Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |  |
| --- | --- |
| Допускаю к защите |  |
| Руководитель |  |
|  | подпись |
|  | Л.С. Вахрушева |
|  | И.О. Фамилия |

|  |
| --- |
| Разработка программного обеспечения для информационной системы |
| художественной школы |

наименование темы

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине

|  |  |  |
| --- | --- | --- |
| ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНЫХ КОМПЛЕКСОВ | | |
|  | 1.008.00.00 - ПЗ (номварианта - № |  |

обозначение документа

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | АСУб-20-2 |  |  |  | М.С. Выборова |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | Л.С. Вахрушева |
|  |  |  |  | подпись |  | И.О. Фамилия |

Курсовой проект защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Иркутск 2023 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

ЗАДАНИЕ

НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| По курсу | | Технологии разработки программных комплексов | | |
| Студенту | | Выборовой М.С. | | |
|  | | (фамилия, инициалы) | | |
| Тема проекта | Разработка программного обеспечения для | | | |
| информационной системы художественной школы | | | | |
| Исходные данные: | | | | |
| Предпроектное обследование объекта автоматизации и основные проектные | | | | |
| решения по серверной части, а также разработка | | |  | |
| информационной системы для художественной школы | | | | |
|  | | | | |
| |  |  | | --- | --- | | Рекомендуемая литература: |  |  |  | | --- | | 1. Гутгарц Р.Д Проектирование автоматизированных систем обработки информации и управления: учебное пособие для академического бакалавриата. – М.: Издательство Юрайт, 2019. | | 1. Node.js. Разработка серверных веб-приложений в JavaScript / Хэррон Д. А. А. – М.: ДМК Пресс, 2012. – 144 с.: ил. | | 1. Проектирование информационных систем : учебник и практикум для академического бакалавриата / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общ. ред. Д. В. Чистова. – М. : Издательство Юрайт, 2016. | | 1. Рудинский И.Д. Технология проектирования автоматизированных систем обработки информации и управления : учебное пособие для вузов / И. Д. Рудинский. - Москва : Горячая линия - Телеком, 2011. | | | | |  |

Графическая часть на \_\_\_\_\_\_\_\_\_\_\_\_\_\_ листах.

Дата выдачи задания « 18 » сентября 2023 г.

|  |  |  |  |
| --- | --- | --- | --- |
| Задание получил |  |  | М.С. Выборова |
|  | подпись |  | И.О. Фамилия |

Дата представления проекта руководителю « 20 декабря » 2023 г.

|  |  |  |
| --- | --- | --- |
| Руководитель курсового проектирования |  | Л.С. Вахрушева |
|  | подпись | И.О. Фамилия |

# Содержание

[Введение 3](#_Toc153637454)

[1 Анализ 4](#_Toc153637455)

[1.1 Описание предметной области 4](#_Toc153637456)

[1.1.1 Введение в предметную область 4](#_Toc153637457)

[1.1.2 Цель проекта 4](#_Toc153637458)

[1.1.3 Задачи для достижения цели 4](#_Toc153637459)

[1.1.4 Проблема 5](#_Toc153637460)

[1.1.5 Актуальность 5](#_Toc153637461)

[1.2 Обзор существующих программных средств 6](#_Toc153637462)

[1.3 Процессы AS-IS и TO-BE 18](#_Toc153637463)

[1.4 Описание вариантов использования 29](#_Toc153637464)

[1.5 Выработка требований 32](#_Toc153637465)

[2 Проектирование 33](#_Toc153637466)

[2.1 Выбор и обоснование средств проектирования и реализации 33](#_Toc153637467)

[2.1.1 Средства проектирования 33](#_Toc153637468)

[2.1.2 Средства реализации 33](#_Toc153637469)

[2.2 Проектирование архитектуры приложения 34](#_Toc153637470)

[2.3 Проектирование хранилища данных 35](#_Toc153637471)

[2.3.1 Основные сущности 35](#_Toc153637472)

[2.3.2 Логическая модель данных 35](#_Toc153637473)

[2.4 Проектирование пользовательского интерфейса 35](#_Toc153637474)

[3 Разработка 48](#_Toc153637475)

[3.1 Структура проекта 48](#_Toc153637476)

[3.2 Frontend 48](#_Toc153637477)

[3.3 Backend 54](#_Toc153637478)

[Заключение 59](#_Toc153637479)

[Список использованных источников 60](#_Toc153637480)

# Введение

В настоящее время внедрение информационных технологий в образовательные учреждения становится неотъемлемой частью повседневной деятельности, направленной на повышение эффективности и качества образовательного процесса. Художественные школы, ориентированные на развитие творческих навыков и талантов учащихся – не исключение. Для оптимизации управленческих и педагогических процессов, а также обеспечения удобства взаимодействия сотрудников и обучающихся представляется актуальной задачей внедрение информационной системы.

