var value = $(this).find(".filter-value").val();
                        if (field && type && value) {
                            filters.push({field: field, type: type, value:
value});
                        }
```

```
                    });
                    table.setFilter(filters);
                    if (!$('#filter-clear').is(':hidden')) {
                        let notHiddenFlag = 0;
                        $('.filter-rows-container .filter-row').each(func-
tion () {
                            if (!$(this).is(':hidden')) {
                                notHiddenFlag = 1;}});
                        if (!notHiddenFlag) {
                            $('#filter-clear').attr('hidden', true);}}}
                $(".filter-rows-container").on("change input", "select, in-
put", function () {
                    updateFilters();
                });
                $("#filter-clear").on("click", function () {
                    var firstValue = $(".filter-field option:se-
lected").val();
                    var secondValue = $(".filter-type option:se-
lected").val();
                    $(".filter-field").val(firstValue);
                    $(".filter-type").val(secondValue);
                    $(".filter-value").val("");
                    table.clearFilter();
                });
                $("#add-filter").on("click", function () {
                    if ($('.filter-rows-container').length && $('#filter-
clear').is(':hidden')) {
                        $('#filter-clear').removeAttr('hidden');
                    }
                    var newRow = $(".filter-row:first").clone();
                    newRow.find("input").val("");
                    newRow.appendTo(".filter-rows-container");
                    newRow.fadeIn(); // Покажем новую строку
                    updateFilters();});
                $(".filter-rows-container").on("click", ".remove-filter",
function () {
                    $(this).closest(".filter-row").fadeOut(function () {
                        $(this).remove();
                        updateFilters();});});
                $(".filter-rows-container").on("click", ".clear-filter",
function () {
                    var row = $(this).closest(".filter-row");
                    var firstValue = $(".filter-field option:se-
lected").val();
                    var secondValue = $(".filter-type option:se-
lected").val();
                    row.find(".filter-field").val(firstValue);
                    row.find(".filter-type").val(secondValue);
                    row.find(".filter-value").val("");
                    updateFilters();
                });
                $("#export-button").click(function () {
                    exportFilteredData();
                });
                function exportFilteredData() {
                    window.location.href = '{{route('eventsExportFil-
tered')}}?filters=' + encodeURIComponent(JSON.stringify(table.getFil-
ters()));}});</script>
```

Приложение Б. Код основного макета

Листинг 2 – Код основного макета

```
<!doctype html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- CSRF Token -->
    <meta name="csrf-token" content="{{ csrf_token() }}">

    <title>{{ config('app.name', 'Laravel') }}</title>

    <!-- Fonts -->
    <link rel="dns-prefetch" href="//fonts.bunny.net">
    <link href="https://fonts.bunny.net/css?family=Nunito"
rel="stylesheet">

    <!-- Scripts -->
    <!-- Bootstrap CSS (jsDelivr CDN) -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/boot-
strap.min.css" rel="stylesheet" integrity="sha384-9ndCyUaIbzAi2FUVXJi0Cjm-
CapSmO7SnpJef0486qhLnuZ2cdeRhO02iuK6FUUVM" crossorigin="anonymous">
    <!-- Bootstrap Bundle JS (jsDelivr CDN) -->
    <script defer src="https://cdn.jsdelivr.net/npm/boot-
strap@5.3.0/dist/js/bootstrap.bundle.min.js" integrity="sha384-
geWF76RCwLtnZ8qwWowPQNguL3RmwHVBC9FhGdlKrxdiJJigb/j/68SIy3Te4Bkz" cros-
sorigin="anonymous"></script>

    <!-- Tabulator.js -->
    <link href="https://unpkg.com/tabulator-tables@5.5.2/dist/css/tabula-
tor.min.css" rel="stylesheet">
    <script type="text/javascript" src="https://unpkg.com/tabulator-ta-
bles@5.5.2/dist/js/tabulator.min.js"></script>
    <link href="https://cdnjs.cloudflare.com/ajax/libs/tabula-
tor/5.5.2/css/tabulator_bootstrap5.min.css" rel="stylesheet">
    <script type="text/javascript"
src="https://oss.sheetjs.com/sheetjs/xlsx.full.min.js"></script>

    <script src="https://ajax.goog-
leapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

    <!-- DataRangePicker.js -->
    <script type="text/javascript" src="https://cdn.jsdelivr.net/mo-
mentjs/latest/moment.min.js"></script>
    <script type="text/javascript" src="https://cdn.jsdelivr.net/npm/dat-
erangepicker/daterangepicker.min.js"></script>
    <link rel="stylesheet" type="text/css" href="https://cdn.jsde-
livr.net/npm/daterangepicker/daterangepicker.css" />


</head>
<body>
    <div id="app">
        <nav class="navbar navbar-expand-md navbar-light bg-white shadow-
sm">
            <div class="container-fluid">
                <a class="navbar-brand" href="{{ url('/') }}">
                    <img src="{{asset('img/minstroylogo.jpg')}}"
class="img-fluid" style="border-radius: 50%;" alt={{ config('app.name',
'Laravel') }}>
                </a>
```

```
                <button class="navbar-toggler" type="button" data-bs-tog-
gle="collapse" data-bs-target="#navbarSupportedContent" aria-con-
trols="navbarSupportedContent" aria-expanded="false" aria-label="{{
__('Toggle navigation') }}">
                    <span class="navbar-toggler-icon"></span>
                </button>

