ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(НИУ «БелГУ»)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ИНТЕРАКТИВНАЯ СИСТЕМА КОНТРОЛЯ УСПЕВАЕМОСТИ И ПОСЕЩАЕМОСТИ УЧЕНИКОВ СРЕДНЕЙ ШКОЛЫ

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.03 Прикладная информатика
очной формы обучения, группы 07001404
Цейтлиной Натальи Евгеньевны

Научный руководитель:
к.т.н., доцент Маматов Е.М.

БЕЛГОРОД 2018

СОДЕРЖАНИЕ

ВВЕДЕНИЕ……..3

1 Аналитическая часть……..... 6

1.1 Краткая характеристика учебного процесса средней школы……....6

1.2 Постановка задач……... 7

1.2.1 Описание структуры работы системы “КАК ЕСТЬ”.....7

1.2.2 Характеристика выявленных недостатков……...20

1.2.3 Постановка задач для устранения выявленных недостатков…….23

1.3 Характеристика возможных способов решения задач…….24

2 Информационное обеспечение задачи……25

2.1 Информационная модель и ее описание……... 25

2.2 Характеристика интегрируемых систем коммуникации……. 33

2.3 Характеристика базы данных……36

2.3.1 Характеристика инфологической модели БД……..36

2.3.2 Характеристика даталогической модели БД……...40

2.4 Обоснование выбора технологий для разработки……....42

2.4.1 Характеристика технологий разработки серверной части……42

2.4.2 Характеристика визуальной оболочки……44

3 Программная реализация проектных решений……... 45

3.1 Программное обеспечение задачи……. 45

3.1.1 Общие положения……..45

3.1.2 Структурная схема пакета……46

3.2 Описание контрольного примера реализации……..48

3.3 Обоснование эффективности разработки……. 57

ЗАКЛЮЧЕНИЕ……..59

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ……60

ПРИЛОЖЕНИЕ А……. 65

ВВЕДЕНИЕ

Данная работа посвящена интерактивной системе контроля успеваемости и посещаемости учеников средней школы.

Актуальность данной работы заключается в возможности разработать универсальную систему, обеспечивающую взаимодействие всех участников учебного процесса средней школы. Так как в большинстве существующих систем управления деятельностью средней школы не обеспечен необходимый функционал для своевременного точного уведомления родителей об успехах или проблемах, возникающих в учебном процессе их детей, они зачастую не имеют возможности своевременно связаться с преподавателем. Разрабатываемая система позволит автоматически уведомлять родителей об изменениях в успеваемости и посещаемости их детей. Наиболее актуальным является то, что у родителей не будет необходимости использовать сторонние системы при отсутствии большого количества времени на контроль учебного процесса их детей. Родители учащихся смогут получать уведомления по стандартным каналам связи, таким как электронная почта, sms-сообщение. Преподавателям в свою очередь не придется самостоятельно рассылать уведомления родителям, так как система на основании изменений показателей учеников будет предлагать отправить уведомление с шаблонным текстом.

Объектом исследования является средняя школа. Предметом исследования является процесс контроля успеваемости и посещаемости учеников.

Целью данной работы является разработка интерактивной системы контроля успеваемости и посещаемости учеников средней школы.

Для реализации поставленной цели необходимо хорошо знать предметную область, особенно детально знать методы и принципы работы учебной части стандартной средней школы, выявить недостатки в работе используемых в ней систем, разработать варианты устранения, выполнить проектирование выбранного варианта решения, а затем реализовать его.

Следует перечислить задачи, которые необходимо выполнить для достижения поставленной цели и подробно их описать:

− выявить цели и результаты деятельности выбранного типа учреждений, выявить недостатки в используемых системах контроля успеваемости и посещаемости, описать требования к разрабатываемому функционалу, выполнить обзор существующих способов решения проблемы, путем поиска аналогичных программных систем;

− выполнить разработку информационной модели новой программной системы, в которой будет показано, что она решает проблемы, которые были выявлены на предыдущем этапе, и подробно описать ее, разработать структуру базы данных, представить ее в виде инфологической модели, сущности которой должны соответствовать реальным объектам предметной области, также представить даталогическую схему базы данных с привязкой к конкретной выбранной СУБД, описать технологии разработки серверной части и клиентской визуальной оболочки;

− разработать иерархическую структуру работы системы, порядок вызова процедур и функций, принцип работы диалога системы и пользователя, разделить функционал разрабатываемой системы на модули, выбрать технологическое обеспечение, используемое при разработке, привести тестовый пример работы системы, выполнить обоснование работы.

Данная работа имеет следующую структуру:

− Аналитическая часть - подробное описание выбранной предметной области, то есть описание структуры учебной части стандартной средней школы, его целей и задач, выявленных недостатков в существующих системах, а также предложений по их устранению.

− Информационное обеспечение задачи - описание информационной модели, структуры базы данных, ее инфологическая модель и даталогическая модель, а также технологий разработки серверной части и визуальной оболочки.

− Программная реализация проектных решений - описание программной реализации, ее общих положений, структурной схемы пакета и программных модулей, подробное описание технологического обеспечения задачи, технологий сбора, передачи, обработки и выдачи информации, приведение тестового примера реализации проекта, обоснование эффективности разработки.

Данная работа состоит из 3 разделов, 74 страниц, 26 рисунков, 1 таблицы, 1 приложения.

1 Аналитическая часть

1.1 Краткая характеристика учебного процесса средней школы

Для выполнения данной работы необходимо иметь полное представление о структуре автоматизируемой деятельности. В данном случае предметной областью выступает учебный процесс средней школы. В начале необходимо кратко его описать.

Средняя школа - это учреждение, предназначенное для получения общего образования, которое в последствии дает возможность продолжить обучение в высшем учебном заведении.

В основном качество получаемого учеником образования оценивается в разрезе успеваемости и посещаемости.

Успеваемость - это глубина освоения учеником материала, преподаваемого на уроках, установленного учебной программой, качество полученных знаний, их осмысленность и полнота, умение применять эти знания на практике. Успеваемость обычно измеряется в количественном выражении, то есть за свои знания ученики получают так называемые оценки (в РФ по пятибалльной шкале).

Посещаемость - это величина, определяющая насколько регулярно ученик присутствует на уроках. Количественно посещаемость можно выразить как отношения числа посещенных учеником уроков за определенный период к общему числу уроков, которые были проведены за этот же период.

В данном подразделе дана краткая характеристика учебного процесса средней школы, основных показателей оценки знаний учащихся, их количественное выражение.

1.2 Постановка задач

1.2.1 Описание структуры работы системы “КАК ЕСТЬ”

После технико-экономической характеристики работы учебной части стандартной средней школы следует составить более точную формализованную модель работы данной структуры, которая позволит наглядно представить существующие процессы, осуществляемые для получения необходимых результатов деятельности. Схематичная модель деятельности представлена с помощью диаграмм в нотации DFD и IDEF0. Наглядное представление позволит адекватно оценить достоинства и недостатки существующего образа осуществления конкретных процессов. Диаграммы позволяют оценить распределение обязанностей между сотрудниками, необходимые материалы и данные, документы и организации, осуществляющие контроль деятельности, выходящие результаты выполнения процессов.

Изначально необходимо представить наиболее глобальную картину осуществления деятельности. Это поможет глобально оценить все материальные, информационные, контролирующие и человеческие ресурсы, используемые при осуществлении рассматриваемых процессов. Более глобальные диаграммы представлены в методологии IDEF0, так как именно данная методология используется для построения диаграмм, в которых нет необходимости максимально детального отображения всех тонкостей осуществляемой деятельности. В IDEF0 и подобных методологиях наиболее глобальная картина осуществляемой деятельности отображается в виде контекстной диаграммы.

Контекстная диаграмма контроля посещаемости и успеваемости учеников средней школы представлена на рисунке 1.1.

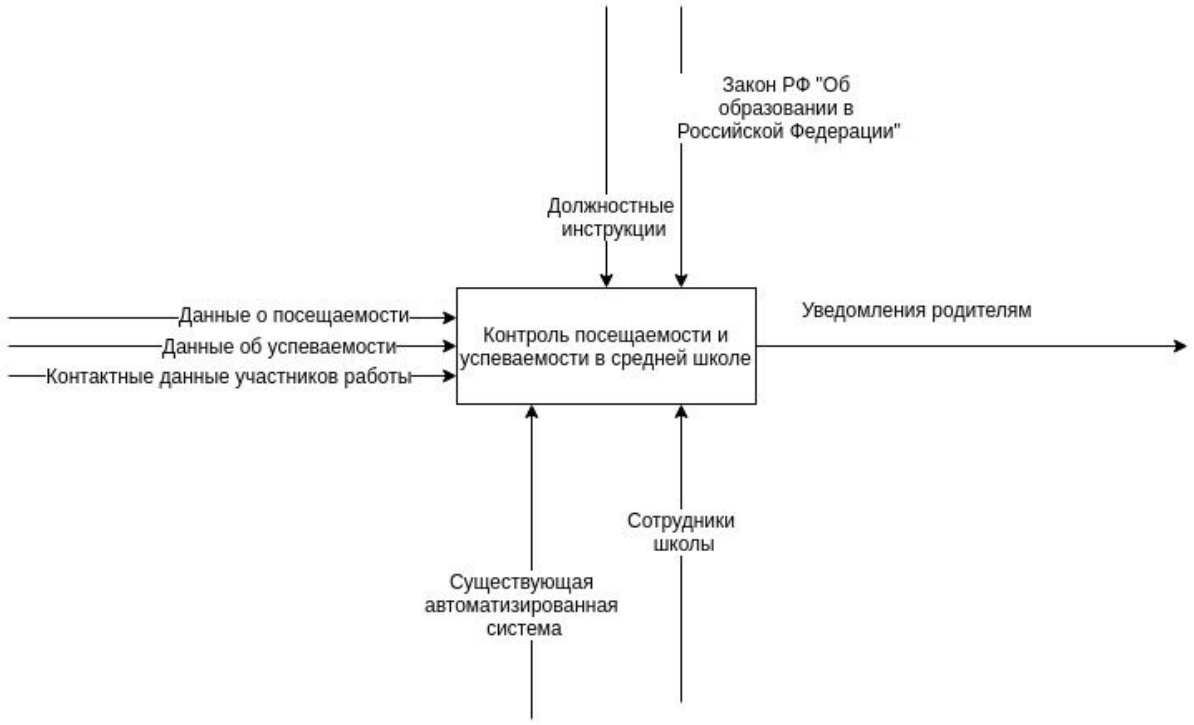

Рисунок 1.1 - Контекстная диаграмма контроля посещаемости и успеваемости и успеваемости в средней школе

На контекстной диаграмме представлен главный блок “Контроль посещаемости и успеваемости в средней школе”. Данный блок обобщает всю рассматриваемую деятельность. Соответственно положениям стандарта IDEF0 у данного блока присутствуют три входа и один выход. Верхние стрелки входа символизируют контроль осуществляемой деятельности. В данном случае это должностные инструкции и закон РФ "Об образовании в Российской Федерации". Должностные инструкции определяются внутренними правилами действующими в средней школе, а закон РФ "Об образовании в Российской

Федерации" определяется законодательными нормами, призванными контролировать учебный процесс.

Нижние стрелки символизируют механизм осуществления деятельности. На контекстной диаграмме они представлены в виде существующей автоматизированной информационной системы и сотрудников школы. Существующая автоматизированная информационная система определяет собой любые из разработанных и представленных для использования системы, которые используют в своей деятельности стандартные средние школы в РФ. Сотрудниками школы соответственно являются преподаватели и другие работники, задействованные в выполнении выбранных для исследования процессов.

Стрелки, входящие в блок контекстной диаграммы слева символизируют собой входящие данные и материалы, то есть все то, что непосредственно обрабатывается в процессе работы. В данной деятельности входящие данные представлены в виде данных о посещаемости, данных об успеваемости и контактных данных участников работы. Данные о посещаемости представляют собой подробную информацию о присутствии каждого конкретного ученика в определенный день на уроке по каждому предмету из обязательного курса обучения данного ученика. Данные об успеваемости представляют собой информацию об оценках, полученных учеником по каждому из его обязательных предметов в процессе обучения. Контактные данные участников работы представляют собой такую информацию как электронная почта или номер телефона, необходимые для взаимодействия между сотрудниками учениками и родителями во время учебного процесса.

Выходящие данные представлены на контекстной диаграмме в виде уведомлений родителям. Уведомления родителям представляют собой

сообщения, которые рассылаются родителям по результатам учебного процесса их детей в виде сообщений по электронной почте или sms-сообщений.

Для того, чтобы оценить полную картину деятельности, направленной на контроль посещаемости и успеваемости учеников средней школы, естественно недостаточно лишь глобальной формулировки. Необходимо рассмотреть детали осуществляемой деятельности. Для этого было произведена декомпозиция описанной выше контекстной диаграммы на более детальные компоненты.

Декомпозиция контекстной диаграммы представлена на рисунке 1.2.

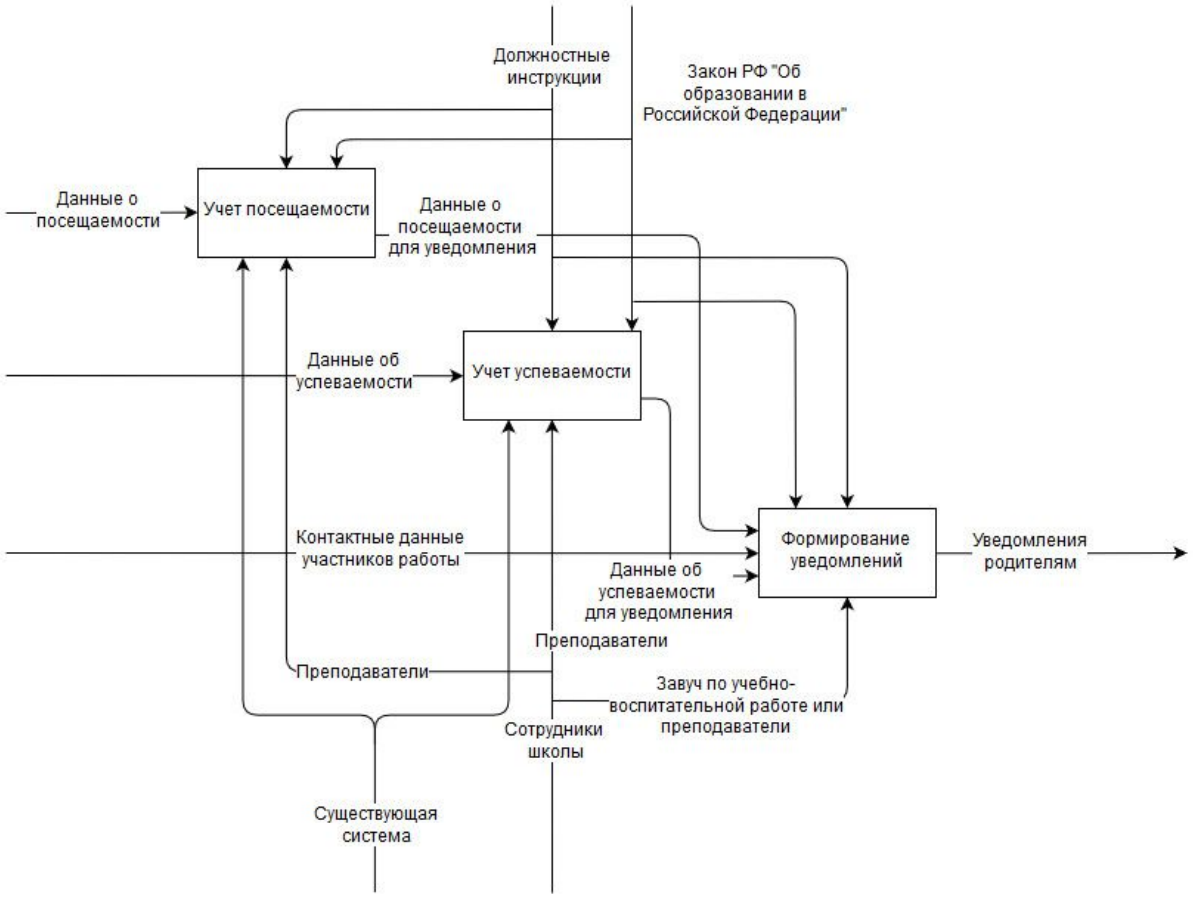

Рисунок 1.2 - Декомпозиция контекстной диаграммы

Как видно на представленной выше диаграмме, стрелка механизма, символизирующая сотрудников школы разветвляется на механизмы с различным названием, а именно на преподавателей и завуча по учебно-воспитательной работе.

На декомпозиции контекстной диаграммы представлены три блока, обозначающие обособленные процессы, такие как учет посещаемости, учет успеваемости и формирование уведомлений.

Учет посещаемости, представленный на декомпозиции контекстной диаграммы отдельным блоком, выполняется в виде фиксации информации о присутствии или отсутствии ученика на конкретном занятии в каждый день его проведения. Стрелки контроля данного блока представлены в виде должностных инструкций и закона РФ "Об образовании в Российской Федерации". Стрелки механизма представлены в виде существующей системы и ответвления от стрелки сотрудники стрелки преподаватели. Существующая система является механизмом, как именно в ней ведется учет успеваемости, а в ее базе данных хранятся все данные о посещаемости. Входными данными соответственно являются данные о посещаемости учеников, а выходными данными являются данные о посещаемости учеников для уведомлений, то есть данные о посещаемости, обработанные существующей используемой системой.

Отдельным процессом был выделен учет успеваемости. Контролем его блока являются должностные инструкции и закон РФ "Об образовании в Российской Федерации", а механизмом существующая система и преподаватели. Структура поступающей в этот блок информации родственна со структурой информации, поступающей в блок учета посещаемости, следовательно, схематичное представление данных процессов во многом аналогично. Входными данными этого блока являются данные об успеваемости, а выходными, следовательно, данные об успеваемости для

уведомлений, то есть данные об успеваемости, обработанные используемой системой.

Заключительным процессом при выполнении контроля посещаемости и успеваемости учеников средней школы является формирование уведомлений, представленное отдельным блоком. Контролем этого процесса является должностные инструкции и закон РФ "Об образовании в Российской Федерации", а механизмом завуч по учебно-воспитательной работе или преподаватели средней школы. Из этого очевидно, что существующие системы не обеспечивают функционала для оперативного формирования уведомлений. Входными данными являются данные о посещаемости для уведомлений, приходящие из процесса учета посещаемости, и данные об успеваемости для уведомлений, приходящие из процесса учета успеваемости. Выходными данными являются уведомления родителям.

Для понимания всех деталей работы выделенных процессов проведены более детальные исследования каждого из них. Следовательно, были составлены диаграммы декомпозиций.

Они были изображены в методологии DFD, так как данная нотация более применима к отображению детальных процессов, чем предназначенная для отображения глобальных процессов методология IDEF0. В связи использованием методологии DFD стрелки контроля, механизма и входных данных уже не будут разделяться на данные категории, а будут просто являться входными стрелками. Появляется возможность использования такого элемента как хранилище, которое может символизировать базу данных существующей используемой автоматизируемой системы.

Декомпозиция блока учета посещаемости представлена на рисунке 1.3.