Целью данного курсового проекта является проведение предпроектного обследования объекта автоматизации – художественной школы, а также разработка основных проектных решений по серверной части информационной системы. В рамках исследования будет осуществлен анализ текущих процессов управления и обучения в школе, выявлены существующие проблемы и потребности, определены ключевые требования к будущей информационной системе, а также предложены принципиальные проектные решения по интерфейсу и базе данных.

# Анализ

# Описание предметной области

# Введение в предметную область

В современном мире искусство остается одним из наиболее важных и вдохновляющих аспектов культуры. Оно не только обогащает наши жизни, но и способствует развитию творческого мышления и самовыражения. В этом контексте художественные школы играют решающую роль, предоставляя молодым талантам и любителям искусства возможность развивать свой потенциал и открывать мир художественных выражений.

Художественная школа представляет собой специализированное образовательное учреждение, ориентированное на развитие и совершенствование художественных навыков и творческого потенциала своих учеников.

В школе есть директор, который отвечает за общее управление учреждением, организацию финансовых вопросов и взаимодействие с родителями. Также она имеет профессиональных учителей, специализирующихся в разных областях искусства. Они проводят занятия, разрабатывают учебные программы и следят за успехами учеников. Основными участниками образовательного процесса являются ученики, которые обычно делятся на различные группы, в которые они самостоятельно записываются.

Художественная школа разрабатывает учебные программы, которые охватывают различные аспекты искусства. Эти программы включают «Живопись», «Декоративно-прикладное творчество» и «Изобразительное искусство».

Основные процессы, которые протекают в школе, включают в себя:

1. обучение учеников;
2. учет успеваемости и посещаемости;
3. мониторинг успеваемости и посещаемости;
4. взаимодействие с родителями несовершеннолетних учеников;
5. оплата обучения.

Все эти компоненты художественной школы взаимодействуют для обеспечения качественного художественного образования, развития художественных способностей и формирования творческой личности учащихся.

# Цель проекта

Цель данного проекта – создание информационной системы, которая способствует улучшению качества образования и оптимизации бизнес-процессов художественной школы, что в итоге приводит к повышению ее эффективности.

# Задачи для достижения цели

Для достижения цели необходимо выполнить ряд задач:

1. Изучить существующие бизнес-процессы художественной школы: провести анализ текущих учебных и административных процессов, выявив основные слабые места и узкие места, которые могут быть улучшены.
2. Проанализировать потребности пользователей: провести опросы среди администрации, учителей, учеников и родителей, чтобы определить их ожидания от новой информационной системы и функциональные требования.
3. Исследовать рынок и доступные решения: изучить существующие информационные системы, предназначенные для художественных школ, оценить их возможности и сравнить с потребностями школы.
4. Разработать техническое задание: сформулировать требования к функциональности, интерфейсу и техническим характеристикам информационной системы.
5. Разработать и протестировать систему: создать информационную систему с учетом собранных требований, провести тестирование на соответствие и эффективность.
6. Внедрить систему и обучить персонал: запустить систему в эксплуатацию, обеспечить поддержку пользователей и обучить административный и учебный персонал работе с ней.

# Проблема

Разработка проекта "Информационная система для художественной школы" была спровоцирована двумя ключевыми проблемами, влияющими на эффективность управления и образовательных процессов в данном учебном учреждении.

Первая проблема заключает в том, что художественные школы часто сталкиваются с высокой административной нагрузкой, включая учет учеников, учет посещаемости и успеваемости, вопросы с оплатой обучения и составление расписания занятий. Эти процессы часто выполняются вручную или с использованием устаревших систем, что может привести к ошибкам и замедлению работы.

И вторая проблема – это сложности в мониторинге успеваемости учеников. Отслеживание успехов и прогресса учеников требует значительных усилий со стороны учителей и администрации. Недостаток системы для автоматизации мониторинга успеваемости затрудняет оперативное принятие решений и предоставление обратной связи ученикам.

# Актуальность

Создание информационной системы для художественной школы имеет высокую актуальность в современном образовательном и технологическом контексте.