                <div class="collapse navbar-collapse" id="navbarSupported-
Content">
                    <!-- Left Side Of Navbar -->
                    <ul class="navbar-nav me-auto">
                        @guest
                        @else
                            <li class="nav-item">
                                <a class="nav-link"
href="/events">События</a>
                            </li>
                            <li class="nav-item">
                                <a class="nav-link" href="/ap-
peals">Обращения</a>
                            </li>
                            <li class="nav-item">
                                <a class="nav-link" href="/ser-
vice">Услуги</a>
                            </li>
                            <li class="nav-item">
                                <a class="nav-link" href="/appealsBy-
Months">Доля обращений по месяцам</a>
                            </li>
                        @endguest
                    </ul>

                    <!-- Right Side Of Navbar -->
                    <ul class="navbar-nav ms-auto">
                        <!-- Authentication Links -->
                        @guest
                            <!--@if (Route::has('login'))
                                <li class="nav-item">
                                    <a class="nav-link" href="{{
route('login') }}">{{ __('Login') }}</a>
                                </li>
                            @endif-->

                            <!--@if (Route::has('register'))
                                <li class="nav-item">
                                    <a class="nav-link" href="{{
route('register') }}">{{ __('Register') }}</a>
                                </li>
                            @endif-->
                        @else
                            <li class="nav-item dropdown">
                                <a id="navbarDropdown" class="nav-link
dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-
haspopup="true" aria-expanded="false" v-pre>
                                    {{ Auth::user()->name }}
                                </a>
                                <div class="dropdown-menu dropdown-menu-
end" aria-labelledby="navbarDropdown">
                                    <a class="dropdown-item" href="{{
route('logout') }}"
                                       onclick="event.preventDefault();
```

Окончание листинга 2 приложения В

```
                                                document.getEle-
mentById('logout-form').submit();">
                                                    {{ __('Logout') }}
                                                </a>

                                                <form id="logout-form" action="{{
route('logout') }}" method="POST" class="d-none">
                                                    @csrf
                                                </form>
                                            </div>
                                        </li>
                                    @endguest
                                </ul>
                            </div>
                        </div>
                    </nav>

                    <main class="py-4">
                        @yield('content')
                    </main>
                </div>
            </body>
            </html>
```

Приложение В. Код класса экспорта EventExport

Листинг 3 – Код класса экспорта `EventExport`.

```
class EventExport implements FromCollection, WithHeadings, WithMapping,
WithColumnWidths, WithStyles, WithStartRow
{
    private $rowNumber= 0;
    protected $filteredData;
    public function __construct($filteredData = null)
    {
        $this->filteredData = $filteredData;
    }

    public function collection()
    {
        if ($this->filteredData)
        {
            return $this->filteredData;
        }

        $query = Event::select('MR_from_registry',
                               'MO_NP',
                               'address_of_object',
                               'sector_of_JKH',
                               'type_of_event',
                               'short_event',
                               'name_of_expl_org',
                               'date_incident_local_time',
                               'date_incident_elimination_local_time',
                               'solution_status',
                                );
        return $query->get();
    }

    public function headingRow(): int
    {
        return 1;
    }
    public function headings(): array
    {
        return [
            'Номер п/п',
            'Муниципальное образование',
            'Населенный пункт',
            'Адрес производства работ',
            'Сфера (теплоснабжение/газоснабжение/электроснабжение/водоснаб-
жение/водоотведение)',
            'Характер производства работ (плановые/аварийные/с отключе-
нием/со снижением)',
            'Описание производства работ',
            'Производитель работ',
            'Период выполнения работ(Начало)',
            'Период выполнения работ(Конец)',
            'Отключение(МКД)',
            'Количество домовладений(Отключено)',
            'Количество домовладений(Снижены параметры)',
            'Отключение(СЗО)',
            'Количество домовладений(Отключено)',
            'Количество домовладений(Снижены параметры)',
            'Отключение(ИЖС/ЧД)',
            'Количество домовладений(Отключено)',
            'Количество домовладений(Снижены параметры)',
```

```
            'Принятые меры',
            'Примечание',
        ];
    }
    public function map($row): array
    {
        return [
            sprintf('%d.', $this->getRowCount()),
            $row->MR_from_registry,
            $row->MO_NP,
            $row->address_of_object,
            $row->sector_of_JKH,
            $row->type_of_event,
            $row->short_event,
            $row->name_of_expl_org,
            $row->date_incident_local_time,
            $row->date_incident_elimination_local_time,
            '',
            '',
            '',
            '',
            '',
            '',
            '',
            '',
            '',
            '',
            $row->solution_status,
        ];
    }

    public function styles(Worksheet $sheet)
{
    $highestColumn = Coordinate::columnIndexFromString($sheet->getHigh-
estColumn());

    $cellRange = 'A1:' . $sheet->getHighestColumn() . $sheet->getHigh-
estRow();
    $styleArray = [
        'borders' => [
            'allBorders' => [
                'borderStyle' => Border::BORDER_THIN,
                'color' => ['rgb' => '000000'],
            ],
        ],
    ];
    $sheet->getStyle('A')->getAlignment()->setHorizontal(Alignment::HORI-
ZONTAL_CENTER);
    $sheet->getStyle($cellRange)->applyFromArray($styleArray);
    $sheet->getStyle($cellRange)->getAlignment()->setWrapText(true);

    $styles = [];

    for ($row = 1; $row <= $sheet->getHighestRow(); $row++) {
        $styles[$row] = [
            'alignment' => [
                'horizontal' => Alignment::HORIZONTAL_LEFT,
                'vertical' => Alignment::VERTICAL_CENTER,
                'wrapText' => true,
            ],
        ];
    }
```

