

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»**  
( **Н И У « Б е л Г У »** )

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ  
КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

**ПОДСИСТЕМА АВТОМАТИЗИРОВАННОГО ФОРМИРОВАНИЯ  
ОТЧЁТНОСТИ НА ОСНОВЕ КОНСТРУКТОРА ЗАПРОСОВ**

Выпускная квалификационная работа  
обучающегося по направлению подготовки  
09.03.02 Информационные системы и технологии  
очной формы обучения, группы 12001509  
Кифак Викторьен Яника

Научный руководитель  
ст. преподаватель  
Федоров В. И.

БЕЛГОРОД 2019

## РЕФЕРАТ

Подсистема автоматизированного формирования отчётности на основе конструктора запросов. Кифак Викторьен Яник, выпускная квалификационная работа бакалавра Белгород, Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), количество страниц 53, включая приложения 58, количество рисунков 48, количество использованных источников 20.

**КЛЮЧЕВЫЕ СЛОВА:** автоматизация, информационная система, база данных, SQL-запросы, конструктор запросов, отчёты, приложение.

**ОБЪЕКТ ИССЛЕДОВАНИЯ:** процессы создания, хранения, обработки и формирования отчётов с помощью конструктора запросов.

**ПРЕДМЕТ ИССЛЕДОВАНИЯ:** методы генерации SQL-запросы на основе графического представления таблицы базы данных.

**ЦЕЛЬ РАБОТЫ:** совершенствование процесса поиска информации и её наглядного представления.

**ЗАДАЧИ ИССЛЕДОВАНИЯ:** анализ предметной области; анализ подходов к формированию отчётов; проектирование базы данных и диаграммы UML; разработка конструктора запросов; разработка подсистемы для генерации отчёта в разных видах; реализация программы, тестирование разработанной подсистемы формирования отчётности на основе конструктора запросов.

**ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ:** в результате работы была спроектирована и реализована подсистема автоматизированного формирования отчётности на основе конструктора запросов.

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	4
1 Анализ предметной области .....	6
1.1 Обзор существующих подходов для решения задачи.....	7
1.2 Метод для решения задачи.....	10
2 Проектирование подсистемы формирования отчётности.....	12
2.1 Определение функций систем .....	12
2.2 Моделирование потоков данных.....	13
2.3 Разработка диаграмм UML .....	18
2.4 Разработка алгоритма работы конструктора запросов. ....	22
3 Программная реализация подсистемы автоматизированного формирования отчётности на основе конструктора запросов.....	24
3.1 Выбор инструментов для разработки автоматизированной системы ....	24
3.2 Разработка базы данных.....	27
3.3 Тестирование разработанной автоматизированной системы.....	32
3.2.1 Пользовательский раздел.....	33
3.2.2 Административный раздел .....	41
ЗАКЛЮЧЕНИЕ .....	51
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	52
ПРИЛОЖЕНИЕ .....	54

## ВВЕДЕНИЕ

Информационная система – это совокупность программных средств предназначена для обработки, хранения, поиска и эксплуатации информации, и соответствующие ресурсы. В современных информационных системах стоит вопрос о быстром поиске информации и её получения в настраиваемом и удобном виде для пользователя [2].

Актуальность выбранной темы обусловлена тем, что на сегодняшний день информационные системы работают с большим количеством разной информации. При этом, зачастую, пользователям этих систем необходимо получение структурированной статистики и данных в наглядном виде. Если привести пример, любой сайт имеет в своей базе много данных и больше количество разделов в самом сайте. Для обычного пользователя, для того чтобы получить определённую информацию, необходимо посмотреть все разделы сайта. Такой метод поиска по сайту не гарантирует нахождение точной информации при этом на это уходит много времени.

Данная выпускная квалификационная работа посвящена изучению принципов создания информационных систем, разработка начинается с моделирования с помощью разных инструментов с целью описать все процессы работы, а также изучения принципов формирования отчётов, принципы работы с реляционными базами данных – база данных, построенная на реляционные модели данных.

Целью работы является совершенствование процесса поиска информации и её наглядного представления за счёт разработки подсистемы автоматизированного формирования отчётности на основе конструктора запросов.

В соответствии с поставленной целью необходимо в течение реализации этого проекта решить следующие задачи:

- проанализировать предметную область;

- проектировать базу данных, модели и диаграмму UML;
- разработать конструктор запросов;
- разработать подсистему для генерации отчёта в разных видах;
- реализовать программу;
- протестировать разработанную подсистему отчётности.

Предметом исследования работы является реализация автоматизированной подсистемы формирования отчётности на основе конструктора запросов. Объект исследования – полученная информационная система с готовым конструктором запросов.

Выпускная квалификационная работа состоит из 51 страницы, включая приложения 53 страниц, 50 рисунков, 20 использованных источников.

В первом разделе производится постановка задачи, анализ предметной области, исследование существующих систем, теоретические основы, а именно основные понятия, необходимые для проектирования и реализация этого проекта.

Во втором разделе будут разработаны диаграммы потоков данных, диаграммы «сущность-связь» для представления данных в базе данных, диаграмма UML для программной реализации и алгоритмы работы конструктора запросов и формирования запроса.

В третьем разделе описана программная реализация подсистемы в основном на языках PHP и Javascript с использованием разных технологий и библиотек, проведено тестирование разработанного программного продукта, проведен анализ полученных результатов, которые укажут на степень функционирования и быстродействия программы.

## 1 Анализ предметной области

Реализация данной информационной системы требует достаточно знаний и пониманий о принципах формирования SQL-запроса из таблиц баз данных, моделировании процессов и управлении данным. В данном разделе будут рассматриваться все нужные понятия.

Запрос – это средство выбора необходимой информации из базы, данных которая находится на определённом сервере. Вопрос, сформированный по отношению к базе данных, и есть запрос. Применяются два типа запросов: по образцу (QBE – Query by example) и структурированный язык запросов (SQL – Structured Query Language). SQL – запросы – это запросы, которые составляются разработчиками информационных систем из последовательности SQL – инструкций. Эти инструкции задают, что надо сделать с входным набором данных для генерации выходного набора. Для удобной работы с базами данных, используются системы управления данными – это совокупность программных средств которая обеспечивает создание и использование баз данных. Примерами являются MySQL, Posture, SQLite. Во всех системах управления данными, запросы строятся на основе языка SQL или модифицированного языка.

Конструктор запросов (Query Builder) – это инструмент который предоставляет удобный, выразительный графический интерфейс для создания и выполнения запросов к базе данных. Он может использоваться для выполнения большинства типов операций и работает со всеми поддерживаемыми систем управления базами данных. В базах данных использует запросы для фильтрации или сортировки таблиц для отображения записей на локальном компьютере. Представления выполняют то же самое. Если база данных расположена на сервере, который поддерживает представления, можно использовать их для фильтрации записей на сервере и уменьшения времени на обработке и отображении так как представление в

самой базе сохраняется. Для изменения данных в базе используются хранимые процедуры, которые представляют собой функции.

При существовании связей между именем поля в одной таблице и именем поля в другой таблице эти связи можно использовать в запросе. Если, к примеру, имеется электронная таблица для статей под номерами и электронная таблица для пользователей, в которую записываются все статьи, записанные пользователем с использованием соответствующих номеров статей, то существует связь между двумя полями данных "номер статьи". Теперь, чтобы создать запрос всех созданных пользователем статей, необходимо получить данные из двух электронных таблиц. Для этого требуется указать связь, существующую между данными этих электронных таблиц.

Результатом конструктора запросов является готовый SQL-запрос, который можно выполнить на базе данных и получить от этого определённые наборы записей. Следовательно, можно с помощью конструктора запросов сформировать динамические отчёты для любой системы и тогда ускорить процесс поиска информации в данной информационной системе. Задачей этого проекта является реализация графического конструктора запросов, а также реализация системы формирования отчётности на основе данного конструктора.

### 1.1 Обзор существующих подходов для решения задачи

До того, как приступить к реализации подсистемы формирования отчётности, необходимо рассматривать существующие системы и их функционалы. С развитием информационных технологий и увеличением мощностей компьютеров, информационные системы обладают большим количеством разной информации. Пользователям этих систем необходимо получить определённую информацию. Сегодня существуют системы, которые предоставляют информацию в виде отчётов являются: datapine, freshbook.

Независимо от того, какой размер имеет одна компания, анализ информации и их визуализация теперь стали проще и быстрее. Программное обеспечение компании datarpine предоставляет любым пользователем все мощь и гибкость, необходимые для наиболее профессиональной визуализации данных. Datarpine предоставляет широкий спектр возможностей обмена и презентации. Можно с Datarpine легко обмениваться панелями с помощью URL, создавать запланированные отчеты или напрямую общаться с руководством, партнерами и клиентами, используя режим презентации. Или сделайте еще один шаг и непосредственно вставьте наши информационные панели в собственное приложение. На рисунке 1.1 показан пример использования Datarpine [4].



Рисунок 1.1 – Информационная отчётность Datarpine

FreshBooks – это пакет программного обеспечения для бухгалтерского учета, разработанный и реализуемый 2ndsite. Продукты FreshBooks ориентированы в основном на малые предприятия и предлагают облачные бухгалтерские приложения, которые управляют и оплачивают счета, а также функции расчета заработной платы. На рисунке 1.2 показана информационная система отчётности FreshBooks [5].



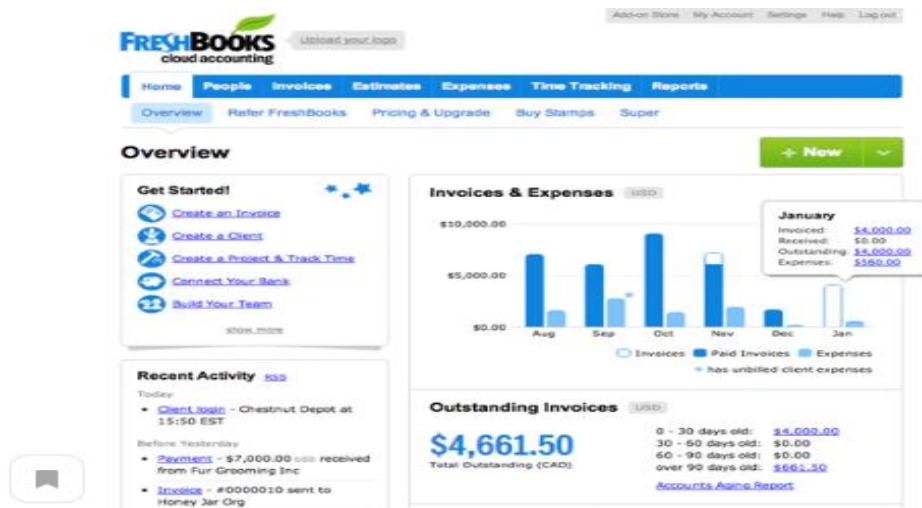


Рисунок 1.2 – информационная отчётность FreshBooks

После обзора и анализа функционалов существующих систем формирования отчетности, можно теперь делать сравнение с разрабатываемой подсистемой. В таблице 1.1 приведено сравнение нашей системы с другими системами.

Таблица 1.1 – Сравнения существующих информационных отчётов

	datapine	freshbook	XLReporting	Данная система
Динамические отчёты	–	–	–	+
Экспорт отчётов	+	+	+	+
Адаптивные отчёты	–	+	–	+
Пользовательские настройки	+	–	+	+
Специализированная система	+	+	+	+

Такие системы чаще всего предоставляют финансовую информации в виде графиков и документов, однако у них имеются недостатки: виды отчётов статические и заранее определены при создании системы. Они решают узкоспециализированные задачи такие как: бухгалтерские, научные, административные.

Поэтому становится актуальной задача разработки подсистемы формирования различной отчетности с возможностью создания динамических шаблонов отчетов всё опираясь на использования конструктора запросов, который делает возможно создание динамических отчетов.

## 1.2 Метод для решения задачи

Для реализации данной подсистемы, необходимо рассмотреть все теоретические аспекты. Графический конструктор запросов состоит из множеств таблиц базы данных на основе которых строится SQL-запроса по определённому алгоритму.

Для перечисления таблиц базы данных и их визуализации, будем использовать библиотеку mxGraph, написанную на JavaScript. MxGraph - это компонент JavaScript, который предоставляет функции, предназначенные для приложений, которые отображают интерактивные диаграммы и графики. Надо заметить, что под графиками подразумевают математические графики, не обязательно диаграммы (хотя некоторые диаграммы являются графиками). mxGraph предоставляет все обычно необходимые функции для рисования, взаимодействия и сопоставления контекста с диаграммой. mxGraph поставляется с несколькими примерами, которые помогают объяснить, как составляется базовое приложение, и демонстрирует отдельные функции библиотеки. На рисунке 1.3 показаны компоненты mxGraph и их взаимодействия [7].

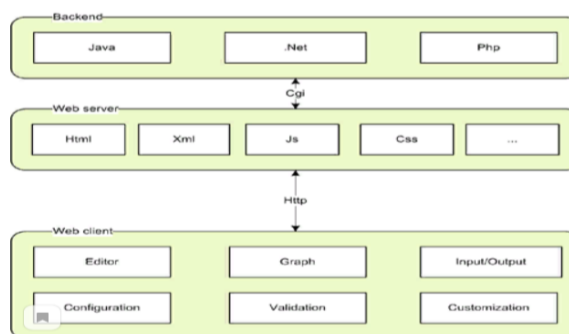


Рисунок 1.3 – компоненты mxGraph

CodeMirror – это компонент JavaScript, который предоставляет редактор кода в браузере. Он имеет богатый программный API и фокусируется на расширяемости. Это универсальный текстовый редактор, который реализован на JavaScript для браузера. Он специализируется на редактировании кода и поставляется с рядом языковых режимов и дополнений, которые реализуют более продвинутые функциональные возможности редактирования [8].

Богатый API-интерфейс и система тем CSS доступны для настройки CodeMirror в соответствии с вашим приложением и расширения его новыми функциями.

Chart.js – это популярный инструмент, который предназначен для создания графиков и диаграмм. Можно создавать адаптивные диаграммы любой сложности на основе HTML5 Canvas.

Данная библиотека позволяет без особого труда создавать графики и диаграммы любого типа, а также выстраивать данные на диапазоне времени и логарифмической шкале. Также в неё встроены средства работы с анимацией, что позволит эффектно видоизменять графики в зависимости от новых данных, а также экспериментировать с цветом.

Выводы по первому разделу.

Проведя всесторонний анализ систем формирования отчётности, можно однозначно установить и подтвердить актуальность и востребованность реализуемого проекта. В ходе проведенной работы были проведены:

- обзор существующих подходов для решения задачи формирования динамических отчётов;
- анализ материалов для разработки;
- анализ текущих средств решения поставленной задачи.

## 2 Проектирование подсистемы формирования отчётности

### 2.1 Определение функций систем

Под автоматизированной подсистемой формирования отчётности на основе конструктора запросов, подразумевается система, которая позволяет создать динамические отчёты на основе графического редактора запроса, который из выбранных таблиц генерирует SQL-запрос для последующего использования определённым пользователем. Основанием для разработки является задание в рамках выпускной квалификационной работы. Данная автоматизированная подсистема предназначена для решения следующих задач:

- предоставления информации в виде отчётов пользователям главной системы;
- упрощение процесса формирования запроса администратором системы через конструктор запросов;
- хранение информации о желаемом отчёте пользователю;
- хранение данных о сеансах фильмов.

Система должна обеспечивать следующие функции:

Авторизацию пользователей системы:

- номер пользователя;
- ФИО;
- электронная почта;
- пароль.

Ввод и хранения информации о желаемом отчёте в виде переписки между пользователем и администратором.

- ФИО отправителя;
- ФИО получателя;
- сообщения.

Ввод, вывод, редактирование, хранение информации о задачах которые стоят перед администратором для создания запроса и его сохранения

- ФИО пользователя;
- описание задачи;
- статус задачи.

Система должна:

- проводить контроль вводимой информации;
- блокировать некорректные действия пользователя при работе с системой;
- обеспечивать защиту данных;
- обеспечивать целостность данных.

Система должна работать под управлением ОС семейства Win32.

## 2.2 Моделирование потоков данных

При построении любой системы, необходимо описывать все функционалы в удобном виде так, чтобы было понятно обычному человеку. Для этого используются нотации IDEF0, DFD и IDF3. Методология функционального моделирования IDEF0 – это технология описания системы в целом как множества взаимозависимых действий, или функций [6].