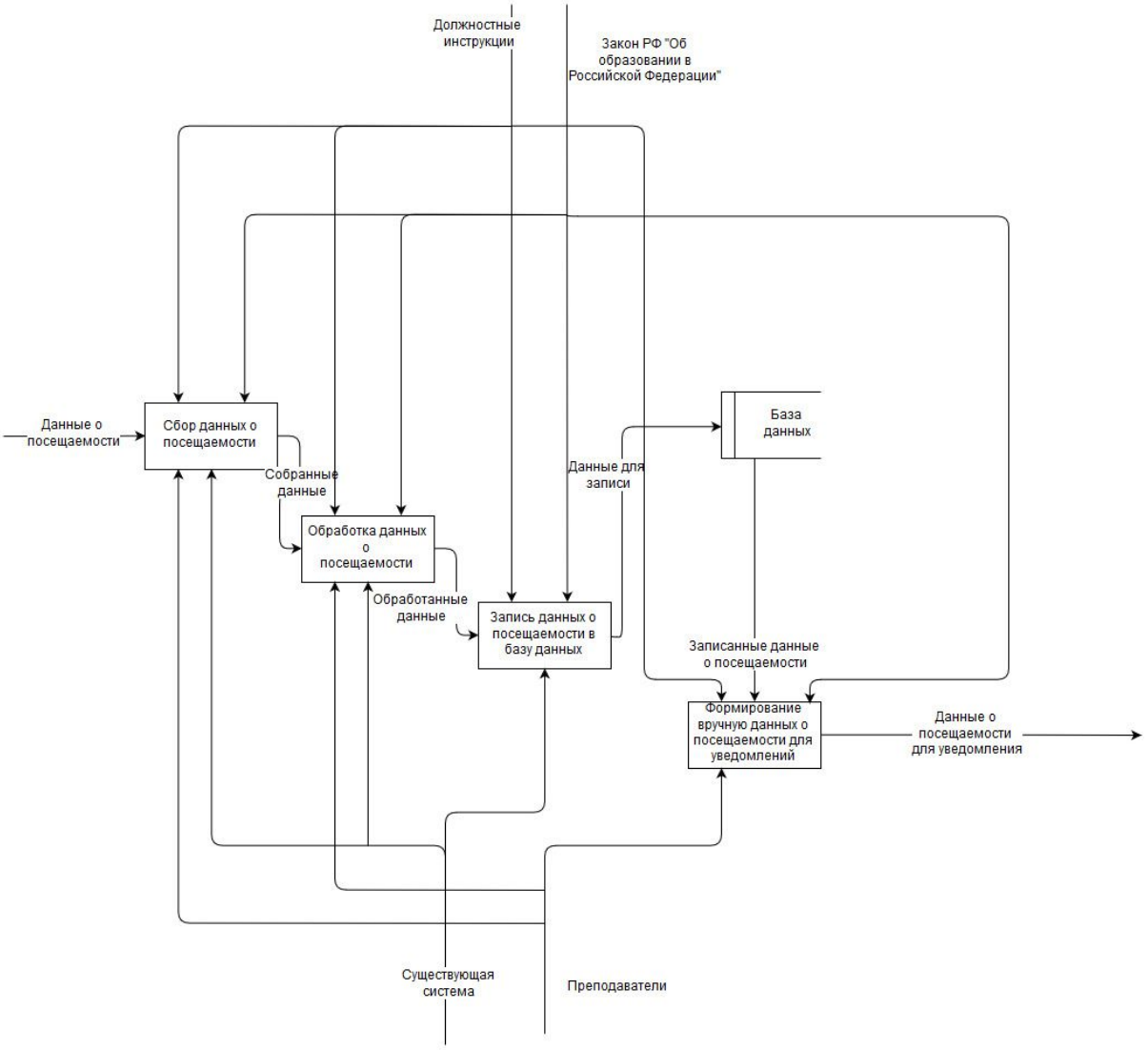

Рисунок 1.3 - Декомпозиция блока учета посещаемости

Учет посещаемости состоит из следующих подпроцессов: сбор данных о посещаемости, обработка данных о посещаемости, запись данных о посещаемости в базу данных, формирование вручную данных о посещаемости для уведомлений. Во время исполнения процесса учета посещаемости используется база данных существующей системы.

Сбор данных о посещаемости представлен отдельным блоком на диаграмме. Сбор данных заключается во вводе преподавателями данных о посещаемости в существующую систему. Входными стрелками представлены

все необходимые компоненты для выполнения процесса, а именно должностные инструкции и закон РФ "Об образовании в Российской Федерации", существующая автоматизированная система и преподаватели, данные о посещаемости. Выходные представлены в виде собранных данных.

Обработка данных о посещаемости представляет собой структурирование данных автоматизированной системой в соответствии с параметрами, задаваемыми преподавателями. Входными стрелками являются должностные инструкции и закон РФ "Об образовании в Российской Федерации", существующая система и преподаватели, собранные данные. Выходные данные являются обработанные данные.

Блок записи данных о посещаемости в базу данных используется для формирования запросов на добавление, изменение и удаление в базу данных. Входные стрелки представлены в виде должностных инструкций и закона РФ "Об образовании в Российской Федерации", существующей системы и обработанных данных. На выходе представлены данные для записи.

Формирование вручную данных о посещаемости для уведомлений заключается в использовании преподавателями сохраненных данных, чтобы самостоятельно проанализировать посещаемость каждого ученика и затем решить кому из родителей следует отправить уведомления, и составить соответствующий список. Входные стрелки в блок: должностные инструкции и закон РФ "Об образовании в Российской Федерации", преподаватели, записанные данные о посещаемости. Выходные стрелки: данные о посещаемости для уведомлений.

Далее рассмотрена декомпозиция учета успеваемости во многом аналогичная декомпозиции учета посещаемости. Диаграмма декомпозиции блока учета успеваемости представлена на рисунке 1.4.

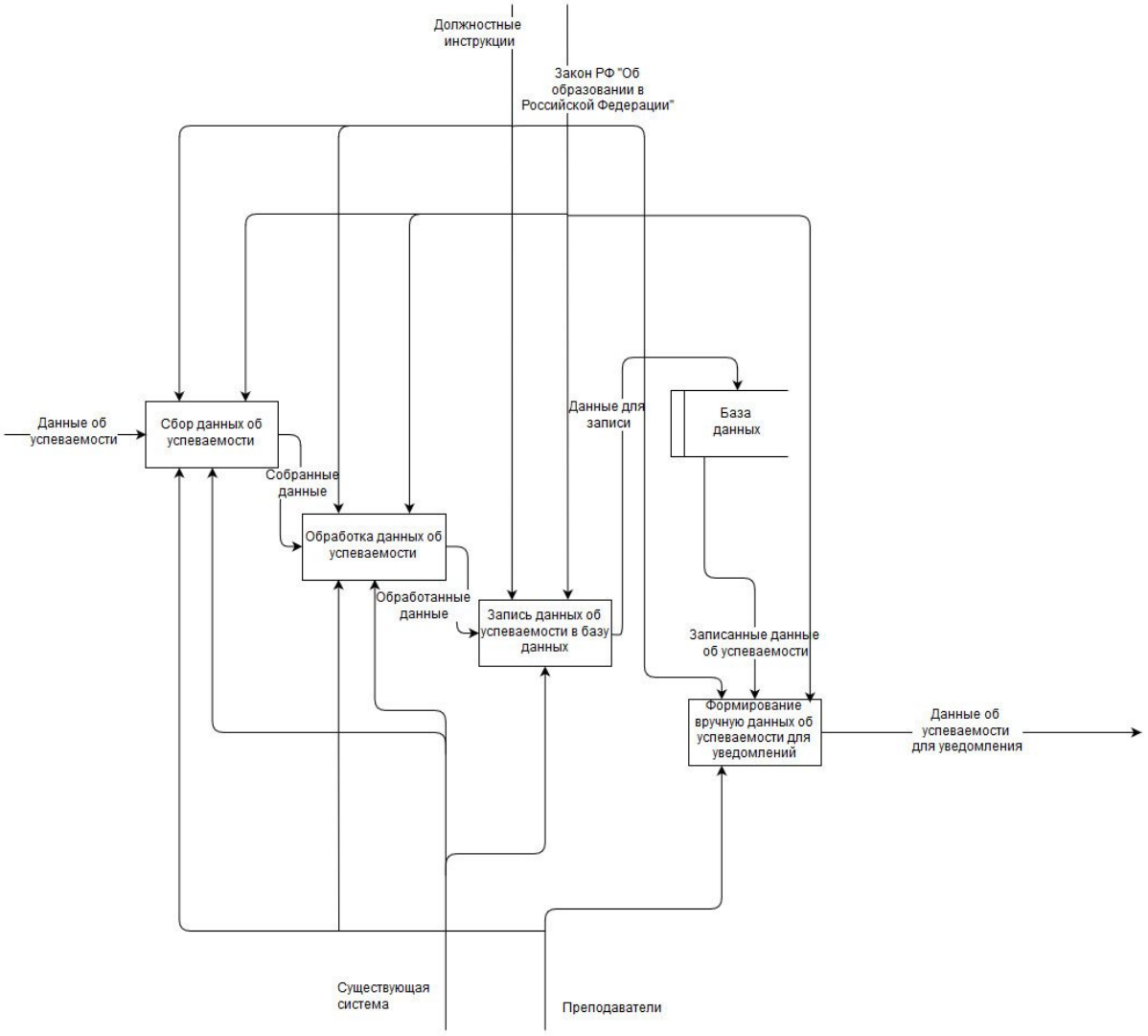

Рисунок 1.4 - Декомпозиция блока учета успеваемости

Процесс учета успеваемости состоит из следующих подпроцессов: сбор данных об успеваемости, обработка данных об успеваемости, запись данных об успеваемости в базу данных, формирование данных об успеваемости для уведомлений. Диаграмма также содержит структурный элемент, представляющий собой хранилище, используемый для описания базы данных существующей системы.

Сбор данных об успеваемости - это подпроцесс, исполняемый в виде ввода преподавателями данных об оценках каждого ученика по каждому из

изучаемых им предметов в каждый из дней, в которые данный предмет проводится. Следовательно, входными компонентами данного блока являются должностные инструкции и закон РФ "Об образовании в Российской Федерации", существующая система, преподаватели, данные об успеваемости, а выходными компонентами будут являться собранные данные.

Обработка данных об успеваемости - это подпроцесс, который заключается в структурировании собранных данных существующей системой в соответствии с параметрами, задаваемыми преподавателями. Входные стрелки данного блока представлены в виде должностных инструкций и закона РФ "Об образовании в Российской Федерации", преподавателей и существующей системы, собранных данных, приходящих из блока сбор данных об успеваемости, а выходные стрелки описаны как обработанные данные.

Запись данных об успеваемости в базу данных - это подпроцесс формирования существующей системой запросов на добавление, изменение и удаление в соответствии со структурой обработанных на предыдущем шаге данных, его входными составляющими являются должностные инструкции и закон РФ "Об образовании в Российской Федерации", существующая система и обработанные данные, а выходными составляющими являются, данные для записи.

Формирование вручную данных об успеваемости для уведомлений - это подпроцесс, где преподаватели на основании сохраненных существующей системой данных принимают решение о необходимости уведомления родителей учеников по результатам их успеваемости. Входные стрелки блока: должностные инструкции и закон РФ "Об образовании в Российской Федерации", преподаватели и записанные данные об успеваемости, выходные стрелки: данные об успеваемости для уведомлений.

Для окончательно полного представления о контроле посещаемости и успеваемости учеников средней школы “КАК ЕСТЬ” далее подробно описан процесс формирования уведомлений.

В отличие от предыдущих диаграмм декомпозиций на диаграмме, представленной ниже отсутствует входящая с верхнего уровня стрелка, обозначающая существующую автоматизированную систему, так как уведомления формируются полностью вручную преподавателями либо завучем по учебно-воспитательной работе.

Процесс формирования уведомлений состоит из следующих подпроцессов: запись данных об успеваемости для уведомлений, запись данных о посещаемости для уведомлений, запись контактов родителей, которых необходимо уведомить, отправка вручную sms-сообщений родителям, отправка вручную email-рассылок родителям. На диаграмме использован структурный элемент “хранилище”, символизирующий бумажный журнал уведомлений, который ведется с целью фиксации истории коммуникаций сотрудников средней школы с родителями.

Диаграмма декомпозиции блока формирования уведомлений представлена на рисунке 1.5.

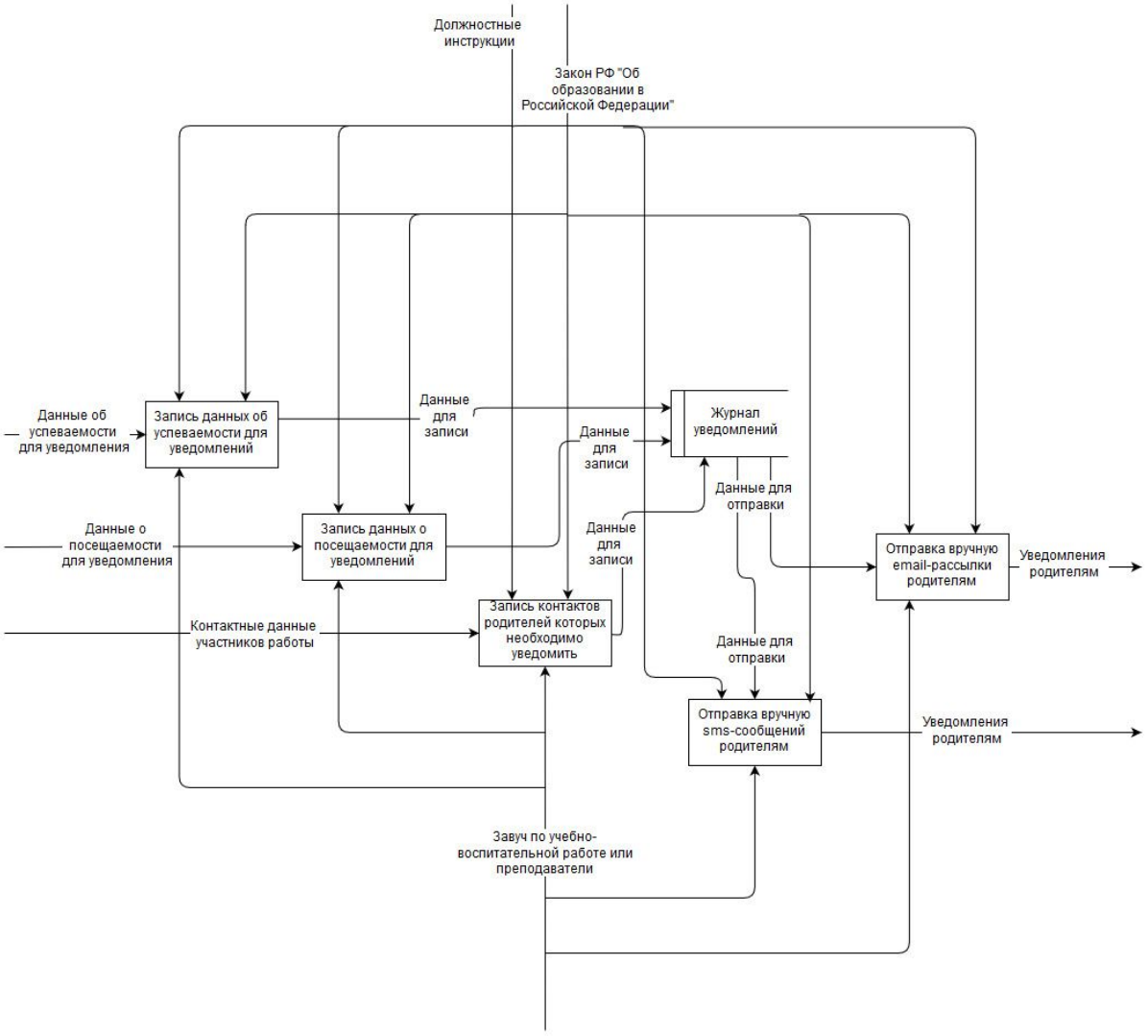

Рисунок 1.5 - Диаграмма декомпозиции блока формирования уведомлений

Отдельным блоком представлен процесс записи данных об успеваемости для уведомлений. Он заключается в записи преподавателями в бумажный журнал информации родителях, которым нужно отправить уведомление, причины уведомления. Входные компоненты: данные о посещаемости для уведомлений, должностные инструкции и закон РФ "Об образовании в Российской Федерации", завуч по учебно-воспитательной работе или преподаватели. Выходные компоненты: данные для записи.

Запись данных о посещаемости для уведомлений существует как отдельный подпроцесс, который заключается во внесении завучем по учебно-воспитательной работе или преподавателями в журнал уведомлений списка родителей, которым нужно отправить уведомление в связи с посещаемостью их детей, причины уведомления. Следовательно, входящие стрелки данного блока выглядят следующим образом: должностные инструкции и закон РФ "Об образовании в Российской Федерации", завуч по учебно-воспитательной работе или преподаватели, данные о посещаемости для уведомлений, а выходящие стрелки формируются как данные для записи.

Запись контактов родителей, которых необходимо уведомить - это подпроцесс внесения новых контактных данных в журнал уведомлений. Его входящие компоненты заключаются в должностных инструкциях и законе РФ "Об образовании в Российской Федерации", завуче по учебно-воспитательной работе или преподавателях, а выходящие заключаются в данных для записи.

Отправка вручную sms-сообщений родителям заключается в формировании и рассылке вручную sms-сообщений родителям на основании данных журнала уведомлений. На входе данного блока отображены должностные инструкции и закон РФ "Об образовании в Российской Федерации", завуч по учебно-воспитательной работе или преподаватели, данные для отправки, а на выходе данного блока отображены уведомления родителям.

Отправка вручную email-рассылки родителям - подпроцесс уведомления вручную родителей с помощью электронной почты. Входящие стрелки: должностные инструкции и закон РФ "Об образовании в Российской Федерации", завуч по учебно-воспитательной работе или преподаватели, данные для отправки. На выходе отображены уведомления родителям.

В данном подразделе выполнено описание структуры работы системы “КАК ЕСТЬ”. На основании полученной информации возможно выполнять характеристику выявленных недостатков.

1.2.2 Характеристика выявленных недостатков

Зная полную картину того, как осуществляется контроль успеваемости и посещаемости учеников средней школы, была составлена характеристика выявленных недостатков. Данная характеристика необходима для составления в дальнейшем задач для автоматизации. Ниже сформулированы основные выявленные недостатки.

– отсутствие модуля автоматического формирования сообщений;

– коммуникации с родителями контролируются только с помощью бумажных источников (полное отсутствие централизованной истории переписки);

– структура существующих модулей контроля успеваемости и посещаемости не комфортна для формирования уведомлений;

– существующие системы не интерактивны.

Следует описать основные достоинства и недостатки существующих систем на конкретных примерах для максимальной очевидности отсутствия в них необходимого для комфортной коммуникации участников учебного процесса и интерактивного ведения учета успеваемости и посещаемости функционала. В таблице 1.1 приведены преимущества и недостатки некоторых типовых систем управления средней школой.

Таблица 1.1 - Характеристика преимуществ и недостатков существующих систем

Название системы	Преимущества	Недостатки
ИСОУ "Виртуальная школа"	Автоматизация основных управленческих процессов в сфере начального общего, основного общего, среднего общего образования; Осуществление аналитических и статистических, мониторинговых исследований по различным срезам базы данных; Электронный документооборот в сфере образования.	Отсутствие интерактивного взаимодействия с родителями учеников; Отсутствие у сотрудника возможности отправить сообщение родителю из системы;
Школьный офис	Обеспечивает эффективное ведение кадрового учёта и учёта учащихся; Позволяет автоматизировать делопроизводство, документооборот и учебное планирование; Помощь в формировании штатного расписания и тарификации, подготовке стандартной и нестандартной отчетности для передачи в контролирующие организации и для внутреннего пользования; Позволяет управляющим организациям объединить все информационные потоки в единую сеть, повысив эффективность управления всей системой образования в целом; Реальный способ вывести процесс управления школой и	Отсутствие автоматической аналитики успеваемости и посещаемости учеников для предложения сотруднику автоматически отправить уведомление. Нет возможности интерактивной коммуникации сотрудников, учеников и родителей.

	всеми образовательными учреждениями на качественно новый уровень, сделав максимально эффективным;	
1С: Образование	Позволяет организовать учебный процесс на основе активного использования ЦОР; Обеспечивает поддержку различных видов учебной деятельности как в классе, так и дома; Возможность настройки на различные уровни оснащения компьютерной техникой и формы организации образовательных учреждений; Использует открытые стандарты хранения, описания и передачи ресурсов; Обеспечивает синхронизацию данных с программным комплексом «1С:Управление школой»; Работает с различными веб-браузерами под управлением Windows и GNU/Linux.	Не обеспечивает отправку уведомлений напрямую из системы; Отсутствие возможности коммуникации участников учебного процесса.

Таким образом становится очевидно, что существующие системы не удовлетворяют требованиям относительно описанного функционала и необходимо подробное описание задач, выполнение которых направлено на устранение выявленных недостатков.