```
    $styles = [
        1 => [
            'font' => ['bold' => true],
            'fill' => ['fillType' => 'solid', 'startColor' => ['rgb' =>
'FFA500']],
            'alignment' => [
                'horizontal' => Alignment::HORIZONTAL_CENTER,
                'vertical' => Alignment::VERTICAL_TOP,
                'wrapText' => true,
            ],
        ],
        'A' => ['alignment' =>['horizontal' => Alignment::HORIZONTAL_CEN-
TER, 'vertical' => Alignment::VERTICAL_CENTER,]],
    ];

    return $styles;
    }
    public function columnWidths(): array
    {
        return [
            'A' => 8,
            'B' => 30,
            'C' => 19,
            'D' => 66,
            'E' => 90,
            'F' => 24,
            'G' => 48,
            'H' => 20,
            'I' => 17,
            'J' => 17,
            'K' => 22,
            'L' => 22,
            'M' => 22,
            'N' => 22,
            'O' => 22,
            'P' => 22,
            'Q' => 22,
            'R' => 22,
            'S' => 22,
            'T' => 20,
            'U' => 20,

        ];
    }
    private function getRowCount(): int {return ++$this->rowNumber;}
    public function startRow(): int{return 3;}
}
```

Приложение Г. Код класса AppealByMonthsImport

Код класса `AppealByMonthsImport` представлен в листинге 4.

Листинг 4 – Код класса `AppealByMonthsImport`

```
class AppealByMonthsImport implements ToCollection, WithCalculatedFormulas
{
    private $month;
    private $year;
    public function __construct($month, $year)
    {
        $this->month = $month;
        $this->year = $year;
    }
    public function collection(Collection $rows)
    {
        $startReadingRow = 2;
        $startReadingItem = 5;
        $sectorInRow = 4;
        $areaStep = 5;
        $areaRow = 0;
        $itemStep = 1;
        $serviceIdQualifier = 0;
        $stopItemReadFlag = 'Итоговое значение';
        $sector = 'стройка';
        $DBdata = DB::table('appeals_by_months')->select('area_id', 'ser-
vice_id', DB::raw('SUM(number_applications_in_electronic_form) AS sum_num-
ber_applications_in_electronic_form'), DB::raw('SUM(number_applications) AS
sum_number_applications'))
                    ->where([['month', '<', $this->month], ['year', '=',
$this->year]])->groupBy('area_id', 'service_id')->get();
        for ($row=$startReadingRow; $row < $rows->count()-2; $row++) { //-2
чтобы исключить итог
            if((strpos(mb_strtolower($rows[$row][$sectorInRow]), $sector)
|| mb_strtolower($rows[$row][$sectorInRow]) == $sector))
            {
                $service = Service::where('name', $rows[$row][$serviceId-
Qualifier])->first();
                for ($item=$startReadingItem; $item < $rows[$row]->count()-
1; $item+=$areaStep) {
                    if (!strpos(mb_strtolower($rows[$areaRow][$item]),
mb_strtolower($stopItemReadFlag)) && mb_strtolower($rows[$area-
Row][$item]) != mb_strtolower($stopItemReadFlag)) {
                        $area = Area::where('name', $rows[$area-
Row][$item])->first();
                        if($this->month != 1){
                            $DBRow = $DBdata->where('area_id', '=', $area-
>id)->where('service_id', '=', $service->id)->first();
                            $DBSumNumberApplicationsInElectronicForm =
$DBRow->sum_number_applications_in_electronic_form;
                            $DBSumNumberApplications = $DBRow->sum_num-
ber_applications;
                            $number_applications_in_electronic_form =
$rows[$row][$item] - $DBSumNumberApplicationsInElectronicForm;
                            $number_applications = $rows[$row][$item+$item-
Step] -$DBSumNumberApplications;
                        }else{
                            $number_applications_in_electronic_form =
$rows[$row][$item];
                            $number_applications = $rows[$row][$item+$item-
Step];
                        }
```

```
                        if($number_applications_in_electronic_form==null ||
$number_applications==null)
                        {
                            $percent_of_electronic_forms_of_total =
round(0, 2);
                        }else{
                            $percent_of_electronic_forms_of_total =
round($number_applications_in_electronic_form/$number_applications * 100,
2);}
                        AppealByMonths::updateOrCreate(
                            ['area_id'=> $area->id, 'service_id'=> $ser-
vice->id, 'month' => $this->month, 'year'=> $this->year],
                            ['number_applications_in_electronic_form' =>
$number_applications_in_electronic_form, 'number_applications' => $num-
ber_applications, '%_of_electronic_forms_of_total' => $percent_of_elec-
tronic_forms_of_total]);
                    }
                }
            }
        }
    }
}
```

Приложение Д. Код файла маршрутизации

Код файла маршрутизации представлен в листинге 5.

Листинг 5 – Код файла маршрутизации

```
<?php

use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Route;