Наиболее часто IDEF0 применяется как технология исследования и проектирования систем на логическом уровне. По этой причине он, как правило, используется на ранних этапах разработки проекта. Результаты IDEF0 анализа могут применяться при проведении проектирования с использованием моделей IDEF3 и диаграмм потоков данных. IDEF0 сочетает в себе небольшую по объему графическую нотацию (блоки и стрелки). Стрелки могут представлять собой людей, места, вещи, идеи или события. Виды стрелок ниже перечислены.

Стрелки входа представляет собой сырье, или информацию, потребляемую или преобразуемую функциональным блоком для производства выхода.

Стрелки управления. Стрелки управления отвечают за регулирование того, как и когда выполняется функциональный блок.

Стрелки выхода – это продукция или информация, получаемая в результате работы функционального блока.

Стрелки механизма исполнения. Механизмы являются ресурсом, который непосредственно исполняет моделируемое действие.

Теперь все понятия о моделировании поток данных известны, можно приступить к моделированию нашей системы. Контекстная диаграмма (диаграмма А-0) разрабатываемой подсистемы приведена на рисунке 2.1.

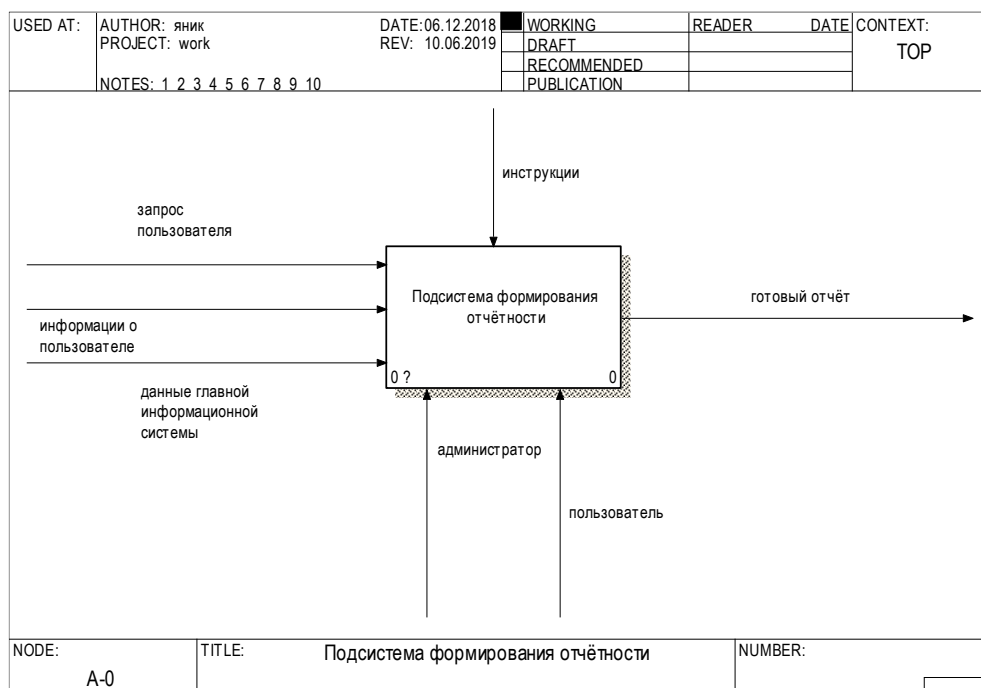


Рисунок 2.1 – Контекстная диаграмма подсистемы

Входными информациями системы являются:

- запрос пользователя, то есть описание того, что ему нужно. Это может быть переписки с администратором или сотрудником главной системы;
- информации о пользователе необходимые для того, чтобы его авторизовать.

В центре находится основной блок, который предназначен для обработки информации и выдачи на выходе результатов. Выходной информацией является отчёты, которые пользователь может визуализировать и скачать в разных форматах данных.

Детализация контекстной IDEF0-диаграммы системы представлена диаграммой A0 на рисунке 2.2. Можно декомпозировать контекстную диаграмму на следующие функциональные блоки:

- обработка запроса пользователя, то есть анализ информации об отчёте. После получения этой информации необходимо создать задачу, которая соответствует требованиям пользователя для дальнейшего выполнения;
- построение шаблона отчёта. На основе созданной задачи с помощью информации, полученной при обработке запроса, администратор создаёт шаблон отчёта, через конструктор запросов;
- настройка отчётов. Так как система предоставляет разные виды отчёта, необходимо их настроить. Настроить отчёт можно либо программно, либо со стороны пользователя;
- подготовка отчёта.

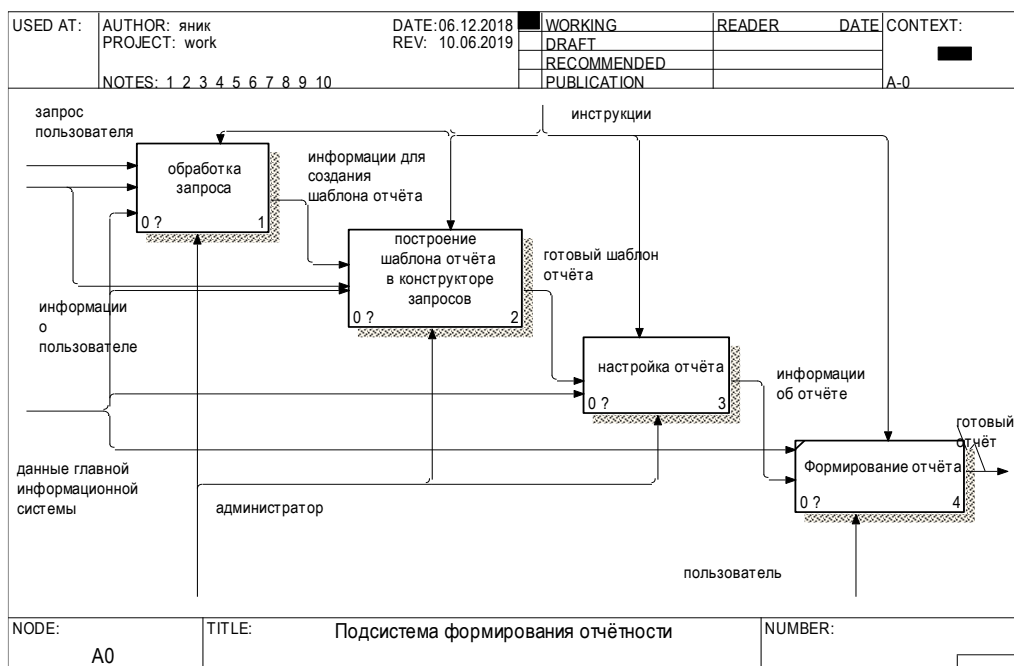


Рисунок 2.2 – Детализация контекстной диаграммы

Для того чтобы понять что происходит в блоке «Обработка запроса», необходимо его декомпозировать используя нотацию IDF3. IDF3 — способ описания процессов, основной целью которого является обеспечение

структурированного метода, используя который эксперт в предметной области может описать положение вещей как упорядоченную последовательность событий с одновременным описанием объектов, имеющих непосредственное отношение к процессу. На рисунке 2.3 представлен декомпозиция этого блока с помощью нотации IDF3.

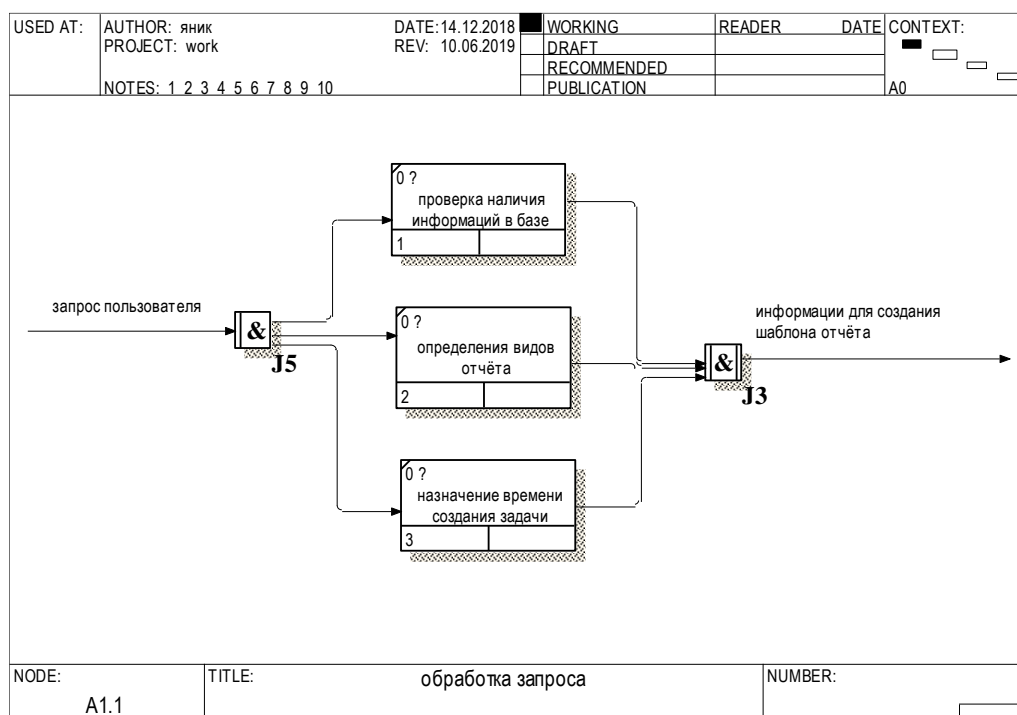


Рисунок 2.3 – Детализация блока «Обработка запроса»

Он содержит следующие функциональные блоки:

- проверка наличия информации в базе;
- определение видов отчёта;
- назначение времени выдачи отчёта. Отчёт – Это составленные по определённой форме сведения, данные о деятельности организации, компании за определённый прошедший период. В этом процессе определяется время, в которое необходимо давать пользователю доступ к сформированным отчётам.



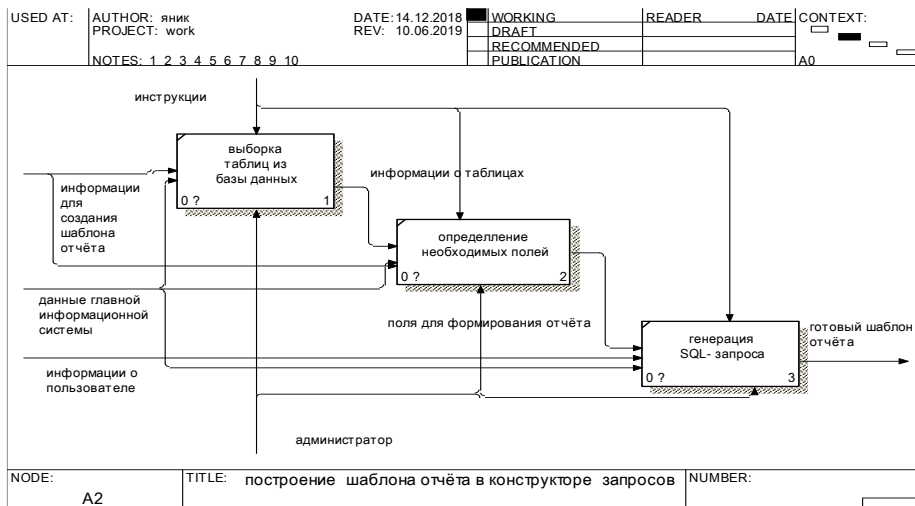


Рисунок 2.4 – Детализация блока «Построение шаблона отчёта»

Рисунок 2.4 содержит следующие функциональные блоки:

- выборка таблиц из базы данных;
- определение необходимых полей;
- генерация SQL-запроса.

Детализация процесса настройки отчета приведена на рисунке 2.5.

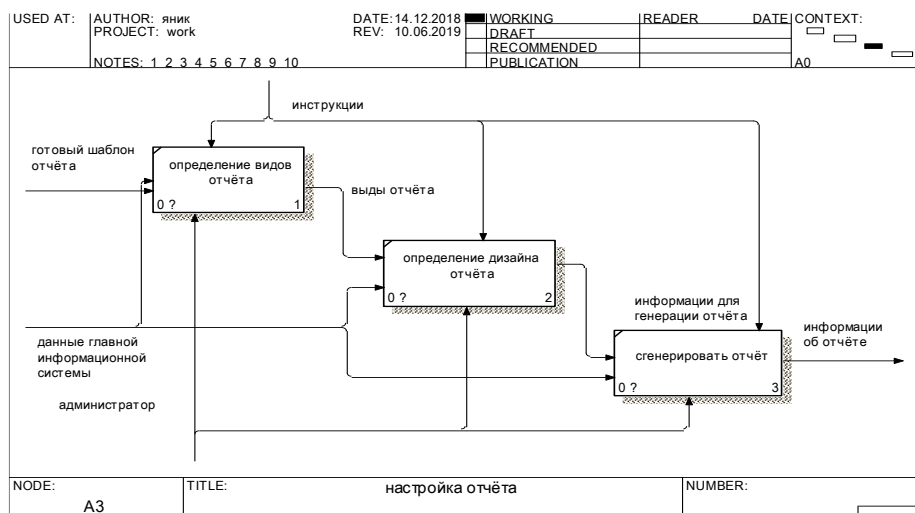


Рисунок 2.5 – Детализация блока «Настройка отчёта»

На этом рисунке представлены следующие функциональные блоки:

- Определение видов отчёта. Отчёт может быть представлен в PDF-документ, в Excel-документ или в виде диаграмм;

- Определение дизайна отчёта;
- Сгенерировать отчёт. С помощью полей запроса сохранённых в базе данных генерируют отчёт.

### 2.3 Разработка диаграмм UML

На этапе проектирования, необходимо создать диаграмму классов на основе которого в дальнейшем будет строится проект. Диаграмма классов – это структурная диаграмма языка моделирования, которая демонстрирует общую структуру иерархии классов разрабатываемой системы, их связи, атрибутов с их типом, методов, интерфейсов и взаимосвязей между ними для того чтобы иметь общие понятия о проекте. Она применяется не только для документирования и визуализации, но также для конструирования [10]. В данной диаграмме, описываются основные классы проекта. На рисунке 2.6 показана диаграмма классов проекта.

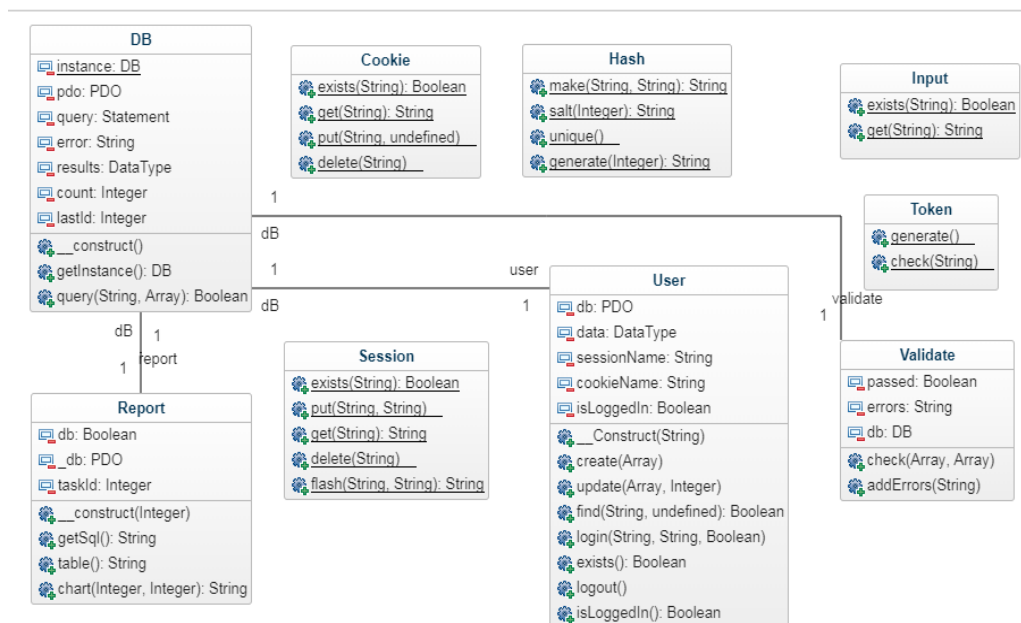


Рисунок 2.6 – Диаграмма классов

Как показана на рисунке, выделяют 9 основных классов, каждый из которых играет важную роль в реализации проекта.