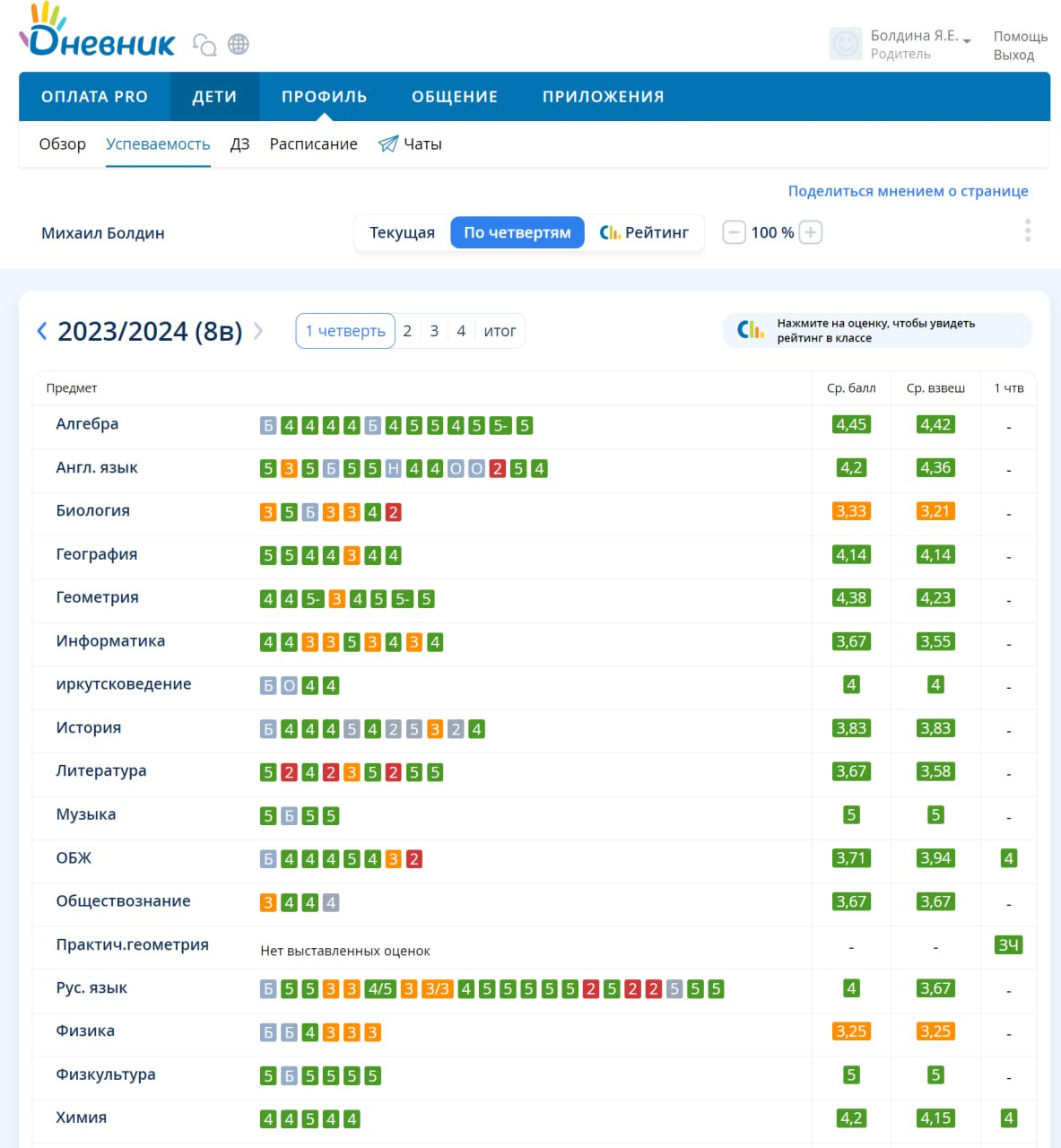
С развитием технологий и изменением образовательных потребностей студентов старые методы управления и обучения становятся устаревшими. Создание информационной системы позволяет оптимизировать учебные процессы, предоставляя учителям и администрации инструменты для более эффективной работы.

Автоматизация учета, финансовых процессов и мониторинга успеваемости облегчит административные задачи, позволит управлять ресурсами более эффективно и сократить операционные издержки.