В данном подразделе была описана характеристика выявленных недостатков, относительно которых возможно составление задач для их устранения и дальнейшее их выполнение.

1.2.3 Постановка задач для устранения выявленных недостатков

Задачи, определенные для устранения выявленных недостатков, определяют дальнейшее развитие работы по автоматизации деятельности. Постановка задач крайне важна со стратегической точки зрения. Краткая формулировка основных задач для устранения выявленных недостатков представлена ниже:

1) организовать комфортный функционал для взаимодействия всех участников учебного процесса;

2) внедрить автоматизированный модуль формирования сообщений;

3) сделать возможным хранение централизованной истории переписки.

Первая задача должна быть выполнена для устранения таких недостатков, как ручное формирование уведомления, отсутствие возможности ведения свободной переписки между участниками учебного процесса.

Вторая задача важна для устранения недостатков отсутствия аналитики данных об успеваемости и посещаемости для возможности автоматической отправки уведомлений, необходимости абсолютно самостоятельного принятия решения о необходимости уведомления.

Третья задача поставлена для решения проблемы отсутствия хранения истории переписки в электронном виде (ведение только бумажного журнала уведомлений).

В данном разделе была описана структура ведения деятельности “КАК ЕСТЬ”, характеристика выявленных недостатков, постановка задач для устранения этих недостатков.

1.3 Характеристика возможных способов решения задач

Поставленные задачи позволяют подобрать способы их решения и выбрать из них максимально оптимальный для реализации. Далее приведены возможные способы решения поставленных задач:

1) приобретение новой системы контроля успеваемости и посещаемости - преимущества данного способа заключаются в отсутствии необходимости новой разработки, а недостаток в том, что ни одна готовая система не будет удовлетворять всех потребностей;

2) внедрение в существующую систему готового модуля уведомлений - данный способ хорош меньшими расходами, чем предыдущий способ, однако плох слишком высокой вероятностью конфликтов реализации между существующей системой и новым модулем;

3) разработка модуля уведомлений для существующей системы - данной способ в отличие от предыдущего имеет меньшую вероятность технических конфликтов, однако требует дополнительной разработки нового функционала;

4) разработка новой системы, удовлетворяющей потребностям учреждений - данный способ не будет иметь значительных технических конфликтов, так как система однородна, но потребует разработки с нуля.

Наиболее удовлетворяющим все условия данной работы является последний способ, так как в рамках данной работы наиболее важным критерием является получение стабильной системы с полностью удовлетворяющим пользователей функционалом.

В данном разделе описана аналитика существующих процессов, выявленные недостатки и способы их устранения.

2 Информационное обеспечение задачи

2.1 Информационная модель и ее описание

Ранее было описано каким образом осуществляется контроль посещаемости и успеваемости в стандартной средней школе, были выявлены конкретные недостатки и предложены способы их устранения, а также был выбран наиболее подходящий способ решения проблемы.

Следует описать как конкретно будут выполнятся выбранные автоматизируемые процессы после разработки информационной системы. Наиболее подходящим способом продемонстрировать различия и изменения является графическое изображение на диаграммах. Наиболее глобальное графическое отображение деятельности “КАК ДОЛЖНО БЫТЬ” представлено на рисунке 2.1.

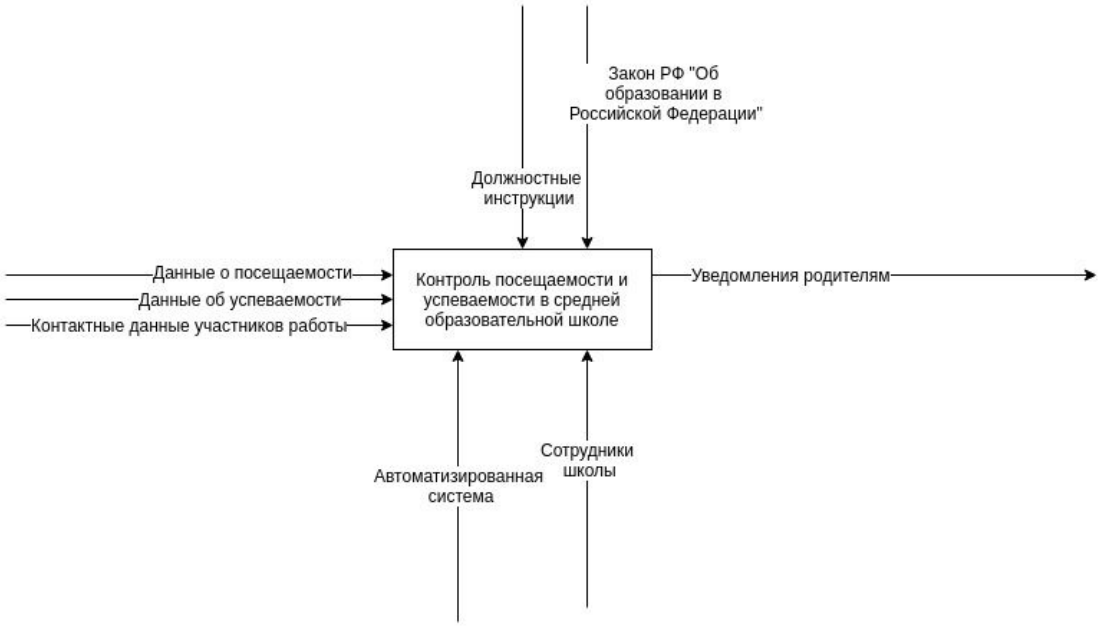

Рисунок 2.1 - Контекстная диаграмма “КАК ДОЛЖНО БЫТЬ”

Входными данными на контекстной диаграмме являются данные о посещаемости, данные об успеваемости, контактные данные участников работы, контроль осуществляется с помощью закона РФ "Об образовании в Российской Федерации" и должностных инструкций, на выходе контекстной диаграммы - уведомления родителям. Механизмом являются сотрудники школы и новая автоматизированная информационная система.

Для полного понимания новой методологии осуществления деятельности необходимо представить и описать декомпозированные представления глобального отображения осуществляемых процессов. Декомпозиция контекстной диаграммы представлена на рисунке 2.2.

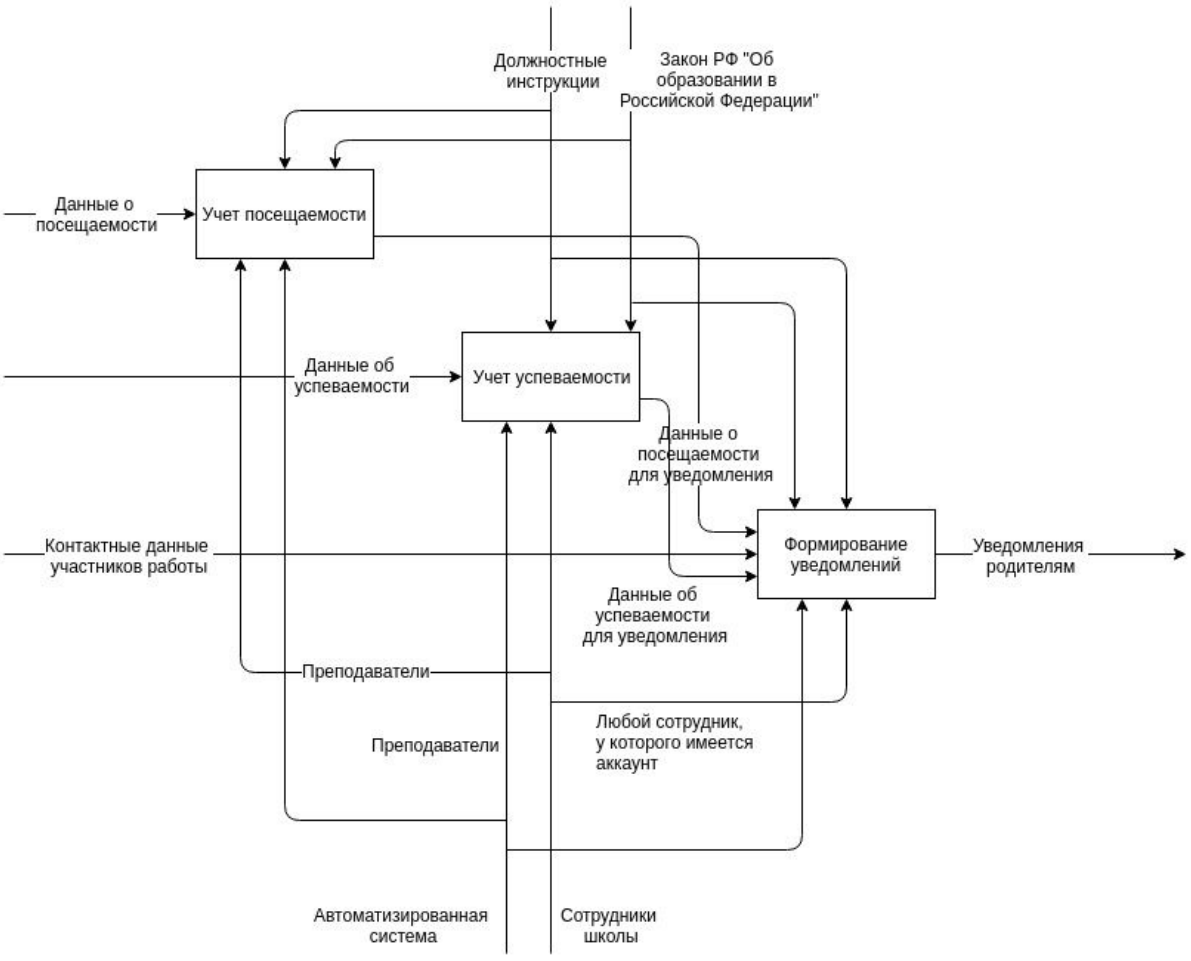

Рисунок 2.2 - Декомпозиция контекстной диаграммы

Контекстная диаграмма декомпозируется на три блока. Над всеми ними осуществляется контроль с помощью закона РФ "Об образовании в Российской Федерации" и должностных инструкций. Первый из них представляет такой обособленный участок деятельности как контроль посещаемости. Его входными данными являются данные о посещаемости, а механизмом автоматизированная система и преподаватели школы. На выходе из него получаются данные о посещаемости для уведомления, то есть данные специально подготовленные системой для отправки уведомлений.

Вторым блоком декомпозиции контекстной диаграммы блок контроля успеваемости. Данные об успеваемости символизируют его входную информацию. Механизмом также являются автоматизированная система и преподаватели школы. На выходе этого блока представлены данные об успеваемости для уведомлений.

Заключающим блоком является блок формирования уведомлений. Его входная информация представлена в виде данных о посещаемости для уведомлений, данных об успеваемости для уведомлений, контактных данных участников работы. Механизм представлен в виде автоматизированной системы и любого сотрудника, у которого имеется аккаунт. Итоговой выходной информацией являются уведомления родителям.

Декомпозиции контекстной диаграммы недостаточно для полноценного понимания осуществляемых процессов. Для полноценного обзора того, как должны осуществляться подобные процессы необходимо декомпозировать их на более детальные диаграммы декомпозиции. В отличие от двух предыдущих диаграмм, выполненных в нотации IDEF0, подходящей для отображения глобальных процессов, диаграммы представленные далее выполнены по методологии DFD. На рисунке 2.3 представлена диаграмма декомпозиции блока контроля посещаемости.

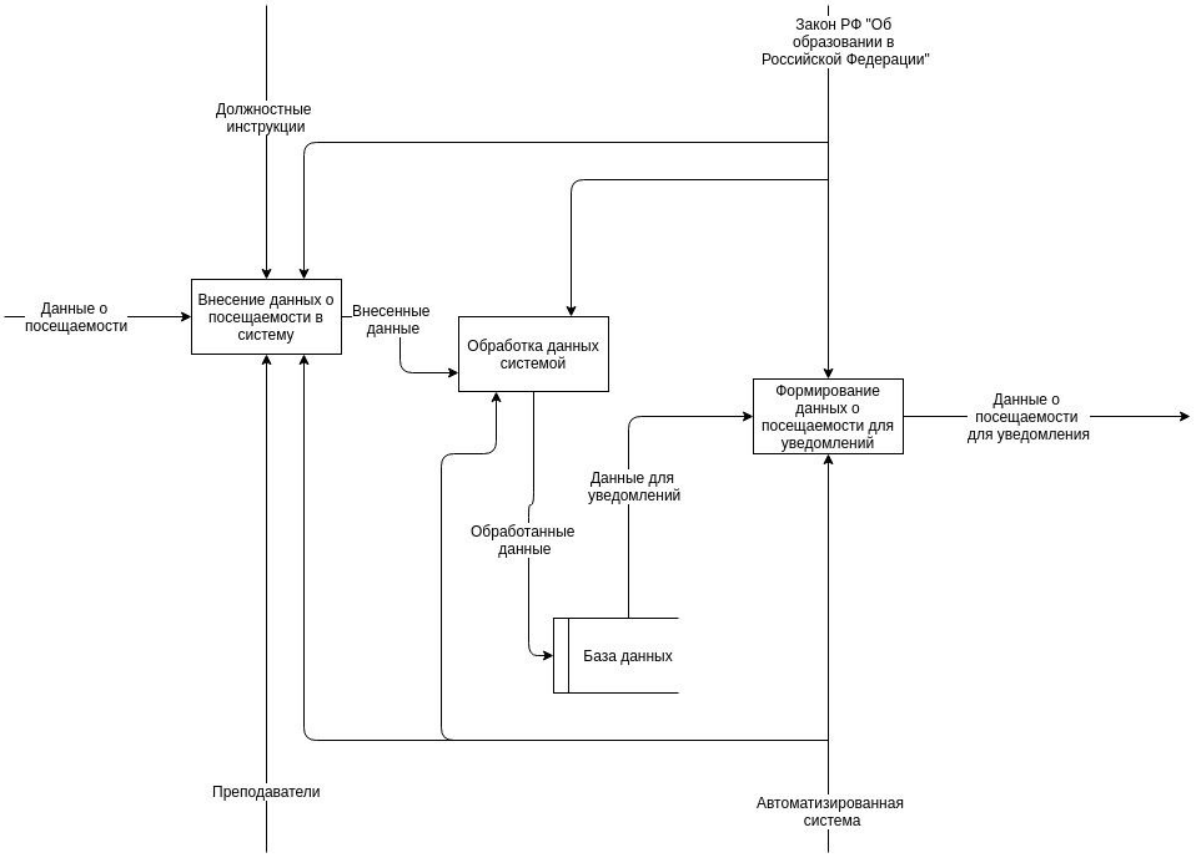

Рисунок 2.3 - Декомпозиция блока “Контроль посещаемости”

На данной диаграмме представлено три блока-процесса и одно хранилище, символизирующее базу данных системы. Первым действием является внесение данных о посещаемости в систему. Его механизмами являются преподаватели и автоматизированная система, входящая информация - данные о посещаемости, контроль осуществляется с помощью должностных инструкций и закона РФ "Об образовании в Российской Федерации". На выходе у данного представлены внесенные данные.

Функционально следующим за внесением данных о посещаемости в систему является блок обработки данных системой. Его входящая информация представлена в виде внесенных данных, контролем является закон РФ "Об образовании в Российской Федерации", на выходе для записи в базу данных -

обработанные данные. Механизмом является только информационная система, так обработка данных осуществляется автоматически без необходимости пользователю осуществлять дополнительные действия.

Последним действием в процессе учета посещаемости является формирование данных о посещаемости для уведомлений. Входные данные процесс получает из базы данных в виде данных для уведомлений, контроль осуществляется законом РФ "Об образовании в Российской Федерации". Итоговыми выходными данными процесса контроля посещаемости, выходящими из блока формирования данных о посещаемости для уведомлений, являются данные о посещаемости для уведомлений. Механизмом, как и в предыдущем блоке, является автоматизированная система и не является пользователь.

Таким образом большая часть действий по учету посещаемости учеников выполняет автоматизированная информационная системы. Это является очень большим преимуществом относительно способов выполнения процессов контроля посещаемости на стадии “КАК ЕСТЬ”.

Процесс контроля успеваемости также необходимо представить в виде более подробной декомпозиции, так как он также является основополагающим в работе интерактивной системы контроля посещаемости и успеваемости учеников стандартной средней школы.

Диаграмма декомпозиции процесса контроля успеваемости учеников стандартной средней школы представлен на рисунке 2.4.

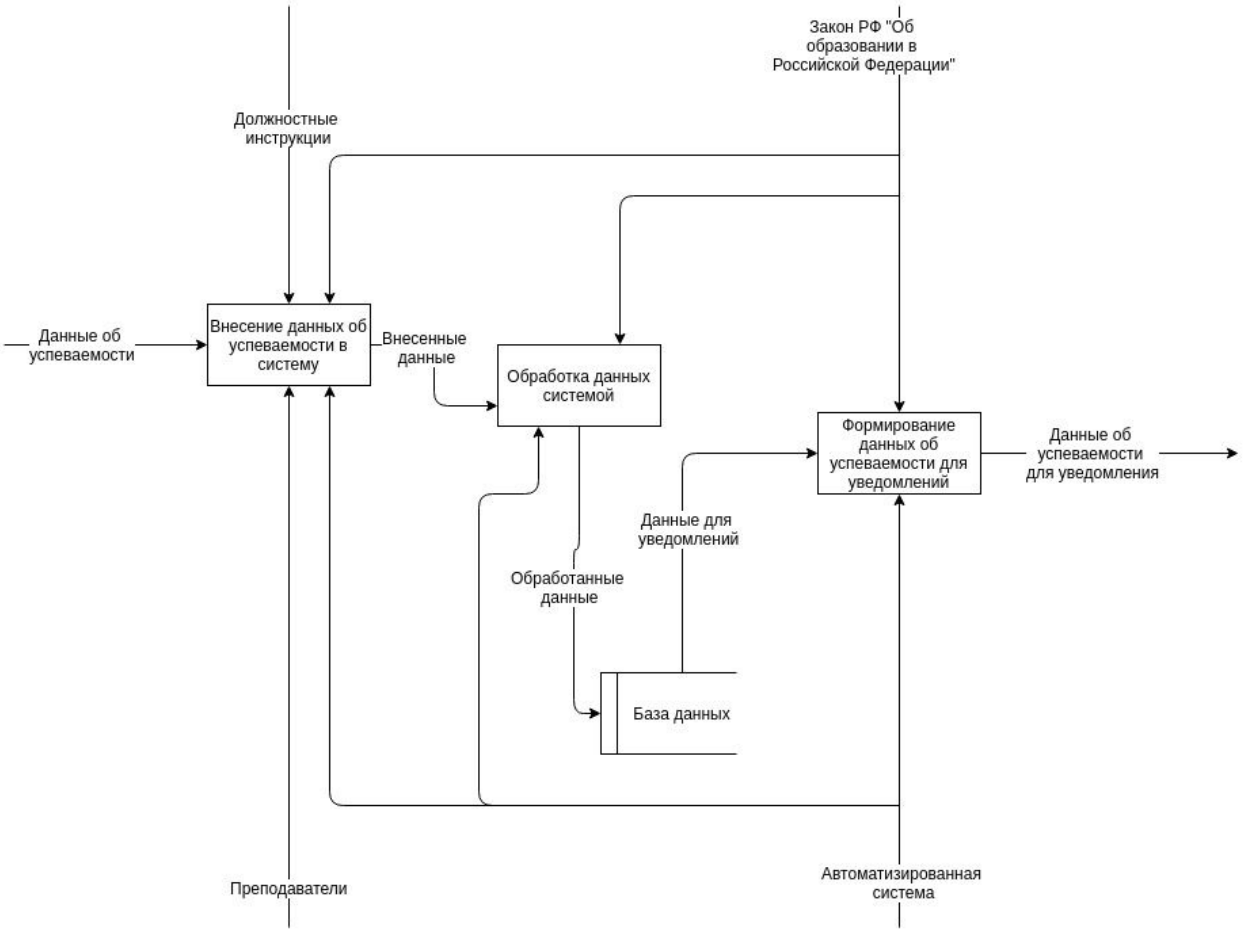

Рисунок 2.4 - Диаграмма декомпозиции блока контроля успеваемости

Процесс контроля успеваемости во многом аналогичен процессу контроля посещаемости. Он также представлен в виде трех блоков-процессов и одного хранилища, представляющего собой базу данных системы.