Класс DB предназначен для подключения к базе данных. В этом классе используется шаблон проектирования «одиночка». Шаблон проектирования – это повторяемая архитектурная конструкция, которая представляет собой решение проблемы проектирования в рамках некоторого часто возникающего контекста. А паттерн одиночки – это порождающий шаблон, который гарантирует, что в однопроцессном приложении будет единственный экземпляр некоторого класса, и является глобальной точкой доступа к этому экземпляру. Класс DB содержит следующие атрибуты:

- статический атрибут `instance`, который является ссылкой на сам объект класса. Он необходим для реализации паттерна одиночки;
- атрибут `pdo` – объект класса PDO, предназначен для подключения к базе данных;
- атрибут `query` объект типа PDOStatement для выполнения запроса в базе данных;
- атрибут – список ошибок при выполнении запроса;
- атрибут `results`, который содержит результаты выполнения запроса;
- атрибут `count` предназначен для сохранения количества записей после выполнения SQL-запроса.

Основные методы этого класса, следующие:

- конструктор для создания экземпляра класса;
- метод `getInstance`, который возвращает экземпляр объекта класса DB;
- метод `query` выполняет запрос. Как аргумент он принимает SQL-запрос и список параметров.

Класс User который имплементирует все функционалы пользователя в системе. Атрибуты этого класса, следующие:

- атрибут `db` – экземпляр класса PDO предназначен для подключения к базе данных;
- атрибут `data` содержит информацию о текущем пользователе. Этой информацией являются его `id`, логин, пароль, электронная почта;

- атрибут `sessionName` содержит название сессии при авторизации пользователя;
- атрибут `cookieName` предназначен для сохранения хэша авторизованного пользователя, чтобы сохранить информацию даже при закрытии браузера;
- атрибут `isLoggedIn` – булева переменная, которая указывает на авторизованность пользователя.

Основные методы этого класса, следующие:

- конструктор для создания экземпляра класса. Он как аргумент принимает идентификатор пользователя (`Id` или логин);
- метод `create` добавляет нового пользователя в базу данных. Аргументы этого метода являются массивом данных;
- метод `update` для изменения данных пользователя. Он принимает массив данных и идентификатор пользователя;
- метод `find` для нахождения пользователя в базе данных. Он возвращает истинно, если пользователь есть и ложь в противном случае;
- метод `login` предназначен для авторизации пользователя. Он принимает массив данных и булеву переменную;
- метод `logout` для выхода авторизованного пользователя из системы.

Класс `Report` который имплементирует все функции формирования отчётов. Он содержит следующие атрибуты:

- атрибут `_db` для подключения к базе данных проектируемой подсистемы;
- атрибут `db` для подключения к базе данных главной системы из которой берутся все данные для создания отчётов;
- атрибут `taskId` идентифицирует шаблон отчёта в базе данных.

Методы этого класса, следующие:

- конструктор для создания экземпляра класса;

- метод `getSql`. При создании шаблона отчёта, все поля запроса сохраняются в таблицах «Поля» и «Связи». При формировании отчёта необходимо сначала сгенерировать SQL – код и затем его выполнить. Данный метод выполняет эту роль генерации SQL – кода;

- метод `table` возвращает данные в виде таблицы. Возвращённая таблица используется дальше для создания либо PDF-документа, либо Excel-документа;

- метод `chart` принимает два аргумента – имена полей. Данный метод возвращает массив данных для создания, диаграммы.

Класс `Validate` предназначен для обработки данных формы. Он содержит следующие атрибуты:

- атрибут `db` для подключения к базе данных и проверки верности введённых данных в форме;

- атрибут `passed` – булева переменная, которая указывает статус валидации данных;

- атрибут `errors` содержит список ошибок при выполнении обработки данных.

Этот класс содержит следующие методы:

- метод `check` выполняет обработку данных формы;

- метод `addError` добавляет ошибки в список.

Класс `Hash` содержит статические методы. Этот класс в основном предназначен для генерации ключей. Методы этого класса, следующие:

- метод `make` предназначен для хеширования строк, то есть создания отпечатков сообщения произвольной длины и последующей проверки их подлинности. Данный метод используется для проверки целостности информации и хранения хешей паролей;

- метод `salt` генерирует строку определённой длины;

- метод `unique` генерирует и возвращает идентификатор;

## 2.4 Разработка алгоритма работы конструктора запросов.

Основной элемент этого проекта является конструктор запросов, на основе которого формируются отчёты. Необходимо в этом разделе рассмотреть алгоритм работы этого конструктора. Подходящий вариант описания системы являются блок-схемы. На рисунке 2.7 представлены блок схемы, которые описывает его функционирование.

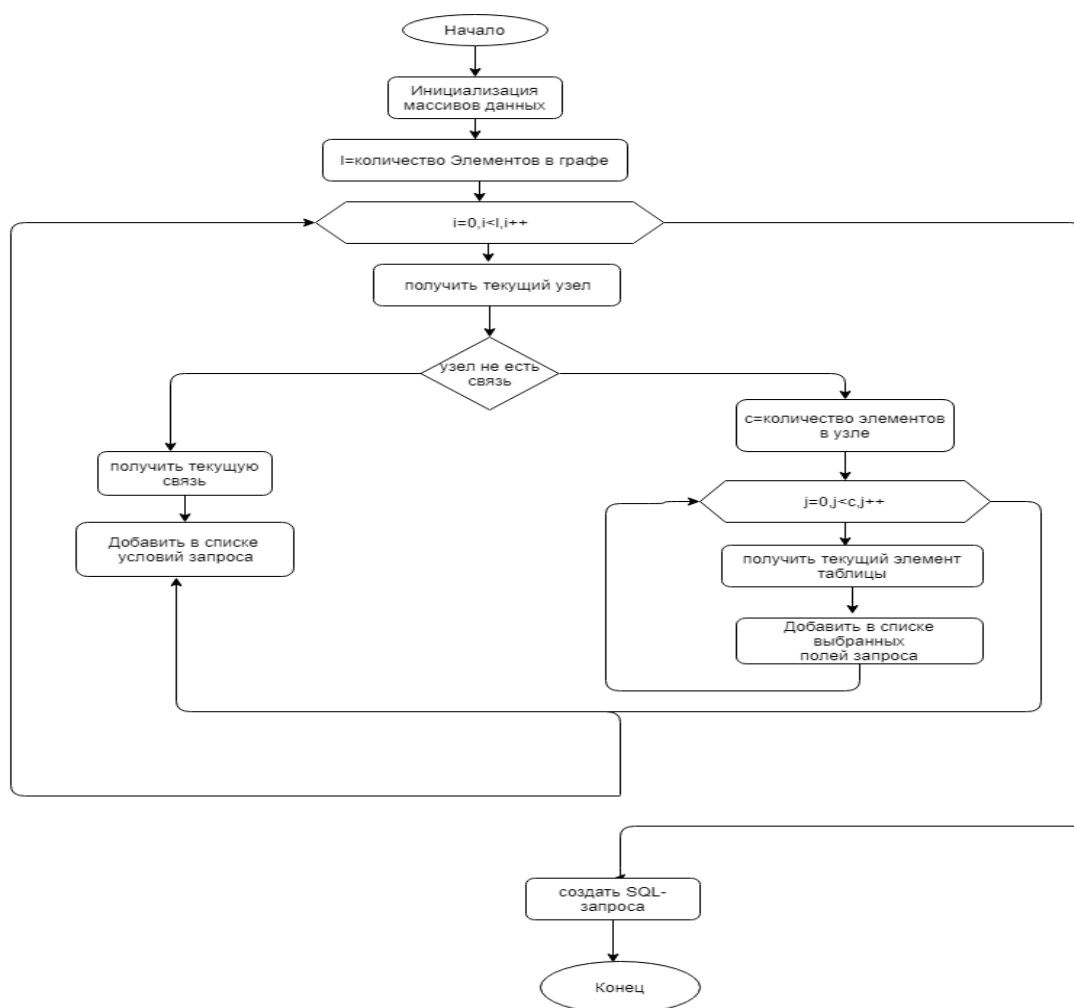


Рисунок 2.7 –Блок схемы конструктора

В программе, графический редактор рассматривается как граф. Граф состоит из ребер и вершин. Вершины являются элемент которые могут в своём очереди содержать элементы расположены в их контейнер по определённой

системе компоновки. В данном случае элемент графа есть таблица базы данных. Одна таблица содержит элементы размещены по принципу работы стека. Элементы таблицы являются полями таблицы в базе, а ребра являются отношениями между таблицами. Для создания отчёта, необходимо сформировать запрос выборки к базе данных – Для этого в SQL существует оператор SELECT. При формировании запроса, важно отмечать для каждого поля следующие пункты:

- название таблицы из которых делается запрос;
- отображается ли это поле в запросе;
- математическая операция, которую необходимо выполнить (<=,>);
- значение поля;
- переименования поля;
- LIKE – операция;
- группировка и сортировка по данному полю;
- агрегатная функция.

Рассмотрены важные параметры для создания запроса, теперь необходимо описывать алгоритм работы графического редактора. Сначала надо инициализировать массивы названия полей, названия таблиц, условия, переименованные поля, поля группировки и сортировки. Затем рассмотреть все объекты графа. Если текущий элемент является таблицей, то добавить её название в массив названия таблиц и рассмотреть все элементы под ним. Если поле выбрано, добавить в массив полей запроса. Если выбрана операция, добавить в списке условий. Если текущий элемент графа является ребром, в массив добавить информацию об отношениях.

Выводы по первому второму разделу.

Таким образом, в ходе проведенной работы были проведены:

- Анализ функций системы;
- проектирование диаграммы потоков и диаграммы UML;
- Разработка алгоритма работы конструктора запросов.

### 3 Программная реализация подсистемы автоматизированного формирования отчётности на основе конструктора запросов

#### 3.1 Выбор инструментов для разработки автоматизированной системы

Перед разработкой автоматизированной системы был проведен анализ требований к данной системе и на основе данного анализа произведен выбор инструментов разработки.

Анализ показал, что автоматизированная подсистема должна работать с операционной системой. Иметь возможность быстрого развертывания на сервере. В связи с этим разработка велась с использованием следующих инструментальных средств.

В качестве языка программирования были выбраны следующие технологии. PHP как самый популярный для один из самых популярных языков разработки веб-приложения. Это тоже современный объектно-ориентированный и тип безопасный язык программирования, относящийся к широко известному семейству интерпретированных языков программирования. Язык поддерживает следующие функции: обработка исключений – дает возможность избегать, отлавливать и обрабатывать ошибки. Javascript с использованием библиотек JQuery, mxGraph и язык разметки HTML.

В качестве среды разработки была выбрана среда Visual Studio 2013 – среда разработки приложений для ОС Windows, как консольных, так и с графическим интерфейсом. Visual Studio 2013 Обладает рядом преимуществ таких как:

- одновременное редактирование;
- высокая степень настраиваемой;
- автозаполнение;

Пример интерфейса Sublime Text представлен на рисунке 3.1.





Рисунок 3.1 – Пример интерфейса среды разработки Sublime Text

В качестве СУБД была выбрана Mysql – это реляционная система управления базами данных. Поддержка большого количества таблиц обеспечивает гибкость системы управления базами данных Mysql. Для развертывания базы данных на сервере используется инструмент PhpMyAdmin – это веб приложения написанное на языке PHP и представляет собой веб-интерфейс для администрирования базы данных. PhpMyAdmin позволяет через браузер осуществлять администрирование сервера MySQL, запускать команды SQL и просматривать содержимое таблиц и баз данных. На рисунке 3.2 представлена база данных этого проекта.

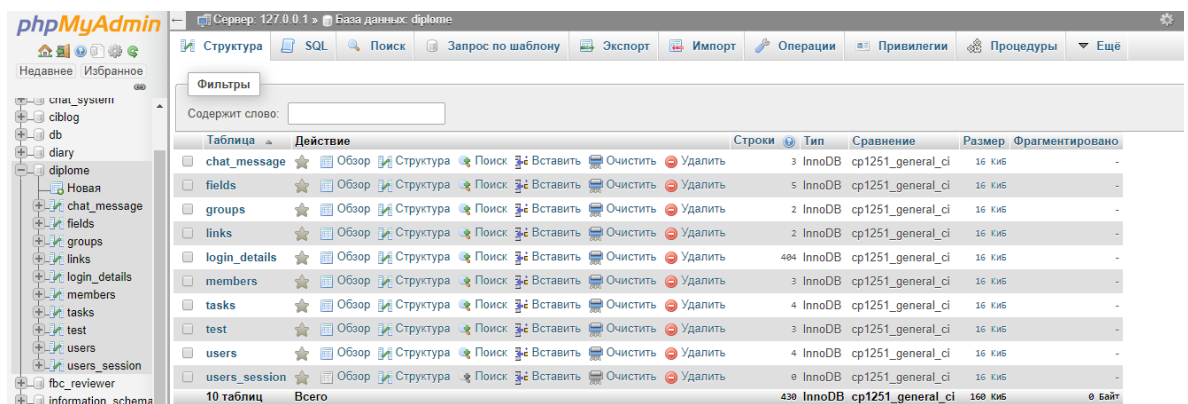


Рисунок 3.2 – Таблицы базы данных в PhpMyAdmin

Соединение с базой данных устанавливается автоматически при создании объекта PDO от его базового класса. Не имеет значения, какой драйвер использовать, вы всегда используете имя базового класса. Конструктор класса принимает аргументы для задания источника данных (DSN), а также необязательные имя пользователя и пароль. Для подключения, необходимо указывать следующие параметры:

- тип базы данных (Mysql,Postgre);
- хост;
- имя базы;
- имя пользователя;
- пароль.

Листинг 1 – скрипт для подключения к базе данных

```
$host = 'db4free.net';  
$db = 'projet';  
$user = 'yanick';  
$pass = 'yanick123';  
$charset = 'utf8';  
$dsn = "mysql:host=$host;dbname=$db;charset=$charset";  
$pdo = new PDO($dsn, $user, $pass, $opt);
```

Перед тем как приступить к разработке данной информационной системы, необходимо дать представление о иерархии проекта, то есть папки, файлы и ресурсы, которые составляют этот проект. На рисунке 3.3 представлена структура проекта. Структура проекта – это организация связей и отношений между ее элементами. Блок – работа, совокупность работ, подпроект относительно независимая часть проекта;

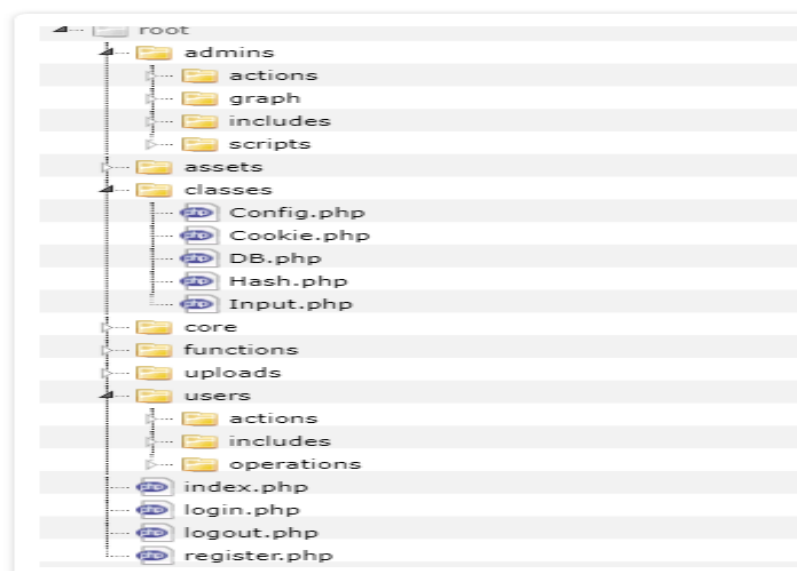


Рисунок 3.3 – Структура проекта

Проект содержит 3 основных папки.

- admins содержит все файлы для административного раздела сайта;
- users для пользовательского раздела сайта;
- classes содержит файлы всех классов необходимых для реализации проекта.

### 3.2 Разработка базы данных

Для разработки базы данных была построена модель ее структуры при помощи нотации сущность-связь. Сущность - связь — это нотация, позволяющая описывать концептуальные схемы предметной области. Нотация используется при высокоуровневом проектировании баз данных. С её помощью можно выделить ключевые сущности и обозначить связи, которые могут устанавливаться между этими сущностями. Для построения концептуальной модели использован инструмент AllFusion ERwin Data Modeler. На рисунке 3.4 продемонстрирована модель сущность-связь разрабатываемой БД.

Сущность «Пользователи» описывает характеристики пользователи-данные которые идентифицирует определённого человека для авторизации. Типы полей представлены в таблице 3.1.

Таблица 3.1 – Типы данных полей сущности «Пользователи»

Название поля	Тип данных
id Пользователи	INTEGER
Ф.и.о	VARCHAR(255)
картинка	VARCHAR(255)
Электронная почта	VARCHAR(255)
пароль	VARCHAR(255)
хэш	VARCHAR(255)

Сущность «Группы» описывает группы пользователей, которые имеют доступ к определённым отчётам. Типы полей представлены в таблице 3.2.