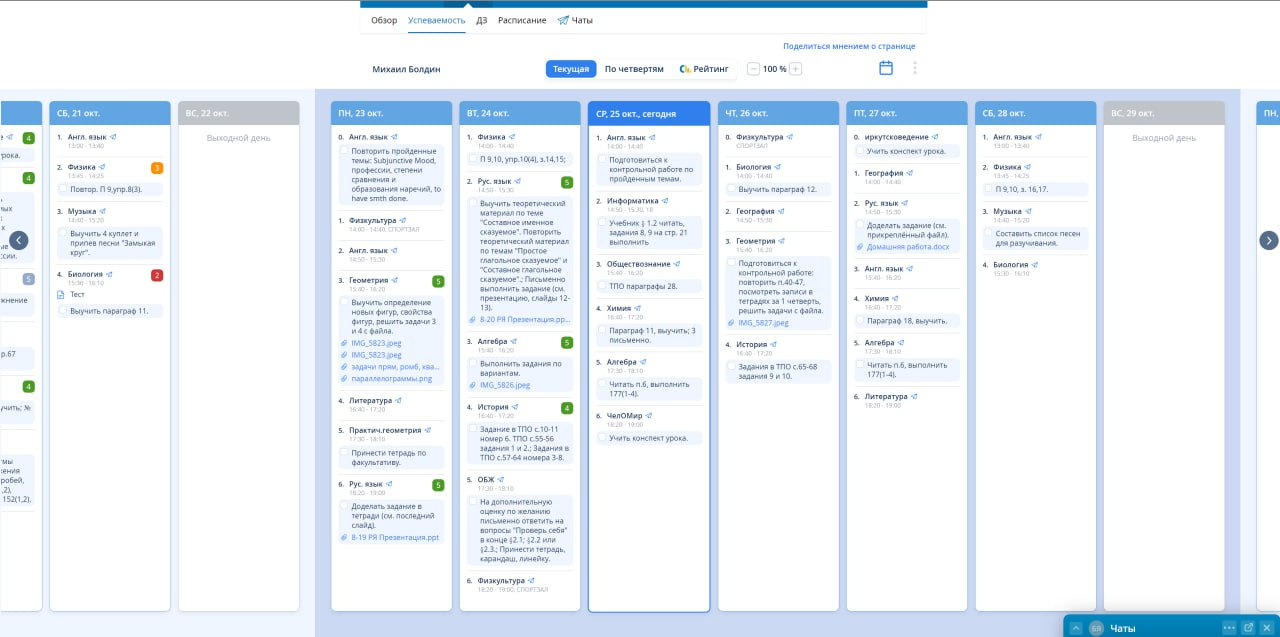
И, поскольку, целью любой образовательной организации является обеспечение высокого качества образования, то внедрение такой системы может поспособствовать этому, так как информационная система поможет школе следить за успехами учеников, анализировать данные и на основе этих данных вносить коррективы в учебный процесс для достижения этой цели.

# Обзор существующих программных средств

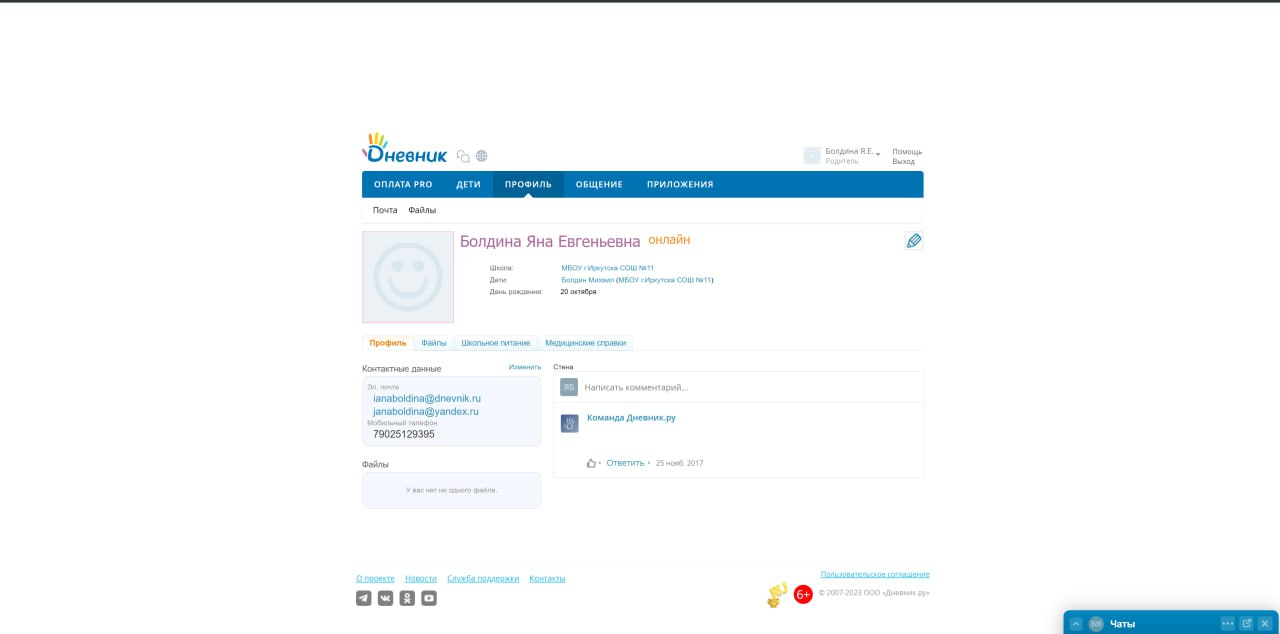
**ООО «Дневник.ру» [1]** – российская IT-компания в сфере образовательных технологий, разработчик решений и единой электронной образовательной среды для учителей, учеников и их родителей, администраций образовательных организаций, а также представителей органов исполнительной власти. Фрагмент интерфейса успеваемости представлен на рисунке 1.1. Интерфейс расписания изображен на рисунке 1.2. Интерфейс личного кабинета показан на рисунке 1.3.