Первый подпроцесс представлен в виде внесения данных об успеваемости в систему. Его входная информация символизирует данные об успеваемости, Контроль выполнен в виде закона РФ "Об образовании в Российской Федерации" и должностных инструкций. Из данного блока выходят внесенные данные. Механизмами являются преподаватели школы и автоматизированная информационная система.

Второй подпроцесс представлен в виде обработки данных системой, обозначающей подготовку данных для записи в базу данных. Его вход представлен в виде внесенных данных, контроль осуществляется законом РФ "Об образовании в Российской Федерации", выходные данные - обработанные данные. Механизм представлен только в виде автоматизированной системы.

Третий и завершающий подпроцесс символизирует формирование данных об успеваемости для уведомлений. Его входные данные формируются из базы данных в виде данных для уведомлений, а выходные данные являются окончательными выходными данными описываемого процесса и обозначены на данные об успеваемости для уведомления. Контроль выполнен законом РФ "Об образовании в Российской Федерации". Механизм представлен только в виде автоматизированной системы.

Завершающим процессом контекстной диаграммы является формирование уведомлений. Декомпозиция блока формирования уведомлений представлена в виде четырех подпроцессов и одного хранилища, обозначающего базу данных системы уведомлений.

Следует отметить, что хранилища, представленные на диаграммах декомпозиций процессов контроля посещаемости и контроля успеваемости, не являются общей базой данных с хранилищем, изображенном на диаграмме декомпозиции блока-процесса формирования уведомлений. Такое распределение используемых данных необходимо для достижения микросервисной архитектуры в системе, так как данные процессы выполняют абсолютно различный функционал и используют сущности, которые имеет смысл разделить на две различных базы данных.

Декомпозиция блока формирования уведомлений представлена на рисунке 2.5.

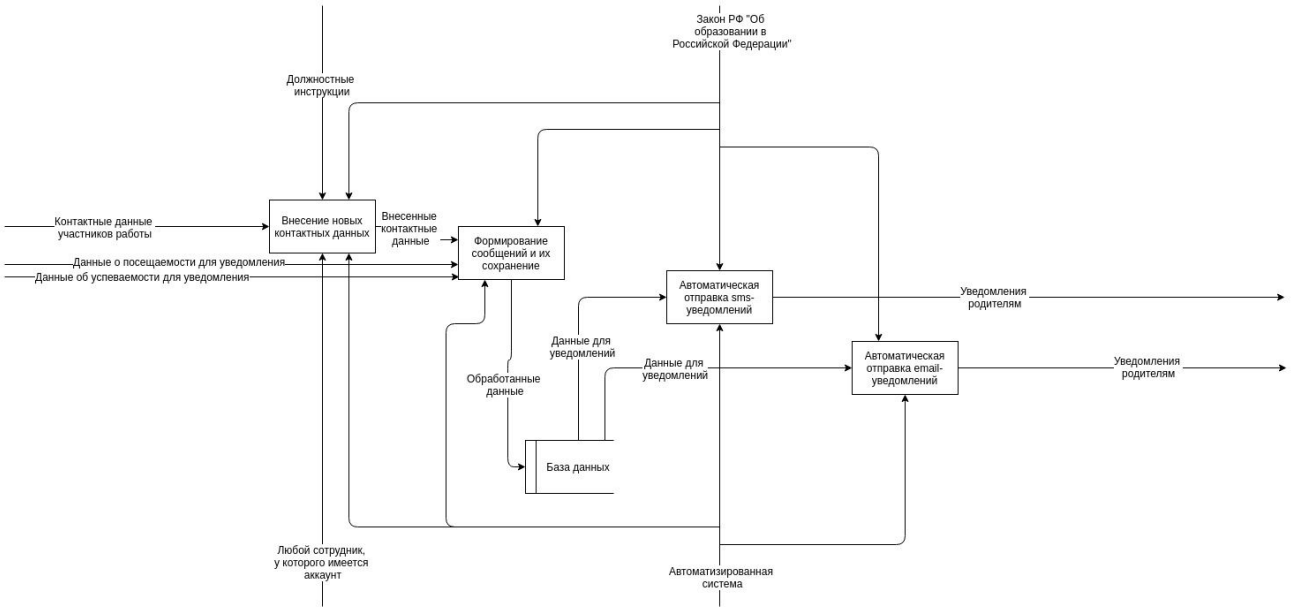

Рисунок 2.5 - Декомпозиция блока формирования уведомлений

Функционально начальным блоком является внесение новых контактных данных. Следовательно, его входными данными являются контактные данные участников работы, а выходными внесенные контактные данные. Данное действие может выполнять любой сотрудник, у которого имеется аккаунт, следовательно, он и является механизмом. Также механизмом является автоматизированная система. Контроль выполняют закон РФ "Об образовании в Российской Федерации" и должностные инструкции.

Далее структурно следует формирование сообщений и их сохранение. Входными данными являются данные о посещаемости для уведомления, данные об успеваемости для уведомления, внесенные контактные данные. Выходной информацией для записи в базу данных являются обработанные данные. Контролем является только закон РФ "Об образовании в Российской Федерации", а механизмом - автоматизированная информационная система.

Структурно завершающими итерацию работу системы являются блоки-процессы автоматической отправки sms-уведомлений и автоматической отправки email-уведомлений. Их входными данными являются данные для уведомлений из базы данных, а выходными - уведомления родителям. Контроль исполняет закон РФ "Об образовании в Российской Федерации", а механизмом является автоматизированная система.

В данном разделе описана информационная модель. На диаграммах “КАК ДОЛЖНО БЫТЬ” показана работа системы, помогающая автоматизировать процесс контроля посещаемости и успеваемости учеников средней школы и уведомления родителей при необходимости. Как видно из графического отображения и описания подобный подход облегчит работу пользователя, избавит его от рутинных действий.

2.2 Характеристика интегрируемых систем коммуникации

После определения верных способов работы системы контроля посещаемости и успеваемости учеников средней школы следует описать выбор средств технической реализации. Так как разработанная система интерактивна, следует описать какие средства коммуникации ею используются.

Как представлено на диаграммах выше, уведомление родителей о посещаемости и успеваемости учеников происходит по двум каналам связи, а именно sms-уведомления и email-рассылка. Следует подробнее описать выбранные технологии.

Электронная почта, коротко называемая email, используется для отправки и получения сообщений, также называемых письмами по аналогии с

обыкновенной почтой, использующей бумажной почтой, пользователями чаще всего Интернет или какой-либо другой компьютерной сети.

От обыкновенной почты электронная почта унаследовала не только основные определения, такие как письмо, конверт, ящик, почта, вложение, но и высокую надежность, но при этом отсутствие гарантии доставки, временные задержки доставки, а также достаточно простой функционал для использования даже неопытными пользователями персонального компьютера.

Для полноты понимания функциональности работы автоматизируемой системы следует перечислить основные достоинства и недостатки электронной почты.

Основные достоинства электронной почты заключаются в следующем:

1) адреса электронной почты являются легкими для запоминания и восприятия;

2) возможность отправки и получения файлов различного типа наравне с простым текстом, также существует возможность отправки форматированного текста;

3) сервера непосредственно обращаются друг к другу, таким образом определяется некоторая независимость серверов;

4) вероятность того, что письмо будет доставлено крайне высока, таким образом можно считать, что технология крайне надежна;

5) интерфейс взаимодействия между электронной почтой и пользователем, а также между электронной почтой и другими программными средствами крайне прост;

6) скорость получения и отправки писем считается очень высокой среди современных технологий обмена сообщениями.

Как и у любых других технологий у электронной почты имеются свои недостатки. Следует их перечислить:

1) среди полученных писем могут присутствовать вирусные и рекламные рассылки, также называемые спам;

2) во время сбоев работы серверов возможны длительные задержки доставки и получения сообщений;

3) отдельно для различных пользователей могут накладываться ограничения на длину сообщения;

4) также в зависимости от аккаунта пользователя может быть ограничена память, выделенная для хранения определенного количества сообщений.

Для начала работы с электронной почтой пользователю достаточно зарегистрироваться в одном из подобных сервисов, что является достаточно необременительной процедурой, которая легко может быть выполнена даже начинающим пользователем.

SMTP является принятым во всем мире как стандартный протоколом пересылки электронных писем, для определения правил пересылки почты в своей реализации согласно стандарту он использует DNS. После того как отправленное электронное письмо оказывается на сервере, он начинает выполнять временное или постоянное хранение полученной почты.

SMS - аббревиатура, обозначающая “short message service” или в переводе на русский язык “служба коротких сообщений”. Данная технология позволяет обмениваться короткими текстовыми сообщениями с использованием сотового телефона. SMS входит в стандарт сотовой связи.

При получении подобного сообщения отправитель имеет возможность получить уведомление, процесс доставки сообщения конечному пользователю зачастую занимает не более десяти секунд. Имеется возможность отправить такое сообщение конечному пользователю, телефон которого на данный момент отключен или находится вне зоны действия сети. Сообщение будет

окончательно доставлено, как только необходимый сотовый телефон будет способен его принять. Таким образом отправитель имеет возможность узнать, в какой момент получатель появился в сети, получив в нужный момент сообщение о доставке сообщения отправителю.

Если во время отправки сообщения получатель осуществляет телефонный разговор, письмо все равно будет доставлено. Существует возможность отправки файлов аудиозаписей и тому подобного при использовании EMS, который является расширенной версией SMS. Передача такого сообщения не нагружает сотовую сеть.

В данном разделе описаны основополагающие каналы связи, позволяющие разработанной системе работать в интерактивном режиме, рассмотрены их основные достоинства и недостатки, разобраны необходимые детали технической реализации. Имея подробное представление об обрабатываемых каналах связи, следует подробно описать внутреннюю архитектуру автоматизированной информационной системы, структуру ее баз данных, серверные решения, технологии разработки клиентского представления.

2.3 Характеристика базы данных

2.3.1 Характеристика инфологической модели БД

После описания глобальной стратегии работы системы и основных каналов связи следует подробно описать внутреннюю архитектуру разработанной автоматизированной информационной системы. В первую

очередь необходимо уделить внимание хранилищам данных пользователей. Так как система имеет микросервисную архитектуру, следовательно, каждый микросервис, выполняющий определенную функциональность будет иметь отдельную базу данных. Следует подробнее написать о структуре микросервисов и их базах данных.

В системе функционирует три микросервиса:

1) сервис аккаунтов - микросервис, отвечающий за хранение данных пользователей, их контактов, регистрацию, авторизацию и тому подобное;

2) сервис уведомлений - микросервис, отвечающий за рассылки уведомлений по выбранным каналам связи;

3) микросервис пользовательского режима - микросервис, в котором реализовано графическое представление, с которым собственно и осуществляется работа пользователя.

Очевидно, что отдельные базы данных будут иметь сервис аккаунтов и сервис уведомлений.

Инфологическая схема базы данных сервиса аккаунтов представлена на рисунке 2.6.

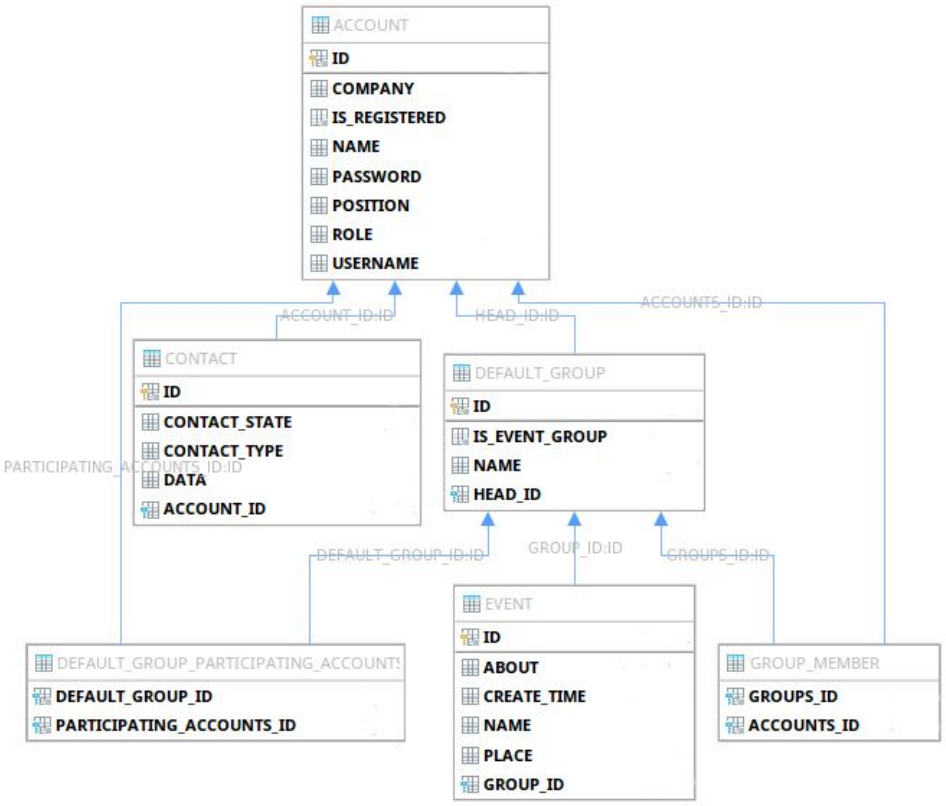

Рисунок 2.6 - Инфологическая схема базы данных сервиса аккаунтов

На схеме выше представлена структура таблиц, полей и связей базы данных для хранения данных об аккаунтах пользователей, их контактах, группах, в которых они принимают участие, событиях, к которым прикреплены отдельные группы.

Для микросервиса уведомлений также необходимо описать структуру базы данных. Логично, что в этой базе данных хранятся данные уведомлений, а также данные о получателях, времени создания уведомления, отправки, а также статус отправки сообщения.

Инфологическая схема базы данных микросервиса уведомлений представлена на рисунке 2.7.

Рисунок 2.7 - Инфологическая схема базы данных сервиса уведомлений

На схеме представленной выше представлена логическая структура таблиц, полей и связей базы данных микросервиса уведомлений. На ней показано, как конкретно взаимосвязаны уведомления и получатели.

В данном разделе описаны инфологические схемы баз данных микросервисов. Зная общую структуру сервисов и их баз данных, необходимо подробно определить структуру полей таблиц и взаимосвязей таблиц с привязкой к конкретным системам управления базами данных.

2.3.2 Характеристика даталогической модели БД

Следует подробно описать техническую сторону реализации баз данных в разработанной автоматизированной информационной системе. В сервисе уведомлений в качестве системы управления базами данных использована Apache Derby, которая является реляционной системой управления базами данных, написана на языке программирования Java и предназначена для встраивания в приложения на платформе Java. База данных сервиса аккаунтов выполнена в системе управления базами данных PostgeSQL, свободной объектно-реляционной СУБД.

Даталогическая схема базы данных сервиса аккаунтов представлена на рисунке 2.8.

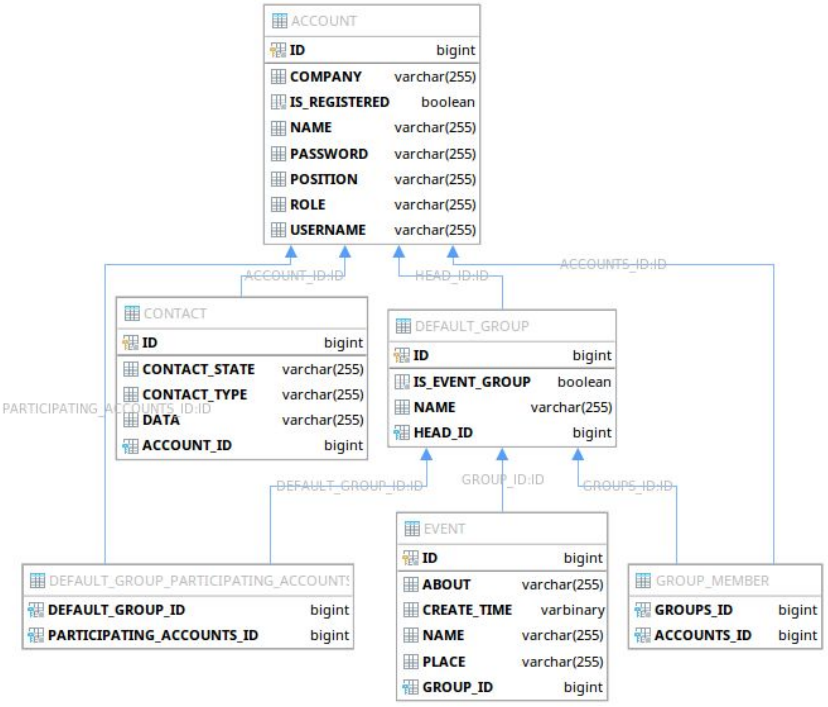

Рисунок 2.8 - Даталогическая схема базы данных сервиса аккаунтов

На схеме выше представлены таблицы, связи и поля с конкретными типами данных с привязкой к системе управления базами данных PostgreSQL.

На рисунке 2.9 представлена даталогическая схема базы данных сервиса уведомлений.

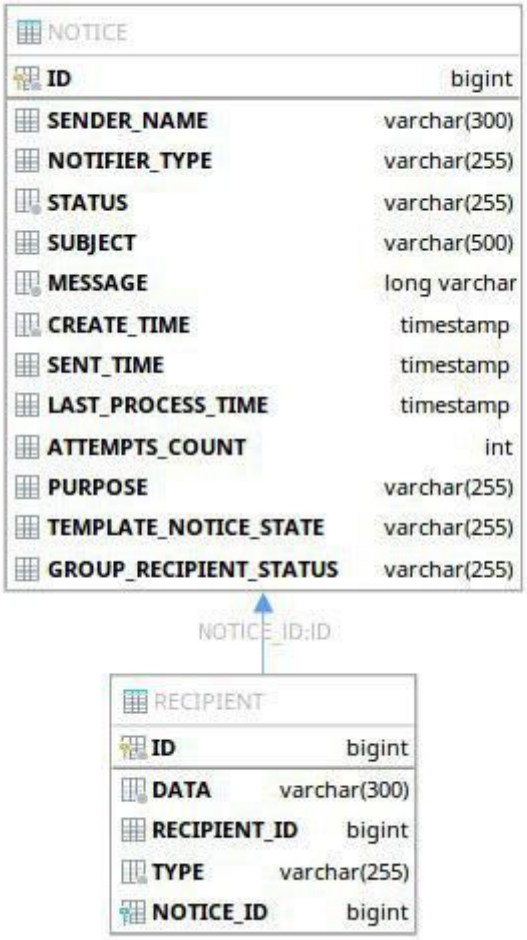

Рисунок 2.9 - Даталогическая схема базы данных сервиса уведомлений

На рисунке выше представлена схема, которая содержит в себе конкретные типы данных для полей с привязкой к системе управления базами данных Apache Derby.

В данном разделе описана структура баз данных микросервисов с привязкой к конкретным системам управления базами данных. Достаточно

полно описав архитектуру хранения данных в системе, имеет смысл представить полноценное описание технологий серверной и клиентской частей.

2.4 Обоснование выбора технологий для разработки

2.4.1 Характеристика технологий разработки серверной части

Зная структуру хранилищ данных, следует описать основополагающие технологии разработки серверной части. Серверная часть отвечает за загрузку данных из базы данных, формирование данных, пригодных для использования в клиентской части, обработку http-запросов, отправку уведомлений по расписанию, авторизацию, распределение прав пользователей, регистрацию, контролирует использование ссылок для регистрации.

Серверную часть образуют сервис аккаунтов и сервис уведомлений. Они используют однородные серверные технологии. Серверная часть организована с помощью фреймворка Spring, написанного на языке программирования Java.

Java - строго-типизированный объектно-ориентированный язык программирования. Написанные на нем приложения транслируются в байт-код, следовательно, они могут быть исполнены на устройстве с любой операционной системой, поддерживающей работу с виртуальной системой.