Таблица 3.2 – Типы данных полей сущности «Группы»

Название поля	Тип данных
id группы	INTEGER
название	VARCHAR(255)

Сущность «Задача» описывает работу, которую должен выполнить администратор – основные информации для создания отчёта, с помощью конструктора. Типы полей представлены в таблице 3.3.

Таблица 3.3 – Типы данных полей сущности «Задача»

Название поля	Тип данных
id задачи	INTEGER
описание	TEXT
заголовок	VARCHAR(255)
дата создания	DATETIME
статус	INTEGER
id группы	INTEGER
id пользователя	INTEGER

Сущность «Поля» описывает определённое поле отчёта – содержит информации для формирования SQL-запроса. Id поля является ключевым полем. Типы полей представлены в таблице 3.4.

Таблица 3.4 – Типы данных полей сущности «Поля»

Название поля	Тип данных
id поля	INTEGER
Название таблица	VARCHAR(255)
название поля	VARCHAR(255)
Переименованное поле	VARCHAR(255)
Как поле	VARCHAR(255)
значение	VARCHAR(255)
операция	VARCHAR(255)
Поле сортировки	VARCHAR(255)
Поле группировки	VARCHAR(255)
функция	VARCHAR(255)
id задачи	INTEGER

Сущность «Связи» описывает связь между таблицами. При построении запроса можно установить между таблицами связь и сохранить информацию об этом. Типы полей представлены в таблице 3.5.

Таблица 3.5 – Типы данных полей сущности «Связи»

Название поля	Тип данных
id связи	INTEGER
Таблица 1	VARCHAR(255)
Таблица 2	VARCHAR(255)
Поле 1	VARCHAR(255)
Поле 2	VARCHAR(255)
id задачи	INTEGER

Сущность «Чат сообщения» содержит информации о переписках между администратором и определённым пользователем. Типы полей представлены в таблице 3.6.

Таблица 3.6 – Типы данных полей сущности «Чат сообщения»

Название поля	Тип данных
id чат	INTEGER
id отправителя	INTEGER
id получателя	INTEGER
сообщение	VARCHAR(255)
статус	INTEGER

Сущность «Члены» содержит информации о пользователях определённой группы. Типы полей представлены в таблице 3.7.

Таблица 3.7 – Типы данных полей сущности «Члены»

Название поля	Тип данных
id члена	INTEGER
id пользователя	INTEGER
id группы	INTEGER
сообщение	VARCHAR(255)
статус	INTEGER

Сущность «Сеанс пользователя» описывает даты входа определённого пользователя в систему для того, чтобы его запомнить. Типы полей представлены в таблице 3.8.

Таблица 3.8 – Типы данных полей сущности «Группы».

Название поля	Тип данных
id группы	INTEGER
название	VARCHAR(255)

Связь между сущностями «Пользователь» и «Задача». Поскольку создаётся задача для определённого пользователя, то тип данной связи «один-ко-многим».

Связь между сущностями «Задача» и «Поля». Для сохранения полей запросов из конструктора, используется таблица «Поля». У одной задачи есть несколько полей запроса.

Связь между сущностями «Задача» и «Связи». Для сохранения информации о связях между таблицами конструктора используется таблица «Связи». И связи тоже входят в состав задачи.

Связь между сущностями «Группы», «Пользователь» и «Члены». В одной группе входят несколько члены и каждый член является пользователем. На рисунке 3.4 представлена логическая модель, а на рисунке 3.5 представлена физическая структура базы данных – модель базы данных, выраженная в терминах языка описания данных конкретной базы, с латинскими буквами.

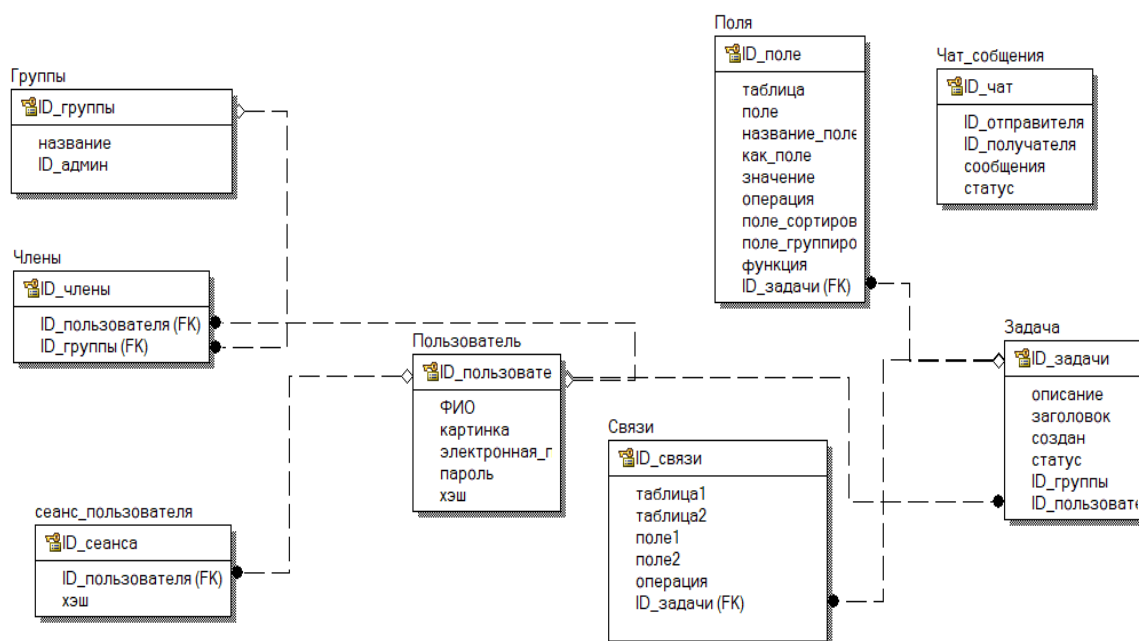


Рисунок 3.4 – Модель структуры базы данных

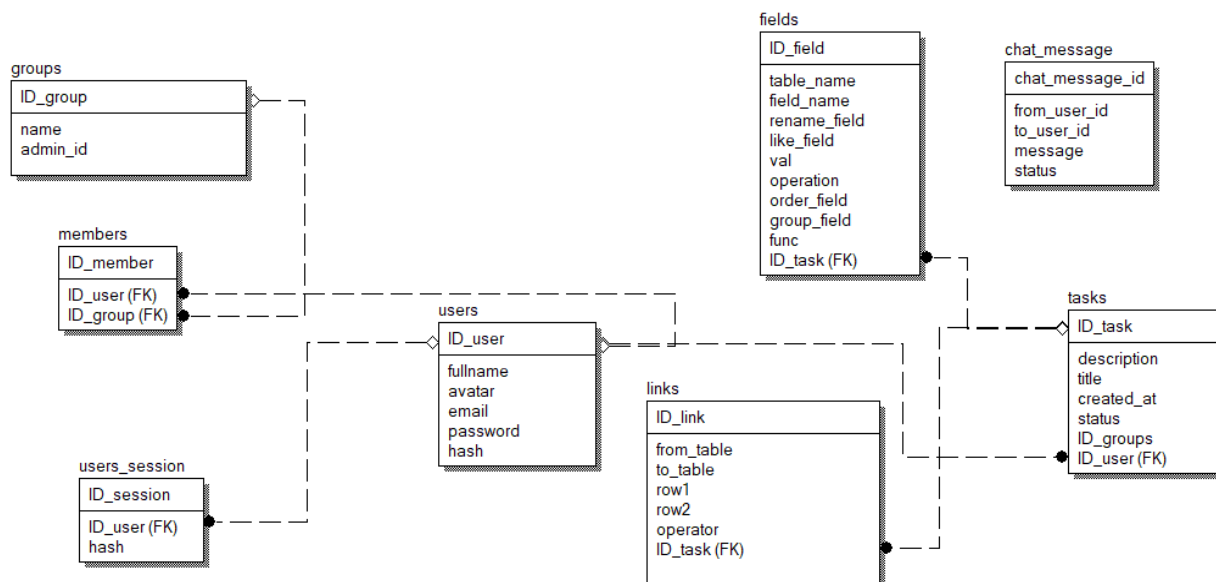


Рисунок 3.5 – Физическая модель

Физическая модель базы данных содержит все детали, необходимые для создания базы. В приложении приведён скрип конструктора запросов и основной системы.

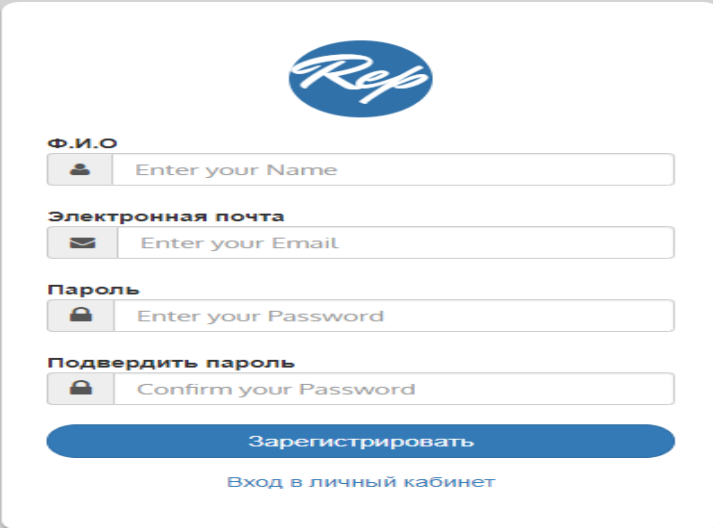
### 3.3 Тестирование разработанной автоматизированной системы

Было проведено тестирование разработанной подсистемы, автоматизированной формирования отчётности на основе конструктора запросов. При запуске автоматизированной системы пользователю предлагается авторизоваться. Экранная форма авторизации представлена на рисунке 3.6. Пока пользователь не пройдет авторизацию ему недоступны никакие другие действия.

Рисунок 3.6 – Форма авторизации



В случае если пользователь ещё не зарегистрирован в системы ,он должен пройти регистрацию через форму которая показана на рисунке 3.7.



The image shows a registration form for a system named 'Rep'. At the top center is the 'Rep' logo, which consists of the word 'Rep' in a stylized blue font inside a blue circle. Below the logo are four input fields, each with a small icon to its left and a label above it. The first field is labeled 'Ф.И.О' and has a person icon; the text inside is 'Enter your Name'. The second field is labeled 'Электронная почта' and has an envelope icon; the text inside is 'Enter your Email'. The third field is labeled 'Пароль' and has a lock icon; the text inside is 'Enter your Password'. The fourth field is labeled 'Подтвердить пароль' and has a lock icon; the text inside is 'Confirm your Password'. Below these fields are two buttons. The first is a large blue button with the text 'Зарегистрировать'. The second is a smaller, lighter blue button with the text 'Вход в личный кабинет'.

Рисунок 3.7 – Форма регистрации

В системе есть две роли для пользователей. Пользователь – ему доступны все виды отчёты, он может создать группы пользователей. Администратор – он получает запрос пользователя используя внедрённую систему чата, создаёт задачу с всеми необходимыми информациями и её выполняет, используя конструктор запросов.

### 3.2.1 Пользовательский раздел

При входе пользователя в систему, отображаются все задачи, созданные администратором для него и их статус, для того чтобы отслеживать процесс обработки. На рисунке 3.8 показан список заявок, то есть обрабатывающиеся отчёты администратором для авторизованного пользователя. Для каждой заявки указан статус, который даёт информации о процессе обработки. Если процесс обработки завершён, тогда отображается ссылка на сформированный отчёт.

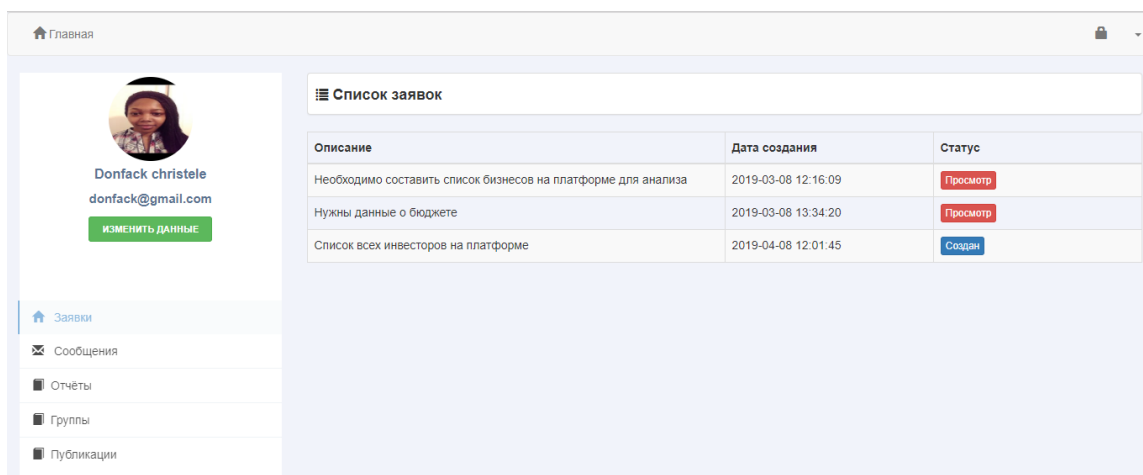


Рисунок 3.8 – Список заявок пользователя

Для того чтобы связываться с администратором, необходимо перейти на страницу чат. На рисунке 3.9 показана внедрённая система чата авторизованного пользователя.

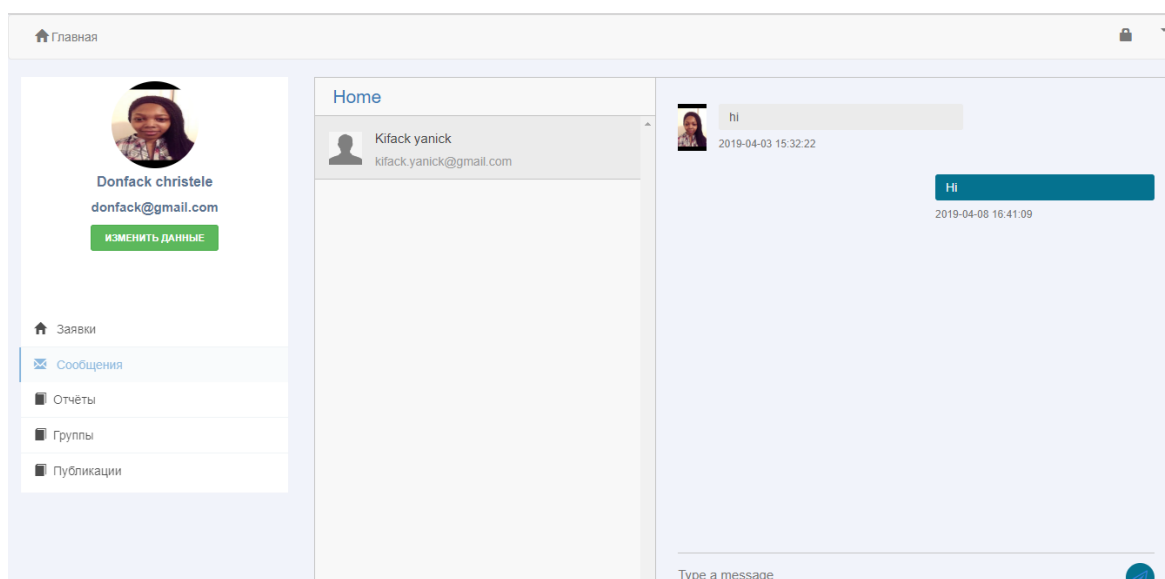


Рисунок 3.9 – Система чата

Для того чтобы посмотреть все готовые отчёты созданы администратором, необходимо переходить в раздел «Отчёты». На рисунке 3.10 показан список всех отчётов авторизованного пользователя.