**Рисунок 1.1 – Фрагмент интерфейса успеваемости «Дневник.ру»**



**Рисунок 1.2 – Интерфейс расписания «Дневник.ру»**



**Рисунок 1.3 – Интерфейс профиля «Дневник.ру»**

Проанализируем интерфейс рассматриваемой системы.

**Общая оценка интерфейса (первое впечатление).** Первое впечатление от пользовательского интерфейса является положительным. Интерфейс приложения обладает интуитивной понятностью, что облегчает навигацию и взаимодействие пользователя с системой. Это способствует быстрому и беспрепятственному освоению приложения, что является важным аспектом комфортного пользовательского опыта.

**Цветовое решение интерфейса.** Преобладающим цветовым решением в пользовательском интерфейсе является гамма синего цвета. Синий цвет часто применяется в веб-дизайне, исходя из его способности визуализировать спокойствие, открытость и ощущение безопасности. Он создает атмосферу комфорта, доверия и ассоциируется с высоким профессионализмом. Учитывая, что пользователи взаимодействуют с системой ежедневно, использование синего цвета способствует легкому восприятию интерфейса и снижает утомляемость глаз.

Дополнительными акцентными цветами в дизайне интерфейса являются бордовый и зеленый. Эти цвета гармонично сочетаются с главным синим цветом и были подобраны в соответствии с принципами цветовой схемы. Особенностью является применение разных цветов для обозначения разных оценок, что обеспечивает визуальное разграничение между ними. Однако стоит отметить, что оценки 4 и 5 отображаются одним и тем же цветом, что может создать путаницу при просмотре и интерпретации оценок.

**Объём и структура представленной информации.** Страницы интерфейса приложения организованы с учетом принципа минимизации избыточной информации. Вся предоставленная информация структурирована и представлена в соответствии с выбранным разделом, что способствует удобству и пониманию пользователей. Важно отметить, что для улучшения удобства восприятия некоторая информация, например, такая как успеваемость, представлена в виде таблицы, что упрощает визуальное восприятие и взаимодействие с ней.

**Наличие и структура меню.** На каждой странице интерфейса присутствует горизонтальное меню, которое содержит перечень основных модулей системы. Это меню обеспечивает навигацию и быстрый доступ к ключевым функциональным разделам приложения.

**Удобство навигации.** Навигация по страницам сделана достаточно удобно. Главные разделы отображаются в верхнем меню, а соответствующие подразделы представлены непосредственно под главным меню. Это обеспечивает легкость доступа к необходимым функциональным разделам приложения. Кроме того, на каждой странице отображается путь, указывающий текущее местоположение пользователя в системе, что способствует легкой ориентации.

**Наличие и информативность подсказок (помощь).** В интерфейсе приложения реализована функциональность подсказок, предназначенная для предоставления пользователю дополнительной информации и рекомендаций. Кроме того, имеется специальная кнопка «Помощь», которая предоставляет дополнительные средства для получения поддержки и информационной помощи.

**Удобство форм для ввода информации.** Формы для ввода данных в интерфейсе приложения вызывают определенные затруднения в использовании. Например, в полях для ввода номера телефона не предусмотрена маска для упрощения ввода, а поле для имени допускает ввод цифровых символов. Кроме того, отмечается избыточный размер поля для ввода номера телефона, что является неэффективным использованием пространства интерфейса. Эти аспекты требуют коррекции с целью повышения удобства использования форм ввода.