Spring - фреймворк для разработки веб-приложений с открытым исходным кодом, используемый с помощью Java-платформы. Spring можно рассматривать как некоторое семейство фреймворков или взаимно совместимых модулей. Фреймворки Spring можно использовать отдельно друг

от друга, но максимальную программную эффективность они имеют при взаимодействии друг с другом.

Следует описать фреймворки Spring, использованные в разработке автоматизированной информационной системы:

1) Spring Boot - упрощает создание автономных приложений, которые можно «просто запустить». Для большинства приложений Spring Boot требуется очень небольшая конфигурация Spring. Особенности: создание автономных приложений Spring, встроенные Tomcat, Jetty или Undertow, автоматическая настройка Spring, готовые к производству функции, такие как показатели, проверки работоспособности и внешняя конфигурация;

2) Spring Data - фреймворк для взаимодействия с базами данных. Особенности: мощные репозитории и пользовательские абстракции объектов, динамический вывод запроса из имен методов репозитория, внедрение базовых классов домена, обеспечивающих основные свойства, поддержка прозрачного аудита, возможность интегрировать код пользовательского репозитория, расширенная интеграция с контроллерами Spring MVC;

3) Spring MVC - это веб-фреймворк, построенный на API-интерфейсе Servlet и включенный в Spring Framework с самого начала. Формальное имя «Spring Web MVC» происходит от имени исходного модуля spring-webmvc, но его чаще называют «Spring MVC»;

4) Spring Security - это платформа, которая ориентирована на предоставление как аутентификации, так и авторизации для приложений Java. Как и все проекты Spring, эффективность Spring Security заключается в том, как легко ее можно расширить, чтобы удовлетворить пользовательские требования. Особенности: всесторонняя и расширяемая поддержка как аутентификации, так и авторизации, защита от атак, интеграция API сервлета, дополнительная интеграция с Spring Web MVC.

В данном разделе были описаны основные серверные технологии, использованные при разработке. Далее следует описать технологии разработки клиентской части.

2.4.2 Характеристика визуальной оболочки

Технологии разработки серверной части были описаны выше. Заключительным разделом представления используемых технологий является описание технологий клиентской части. Клиентская часть является третьим микросервисом. Она была выполнена с помощью фреймворка Angular, использующего язык программирования TypeScript.

TypeScript - это язык программирования с открытым исходным кодом, разработанный и поддерживаемый Microsoft. Это строгое синтаксическое множество над JavaScript, которое добавляет необязательную статическую типизацию на язык.

Angular - это платформа, которая упрощает построение web-приложений. Angular сочетает декларативные шаблоны, инъекции зависимостей, интегрированные передовые методы решения проблем развития. Angular позволяет разработчикам создавать web-приложения, а также мобильные и настольные приложения.

Выше были описаны структура работы новой системы и технологии ее разработки. Далее следует описать программную реализацию проектных решений.

3 Программная реализация проектных решений

3.1 Программное обеспечение задачи

3.1.1 Общие положения

После описания структуры работы новой системы и технологий ее разработки следует описать программную реализацию проектных решений. В первую очередь описано программное обеспечение задачи. Ниже приведены общие положения по структуре взаимодействия пользователя и разработанной автоматизированной информационной системы.

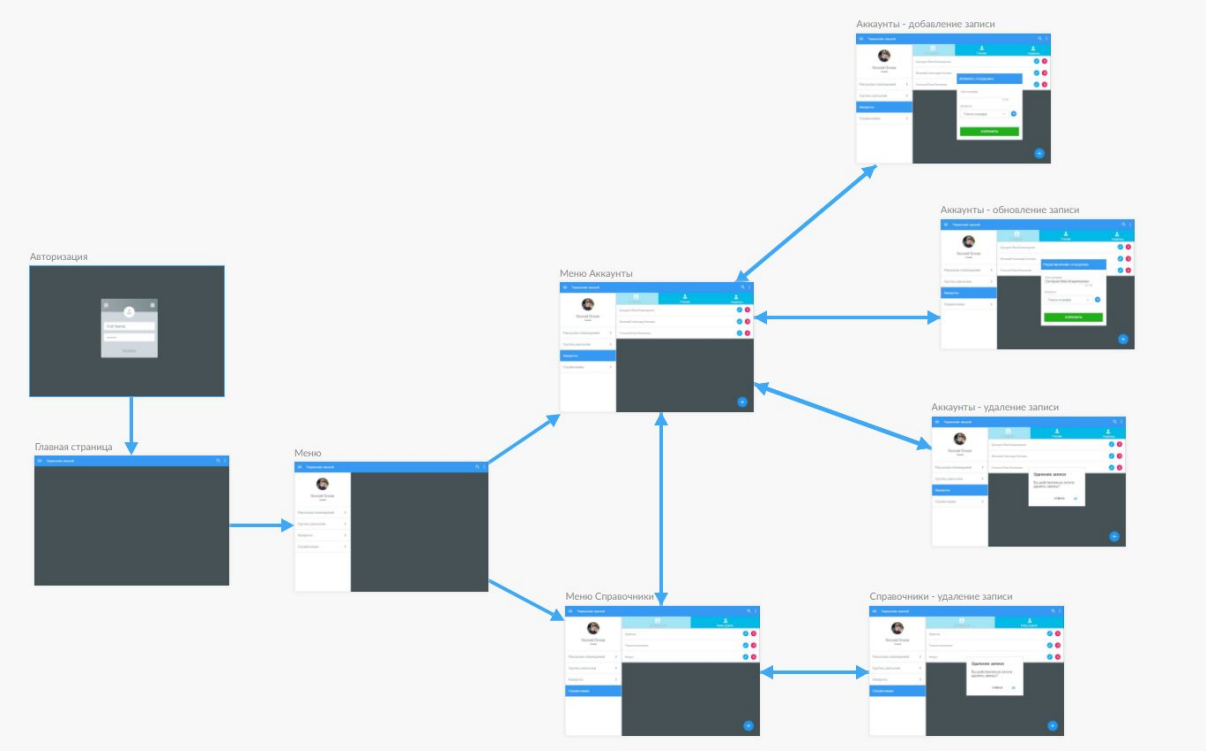

Рисунок 3.1 - Структурная схема работы системы

На рисунке выше приведена схема навигации в разработанной системе. Работа пользователя начинается с ввода логина и пароля в форме авторизации, затем он получает доступ к главному меню, далее из главному меню у него имеется возможность продолжить работу со справочниками, аккаунтами, подсистемами учета успеваемости и посещаемости, подсистемой рассылок.

Схема приведена выше с целью обозначить структуру навигации и не регламентирует обязательный макет дизайна пользовательского интерфейса.

В данном разделе описаны общие положения, а именно описана структура навигации пользовательского пользователя в системе.

3.1.2 Структурная схема пакета

Выше были описаны общие положения, далее следует описать структурную схему пакета. Так как разработанная автоматизированная информационная система имеет микросервисную архитектуру, каждый декомпозируемый элемент функционала представлен в виде отдельной базы данных и серверного компонента, клиентская часть также представлена в виде отдельного микросервиса.

Глобально разработанный пакет можно разделить на три слоя, а именно клиентский слой, слой backend микросервисов и слой базы данных. Доступ к слою backend микросервисов из клиентского слоя происходит в соответствии в соответствии с правами доступа.

Структурная схема пакета автоматизированной информационной системы представлена на рисунке 3.2.

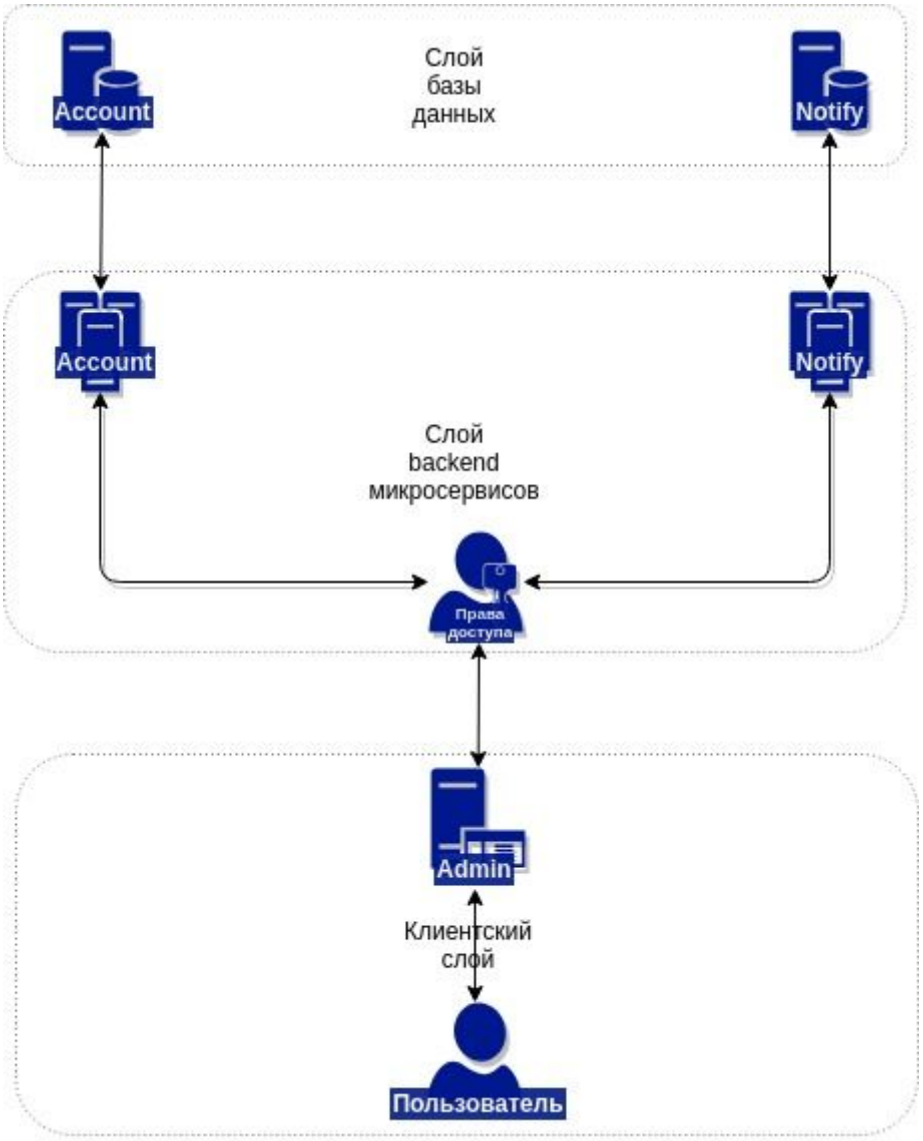

Рисунок 3.2 - Структурная схема пакета

В данном разделе описана структурная схема пакета разработанной информационной системы. Таким образом была описана структура разработанных микросервисов, описана структура взаимодействия между ними.

Выше было представлено программное обеспечение задачи в виде общих положений и структурной схемы пакета. Далее следует описать контрольный пример реализации проекта для окончательного тестирования системы и демонстрации ее работы.

3.2 Описание контрольного примера реализации

После описания программного обеспечения задачи необходимо выполнить контрольный пример реализации проекта. Ниже приведены подробные описания работы системы, также приведены конкретные наглядные примеры.

Начало работы в автоматизированной информационной системе происходит за счет авторизации. Форма авторизации в системе представлена на рисунке 3.3.

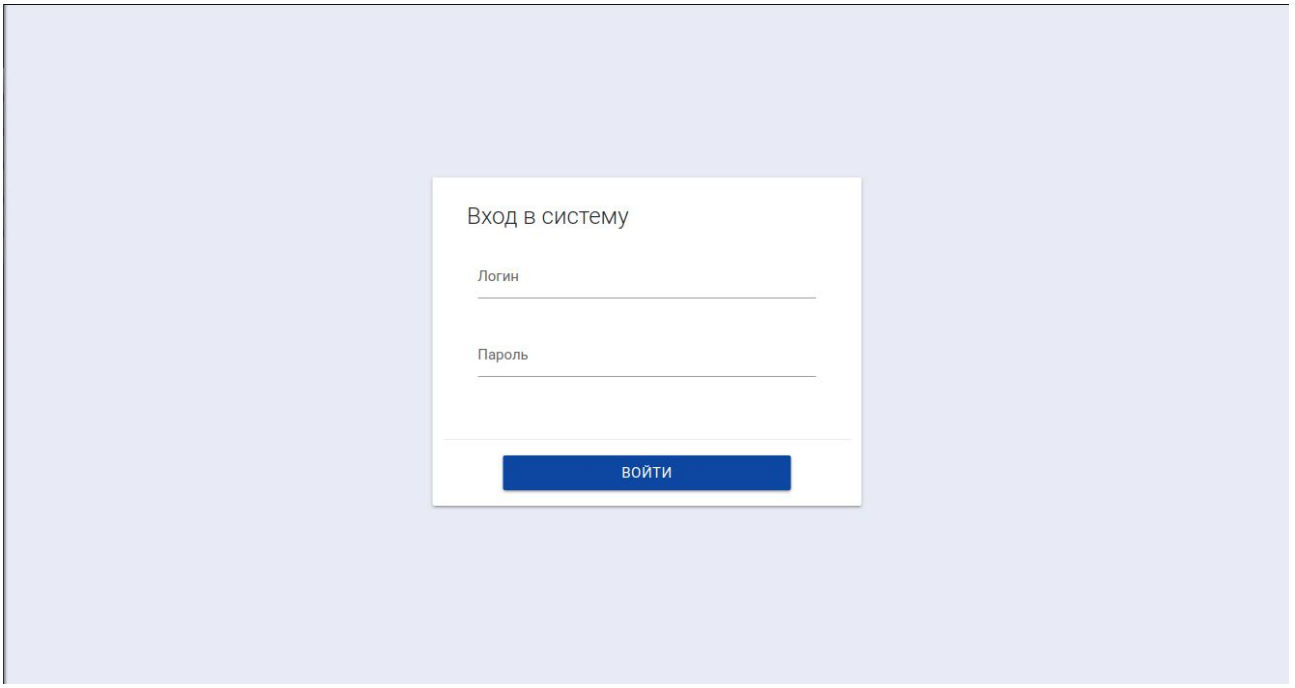

Рисунок 3.3 - Форма авторизации в системе

Каждый пользователь обязан иметь логин и пароль, чтобы иметь возможность работы в системе. Данная процедура необходима для того, чтобы

обеспечить защиту конфиденциальных данных от несанкционированного доступа.

Необходимо продемонстрировать основной функционал автоматизированной системы. В отдельной таблице выводится список контактов. На рисунке 3.4 представлен список контактов родителей.

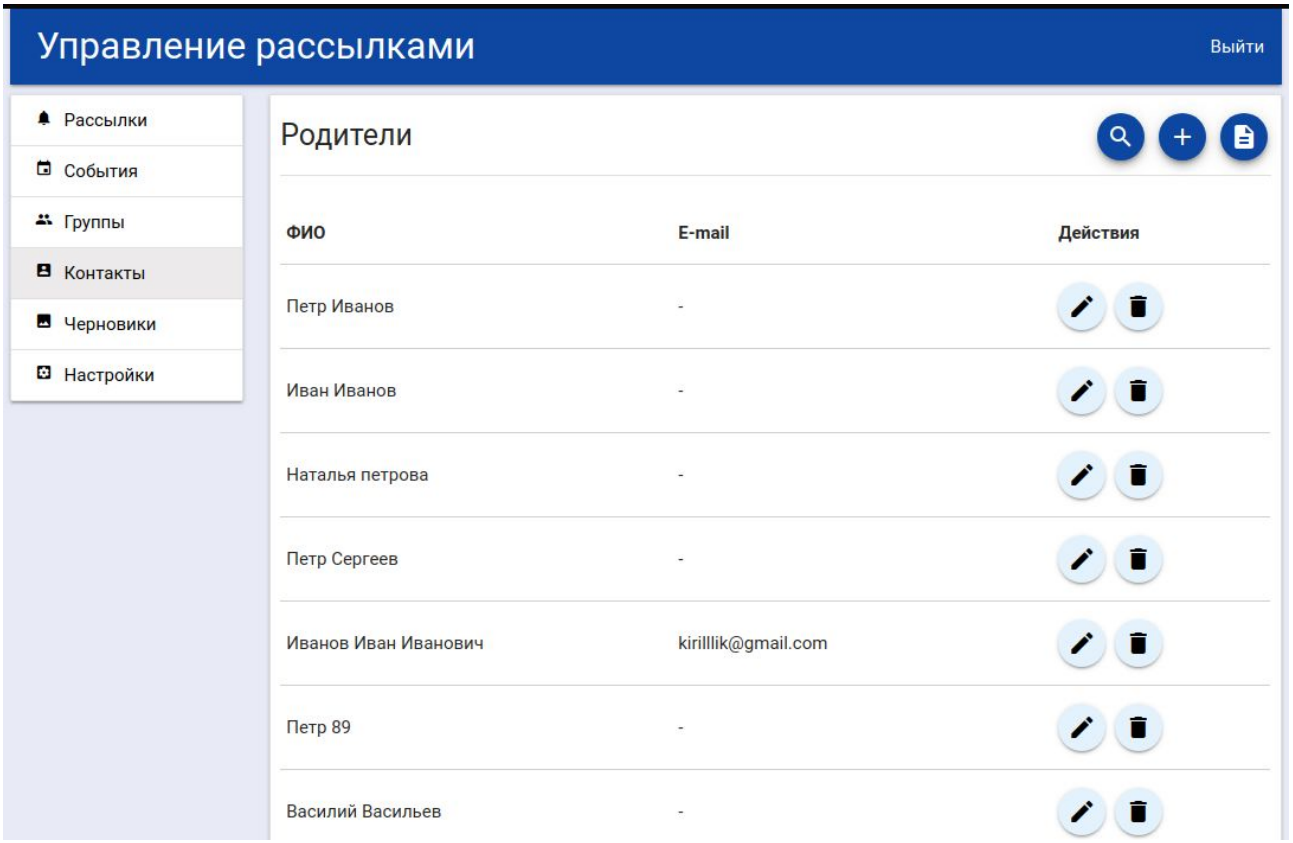

Рисунок 3.4 - Список контактов родителей

Логично, что данный список может видоизменяться со временем, следовательно в системе реализован функционал для добавления, изменения и удаления контактов.

На рисунке 3.5 представлена форма, используемая для добавления и изменения контактов.