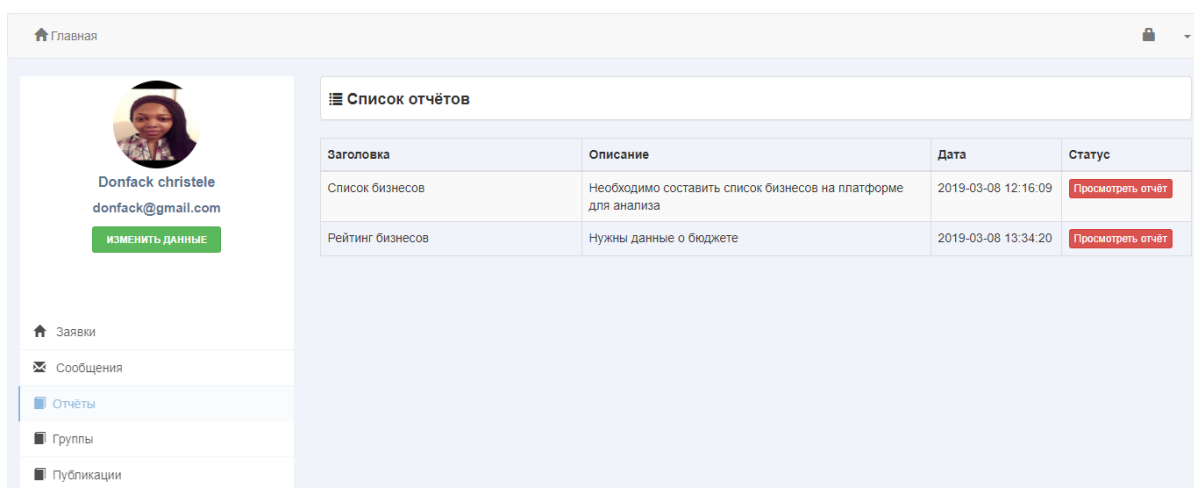


Рисунок 3.10 – Список готовых отчётов

После того как отчёт создан, можно посмотреть все доступные виды отчёта, их визуализировать и скачать. В данной подсистеме доступны 3 вида отчёта: PDF документ; Excel Документ; Граф, который может быть гистограммой, круговой диаграммой и линейной диаграммой.

При нажатии на кнопку «Посмотреть отчёт», отображаются все виды текущего отчёта. Всё это показана на рисунке 3.11.

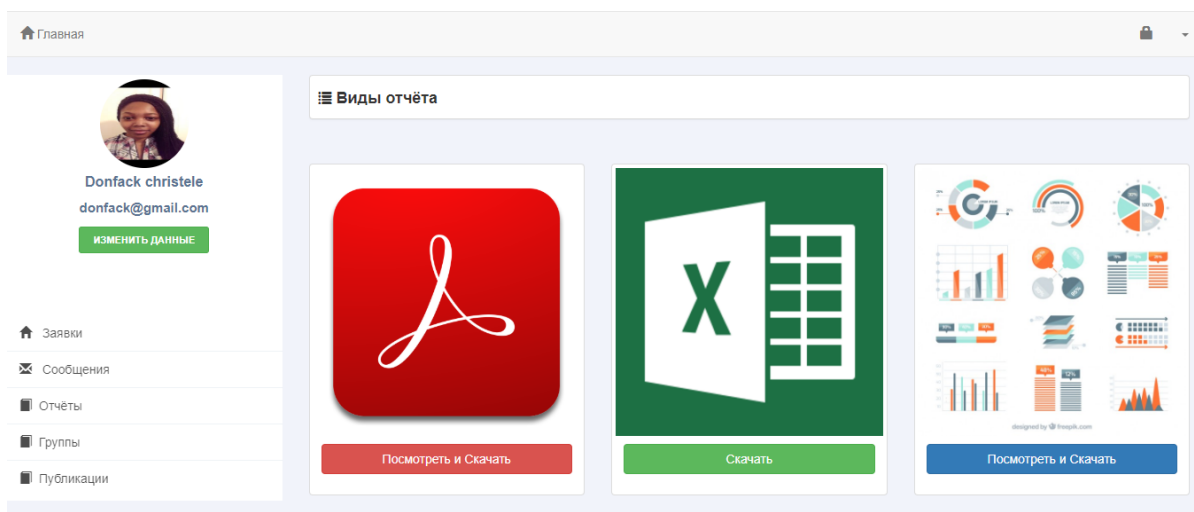


Рисунок 3.11 – Отображение видов текущего отчёта

При нажатии на PDF-документ отображаются данные в виде таблицы с возможностью скачивать. На рисунке 3.12 показаны данные в виде таблицы

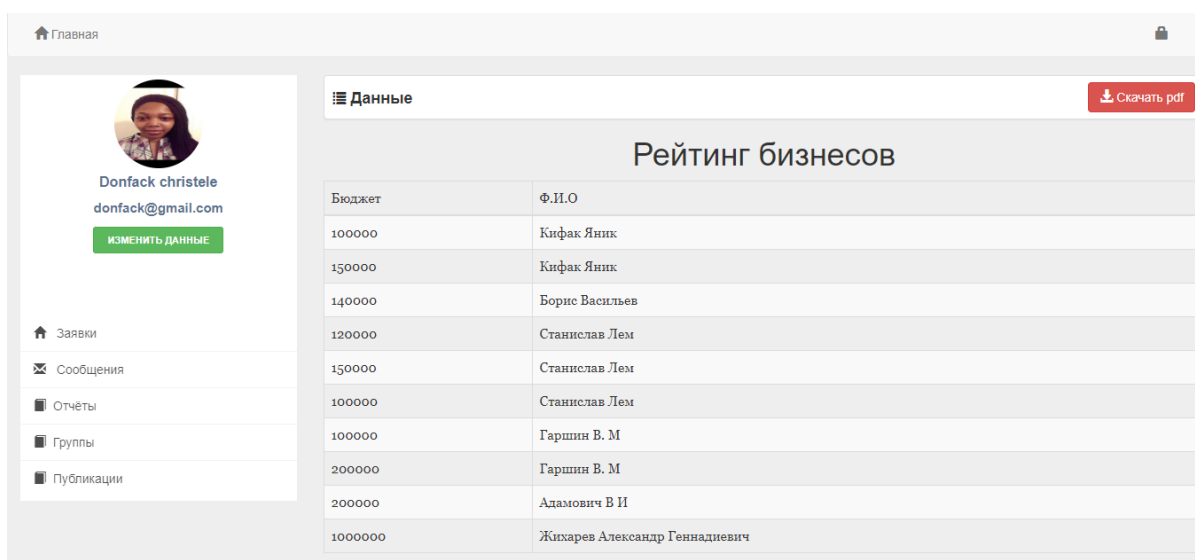


Рисунок 3.12 – Отображение данных в виде таблицы

При нажатии на кнопку «Скачать», таблица данные преобразуются в документ с расширением pdf. Формат переносимых документов (PDF) представляет собой универсальный файловый формат, который позволяет сохранить шрифты, изображения и сам макет исходного документа независимо от того, на какой из множества платформ и в каком из множества приложений такой документ создавался. На рисунке 3.13 просмотр скаченного документа.



Рисунок 3.13 – Скаченный PDF-документ

Для того чтобы отчёт был в виде Excel-документ, необходимо нажать на кнопку «Скачать». На рисунке 3.14 показан результат.

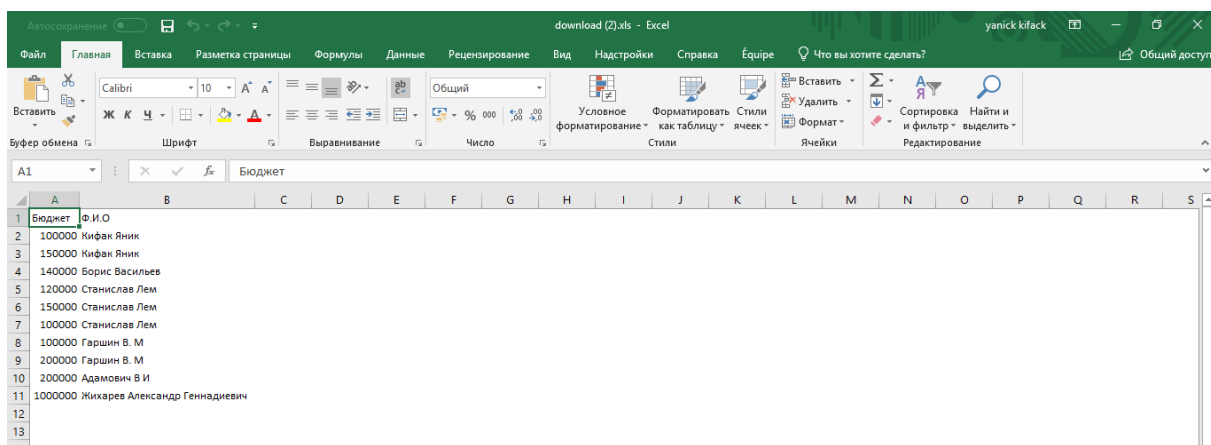


Рисунок 3.14 – Данные в Excel -документ

Третий вид отчёта является графом. Для отображения графа, необходимо выбрать вид графа, нужные поля, ввести название графа и нажать на кнопку «Просмотр». На рисунке 3.15 показаны данные отчёта в виде гистограммы.

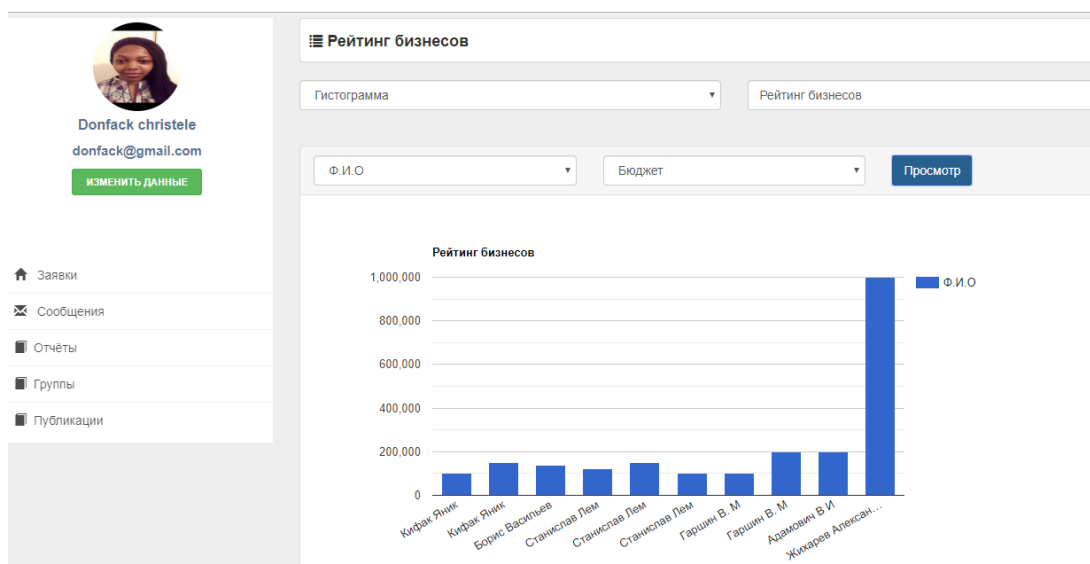


Рисунок 3.15 – Данные в виде гистограммы

Круговая диаграмма – это круговой статистический график, который разделен на срезы для иллюстрации числовой пропорции. На круговой

диаграмме длина дуги каждого среза (и, следовательно, его центральный угол и площадь) пропорциональна величине, которую он представляет. На рисунке 3.16 показаны данные в виде круговой диаграммы.

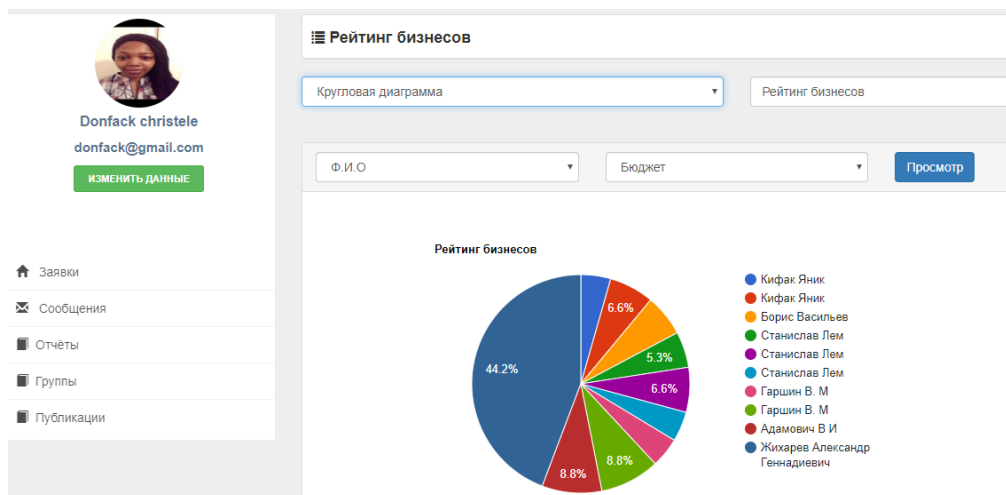


Рисунок 3.16 – Данные в виде круговой диаграммы

Диаграммы-линии или графики — это тип диаграмм, на которых полученные данные изображаются в виде точек, соединённых линиями. Точки могут быть как видимыми, так и невидимыми (ломанные линии). На рисунке 3.15 показаны данные в виде линий диаграммы.

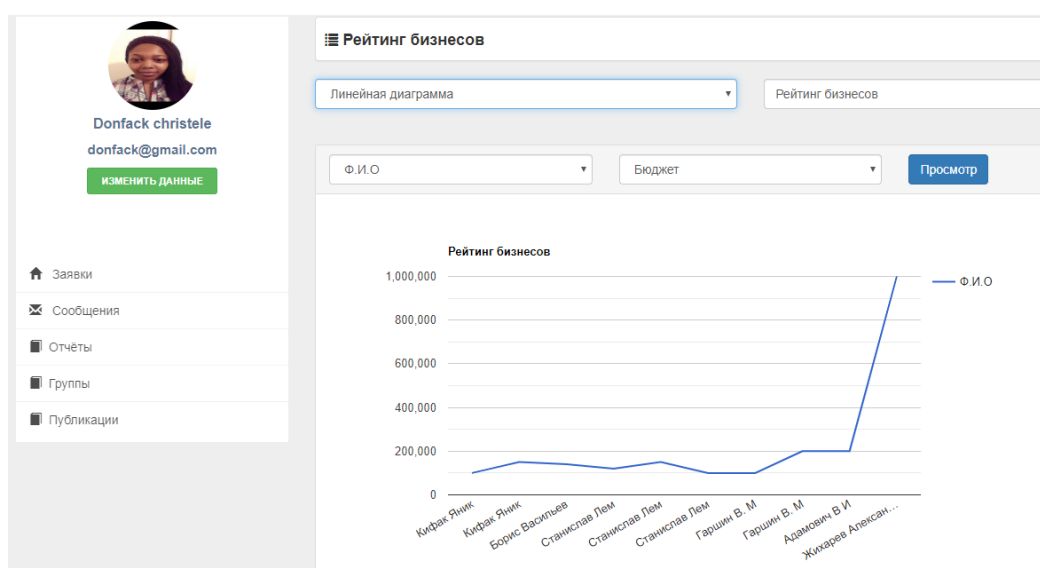


Рисунок 3.17 – Данные в виде линейной диаграммы

Диаграммы-области — это тип диаграмм, схожий с линейными диаграммами способом построения кривых линий. Отличается от них тем, что область под каждым графиком заполняется индивидуальным цветом или оттенком. На рисунке 3.18 показаны данные в виде диаграммы площади.

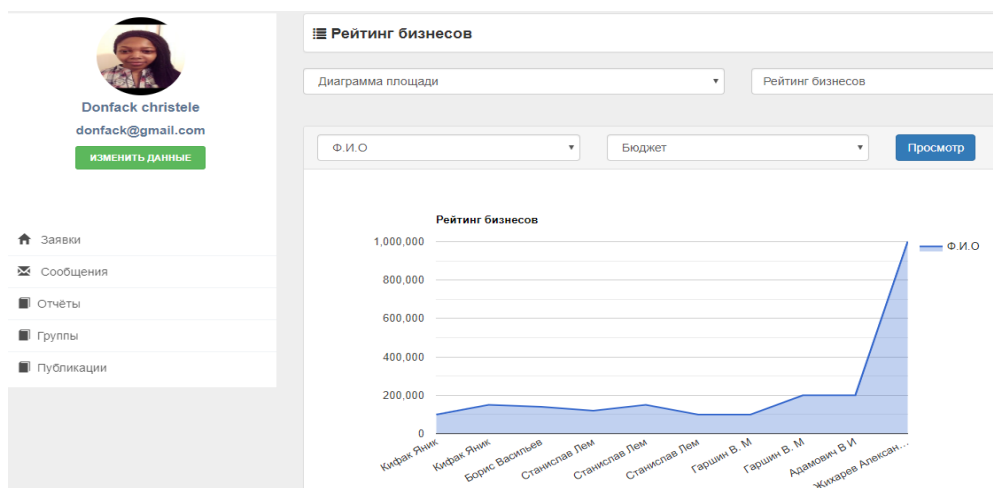


Рисунок 3.18 – Данные в виде диаграммы площади

Данная подсистема тоже позволяет создать группы пользователей и им предоставить возможность поделиться реализованными отчётами. На рисунке 3.19 показана форма добавления новой группы пользователей.