**Возможность поиска информации (технология организации поиска, удобство, понятность, релевантность).** В системе отсутствует возможность поиска информации по странице, это может затруднить пользователям возможность быстрого и точного поиска необходимых данных, что является важным функциональным аспектом для обеспечения удобства взаимодействия с системой.

**Основные функциональные задачи (сформулировать с использованием функций управления) в соответствии с назначением системы анализируемого типа.**

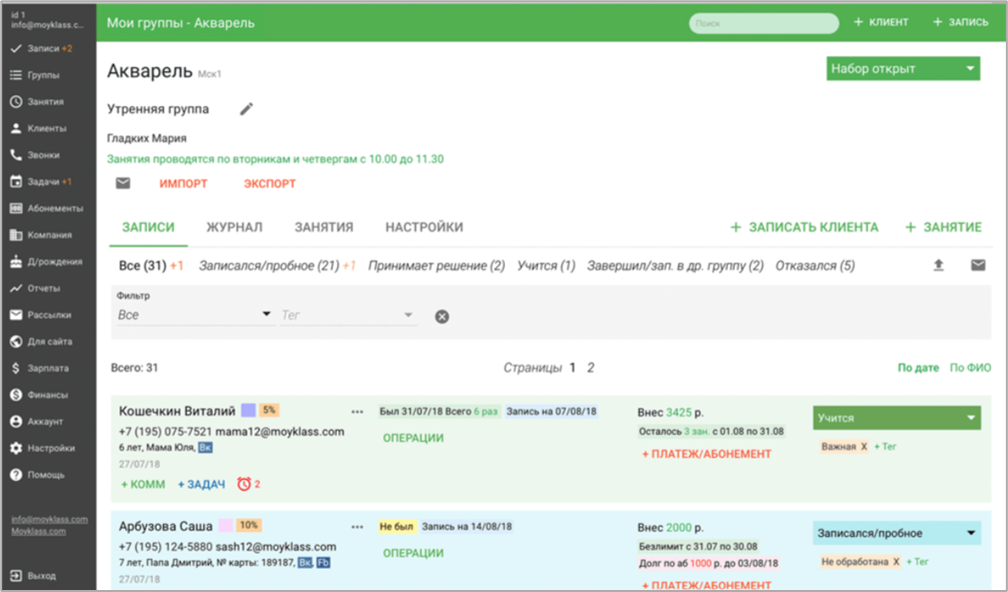
Основные функциональные возможности:

* круглосуточный доступ к оценкам, расписанию и домашним заданиям;
* электронный журнал;
* ввод расписания, возможность его редактирования;
* автоматическое составление отчетов;
* проведение дистанционного обучения;
* уведомления об учебе;
* составление портфолио.

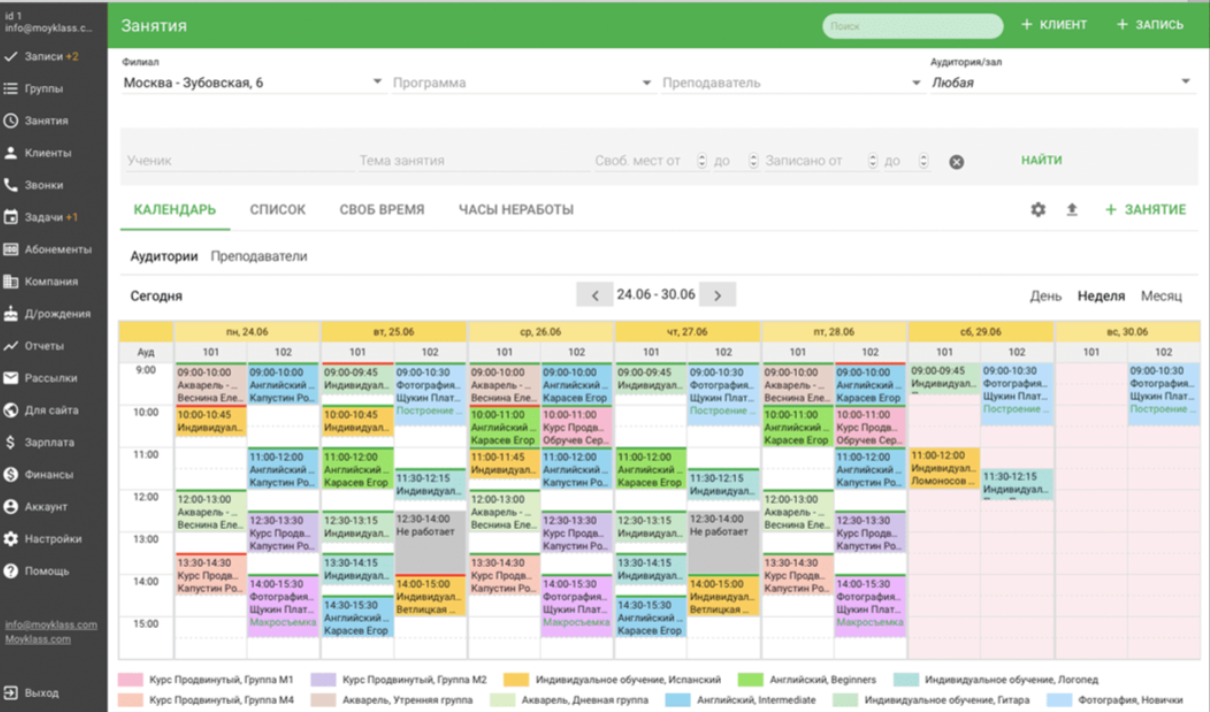
Несмотря на большое количество полезных функций, нет возможности автоматического создания расписания, а также не выделены функции для роли «Директор школы». Также для преподавателей представление отчета возможно только в табличном виде, без графиков.