Route::middleware('auth')->group(function () {
Route::get('/events',
[App\Http\Controllers\EventController::class,'index'])->name('events');
Route::post('/events',
[App\Http\Controllers\EventController::class,'import'])-
>name('eventsImport');
Route::get('/events',
[App\Http\Controllers\EventController::class,'viewDataTable'])-
>name('eventsViewData');
Route::get('/events/export',
[App\Http\Controllers\EventController::class,'export'])-
>name('eventsExport');
Route::get('/events/exportFilteredData',
[App\Http\Controllers\EventController::class,'exportFilteredData'])-
>name('eventsExportFiltered');
Route::get('/appeals',
[App\Http\Controllers\AppealController::class,'index'])->name('appeals');
Route::post('/appeals',
[App\Http\Controllers\AppealController::class,'import'])-
>name('appealsImport');
Route::get('/appeals',
[App\Http\Controllers\AppealController::class,'viewDataTable'])-
>name('appealsViewData');
Route::get('/appeals/export',
[App\Http\Controllers\AppealController::class,'export'])-
>name('appealsExport');
Route::get('/appeals/exportFilteredData',
[App\Http\Controllers\AppealController::class,'exportFilteredData'])-
>name('appealsExportFiltered');
Route::get('/service',
[App\Http\Controllers\ServiceController::class,'index'])->name('service');
Route::post('/service',
[App\Http\Controllers\ServiceController::class,'import'])-
>name('serviceImport');
Route::get('/appealsByMonths',
[App\Http\Controllers\AppealByMonthsController::class,'index'])-
>name('appealsByMonth');
Route::post('/appealsByMonths',
[App\Http\Controllers\AppealByMonthsController::class,'import'])-
>name('appealsByMonthImport');
});
Auth::routes();
Route::get('/', [App\Http\Controllers\HomeController::class, 'index'])-
>name('home');
```

Приложение Е. Код файла миграции для таблицы for_omas_control

Код файла миграции представлен в листинге 6.

Листинг 6 – Код файла миграции

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('for_omas_control', function (Blueprint $table) {
            $table->id();
            $table->string('solution_status')->nullable();
            $table->dateTime('date_incident_local_time')->format('d.m.Y
H:i')->nullable();
            $table->dateTime('date_incident_MSK_time')->format('d.m.Y
H:i')->nullable();
            $table->string('federal_district')->nullable();
            $table->string('subject_RF')->nullable();
            $table->string('MR_from_registry')->nullable();
            $table->string('SP_GP')->nullable();
            $table->string('MO_NP')->nullable();
            $table->text('short_event')->nullable();
            $table->string('type_of_event')->nullable();
            $table->string('sector_of_JKH')->nullable();
            $table->string('process_stage')->nullable();
            $table->string('object_of_incident')->nullable();
            $table->text('type_of_incident')->nullable();
            $table->string('object_elements')->nullable();
            $table->string('view_of_object_until_30_02_2022')->nullable();
            $table->string('type_of_object_until_30_02_2022')->nullable();
            $table->string('subtype_of_object_until_30_02_2022')->nulla-
ble();
            $table->string('name_of_object')->nullable();
            $table->string('address_of_object')->nullable();
            $table->string('coordinates')->nullable();
            $table->string('name_owner')->nullable();
            $table->string('name_of_expl_org')->nullable();
            $table->string('weather')->nullable();
            $table->text('type_of_accident')->nullable();
            $table->string('subtype_of_accident')->nullable();
            $table->integer('number_of_death')->nullable();
            $table->integer('number_of_victims')->nullable();
            $table->string('P_list_of_NP')->nullable();
            $table->string('P_first_category_of_objects')->nullable();
            $table->integer('P_number_of_social_objects')->nullable();
            $table->integer('P_number_of_MKD')->nullable();
            $table->integer('P_number_of_residents')->nullable();
            $table->integer('P_number_of_IJS')->nullable();
            $table->integer('P_number_of_residents_in_IJS')->nullable();
            $table->string('P_list_of_other_objects')->nullable();
            $table->string('CH_list_of_NP')->nullable();
            $table->string('CH_first_category_of_objects')->nullable();
            $table->integer('CH_number_of_social_objects')->nullable();
```

```
            $table->integer('CH_number_of_MKD')->nullable();
            $table->integer('CH_number_of_residents')->nullable();
            $table->integer('CH_number_of_IJS')->nullable();
            $table->integer('CH_number_of_residents_in_IJS')->nullable();
            $table->string('CH_list_of_other_objects')->nullable();
            $table->string('Sv_sector_of_JKH')->nullable();
            $table->string('Sv_list_of_NP')->nullable();
            $table->string('Sv_first_category_of_objects')->nullable();
            $table->integer('Sv_number_of_social_objects')->nullable();
            $table->integer('Sv_number_of_MKD')->nullable();
            $table->integer('Sv_number_of_residents')->nullable();
            $table->integer('Sv_number_of_IJS')->nullable();
            $table->integer('Sv_number_of_residents_in_IJS')->nullable();
            $table->string('Sv_list_of_other_objects')->nullable();
            $table->dateTime('date_incident_elimination_local_time')->for-
mat('d.m.Y H:i')->nullable();
            $table->dateTime('date_incident_elimination_MSK_time')->for-
mat('d.m.Y H:i')->nullable();
            $table->integer('event_id')->nullable();
            $table->dateTime('date_post_creation')->format('d.m.Y H:i')-
>nullable();
            $table->string('per_planing_shutdown_from')->nullable();
            $table->string('per_planing_shutdown_to')->nullable();
            $table->dateTime('date_post_update')->format('d.m.Y H:i')->nul-
lable();
            $table->string('user_login')->nullable();
            $table->string('user_name')->nullable();
            $table->string('organization_name')->nullable();
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('for_omas_control');
    }
};
```

Приложение Ж. Код класса AppealExport

Код класса `AppealExport` представлен в листинге 7.