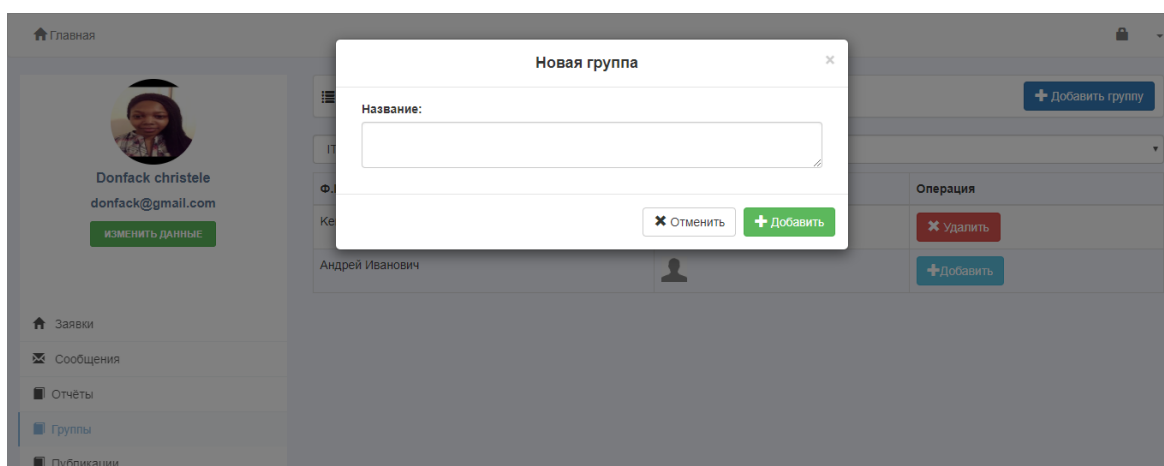


Рисунок 3.19 – Создание новой группы пользователей

После добавления новой группы, доступен список пользователей которые есть в группе. Для добавления нового пользователя ,надо нажать на кнопку «Добавить». На рисунке 3.20 показан список пользователей группы «IT engineer».

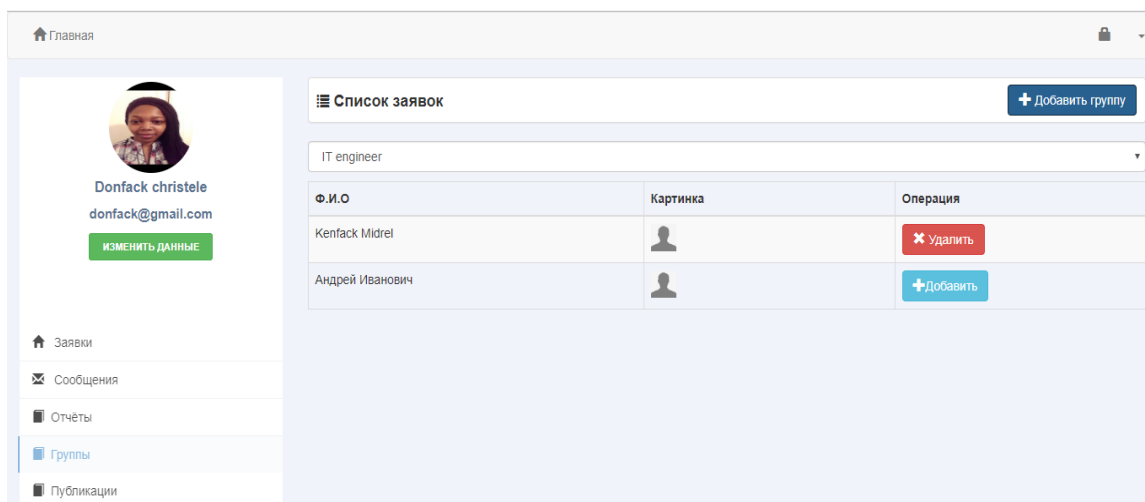


Рисунок 3.20 – Список пользователей группы «IT engineer»

Для того чтобы посмотреть все отчёты групп, которым принадлежит авторизованный пользователь, необходимо перейти на страницу «Публикации». На рисунке 3.21 показаны все публикации пользователя.

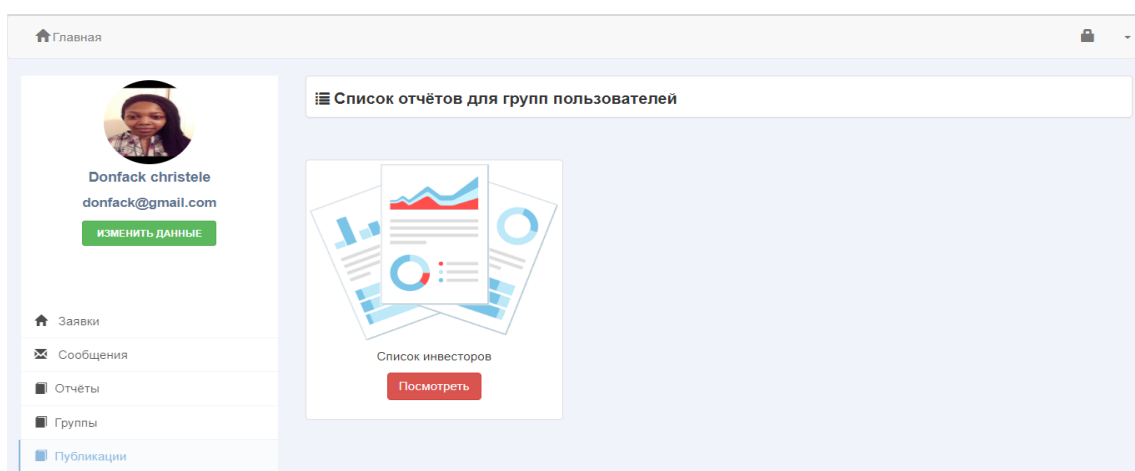


Рисунок 3.21 – Публикации авторизованного пользователя



### 3.2.2 Административный раздел

В административной части, всё начинается с создания задачи при поступлении сообщения пользователя. На рисунке 3.22 показан список пользователей.

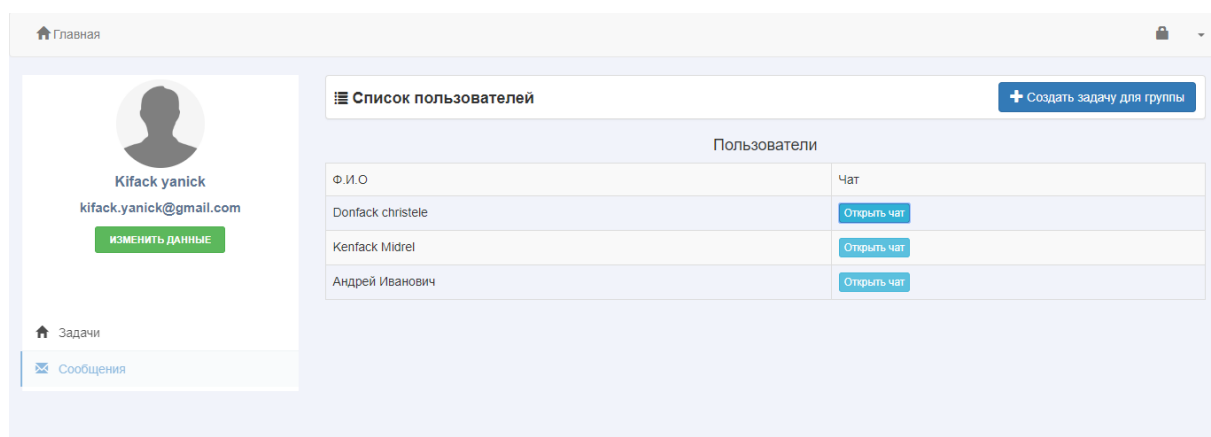


Рисунок 3.22 – Список всех пользователей

Для того чтобы начать чат с определённым пользователем, необходимо нажать на кнопку «Открыть чат». На рисунке 3.23 показаны переписки с пользователем.

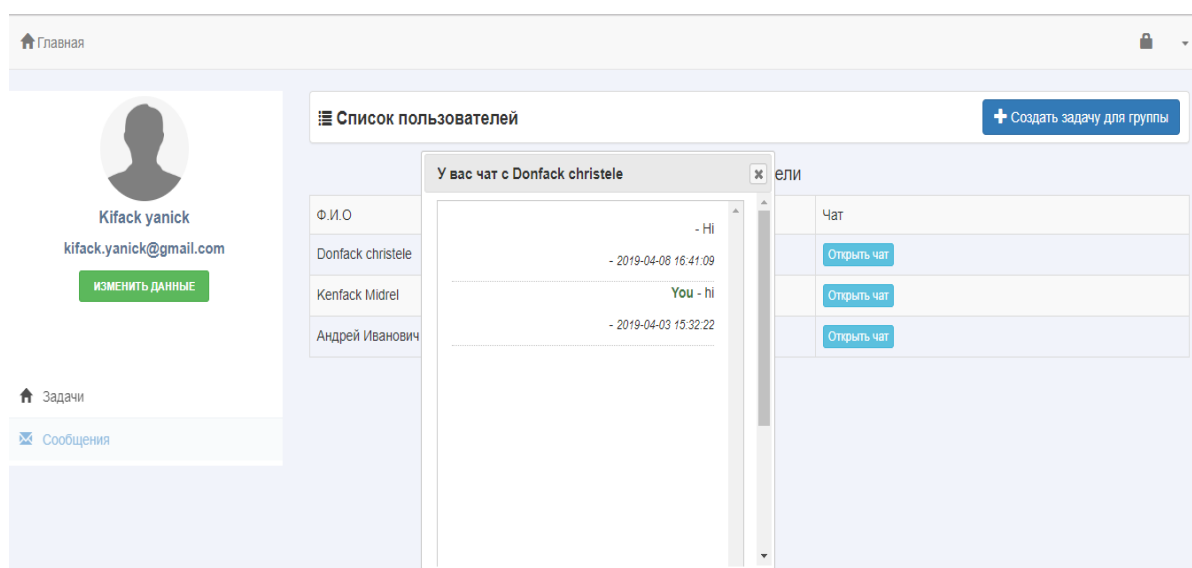


Рисунок 3.23 – Переписки с пользователем

Для того чтобы создать отчёт для пользователя, в окне чата нажать на кнопку «Задача» и ввести описание задачи. На рисунке 3.24 показана форма создания задачи.

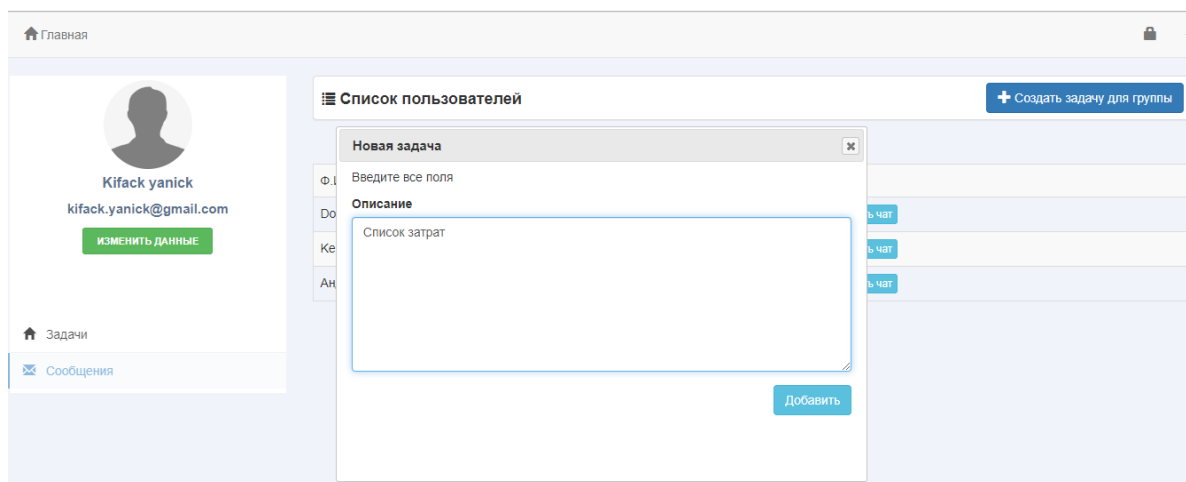


Рисунок 3.24 – Форма создания задачи

После создания задачи, можно приступить к её выполнению. В разделе «Задачи» находятся список всех созданных задач (рисунок 3.25). У каждой задачи есть 3 статуса: Создан – это после создания задачи; В обработке в случае когда уже начался процесс создания отчёта чере конструктор; Завершён.

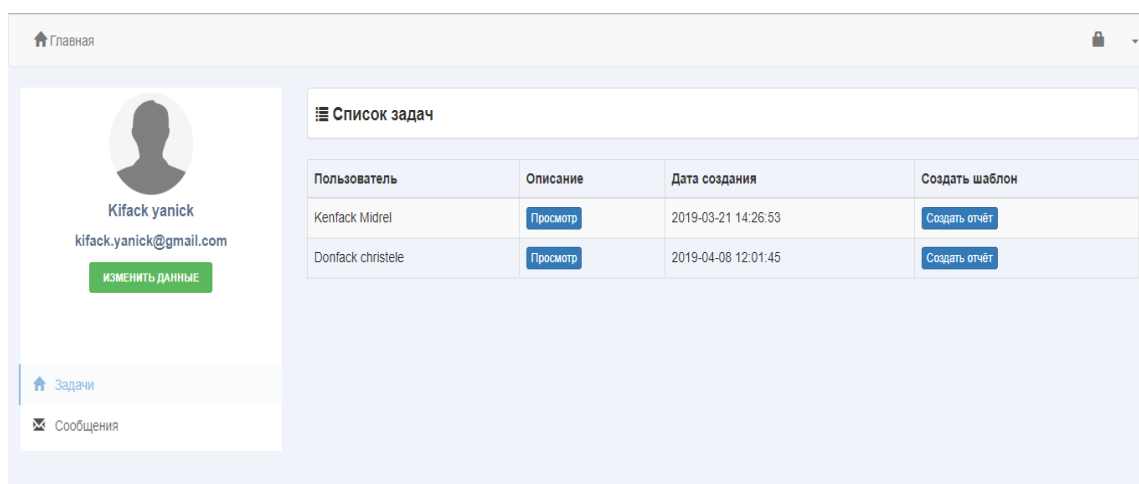


Рисунок 3.25 – Список созданных задач с статусом

После нажатия на кнопке «Создать отчёт», открывается конструктор – основной компонент этого проекта. На рисунке 3.26 показан разработанный конструктор запросов.

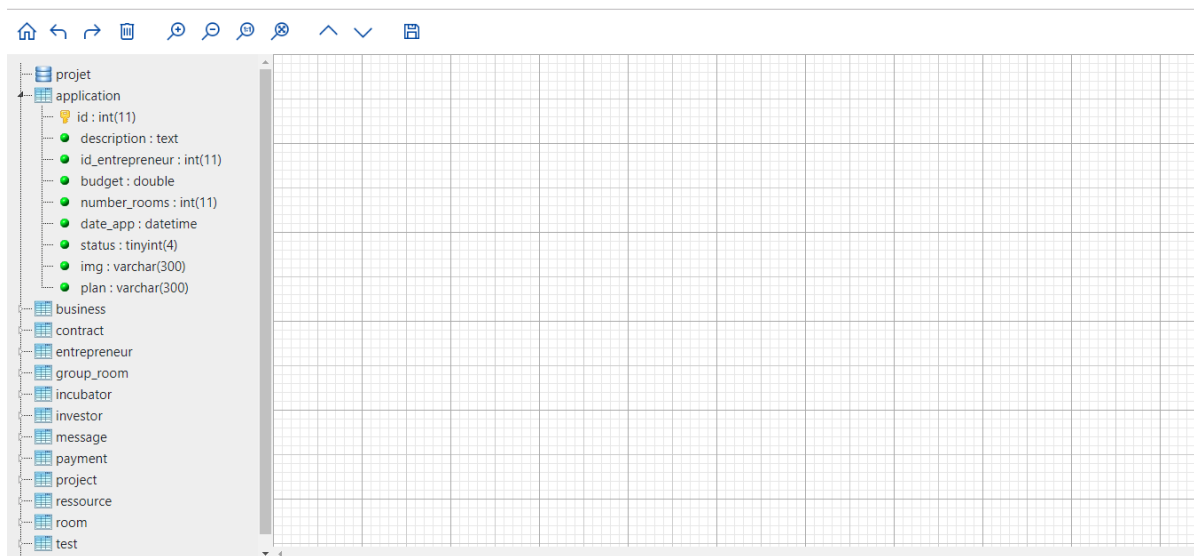


Рисунок 3.26 – Конструктор запросов

Слева находятся все таблицы базы данных главной системы с их атрибутами. Для того, чтобы отобразить определённую таблицу, необходимо её перетащить в сетку и она автоматически создастся. Результат этого действия можно посмотреть на рисунке 3.27.