**Другие функциональные возможности (личный кабинет, корзина, отзывы и др.).** Среди дополнительных функциональных возможностей следует выделить наличие личного кабинета для каждого пользователя, предоставляющего доступ к персональным настройкам и данным. Кроме того, в системе имеется функциональность для распространения сообщений и уведомлений о различных мероприятиях и событиях. Пользователи могут также обращаться к ссылкам и ресурсам, связанным с образовательными приложениями и дополнительными образовательными ресурсами.

«Мой класс» [2] – CRM для учета клиентов школ и детских центров: занятия, посещения, платежи, продажи. Пример интерфейса для журнала изображен на рисунке 1.4, интерфейс расписания представлен на рисунке 1.5.



**Рисунок 1.4 – Интерфейс ведения групп «Мой класс»**



**Рисунок 1.5 – Интерфейс расписания «Мой класс»**

Проанализируем интерфейс данной системы.

**Общая оценка интерфейса (первое впечатление).** При первом взгляде на пользовательский интерфейс возникают определенные негативные впечатления. Интерфейс выглядит насыщенным информацией и вызывает затруднения в понимании его структуры. Это создает сложности в навигации и взаимодействии пользователя с системой.

**Цветовое решение интерфейса.** Основной цвет, преобладающий в дизайне интерфейса, является зеленым. Этот выбор цвета в дизайне способствует созданию ощущения стабильности и гармонии, что важно для комфортного и ежедневного использования системы.

Дополнительными акцентными цветами в дизайне интерфейса являются серый и оранжевый. Эти цвета гармонично сочетаются с основным зеленым цветом.

**Объём и структура представленной информации.** Страницы интерфейса характеризуются избыточным объемом представленной информации. Несмотря на это, она отображается в разнообразных форматах. Некоторые данные структурированы в виде таблиц, что обеспечивает более удобное восприятие информации.

**Наличие и структура меню.** В интерфейсе приложения представлено вертикальное меню с большим количеством разделов. Каждый раздел снабжен иконкой, которая наглядно отражает его специфику. Эта дополнительная визуальная характеристика облегчает пользователям навигацию и понимание контекста разделов.

**Удобство навигации.** Навигация по страницам сделана удобно. Пользователь выбирает необходимый ему раздел в боковом меню. Если в выбранном разделе есть другие подразделы, то они обычно представлены в виде горизонтального списка. Минусом является то, что пользователю не показывают текущее местоположение на странице.

**Наличие и информативность подсказок (помощь).** В интерфейсе приложения реализованы различные подсказки, которые помогают пользователю при использовании системы. Также имеется специальная кнопка «Помощь», которая предоставляет дополнительные средства для получения поддержки и информационной помощи.

**Удобство форм для ввода информации.** Формы для ввода данных в интерфейсе приложения организованы с учетом удобства пользователей. Для ввода числовых значений предусмотрены специальные поля, что способствует более точному и удобному вводу числовых данных.

**Возможность поиска информации (технология организации поиска, удобство, понятность, релевантность).** В интерфейсе системы предусмотрена функциональность поиска, позволяющая осуществлять поиск информации как на текущей странице, так и в данных с использованием выбранных критериев. Это существенно упрощает взаимодействие пользователей с системой и значительно ускоряет процесс поиска необходимой информации.