Листинг 7 – Фрагмент кода класса экспорта `AppealExport`

```
class AppealExport implements FromCollection, WithHeadings, WithMapping, WithColumnWidths, WithStyles, WithStartRow
{
    private $rowNumber= 0;
    protected $filteredData;
    public function __construct($filteredData = null)
    {
        $this->filteredData = $filteredData;

    }
    public function collection()
    {
    if ($this->filteredData) {
        //dd($this->filteredData);
        return $this->filteredData;
    }
    $query = Appeal::select('district_or_locality', 'start_heating_season', 'incident_text', 'executor','url_post','create_time', 'result','note');

    return $query->get(); // Используйте get() для выполнения запроса и получения коллекции моделей
    }
    public function headingRow(): int
    {
        return 1;
    }
    public function headings(): array
    {
        return [
            'Номер п/п',
            'Муниципальное образование',
            'Населенный пункт',
            'Тема',
            'Текст инцидента',
            'Исполнитель',
            'Урл поста',
            'Дата создания',
            'Приняты меры',
            'Примечание',

        ];
    }
    public function map($row): array
    {
        return [
            sprintf('%d.', $this->getRowCount()),
            '',
            $row->district_or_locality,
            $row->start_heating_season,
            $row->incident_text,
            $row->executor,
            $row->url_post,
            $row->create_time,
            $row->result,
            $row->note,
        ];
    }
```

```
    public function styles(Worksheet $sheet)
{
    // Используем getHighestColumn для получения количества колонок
    $highestColumn        =        Coordinate::columnIndexFromString($sheet-
>getHighestColumn());

    // Мержим ячейки и устанавливаем стили для первых двух строк каждой
колонки
    // for ($col = 1; $col <= $highestColumn; $col++) {
    //     $columnLetter = Coordinate::stringFromColumnIndex($col);
    //          $sheet->mergeCells(sprintf('%s%d:%s%d',  $columnLetter,  1,
$columnLetter, 2));
    //     if($columnLetter === "J") { break; }
    // }

    $cellRange    =    'A1:'    .    $sheet->getHighestColumn()    .    $sheet-
>getHighestRow();
    $styleArray = [
        'borders' => [
            'allBorders' => [
                'borderStyle' => Border::BORDER_THIN,
                'color' => ['rgb' => '000000'],
            ],
        ],
    ];
    $sheet->getStyle('A')->getAlignment()-
>setHorizontal(Alignment::HORIZONTAL_CENTER);
    $sheet->getStyle($cellRange)->applyFromArray($styleArray);
    $sheet->getStyle($cellRange)->getAlignment()->setWrapText(true);
    $styles = [];
    for ($row = 2; $row <= $sheet->getHighestRow(); $row++) {
        $styles[$row] = [
            'alignment' => [
                'horizontal' => Alignment::HORIZONTAL_LEFT,
                'vertical' => Alignment::VERTICAL_CENTER,
                'wrapText' => true,
            ],
        ];
    }

    $styles = [
        1 => [
            'font' => ['bold' => true],
            'fill'  =>  ['fillType'  =>  'solid',  'startColor'  =>  ['rgb'  =>
'FFA500']],
            'alignment' => [
                'horizontal' => Alignment::HORIZONTAL_CENTER,
                'vertical' => Alignment::VERTICAL_TOP,
                'wrapText' => true,
            ],
        ],
        'A' => ['alignment' =>['horizontal' => Alignment::HORIZONTAL_CENTER,
'vertical' => Alignment::VERTICAL_CENTER,]],
    ];

    return $styles;
}

    public function columnWidths(): array
    {
        return [
            'A' => 8,
```

```
            'B' => 28,
            'C' => 21,
            'D' => 66,
            'E' => 150,
            'F' => 38,
            'G' => 48,
            'H' => 18,
            'I' => 17,
            'J' => 150,
        ];
    }
    private function getRowCount(): int
    {
        return ++$this->rowNumber;
    }
    public function startRow(): int
    {
        return 3;
    }
}
```

Приложение 3. Диаграмма вариантов использования

Диаграмма вариантов использования представлена на рисунке 2.