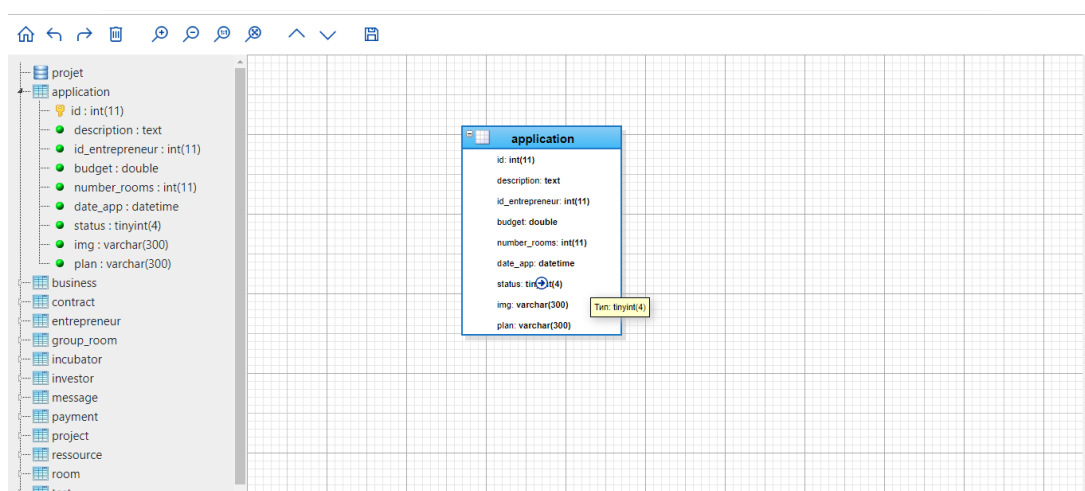


Рисунок 3.27 – Добавление таблицы в конструктор

В конструкторе доступно контекстное меню. При введении поля таблицы и нажатии правой кнопкой мыши, можно увидеть меню (рисунок 3.28).

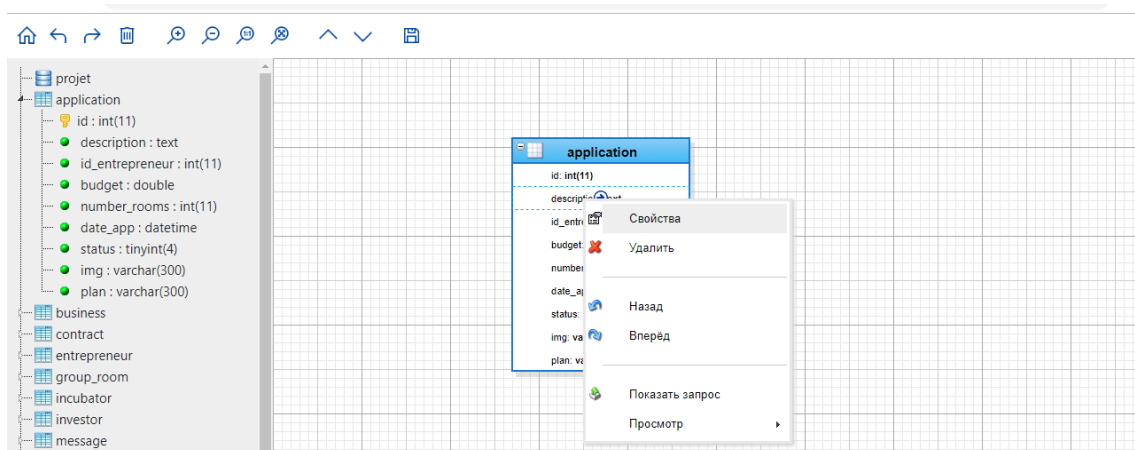


Рисунок 3.28 – Контекстное меню

У каждого поля таблицы доступны следующие свойства:

- опция отображать поле в запросе или нет;
- математическая операция, которую надо выполнить на поле;
- значение, которое можно придать выделенному полю при формировании SQL-запроса;
- SQL операция LIKE;
- Переименование поля;
- Сортировка по данному полю (возрастание или убывание);
- Группировка по данному полю;
- Агрегатная функция над полем.

Агрегатная функция выполняет вычисление на наборе значений и возвращает одиночное значение. Агрегатные функции, за исключением COUNT, не учитывают значения NULL. На рисунке 3.29 показаны все свойства поле «description».

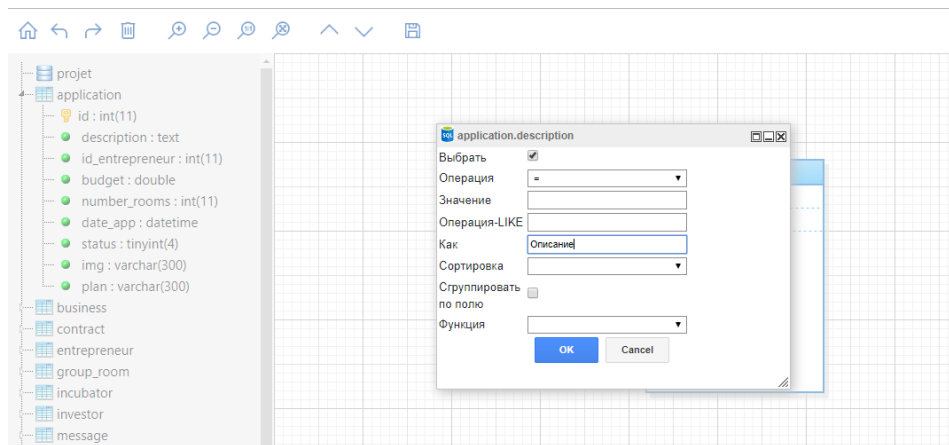


Рисунок 3.29 – Свойства поля

В конструкторе можно установить связь между таблицами при условии, что типы полей связи совпадают. У связи есть следующие свойства:

- исходная таблица, откуда идёт связь;
- конечная таблица;
- поле исходной таблиц;
- поле конечной таблицы;
- математическое отношение между полями таблиц.

На рисунке 3.30 показан пример связи между двумя таблицами с описанными выше информацией.

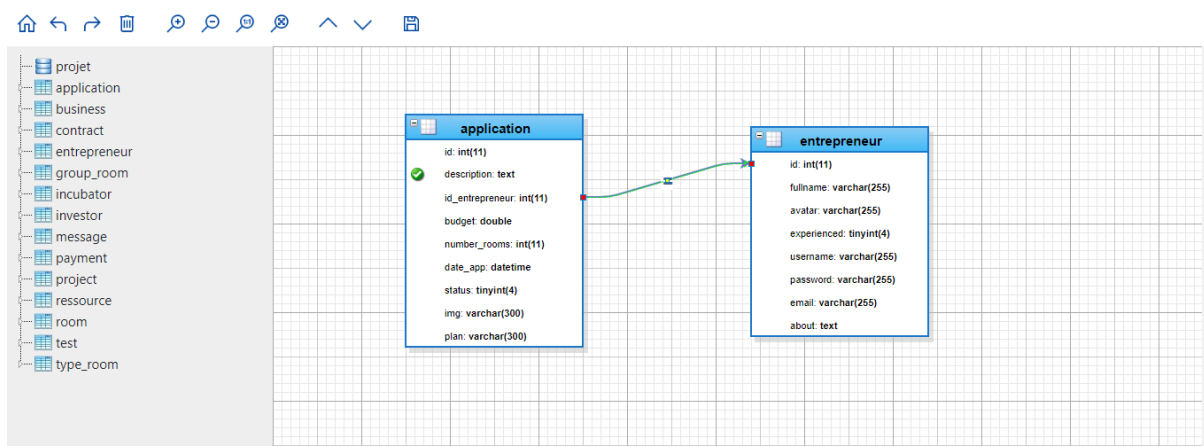


Рисунок 3.30 – Уставление связи между таблицами по «ID»

Для редактирования свойств связи между двумя таблицами, выделить связь, нажать на правой кнопке мыши «Свойства». На рисунке 3.31 показано окно редактирования связи.

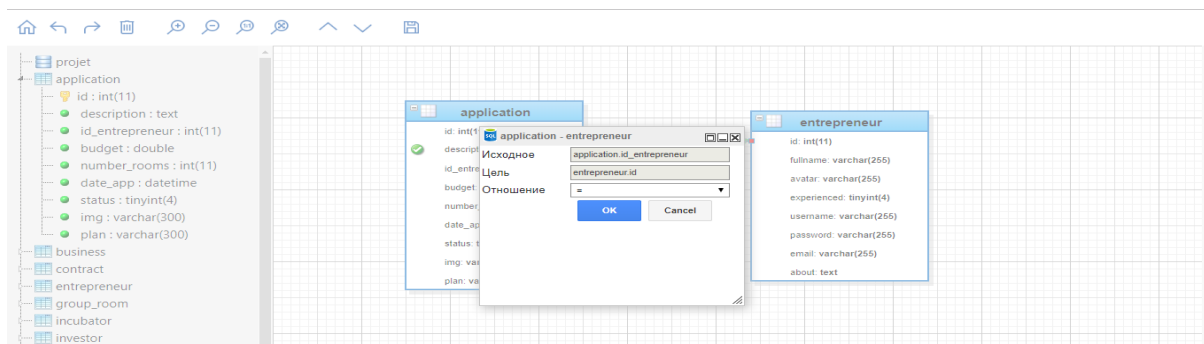


Рисунок 3.31 – Редактирование свойств связи

После выбора таблиц, полей и редактирования их свойств можно посмотреть сгенерированный SQL-запрос, нажимая в контекстном меню «Показать запрос». На рисунке 3.32 показан полученный запрос.



Рисунок 3.32 – Сгенерированный SQL-запрос

Перед тем как сохранить запрос, необходимо посмотреть результат сгенерированного запроса. В контекстном меню имеется подменю (рисунок 3.33).

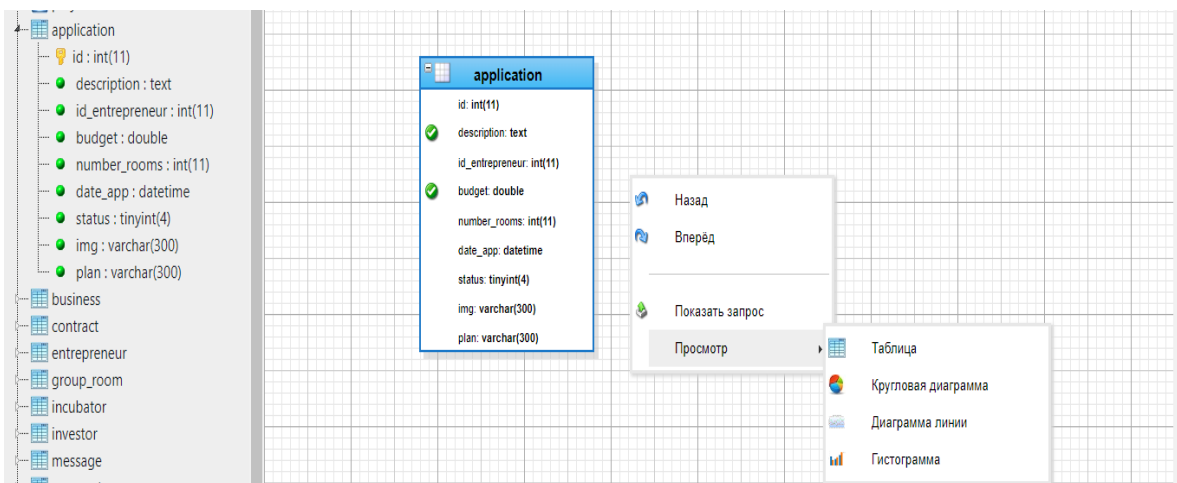


Рисунок 3.33 – Подменю

Результат запроса в конструкторе, можно посмотреть в трёх видах. В виде таблицы при нажатии на кнопке «Таблица» в подменю. (рисунок 3.34)

Описание	Бюджет
Школа для программиста	100000
Лаборатории для физики	80000
Компьютерные классы	30000
Оборудования для офисов	11000
Материал для сборки роботов	120000
Оборудования для нанотехнологии	70000
Центр для обучения языков	150000
Создание Startup	80000
Школа для обучения плавания	50000
Детский сад на дому	100000
Чистка ковров Бизнес на чистке ковров относится к тем видам деятельности, которые требуют минимальных капиталовложений	200000
Изготовление мыла ручной работы Бизнес, который можно открыть с минимумом средств - производство натурального декоративного мыла ручной работы	30000
Передвижной 3D-кинотеатр	200000
Пивной магазин Пиво - напиток номер один в России, который никогда не оставит предпринимателя без прибыли	150000

Рисунок 3.34 – Результаты запроса в виде таблицы

А также в виде круговой диаграммы, нажимая на кнопке «круговая диаграмма» в подменю. Она представляет собой круговую статистическую диаграмму, которая разделена на фрагменты для иллюстрации числовой пропорции (рисунок 3.35).

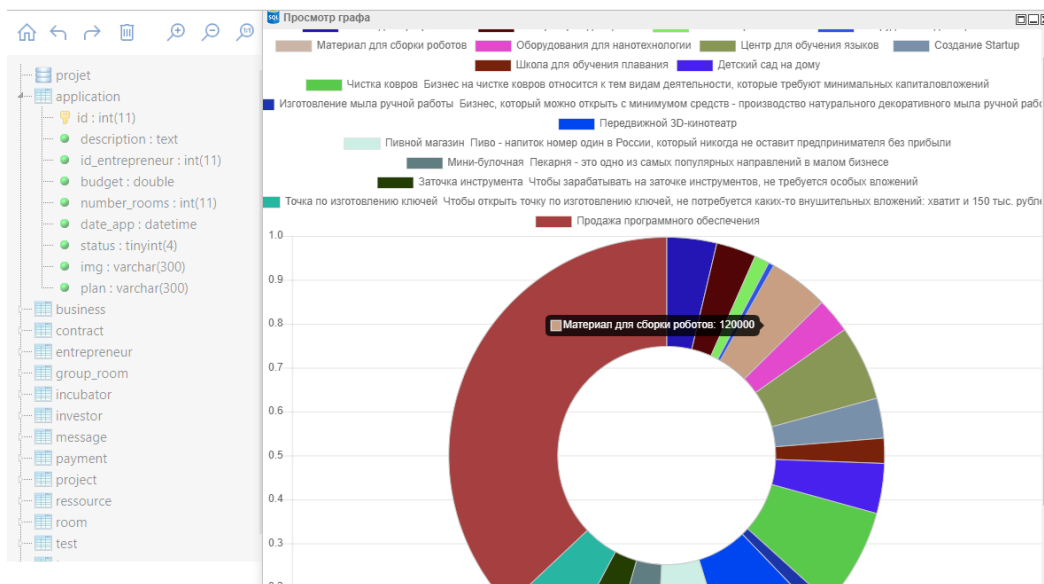


Рисунок 3.35 – Результаты запроса в виде круговой диаграммы

В виде диаграммы линий, нажимая на кнопку «Диаграмма линий» в подменю (рисунок 3.36). Диаграммы-линии или графики – это тип диаграмм, на которых полученные данные изображаются в виде точек, соединённых линиями.