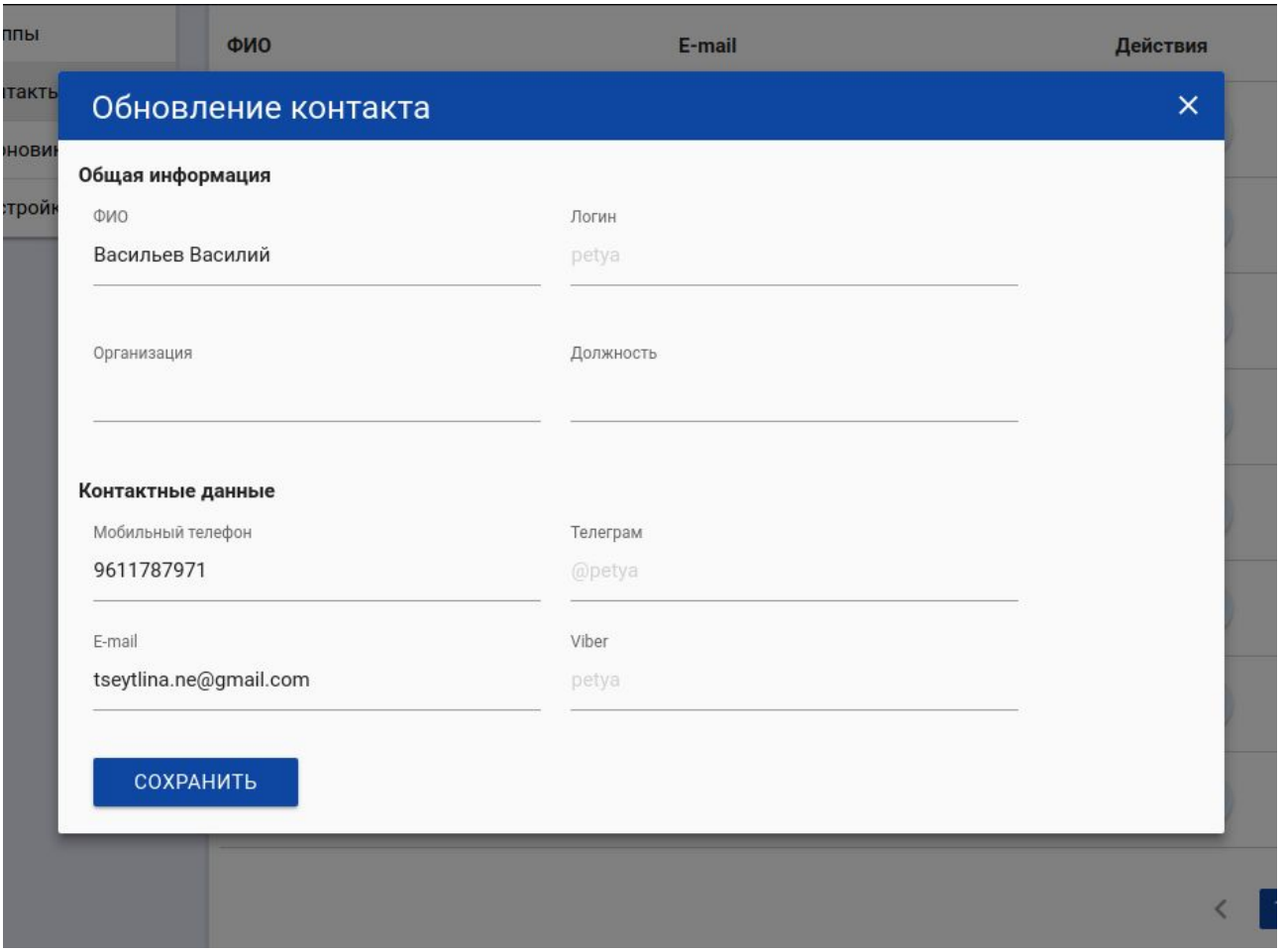

Рисунок 3.5 - Форма добавления и изменения контакта

Как показано на рисунке выше форма добавления и изменения контакта представлена в виде модального окна, однако нет необходимости обязательно добавлять или изменять какие-либо данные, так как по нажатию на кнопку в правом верхнем углу или фон модального окна, оно закроется без изменений.

Сохранение в базу данных происходит сразу же после нажатия на кнопку “Сохранить”, данные в списке обновляются сразу же после сохранения в базу данных.

Для защиты от случайного удаления контакта была введена форма подтверждения удаления контакта. Форма подтверждения удаления контакта представлена на рисунке 3.6.

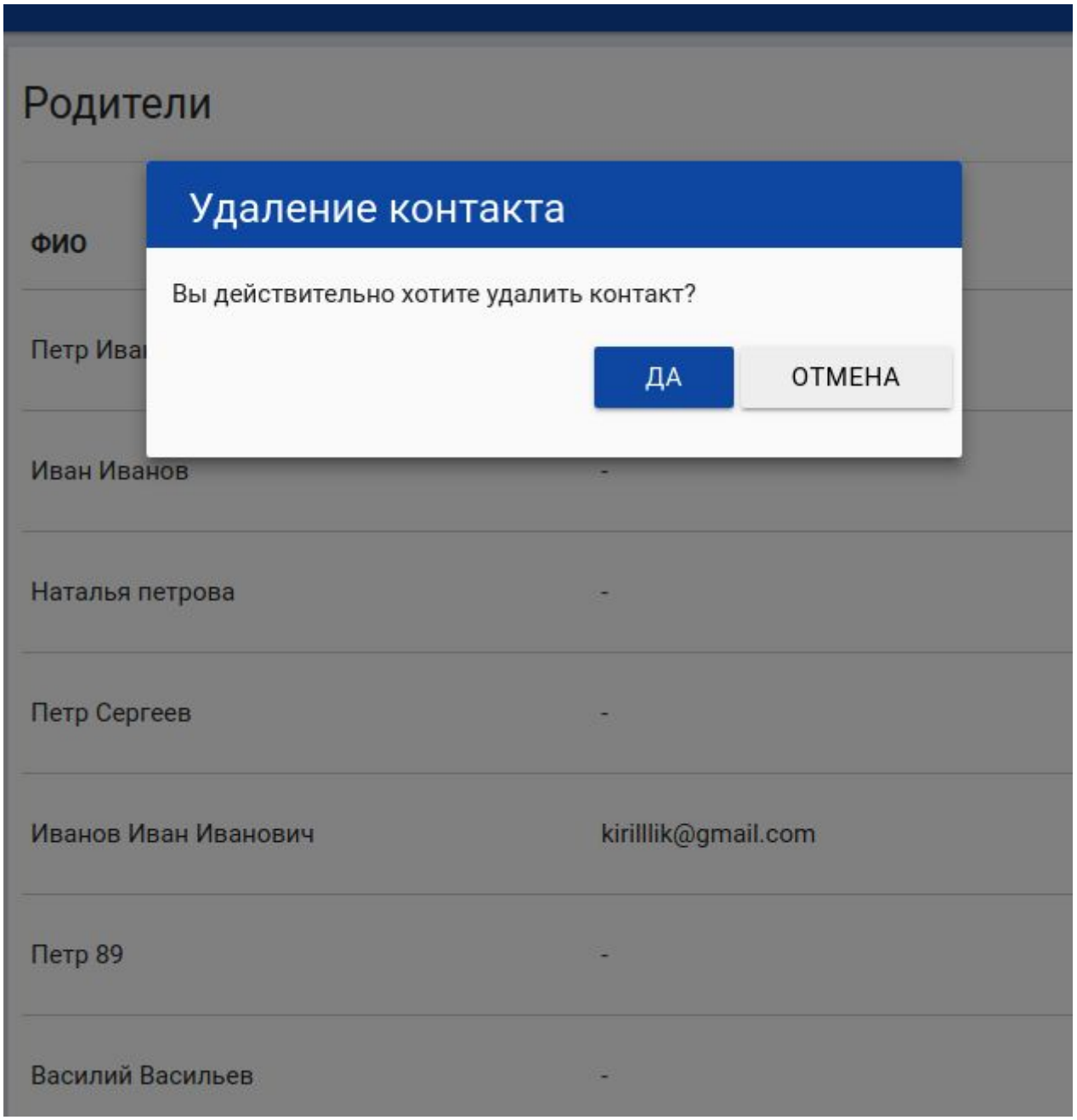

Рисунок 3.6 - Форма подтверждения удаления контакта

В отдельной таблице выводится список школьных групп и количество контактов, которые в них добавлены, а также посвящена ли эта группа какому-либо школьному событию. На рисунке 3.7 представлен список школьных групп.

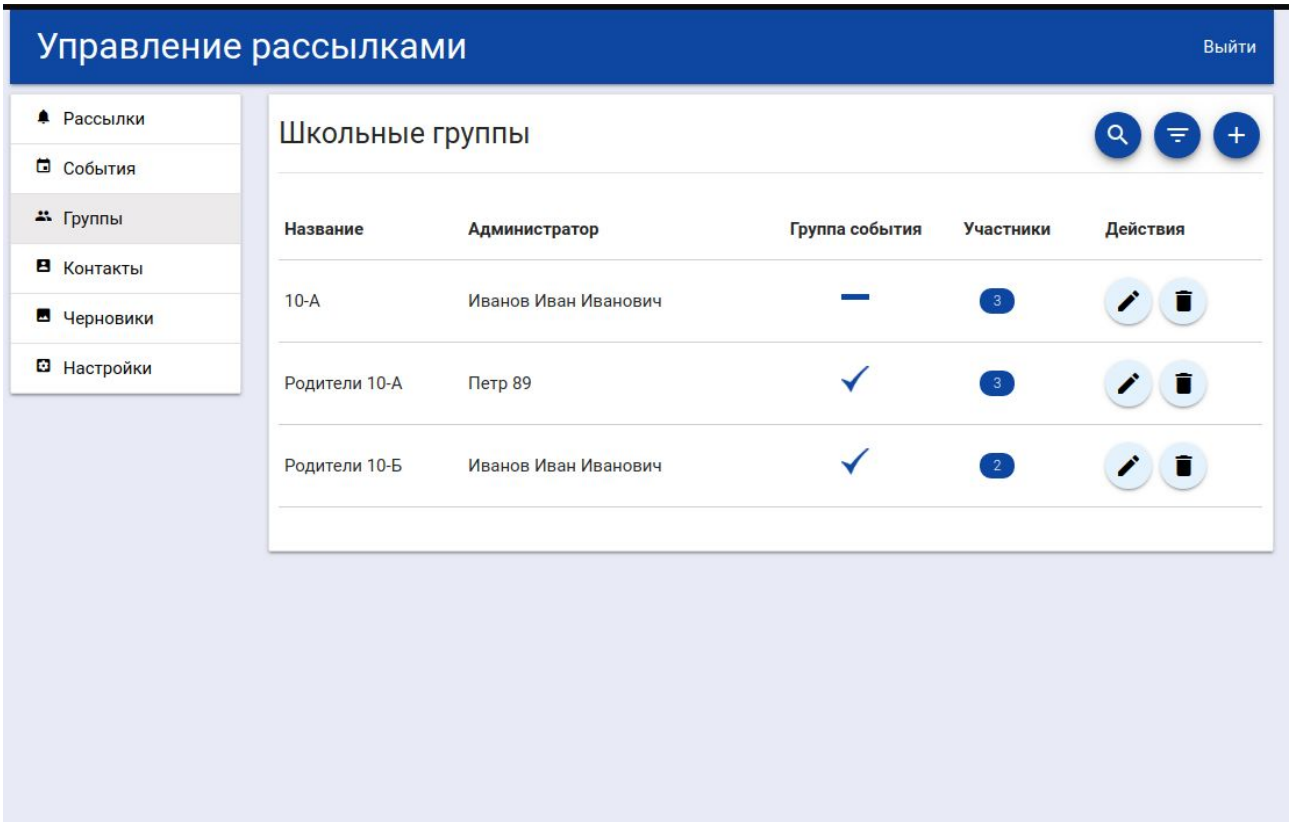

Рисунок 3.7 - Список школьных групп

Логично, что данный список также может видоизменяться со временем, следовательно в системе реализован функционал для добавления, изменения и удаления школьных групп.

Как показано на рисунке ниже форма добавления и изменения школьной группы представлена в виде модального окна, однако нет необходимости обязательно добавлять или изменять какие-либо данные, так как по нажатию на кнопку в правом верхнем углу или фон модального окна, оно закроется без изменений.

Сохранение в базу данных происходит сразу же после нажатия на кнопку “Сохранить”, данные в списке обновляются сразу же после сохранения в базу данных так же как и при использовании функционала для работы с контактами родителей, сотрудников и учеников.

На рисунке 3.8 представлена форма, используемая для добавления и изменения школьных групп.

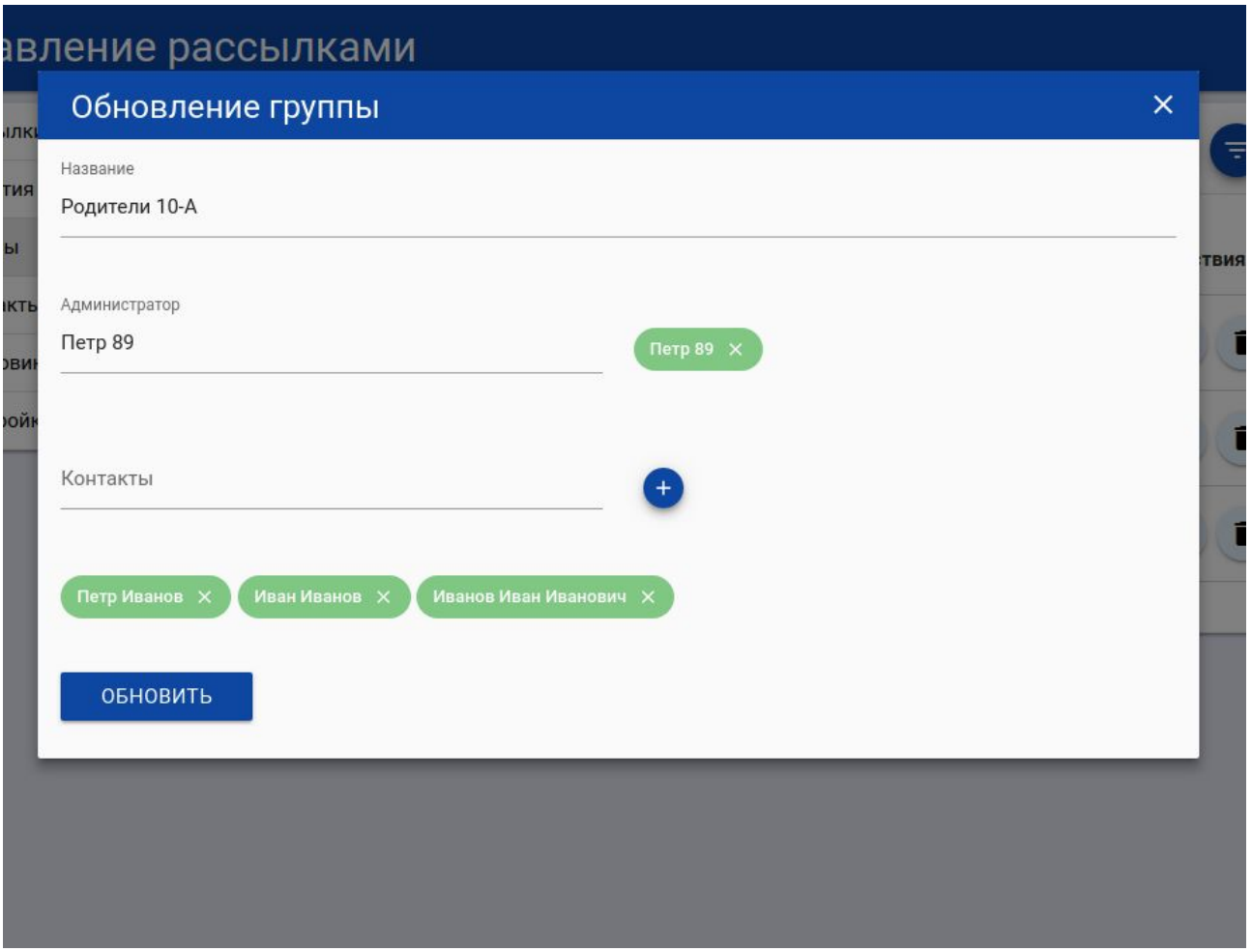

Рисунок 3.8 - Форма добавления и изменения школьных групп

Для защиты от случайного удаления школьной группы была также введена форма подтверждения удаления школьной группы. Форма подтверждения удаления школьной группы представлена на рисунке 3.9.

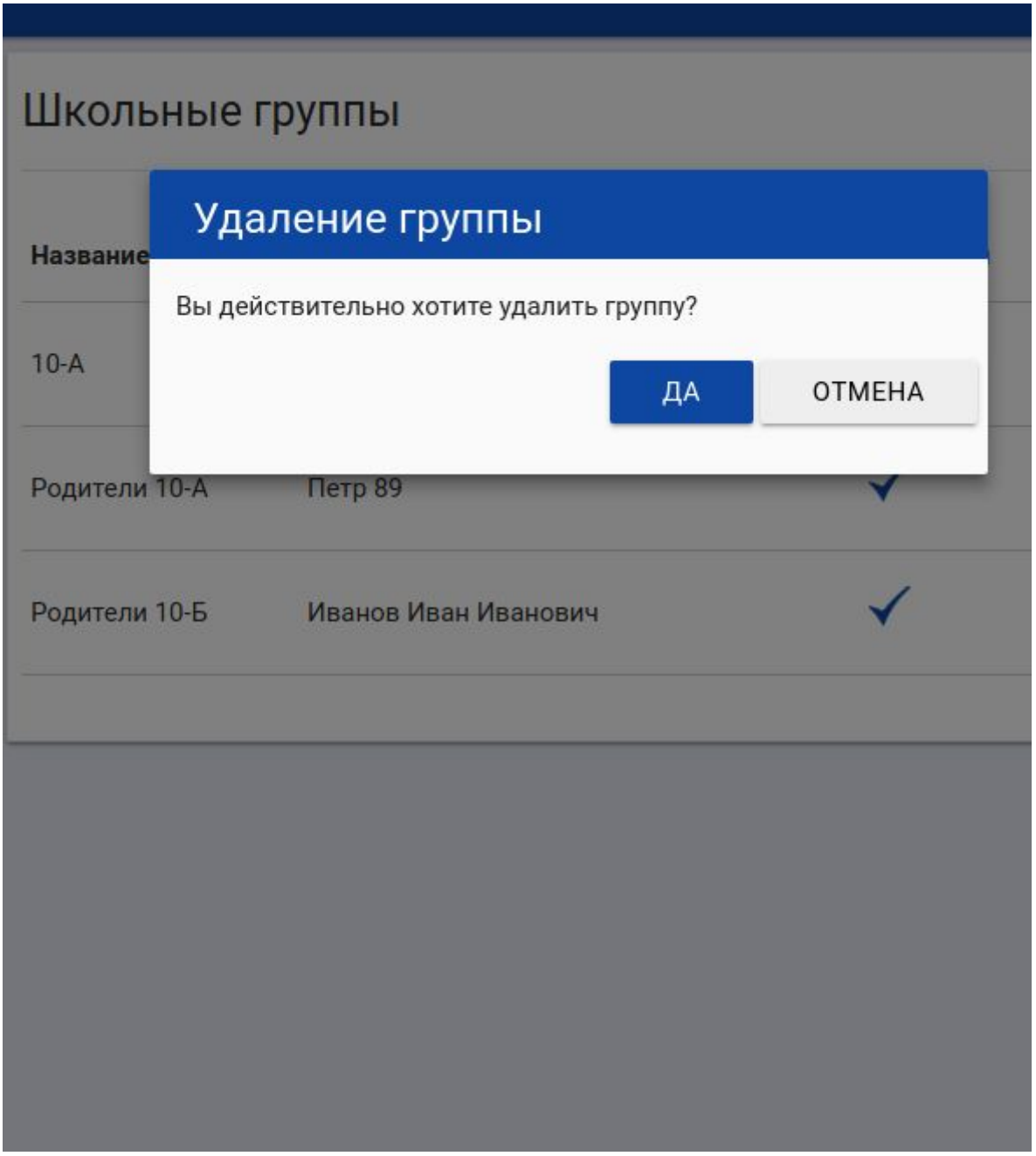

Рисунок 3.9 - Форма подтверждения удаления школьной группы

В разработанной автоматизированной информационной системе имеется возможность автоматически формировать уведомления в зависимости от успеваемости и посещаемости учеников, что делает данную систему интерактивной.

Форма автоматически сформированной рассылки с уведомлениях о проблемах с посещаемость ученика представлена на рисунке 3.10.