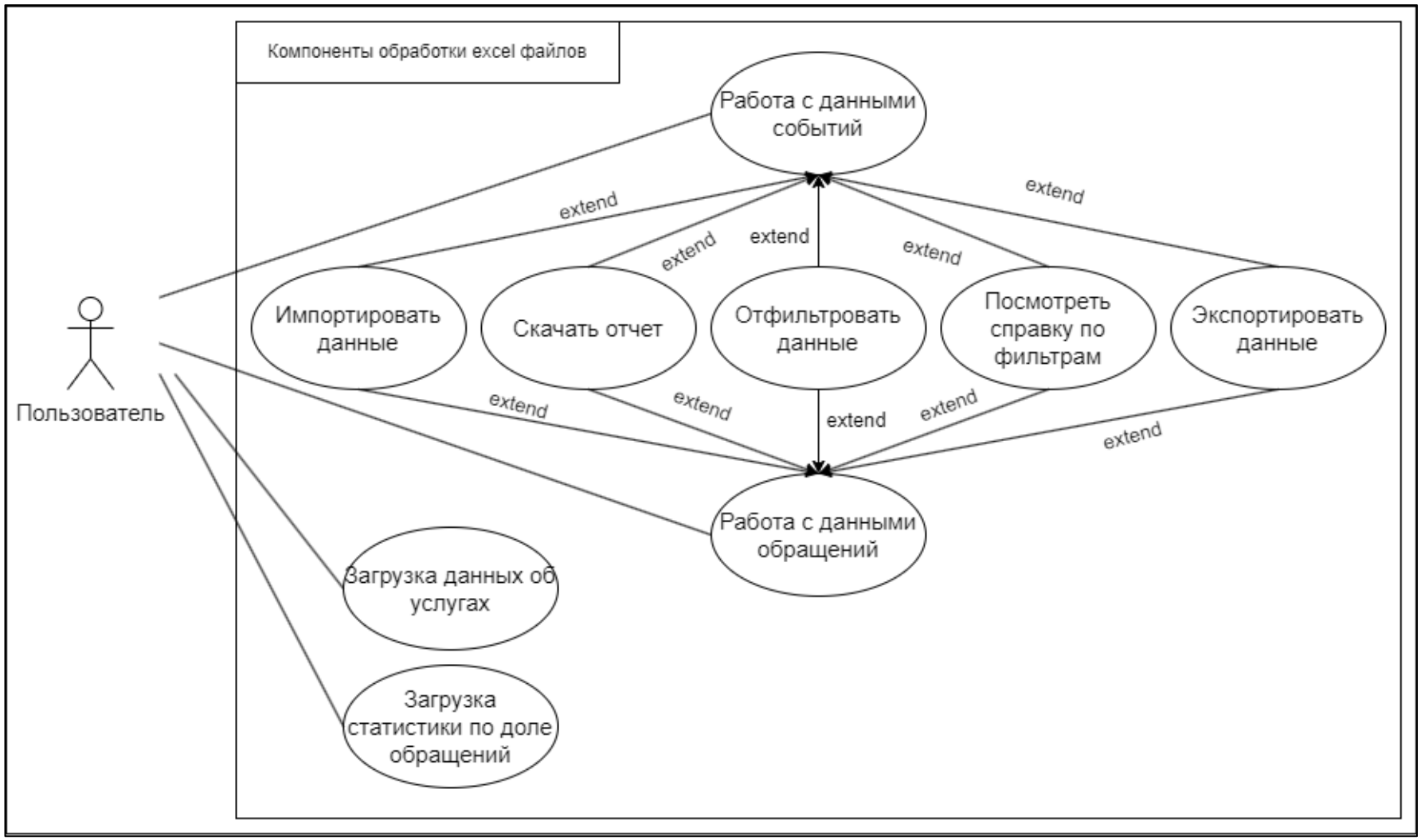

Рисунок 1 – Диаграмма вариантов использования компонентов

Приложение И. Архитектура системы

Архитектура системы представлена на рисунке 3.

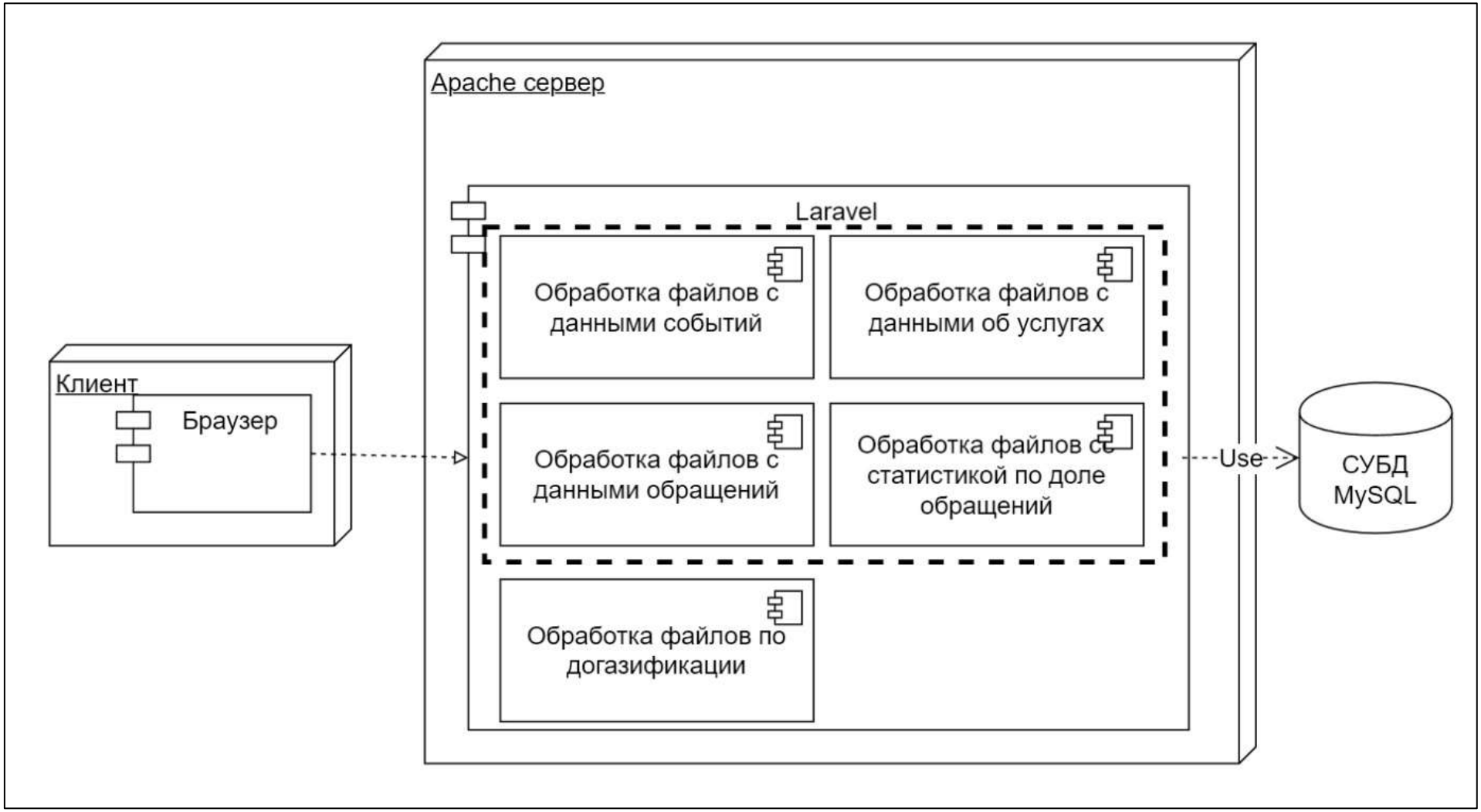

Рисунок 2 – Архитектура системы

Приложение К. Последовательность взаимодействий компонента обработки файлов с данными об услугах с базой данных