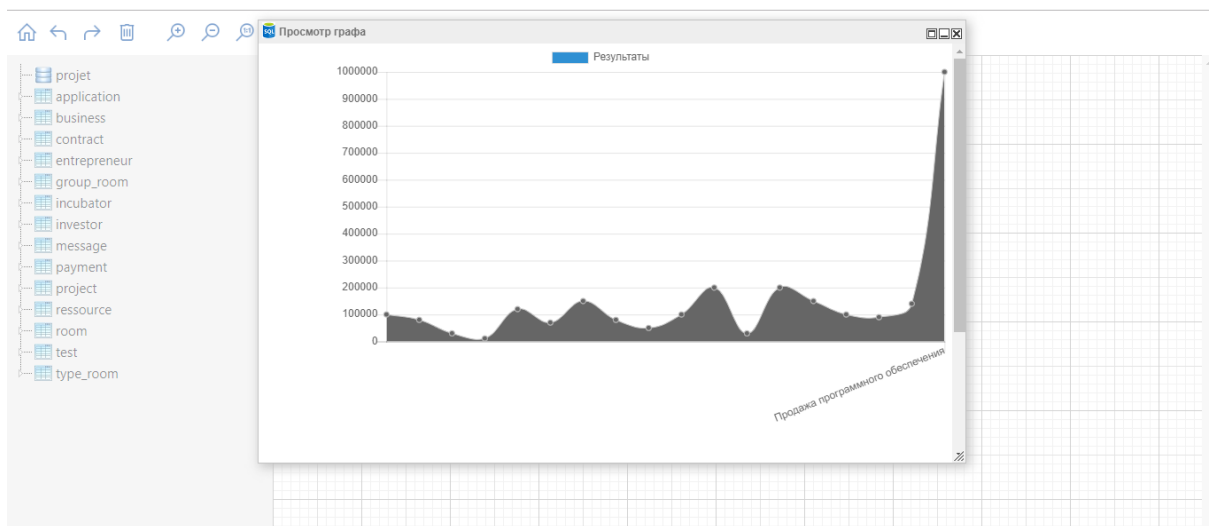


Рисунок 3.36 – Результаты запроса в виде диаграммы линий

В виде гистограммы нажимая на кнопку «Гистограмма» в подменю (рисунок 3.37).



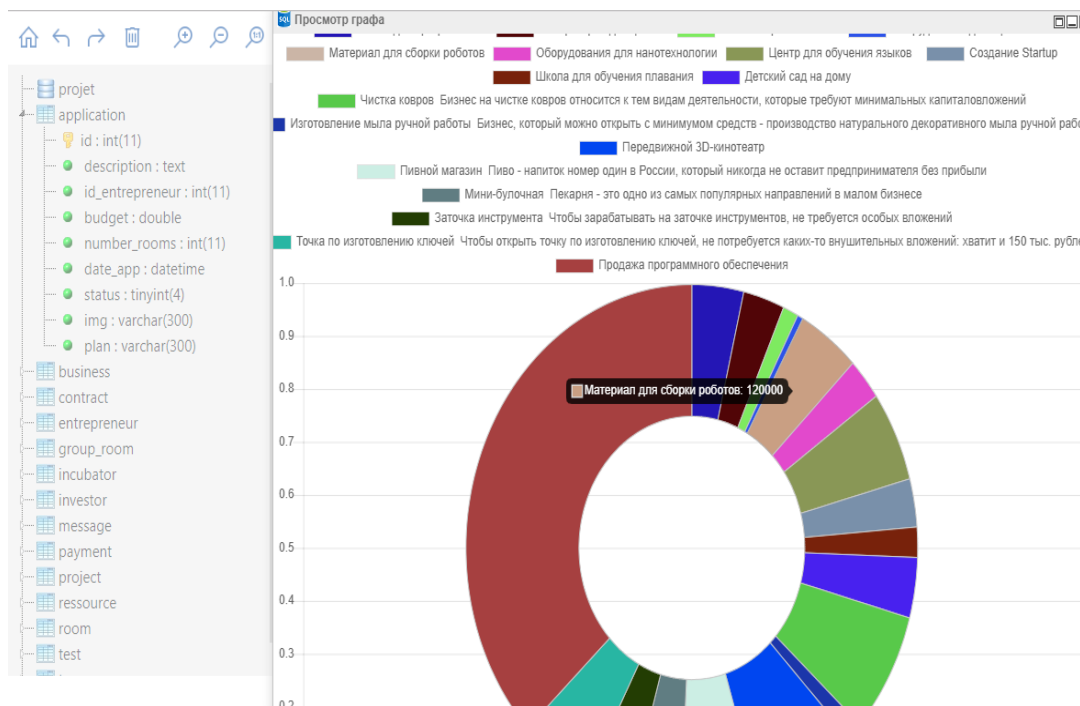


Рисунок 3.37 – Результаты запроса в виде гистограммы

Конструктор так же предоставляет следующие функции которые находятся в меню сверху :

- назад и вперёд;
- удаление таблиц или полей таблиц;
- увеличение размера отображения;
- свернуть таблицы.

Для того чтобы сохранить запрос, который в дальнейшем послужит для формирования отчётов, необходимо нажать на картинку «Сохранить» в меню и ввести заголовок отчёта. Пока не вводится заголовок созданного запроса с помощью конструктора он не сохраняется. После ввода заголовка, отправляют данные о запросе на сервер для того чтобы сохранить в базе данных после того направляют администратора на главную страницу где отображены все задачи созданны им. Процесс сохранения запроса показан на рисунке 3.38.

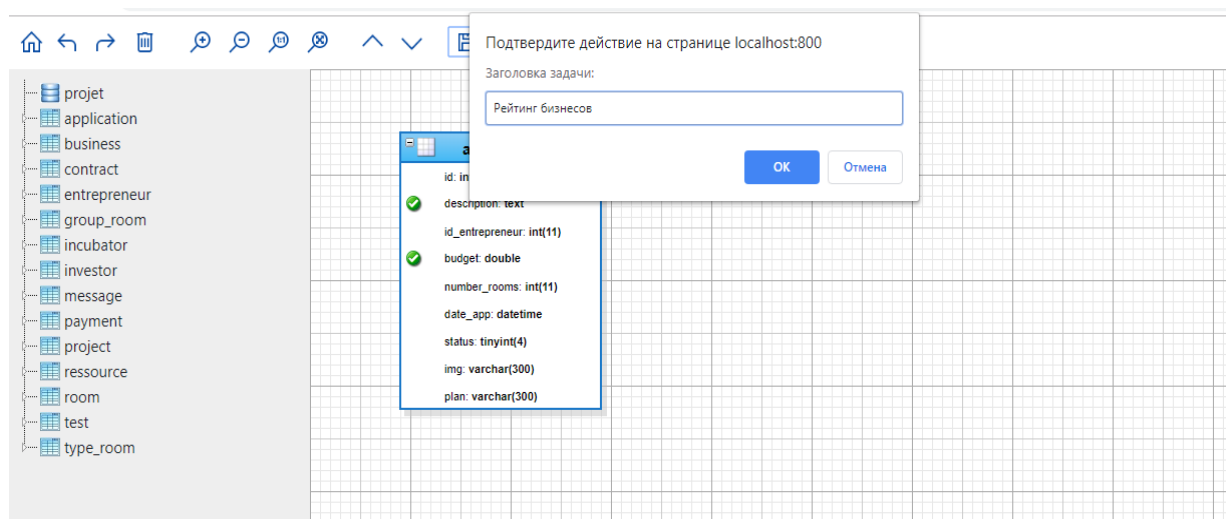


Рисунок 3.38 – Процесс сохранения сформированного запроса

Выводы по третьему разделу.

В данном разделе выпускной квалификационной работы была пошагово разработана подсистема автоматизированного формирования отчётности на основе конструктора запросов. Был разработан конструктор запросов, который позволяет создать динамические отчёты используя графическое представление таблиц базы данных. Показана разработка механизма формирования отчётов. Также проведено успешное функциональное тестирование системы начиная от регистрации нового пользователя, работы с заявками и до просмотра сформированного отчёта в разных видах. По завершению разработки программный продукт передан заказчику с целью внедрения и интеграционного тестирования.

## ЗАКЛЮЧЕНИЕ

В результате выполнения работы были получены следующие основные результаты:

- проведен обзор существующих автоматизированных информационных систем формирования отчетности и инструментов создания отчетов;

- разработан алгоритм, по которому работает конструктор запросов, используя принципы и синтаксис языка программирования SQL. Так же были разработаны алгоритмы сохранения запросов в базе данных и алгоритм создания динамических отчетов, используя сохраненные поля запроса в базе;

- разработана модель процесса подсистемы формирования отчетности на основе конструктора запросов. Данная модель позволяет использовать разработанную подсистему на различных информационных системах, которые обладают большими объемами данных;

- проведено проектирование и реализация автоматизированной подсистемы формирования отчетности на основе конструктора запросов, была продемонстрирована работа с разработанным конструктором запросов, который является основной частью этого проекта, так как такого инструмента создания запроса в виде web-приложения нет.

Анализ решения поставленных задач по полученным результатам исследования позволяет утверждать, что цель достигнута.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Абросимова, М.А. Информационные технологии / М.А. Абросимова – М.: КноРус, 2013. – 248 с.
2. Информационная система [Электронный ресурс] Режим доступа: [https://ru.wikipedia.org/wiki/Информациоанная\\_система](https://ru.wikipedia.org/wiki/Информациоанная_система)
3. Венделева, М.А. Информационные технологии: Учебное пособие для магистрантов / М.А. Венделева, Ю.В. Вертакова – М.: Юрайт, 2013. – 462 с.
4. Datarpine система [Электронный ресурс] Режим доступа: <https://www.datapine.com/>
5. FreshBooks [Электронный ресурс] Режим доступа: <https://www.freshbooks.com/>
6. Дубейковский, В.И. Эффективное моделирование AllFusionProcessMockler (Bpwin) / В.И. Дубейковский – М.: Диалог–МиФи, 2016. – 384 с.
7. mxGraph [Электронный ресурс] Режим доступа: <https://jgraph.github.io/mxgraph/docs/manual.html>
8. CodeMirror [Электронный ресурс] Режим доступа: <https://codemirror.net/>
9. Дубейковский, В.И. Практика функционального моделирования / В. И. Дубейковский– М.: Диалог–МиФи, 2016. – 464 с.
10. Диаграмма классов [Электронный ресурс] Режим доступа: [https://ru.wikipedia.org/wiki/Диаграмма\\_классов](https://ru.wikipedia.org/wiki/Диаграмма_классов)
11. Репин, В.В. Процессный подход к управлению. Моделирование бизнес-процессов / В.В. Репин, В.Г. Елиферов – М.: РИА «Стандарты и качество», 2014. – 398 с.
12. Романов, В.П. Информационные технологии моделирование / В.П. Романов, М.В. Бадрина – М.: ФиС, 2015. – 288 с.

13. Синаторов, С.В. Информационные технологии / С.В. Синаторов – М.: Альфа – М, НИЦ ИНФРА – М, 2013. – 336 с.
14. Федотова, Е.Л. Информационные технологии и системы / Е.Л. Федотова – М.: ИД ФОРУМ, НИЦ ИНФРА–М, 2013. – 352 с.
15. Хаггард, Г. Автоматизация для программистов: Учебное пособие / Г. Хаггард, Д. Шлипф, С. Уайтсайдс; Пер. с англ. Н.А. Шихова; Под ред. А.А. Сапоженко– М.: БИНОМ. ЛЗ, 2013. – 627 с.
16. Черемных, С.В. Структурный анализ систем: IDEF-технологии / С.В. Черемных, И.О. Семенов, В.С. Ручкин – М.: Финансы и статистика, 2015. – 208 с.
17. . Агуров, Павел С#. Сборник рецептов / Павел Агуров. - М.: "БХВ-Петербург", 2012. - 432 с.
18. Албахари, Джозеф С# 3.0. Справочник / Джозеф Албахари , Бен Албахари. - М.: БХВ-Петербург, 2012. - 944 с.
19. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - М.: Вильямс, 2015. - 266 с.
20. Ишкова, Э. А. Самоучитель С#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 с

## ПРИЛОЖЕНИЕ

### Листинг 1 – Конструктор запросов

```
function get_data(graph) {

    var parent = graph.getDefaultParent();
    var childCount = graph.model.getChildCount(parent);

    var links=[],fields=[];

    for (var i=0; i<childCount; i++)
    {
        var child = graph.model.getChildAt(parent, i);

        if (!graph.model.isEdge(child))
        {
            var columnCount = graph.model.getChildCount(child);

            for (var j=0; j<columnCount; j++)
            {
                var tmp=[];
                var column = graph.model.getChildAt(child, j).value;

                if(column.choosed){
                    tmp.push(<?= $id ?>);
                    tmp.push(child.value.name);
                    tmp.push(column.name);
                    tmp.push(column.as);
                    tmp.push(column.like);
                    tmp.push(column.val);
                    tmp.push(column.relation);
                    var o=null,g=null;

                    tmp.push(column.order==""?0:column.order=="ASC"?1:2);

                    tmp.push(column.group?1:0);
                    tmp.push(column.func);

                    fields.push(tmp);
                }
            }

        }else{
            var val=child.value;
            var tmp=[val.from.table,val.to.table,val.from.field,val.to.field,val.type,<?= $id ?>];

            links.push(tmp);
        }
    }
    return [fields,links];
}function showProperties(graph, cell)
```

```

{
    // Creates a form for the user object inside
    // the cell
    var form = new mxForm('properties');
    var chooseField = form.addCheckbox('Выбрать', cell.value.choosed);
    var relationField=form.addCombo("Операция",false,1);
    var valField = form.addText('Значение', cell.value.val);
    var likeField = form.addText('Операция-LIKE', cell.value.like);
    var asField = form.addText('Как', cell.value.as);
    // var sortByField = form.addCheckbox('Sort by field name', cell.value.by);
    var orderField=form.addCombo("Сортировка",false,1);
    var orderCell=cell.value.order;
    form.addOption(orderField,"", "");
    form.addOption(orderField,"По возрастанию","ASC",orderCell=="ASC");
    form.addOption(orderField,"По убыванию","DESC",orderCell=="DESC");
    var arr=["","AVG","COUNT","MAX","MIN","SUM"];
    var arr1=["=","<=",">=","<",">","<>"];
    var groupByField = form.addCheckbox('Сгруппировать по полю', cell.value.group);
    var funcField=form.addCombo("Функция",false,1);
    var funcCell=cell.value.func;
    var relCell=cell.value.relation;
    for(var i=0;i<arr.length;i++){
        form.addOption(funcField,arr[i],arr[i],funcCell==arr[i]);
    }
    for(var i=0;i<arr1.length;i++){
        form.addOption(relationField,arr1[i],arr1[i],relCell==arr1[i]);
    }
    var wnd = null;
    var okFunction = function()
    {
        var clone = cell.value.clone();
        clone.val = valField.value;
        clone.like = likeField.value;
        clone.as = asField.value;
        clone.order = orderField.value;
    }
}

```

```

clone.relation = relationField.value;
clone.group = groupByField.checked;
clone.choosed = chooseField.checked;
clone.func=funcField.value;
graph.model.setValue(cell, clone);
wnd.destroy();
}
var cancelFunction = function()
{
    wnd.destroy();
}
form.addButtons(okFunction, cancelFunction);
var parent = graph.model.getParent(cell);
var name = parent.value.name + '.' + cell.value.name;
wnd = showModalWindow(name, form.table, 400, 300);
};
function showLinkProperties(graph,cell){

    var form = new mxForm('Properties link');

    var fromField = form.addText('Исходное', cell.value.from.table+"."+cell.value.from.field);
    var toField = form.addText('Цель', cell.value.to.table+"."+cell.value.to.field);
    var typeField = form.addCombo("Отношение",false,1);
    fromField.disabled = true;
    toField.disabled = true;
    var arr=["=", "<=", ">=", "<", ">", "<>"];
    var typeCell=cell.value.type;

    for(var i=0;i<arr.length;i++){
        form.addOption(typeField,arr[i],arr[i],typeCell==arr[i]);
    }
    var wnd = null;
    var okFunction = function()
    {
        var clone = cell.value.clone();

```



```

    clone.type=typeField.value;
    console.log(clone.type)
    // cell.value.type=typeField.value;
    graph.model.setValue(cell, clone);
    wnd.destroy();
  }
  var cancelFunction = function()
  {
    wnd.destroy();
  }
  form.addButtons(okFunction, cancelFunction);
  var name = cell.value.from.table + ' - ' + cell.value.to.table;
  wnd = showModalWindow(name, form.table, 300, 250);
}
function fromArrayToSql(arr){
  var select=arr[0],
    from=arr[1],
    where=arr[2],
    order=arr[3],
    groupBy=arr[4];
  if(select.length>0)select="SELECT "+select.join(" , ")+"\n";
  else select="SELECT "+arr[5].join(" , ")+"\n";
  if(from.length>0)from="FROM "+from.join(" , ")+"\n";
  else from="";
  if(where.length>0) where="WHERE "+where.join(" AND ")+"\n";
  else where="";
  if(order.length>0)order="ORDER BY "+order.join(" , ")+"\n";
  else order="";
  if(groupBy.length>0) groupBy="GROUP BY "+groupBy.join(" , ")+"\n";
  else groupBy="";

  return select+from+where+order+groupBy;
}

```

Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

« \_\_\_\_ » \_\_\_\_\_ Г.

\_\_\_\_\_  
(подпись)

Кифак Викторьен Яник