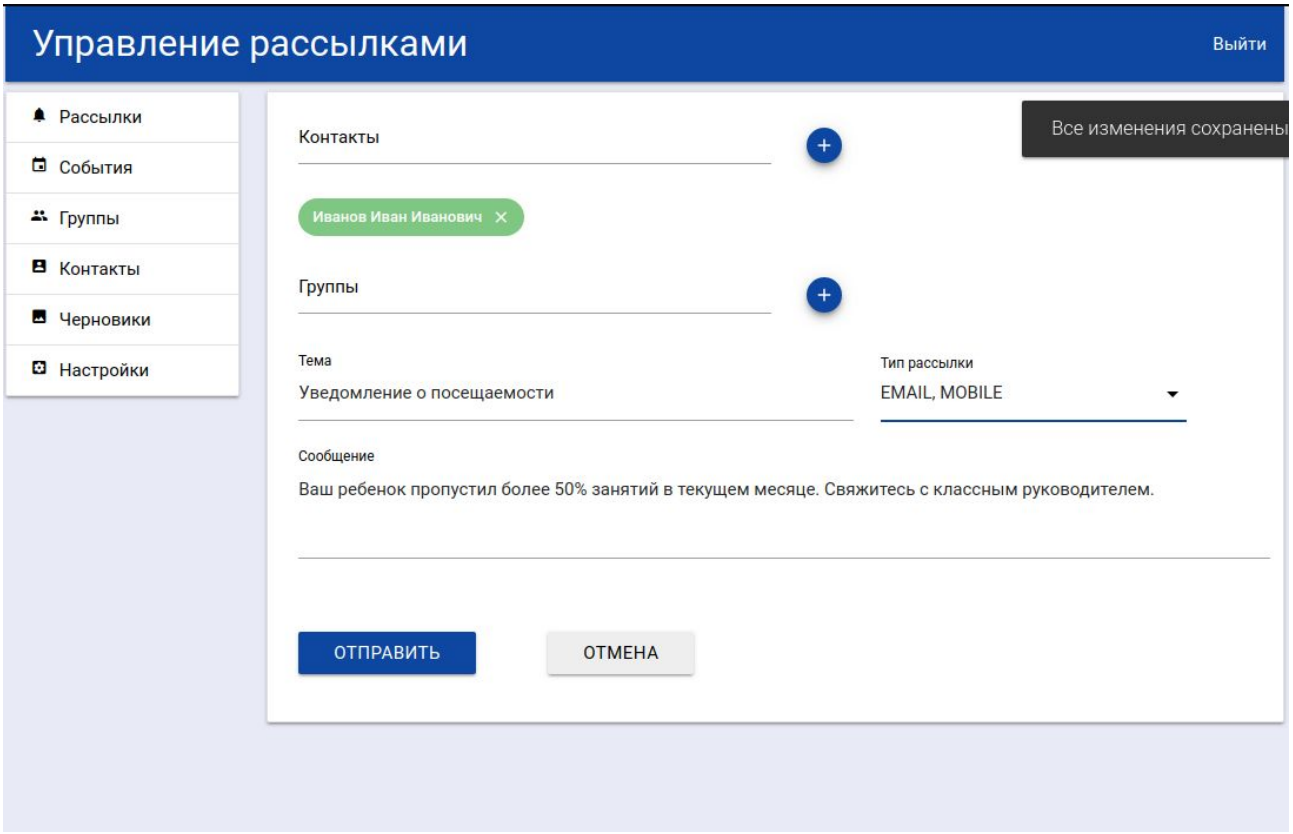

Рисунок 3.10 - Форма автоматически сформированной рассылки

Рассылки в системе можно формировать автоматически, однако существует возможность редактировать список получателей рассылки. Пользователь может самостоятельно добавить или удалить получателя рассылки, а также отредактировать текст сообщения и выбрать каналы связи, если у него имеется такая необходимость. Также возможно отредактировать контактные данные получателя, не покидая форму рассылки.

Форма выбора одиночных получателей рассылки представлена на рисунке 3.11.

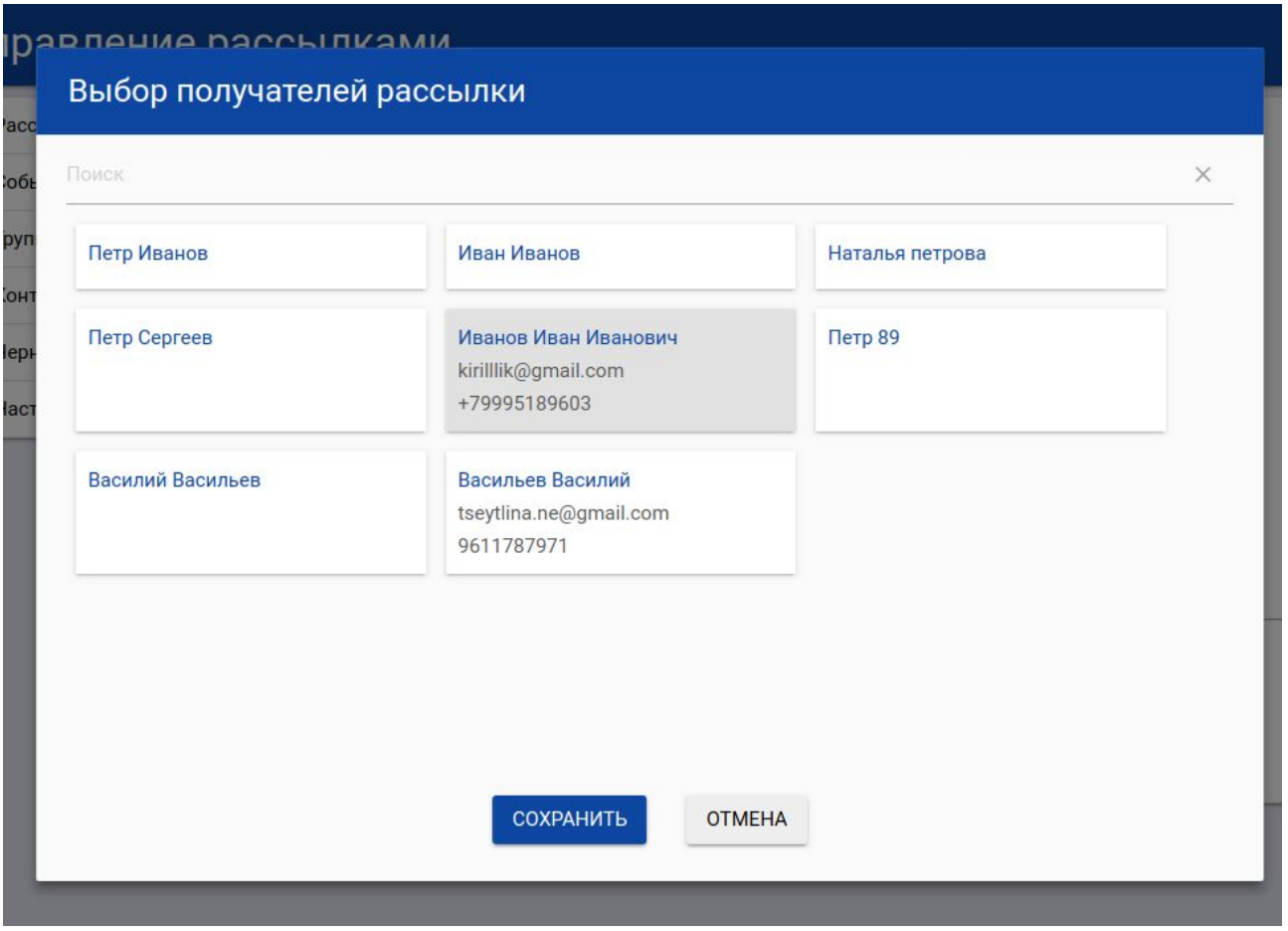

Рисунок 3.11 - Форма выбора одиночных получателей рассылки

Также в системе существует возможность отправить уведомление всем участникам школьной группы. Данные школьной группы также можно отредактировать, не покидая формы рассылки.

В форме школьной группы существует возможность сразу просмотреть список контактов, добавленных как участников в группу.

Форма выбора школьной группы получателей рассылки представлена на рисунке 3.12.

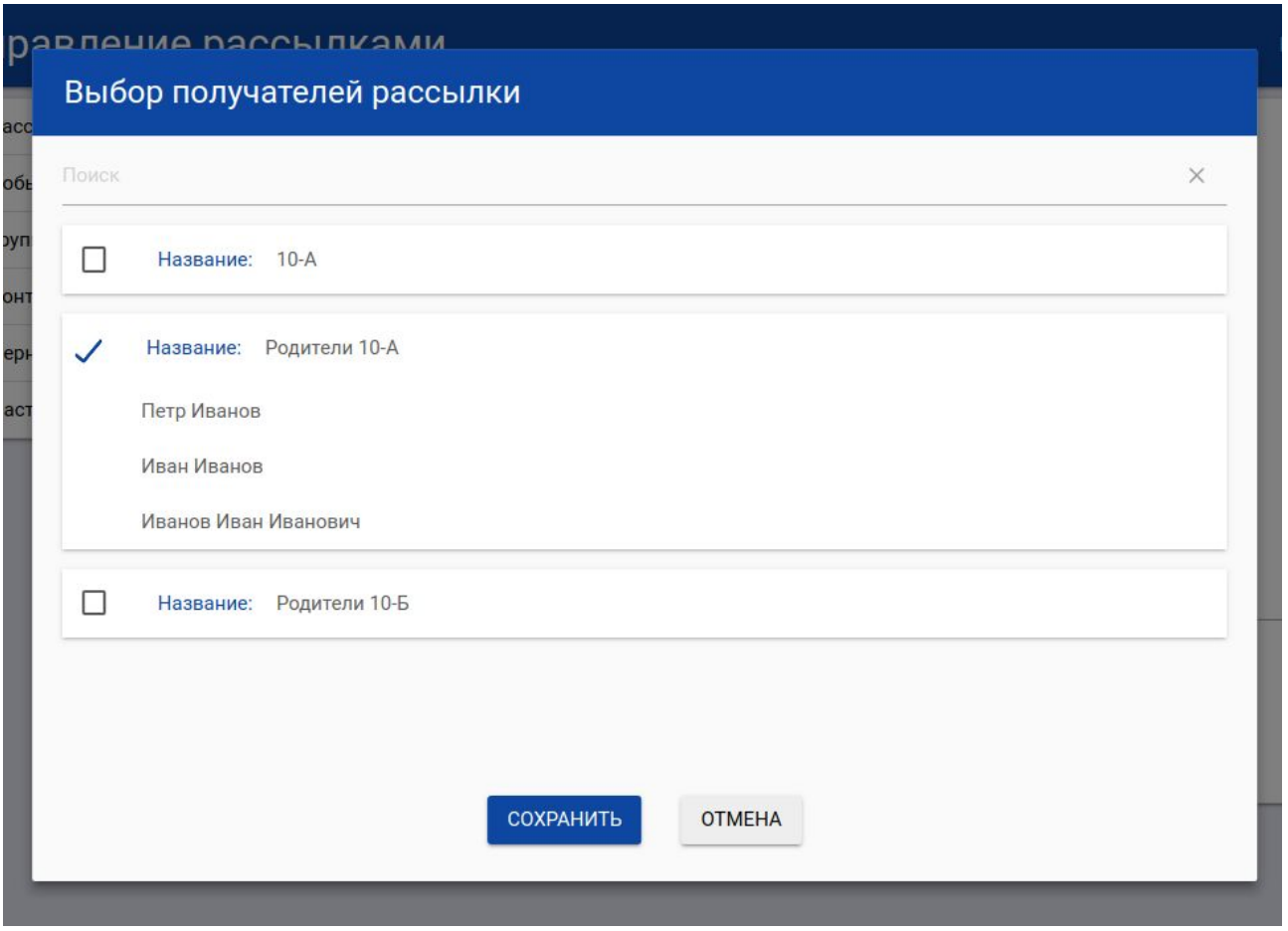

Рисунок 3.12 - Форма выбора школьных групп получателей рассылки

В данном разделе был описан контрольный пример реализации, который наглядно демонстрирует работу системы и позволил окончательно завершить тестирование разработанной автоматизированной информационной системы.

3.3 Обоснование эффективности разработки

В заключительном разделе следует обосновать эффективность разработки. Это является важнейшим этапом, так как именно данный раздел

помогает ответить на вопрос о том, для чего конкретно проводилось исследование и что конкретно улучшилось в результате.

Полученная автоматизированная информационная система наиболее всего проявила себя как социально эффективная, так как ее основной задачей было комфортной взаимодействие всех участников учебного процесса.

Социальная эффективность разработки заключается в том, что она способствует большему взаимопониманию абсолютно разносторонних участников учебного процесса, позволила сотрудникам комфортнее и быстрее работать с учетом посещаемости и успеваемости, а также формированием рассылок, что в свою очередь помогает экономить временные ресурсы и больше внимания уделять ученикам.

Родители также намного быстрее получают уведомления о достижениях и неудачах их детей и могут своевременно на них реагировать.

Таким образом разработанная автоматизированная информационная система является социально эффективной.

В данном разделе описаны общие положения в виде схемы навигации в системе, структурная схема пакета в виде структуры взаимодействия микросервисов автоматизированной информационной системы, контрольный пример реализации, обоснована эффективность разработки в виде социальной эффективности.

Все основные действия, необходимые в данной работе были выполнены, следует подвести итоги в виде заключения.

ЗАКЛЮЧЕНИЕ

В заключении следует подвести итоги совершенной работы. Были закреплены теоретические знания, полученные в процессе обучения в университете. Были выполнены и подробно описаны краткая характеристика учебного процесса средней школы, постановка задач, описание структуры работы системы “КАК ЕСТЬ”, характеристика выявленных недостатков, постановка задач для устранения выявленных недостатков, характеристика возможных способов решения задач, информационное обеспечение задачи, информационная модель и ее описание, характеристика интегрируемых систем коммуникации, характеристика базы данных, характеристика инфологической модели БД, характеристика даталогической модели БД, обоснование выбора технологий для разработки, характеристика технологий разработки серверной части, характеристика визуальной оболочки, программная реализация проектных решений, общие положения разработки, структурная схема пакета, контрольный пример реализации, приведено обоснование эффективности разработки. Цель была достигнута.

В данной работе были подробно описаны все этапы, которые необходимо реализовать на пути к получению программного продукта - интерактивной системы контроля успеваемости и посещаемости учеников средней школы.

В дальнейшем данное приложение может быть использовано для контроля успеваемости и посещаемости учеников в средней школе.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ибрагимов, И.Д. Многокомпонентная система контроля и оценки знаний и компетенций студентов лингвистического вуза [Текст] / И.Д. Ибрагимов. – Липецк: Педагогическое образование в России, 2014. – № 3.

2. Клир, Д.А. Системология. Автоматизация решения системных задач: пер. с англ. [Текст] / Д.А. Клир. – Липецк: Книжный бульвар, 2014. – 341с.

3. Левинская, М.А. Автоматизированная генерация заданий по математике для контроля знаний учащихся [Текст] / М.А. Левинская. – Москва: Образовательные технологии, 2013. – 345с.

4. Конопля, А.И. Компетентностная модель подготовки специалиста-медика [Текст] / А.И. Конопля. – Москва: Книжный бульвар, 2015. – 258с.

5. Гавронская, Ю.Б. Интерактивность» и интерактивное обучение [Текст] / Ю.Б. Гавронская. – Уфа: Образование, 2014. – 126с.

6. Роберт, И.В. Информационные и коммуникационные технологии в образовании [Текст] / И.В. Роберт. – Москва: Дрофа, 2015. – 147с.

7. Катасонова, Г. Р. Интерактивные технологии в обучении [Текст] / Г.Р. Катасонова. – Екатеринбург: Труды и искусство, 2013. – 125с.

8. Бойцова, Е.П. Модульно-реитинговая система на базе тестовых технологий [Текст] / Е.П. Бойцова, В.Г. Дроздов. – Москва: Образование, 2015. – 180с.

9. Казарин, О.В. Интерактивная система доказательств для интеллектуальных средств контроля доступа к

информационно-вычислительным ресурсам [Текст] / О.В. Казарин, Л.М. Ухлинов. – Самара: Автоматика и телемеханика, 2014. – 167с.

10. Артюхина, М.С. Аппаратная составляющая интерактивных технологий образовательного назначения [Текст] / М.С. Артюхина, О.И. Артюхин, И.И. Клешнина. – Казань: Вестник, 2014. – 117с.

11. Гордеева, В.В. Активные и интерактивные формы организации и педагогического сопровождения самостоятельной работы студентов [Текст] / В.В. Гордеева. – Пенза: Известия, 2013. – 212с.

12. Красильникова, В.В. Теория и технологии компьютерного обучения и тестирования [Текст] / В.В. Красильникова. – Москва: Litres, 2017. – 256с.

13. Артюхина, М.С. Особенности современных средств обучения в контексте интерактивных технологий [Текст] / М.С. Артюхина. – Москва: Вестник, 2015. – 235с.

14. Богданов, А.А. Создание и исследование робототехнической системы с интерактивным управлением [Текст] / А.А. Богданов. – Уфа: Чтение, 2013. – 147с.

15. Рудометов, С.В. Визуально-интерактивная система имитационного моделирования технологических систем [Текст] / С.В. Рудометов. – Москва: Вестник, 2013. – 97с.

16. Бегг, К.Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. [Текст] / К.Т. Бегг. – Екатеринбург: Издательство АРМ, 2016. – 345с.

17. Корнеев, В.В. Базы данных. Интеллектуальная обработка информации. [Текст] / В.В. Корнеев, А.Ф. Гареев, С.В. Васютин. – Москва: Нолидж, 2015. – 258с.

18. Голицына, О.Л. Базы данных [Текст] / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - Москва: Форум, 2014. – 289с.

19. Варламов, О.О. Эволюционные базы данных и знаний для адаптивного синтеза интеллектуальных систем. Миварное информационное пространство [Текст] / О.О. Варламов. – Москва: Радио и связь, 2014. – 341с.

20. Советов, Б.Я. Базы данных: теория и практика. Общество с ограниченной ответственностью [Текст] / Б.Я. Советов, В.В. Цехановский, В.Д. Чертовской. – Москва: Издательство ЮРАЙТ, 2016. – 439с.

21. Дюбуа, П.С. MySQL: Полное и исчерпывающее руководство по применению и администрированию баз данных MySQL 4, а также программированию приложений. [Текст] / П.С. Дюбуа. – Москва: Издательский дом Вильямс, 2014. – 532с.

22. Яргер, Р.Д. MySQL и mSQL. Базы данных для небольших предприятий и Интернета. [Текст] / Р.Д. Яргер, Д.В. Риз, Т.Л. Кинг. – СПб: Символ-Плюс, 2017. – 538с.

23. Миронов, В.В. Ситуационно-ориентированные базы данных: концепция, архитектура, XML-реализация. [Текст] / В.В. Миронов, Н.И. Юсупова, Г.Р. Шакирова. – Уфа: Вестник, 2015. – 376с.

24. Маклаков, С.В. BPwin и ERwin. CASE-средства разработки информационных систем. [Текст] / С.В. Маклаков. – Москва: Диалог-МИФИ, 2017. – 412с.

25. Грекул, В.И. Проектирование информационных систем. [Текст] / В.И. Грекул, Г.Н. Денищенко, Н.Л. Коровкина. – Москва: Интернет-университет информационных технологий-ИНТУИТ, 2015.

26. Якимов, И.М. Комплексный подход к моделированию сложных систем в системе BPwin-Arena. [Текст] / И.М. Якимов. – Казань: Вестник Казанского технологического университета, 2014. – № 6.

27. Маклаков, С.В. Моделирование бизнес-процессов. [Текст] / С.В. Маклаков. – Москва: Диалог, 2016. – 258с.

28. Репин, В.В. Сравнительный анализ нотаций ARIS/IDEF и продуктов, их поддерживающих. [Электронный ресурс] / В.В. Репин. – Электрон. текстовые дан. – Москва, 2015. – Режим доступа: Web: http://www. iteam. ru/publications/it/section_51/article_2518, свободный.

29. Атисков, А.Ю. Автоматизированная система трансформации диаграмм бизнес-процессов в диаграммы классов. [Текст] / А.Ю. Атисков. – Самара: Издательства №3, 2016. – 155с.

30. Григорьев, А.В. Анализ существующих способов создания интерфейса «языки формальных спецификаций проблемно-ориентированные языки». [Текст] / А.В. Григорьев. – Москва: Паблик, 2017. – 325с.

31. Программирования Java. [Электронный ресурс] / Официальный сайт языка программирования Java. Электрон. журн., 2015. – Режим доступа: htt p://java. com, свободный.

32. Арнольд, К.Т. Язык программирования JAVA. [Текст] /К.Т. Арнольд, Д.П. Гослинг. – СПб.: Питер-Пресс, 2014. – 269с.

33. Васильев, А.Н. Java: объектно-ориентированное программирование: для магистров и бакалавров: базовый курс по объектно-ориентированному программированию. [Текст] / А.Н. Васильев. – СПб.: Издательский дом «Питер», 2015. – 513с.

34. Романов, В.Ю. Моделирование и верификация архитектуры программного обеспечения, разработанного на языке Java. Современные информационные технологии и ИТ-образование. [Текст] / В.Ю. Романов. – Москва: Издательский дом «Москва», 2014. – 178с.