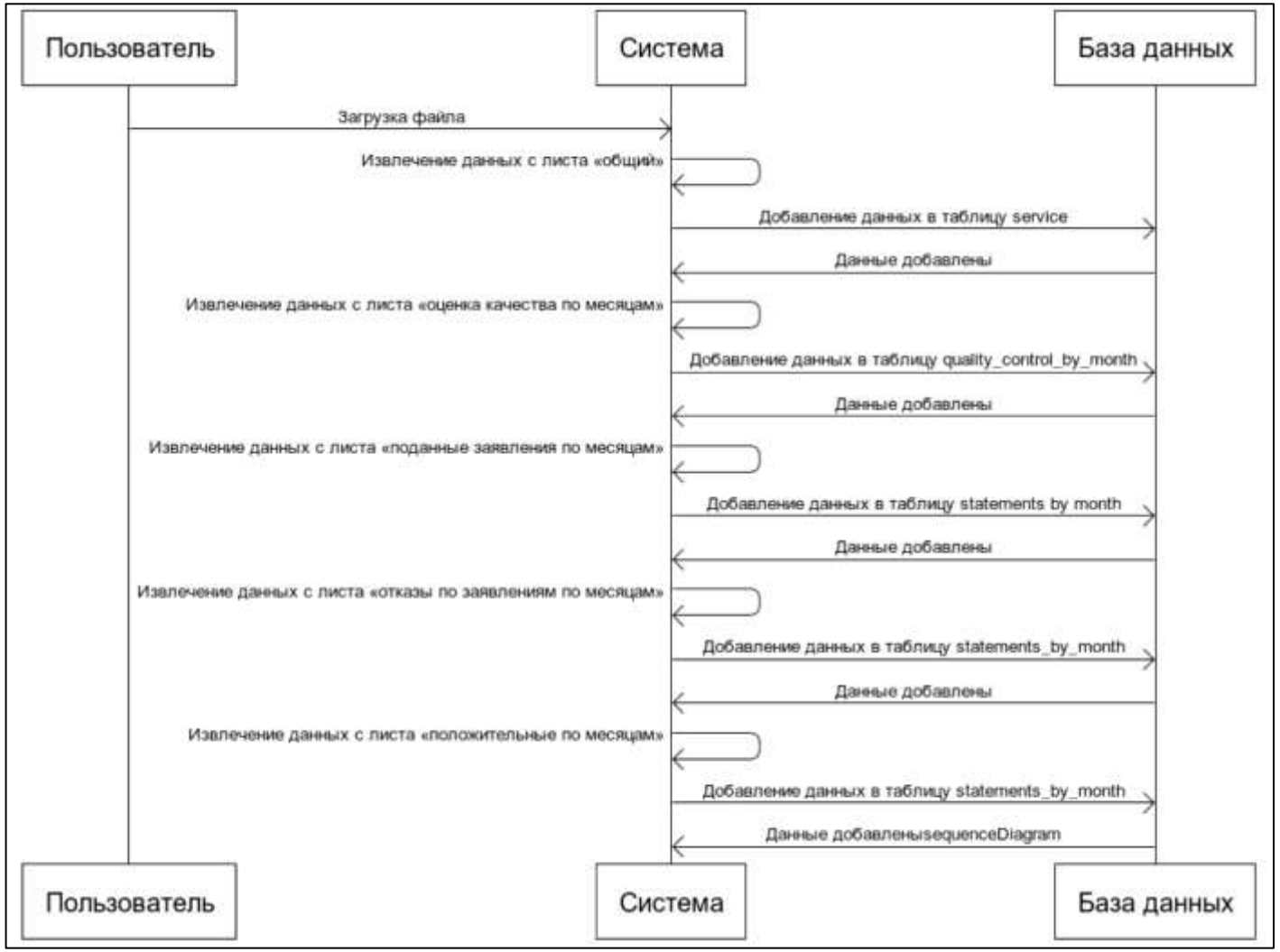

Рисунок 3 – Диаграмма последовательности взаимодействия компонента и базы данных

Приложение Л. Последовательность взаимодействий компонента обработки файлов со статистикой по доле обращений с базой данных

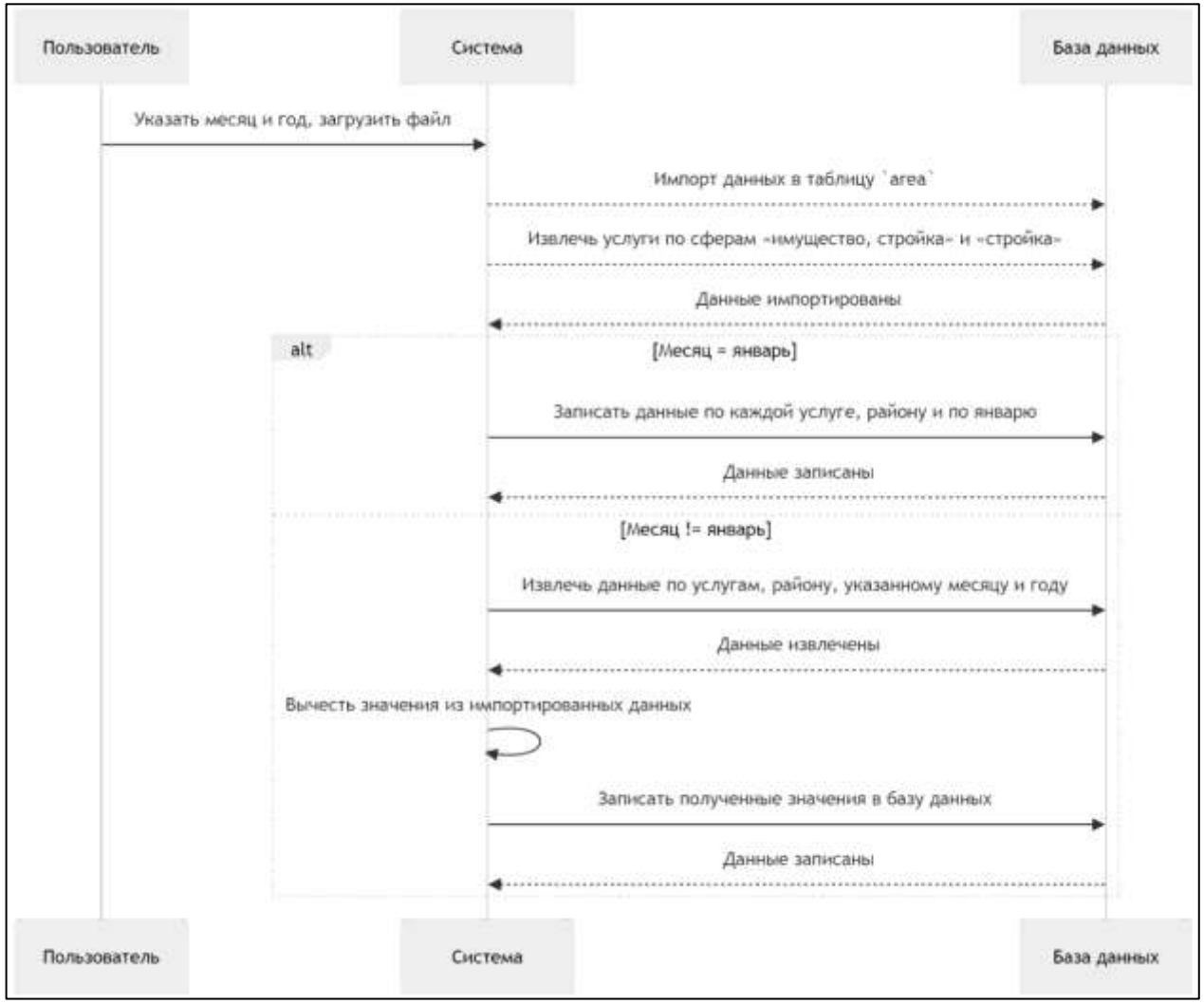

Рисунок 4 – Диаграмма последовательности взаимодействия компонента и базы данных

Приложение М. Макет интерфейса для компонента обработки файлов с данными о событиях и компонента обработки файлов с данными об обращениях

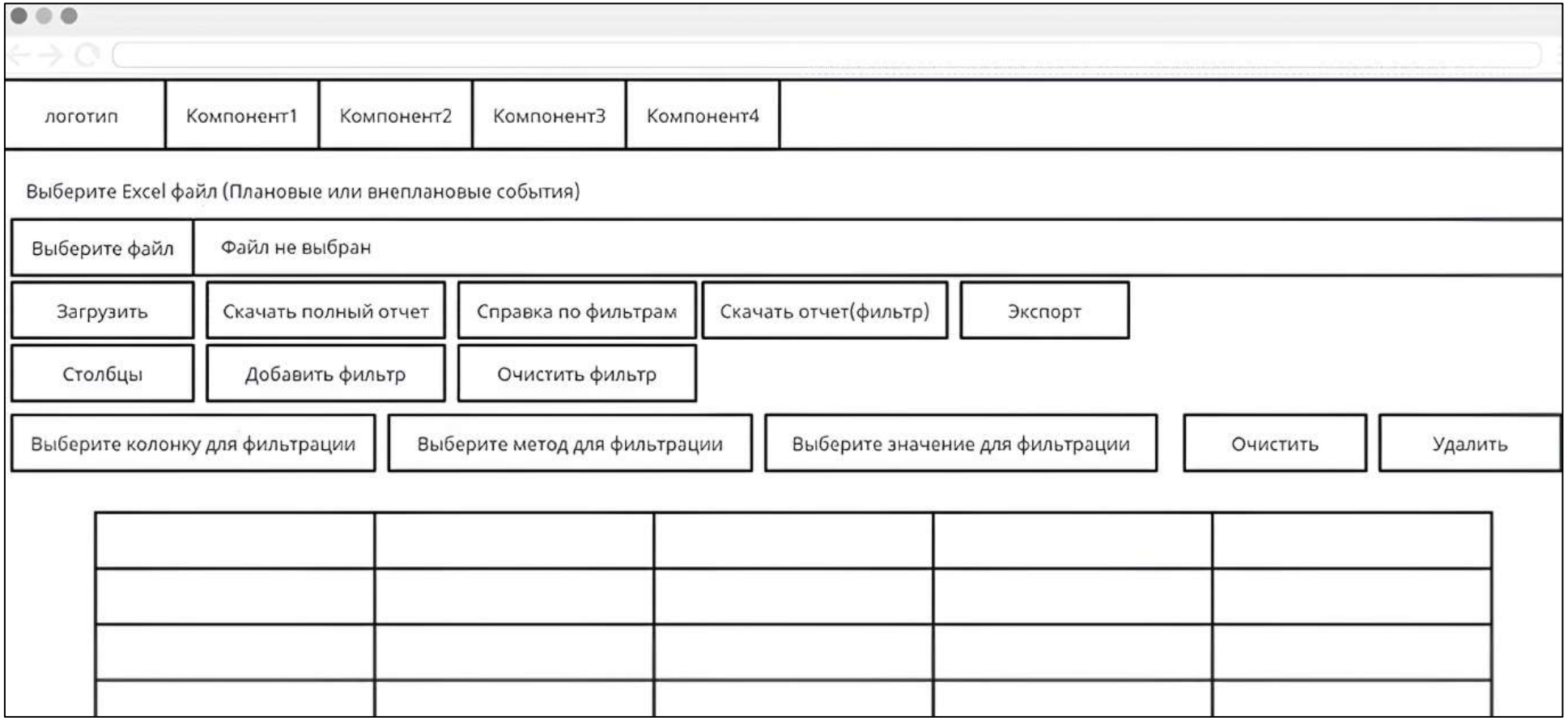

Рисунок 5 – Макет интерфейса