35. Алексанян, Г.К. Анализ возможности применения языка программирования Java в задачах электроимпедансной томографии. [Текст] / Г.К. Алексанян, А.Д. Тарасов, К.В. Клевец. – Москва: Паблик, 2015. – 321с.

36. Allen, L. et al. Orbital angular momentum of light and the transformation of Laguerre-Gaussian laser modes. [Текст] / Allen L. – New-York: Publishing house, 2014. – 357с.

37. Edmonds, A.R. Angular momentum in quantum mechanics. [Текст] / A.R. Edmonds. – New Jersey: Princeton University Press, 2016. – 356с.

38. Williams, M.L. Stress singularities resulting from various boundary conditions in angular corners of plates in extension. [Текст] / M.L. Williams. – New York: Journal of applied mechanics, 2014. – 528с.

39. Mair, A. et al. Entanglement of the orbital angular momentum states of photons. [Текст] / A. Mair. – New Jersey: Nature, 2015. – 313с.

40. Wilczek, F. Magnetic flux, angular momentum, and statistics. [Текст] / F. Wilczek. – Canada: Physical Review Letters, 2016. –1144с.

41. Zubin, J. Vulnerability: a new view of schizophrenia. [Текст] / J. Zubin, B. Spring. – New York: Journal of abnormal psychology, 2016. – 103с.

42. Pasquinelli, A.E. et al. Conservation of the sequence and temporal expression of let-7 heterochronic regulatory RNA. [Текст] / A.E. Pasquinelli, - New York: Nature, 2015. – 86с.

ПРИЛОЖЕНИЕ А

```
import {Component, OnDestroy, OnInit, ViewChild} from '@angular/core';
import {GroupService} from '../../../../../../services/group-service';
import {EventsAccountService} from '../../../../../../services/account-service';
import {NoticeService} from '../../../../../../services/notice-service';
import {ActivatedRoute, Router} from '@angular/router';
import {Response} from '@angular/http';
import {Account} from '../../../../../../models/Account';
import {Notice} from '../../../../../../models/Notice';
import {isNullOrUndefined} from 'util';
import {AddRecipientsDialogComponent} from '../add-recipients-dialog/add-recipients-dialog.component';
import {AccountUpdateDialogComponent} from '../../../account/dialogs/account-update-dialog.component';
import {Contact} from '../../../../../../models/Contact';
import {ContactState} from '../../../../../../models/ContactState';
import {GroupUpdateDialogComponent} from '../../../group/dialogs/group-update-dialog.component';
import {Group} from '../../../../../../models/Group';
import {NgMultipleSelectComponent} from
'../../../../support_components/ng-multiple-select/ng-multiple-select.component';

declare var $: any;
declare var Materialize: any;

@Component({
  selector: 'app-simple-notification-dialog',
  templateUrl: './simple-notification-dialog.component.html',
  styleUrls: ['./simple-notification-dialog.component.css']
})
export class SimpleNotificationDialogComponent implements OnInit, OnDestroy {
  @ViewChild('addRecipientsDialog') addRecipientsDialog: AddRecipientsDialogComponent;
  @ViewChild('updateDialog') updateDialog: AccountUpdateDialogComponent;
  @ViewChild('updateGroupDialog') updateGroupDialog: GroupUpdateDialogComponent;
  @ViewChild('notifierTypesSelect') notifierTypesSelect: NgMultipleSelectComponent;

  public choicedNotifierTypes = [];
  public subject = '';
  public message = '';

  private ids = null;
  private createTime = null;

  private notifierTypes = [];

  public choicedAccounts = [];
  private findedAccounts = [];
  public choicedGroups = [];
  private findedGroups = [];

  private searchTextAccounts: string;
  private searchTextGroups: string;

  private isInProcess = false;
  private notifierTypesLoaded = false;
  private isAddAllowed = false;
```

```
private timer;

constructor(private groupService: GroupService,
            private accountService: EventsAccountService,
            private noticeService: NoticeService,
            private router: Router,
            private route: ActivatedRoute) { }

ngOnInit() {
  $(document).ready(function(){
    $('#message').characterCounter();
  });

  this.notifierTypes = this.route.snapshot.data['notifierTypes'].json() as string[];
  this.choicedNotifierTypes.push(this.notifierTypes[0]);
  this.notifierTypesLoaded = true;

  this.updateDialog.setAfterClosed((isUpdateAllow: boolean) => {
    if (isUpdateAllow) {
      const account = this.updateDialog.currentAccount;
      account.contacts = [];
      const dialogContacts = this.updateDialog.contacts;
      for (const contact in dialogContacts) {
        if (dialogContacts[contact] != '') {
          account.contacts.push(
            new Contact(0, contact, dialogContacts[contact], ContactState.ENABLED)
          );
        }
      }
      account.name = account.name.length == 0 ? dialogContacts['EMAIL'] : account.name;
      this.accountService.updateAccount(account).then((o) => {
        const account = o.json();
        this.choicedAccounts = this.choicedAccounts.map(a => a.id === account.id? account as Account : a);
        this.choicedGroups = this.choicedGroups.map(g => {
          g.accounts = g.accounts.map(a => a.id === account.id? account as Account : a);
          return g;
        });
        Materialize.toast('Контакт "' + account.name + '" обновлен', 2500);
      }).catch((e) => {
        Materialize.toast('Ошибка при обновлении контакта', 2500);
      });
    }
  });

  this.updateGroupDialog.setAfterClosedEvent((isUpdateAllow) => {
    if (isUpdateAllow) {
      const currentGroup = new Group();
      currentGroup.id = this.updateGroupDialog.id;
      currentGroup.name = this.updateGroupDialog.name;
      currentGroup.head = this.updateGroupDialog.head;
      currentGroup.eventGroup = false;
      currentGroup.accounts = this.updateGroupDialog.choicedMembers;
      currentGroup.participatingAccounts = null;
      this.groupService.updateGroup(currentGroup).subscribe(
        (complete: any) => {
          const group = complete.json();
          this.choicedGroups = this.choicedGroups.map(g => g.id === group.id ? group as Group : g);
          Materialize.toast('Группа "' + this.updateGroupDialog.name + '" обновлена', 2500);
```

```
      },
      (error: any) => {
        Materialize.toast('Ошибка при обновлении группы', 2500);
      }
    );
  }
});

this.route.params.subscribe(params => {
  this.ids = isNullOrUndefined(params['ids']) ? null : params['ids'];
  if (params['ids'] !== '' && !isNullOrUndefined(params['ids'])) {
    this.choicedAccounts = [];
    this.choicedGroups = [];
    this.noticeService.getByIds(params['ids'].split(',')).subscribe((data: Response) => {
      const notices = data.json() as Notice[];
      const notice = notices[0];
      for (const recipient of notice.recipients) {
        switch (recipient.type) {
          case 'SINGLE':
            this.accountService.getAccountById(recipient.recipientId)
              .then((data: any) => {
                this.choicedAccounts.push(data.json() as Account);
              })
              .catch(error => {
                Materialize.toast('Ошибка при загрузке данных', 2500);
              });
            break;
          case 'GROUP':
            this.groupService.getGroupById(recipient.recipientId)
              .then((data: any) => {
                this.choicedGroups.push(data.json() as Group);
              })
              .catch(error => {
                Materialize.toast('Ошибка при загрузке данных', 2500);
              });
            break;
        }
      }
      document.getElementById('label-for-subject').className = 'active';
      document.getElementById('label-for-message').className = 'active';
      const notifierTypes = [];
      for (const currentNotice of notices) {
        notifierTypes.push(currentNotice.notifierType);
      }
      console.log(notifierTypes);
      this.notifierTypesSelect.setValues(notifierTypes);
      this.message = notice.message;
      this.subject = notice.subject;
      this.choicedNotifierTypes = notifierTypes;
      this.createTime = notice.createTime;
    }, error => {
      Materialize.toast('Ошибка при загрузке данных', 2500);
    })
  }
}, error => {
  Materialize.toast('Ошибка при загрузке данных', 2500);
});
```

```
  this.refresh();
 }

 ngOnDestroy(): void {
  Materialize.Toast.removeAll();
  clearInterval(this.timer);
  if (!this.isAddAllowed) {
   if (this.choicedNotifierTypes.length !== 0) {
    this.save()
     .subscribe(() => {
      Materialize.toast('Все изменения сохранены', 2500);
     }, () => {
      Materialize.toast('Ошибка при сохранении', 2500);
     });
   }
  }
 }

 changeNotifierType (values) {
  this.choicedNotifierTypes = values;
  console.log(this.choicedNotifierTypes)
 }

 isNotInChoicedAccount(account: any): boolean {
  let exists = false;
  this.choicedAccounts.forEach((o) => {
   if (o.id === account.id) {
    exists = true;
   }
  });
  return !exists;
 }

 onFocusAndChangeAccountInput() {
  if (this.searchTextAccounts !== undefined) {
   if (this.searchTextAccounts.length > 0) {
    this.accountService
     .getAccounts()
     .then((complete: Response) => {
      this.findedAccounts = [];
      const accounts = complete.json();
      for (const account of accounts) {
       const currentAccount = new Account();
       currentAccount.id = account.id;
       currentAccount.name = account.name;
       currentAccount.username = account.username;
       currentAccount.role = account.role;
       currentAccount.groups = account.groups;
       currentAccount.contacts = account.contacts;

       if ((!isNullOrUndefined(currentAccount.name) && currentAccount.name.includes(this.searchTextAccounts))
        && this.isNotInChoicedAccount(account)) {
        this.findedAccounts.push(currentAccount);
        continue;
       }

                                                    if    ((!isNullOrUndefined(currentAccount.username)    &&
currentAccount.username.includes(this.searchTextAccounts))
```

```
            && this.isNotInChoicedAccount(account)) {
            this.findedAccounts.push(currentAccount);
            continue;
           }

                        if (!isNullOrUndefined(currentAccount.contacts.find(c => !isNullOrUndefined(c.data) &&
c.data.includes(this.searchTextAccounts)))
            && this.isNotInChoicedAccount(account)) {
            this.findedAccounts.push(currentAccount);
            continue;
           }
          }
        }).catch((error: any) => {
         console.log(error);
       });
     } else {
      this.findedAccounts = [];
     }
    }
  }

  choiceAccount(o: any) {
    if (!this.choicedAccounts.includes(o)) {
      this.choicedAccounts.push(o);
      this.findedAccounts = this.findedAccounts.filter(x => x !== o);
      this.searchTextAccounts = '';
    }
  }

  deleteChoicedAccount(p: any) {
    this.choicedAccounts = this.choicedAccounts.filter(x => x !== p);
    this.findedAccounts.push(p);
  }

  clickAccountChoiceComponent() {
    event.stopPropagation();
  }

  refreshAccountChoiceComponent(event: any) {
    this.findedAccounts = [];
  }

  prepareAccountText (account: Account) {
    return !isNullOrUndefined(account.name) && account.name !== '' ? account.name :
      (!isNullOrUndefined(account.username) && account.username !== ''  ? account.username :
          account.contacts.map(c => c.contactType.charAt(0) + c.contactType.slice(1).toLowerCase() + ': ' + c.data).join(',
'));
    }

  isNotInChoicedGroup(group: any): boolean {
    let exists = false;
    this.choicedGroups.forEach((o) => {
      if (o.id === group.id) {
        exists = true;
      }
    });
    return !exists;
  }
```

```
onFocusAndChangeGroupInput() {
  if (this.searchTextGroups !== undefined) {
    if (this.searchTextGroups.length > 0) {
      this.groupService.getGroups().then((data: any) => {
        const responseGroups = data.json();
        this.findedGroups = [];
        for (const group of responseGroups) {
          if (group != null
            && group.name.includes(this.searchTextGroups)
            && this.isNotInChoicedGroup(group)) {
            this.findedGroups.push({
              id: group.id,
              name: group.name,
              head: {
                id: group.head.id,
                name: group.head.name,
                username: group.head.username,
                role: group.head.role,
                groups: group.head.groups,
                contacts: group.head.contacts,
                password: ''
              },
              eventGroup: group.isEventGroup,
              accounts: []
            });

            for (const account of group.accounts) {
              const currentAccount = new Account();
              currentAccount.id = account.id;
              currentAccount.name = account.name;
              currentAccount.username = account.username;
              currentAccount.role = account.role;
              currentAccount.groups = account.groups;
              currentAccount.contacts = account.contacts;

              this.findedGroups[this.findedGroups.length - 1].accounts.push(currentAccount);
            }
          }
        }
      }).catch((error: any) => {
        console.log('error ' + error);
      });
    } else {
      this.findedGroups = [];
    }
  }
}

choiceGroup(o: any) {
  if (!this.choicedGroups.includes(o)) {
    this.choicedGroups.push(o);
    this.findedGroups = this.findedGroups.filter(x => x !== o);
    this.searchTextGroups = '';
  }
}

deleteChoicedGroup(p: any) {
```

```
    this.choicedGroups = this.choicedGroups.filter(x => x !== p);
    this.findedGroups.push(p);
  }

  clickGroupChoiceComponent() {
    event.stopPropagation();
  }

  refreshGroupChoiceComponent(event: any) {
    this.findedGroups = [];
  }

  showAddRecipientsDialog(typeOfRecipients: string) {
    switch (typeOfRecipients) {
      case 'ACCOUNT':
        this.addRecipientsDialog.show(typeOfRecipients, this.choicedAccounts);
        break;
      case 'GROUP':
        this.addRecipientsDialog.show(typeOfRecipients, this.choicedGroups);
        break;
    }
  }

  changeRecipients(data) {
    switch (data.recipientsType) {
      case 'ACCOUNT':
        this.choicedAccounts = data.selectedItems.slice();
        break;
      case 'GROUP':
        this.choicedGroups = data.selectedItems.slice();
    }
  }

  showUpdateDialog(account: Account) {
    this.updateDialog.showDialog(account);
  }

  showUpdateGroupDialog (group: Group) {
    this.updateGroupDialog.showDialog(group);
  }

  checkAccountContacts (account) {
      return isNullOrUndefined(this.choicedNotifierTypes.map(type => !isNullOrUndefined(account.contacts.find((c) =>
c.contactType === type))).find(type => type === false));
  }

  checkGroupContacts (group) {
    return isNullOrUndefined(group.accounts.map(a => this.checkAccountContacts(a)).find(c => c === false));
  }

  hide (isAddAllowed?: boolean) {
    this.isAddAllowed = isAddAllowed;
    if (isAddAllowed) {
      this.send();
    } else {
      this.router.navigateByUrl('/admin/notifications');
    }
  }
```

```
private validate (): boolean {
  let isValid = true;
  for (const notifierType of this.choicedNotifierTypes) {
    for (const account of this.choicedAccounts) {
      if (isNullOrUndefined(account.contacts.find(c => c.contactType === notifierType.toUpperCase()))) {
        isValid = false;
        const name = this.prepareAccountText(account);
        Materialize.toast('Для контакта "' + name + '" не указан ' + notifierType, 5000);
      }
    }
    for (const group of this.choicedGroups) {
      for (const a of group.accounts) {
        if (isNullOrUndefined(a.contacts.find(c => c.contactType === notifierType.toUpperCase()))) {
          isValid = false;
          const name = this.prepareAccountText(a);
          Materialize.toast('Для контакта "' + name + '" в группе "' + group.name + '" не указан ' + notifierType, 5000);
        }
      }
    }
    if (notifierType === 'EMAIL' && this.subject.length === 0) {
      isValid = false;
      document.getElementById('subject').classList.add('invalid');
      Materialize.toast('Укажите тему сообщения для Email рассылки', 5000);
    }
  }
  if (this.message.length === 0) {
    isValid = false;
    document.getElementById('message').classList.add('invalid');
    Materialize.toast('Напишите текст сообщения', 5000);
  }
  if (this.message.length > 250) {
    isValid = false;
    document.getElementById('message').classList.add('invalid');
    Materialize.toast('Слишком длинное сообщение', 5000);
  }

  if (this.choicedGroups.length === 0 && this.choicedAccounts.length === 0) {
    isValid = false;
    Materialize.toast('Для рассылки выберите хотя бы одну группу или контакт', 5000);
  }

  return isValid;
}

private send () {
  if (this.validate()) {
    document.getElementById('send').classList.add('disabled');
    this.isInProcess = true;
    const sendNotices: Notice[] = [];
    const choicedAccounts = this.choicedAccounts;
    const choicedGroups = this.choicedGroups;

    const notifierTypes = this.choicedNotifierTypes;
    let noticeCounter = 0;
    let ids;
    if (!isNullOrUndefined(this.ids)) {
      ids = this.ids.split(',');
```

```
    }
    for (const type of notifierTypes) {
      const currentNotification = new Notice();
      console.log(type);
      currentNotification.id = !isNullOrUndefined(ids) && ids.length > noticeCounter ? ids[noticeCounter] : null;
      currentNotification.status = 'NEW';
      currentNotification.message = this.message;
      currentNotification.purpose = 'INFO';
      currentNotification.notifierType = type;
      currentNotification.subject = this.subject;
      currentNotification.createTime = this.createTime;
      currentNotification.recipients = [];
      for (const account of choicedAccounts) {
        currentNotification.recipients.push({
          data: account.contacts.find(c => c.contactType === type.toUpperCase()).data,
          recipientId: account.id,
          type: 'SINGLE'
        })
      }
      for (const group of choicedGroups) {
        currentNotification.recipients.push({
          data: group.accounts.map(a => a.contacts.find(c => c.contactType === type.toUpperCase()).data).join(','),
          recipientId: group.id,
          type: 'GROUP'
        })
      }
      noticeCounter++;
      sendNotices.push(currentNotification);
    }
    this.noticeService.sendListNotices(sendNotices)
      .subscribe(() => {
        this.router.navigateByUrl('/admin/notifications');
      }, () => {
        Materialize.toast('Ошибка при отправке сообщения', 2500);
        this.router.navigateByUrl('/admin/notifications');
      });
  }
}

private save () {
  document.getElementById('send').classList.add('disabled');
  this.isInProcess = true;
  const sendNotices: Notice[] = [];
  const choicedAccounts = this.choicedAccounts;
  const choicedGroups = this.choicedGroups;

  const notifierTypes = this.choicedNotifierTypes;
  let noticeCounter = 0;
  let ids;
  if (!isNullOrUndefined(this.ids)) {
    ids = this.ids.split(',');
  }
  for (const type of notifierTypes) {
    const currentNotification = new Notice();
    console.log(type);
    currentNotification.id = !isNullOrUndefined(ids) && ids.length > noticeCounter ? ids[noticeCounter] : null;
    currentNotification.status = 'NOT_SENT';
    currentNotification.message = this.message;
```

```
    currentNotification.purpose = 'INFO';
    currentNotification.notifierType = type;
    currentNotification.subject = this.subject;
    currentNotification.createTime = this.createTime;
    currentNotification.recipients = [];
    for (const account of choicedAccounts) {
     const contact = account.contacts.find(c => c.contactType === type.toUpperCase());
     currentNotification.recipients.push({
      data: isNullOrUndefined(contact) ? '' : contact.data,
      recipientId: account.id,
      type: 'SINGLE'
     })
    }
    for (const group of choicedGroups) {
     currentNotification.recipients.push({
      data: group.accounts.map(a => {
       const contact = a.contacts.find(c => c.contactType === type.toUpperCase());
       return isNullOrUndefined(contact) ? '' : contact.data;
      }).join(','),
      recipientId: group.id,
      type: 'GROUP'
     })
    }
    noticeCounter++;
    sendNotices.push(currentNotification);
   }
   return this.noticeService.sendListNotices(sendNotices);
  }

  private refresh () {
   this.timer = setInterval(() => {
    if (this.choicedNotifierTypes.length !== 0) {
     this.save()
      .subscribe((data: Response[]) => {
       this.ids = data.map(d => d.json().id).join(',');
       this.createTime = data[0].json().createTime;
       this.isInProcess = false;
       document.getElementById('send').classList.remove('disabled');
       Materialize.toast('Все изменения сохранены', 2500);
      }, () => {
       this.isInProcess = false;
       document.getElementById('send').classList.remove('disabled');
       Materialize.toast('Ошибка при сохранении', 2500);
      });
    }
   }, 60000);
  }
}
```