**Основные функциональные задачи (сформулировать с использованием функций управления) в соответствии с назначением системы анализируемого типа.**

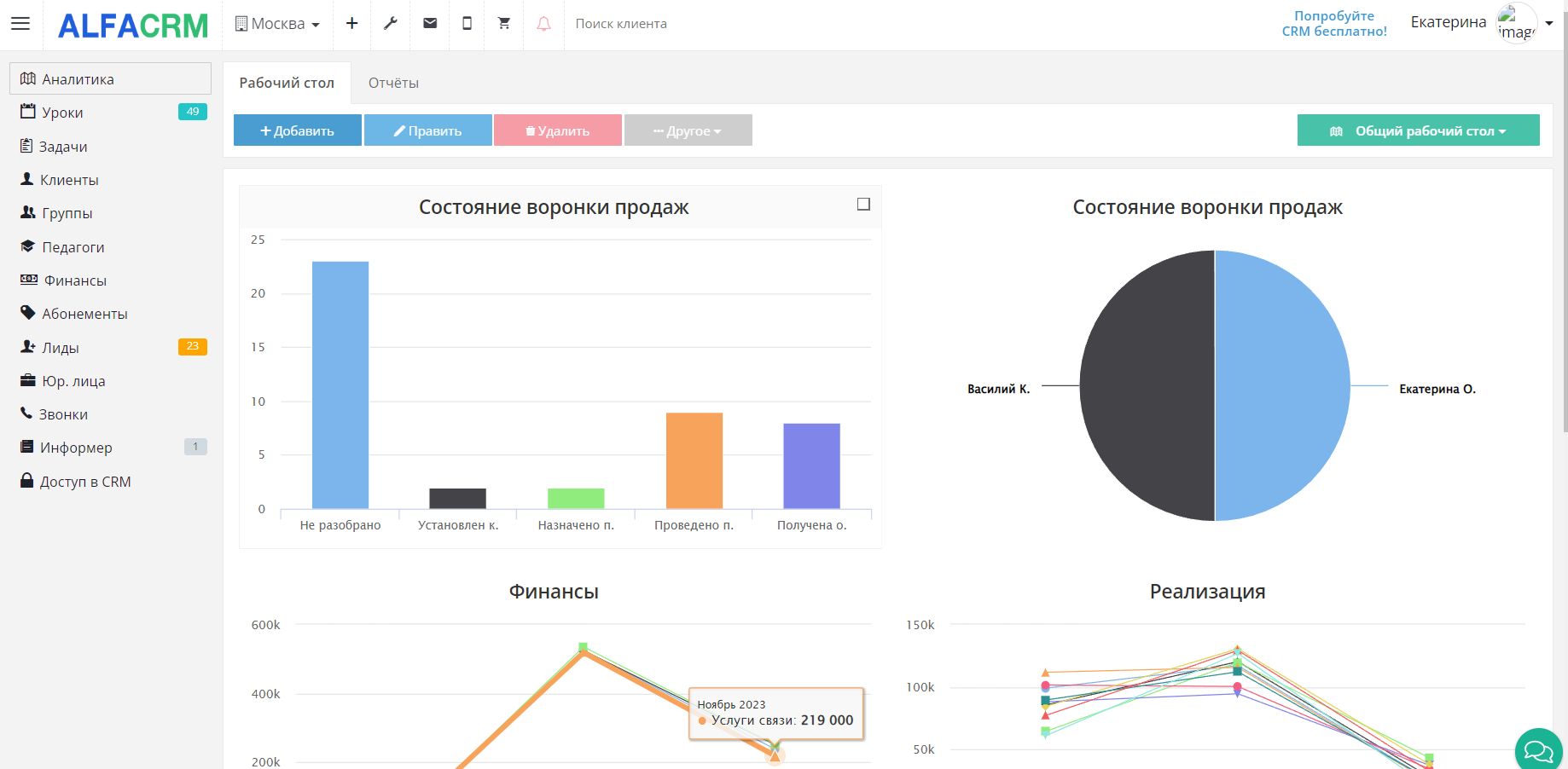
Основные функциональные возможности:

* расписание занятий;
* электронный журнал;
* различные оповещения и рассылки (о новых записях, о начале занятий, о посещениях и пропусках и т.д.);
* проведение дистанционного обучения;
* онлайн-оплата;
* автоматический расчет зарплаты сотрудников;
* автоматическое составление отчетов;
* онлайн-запись на занятия;
* составление портфолио.

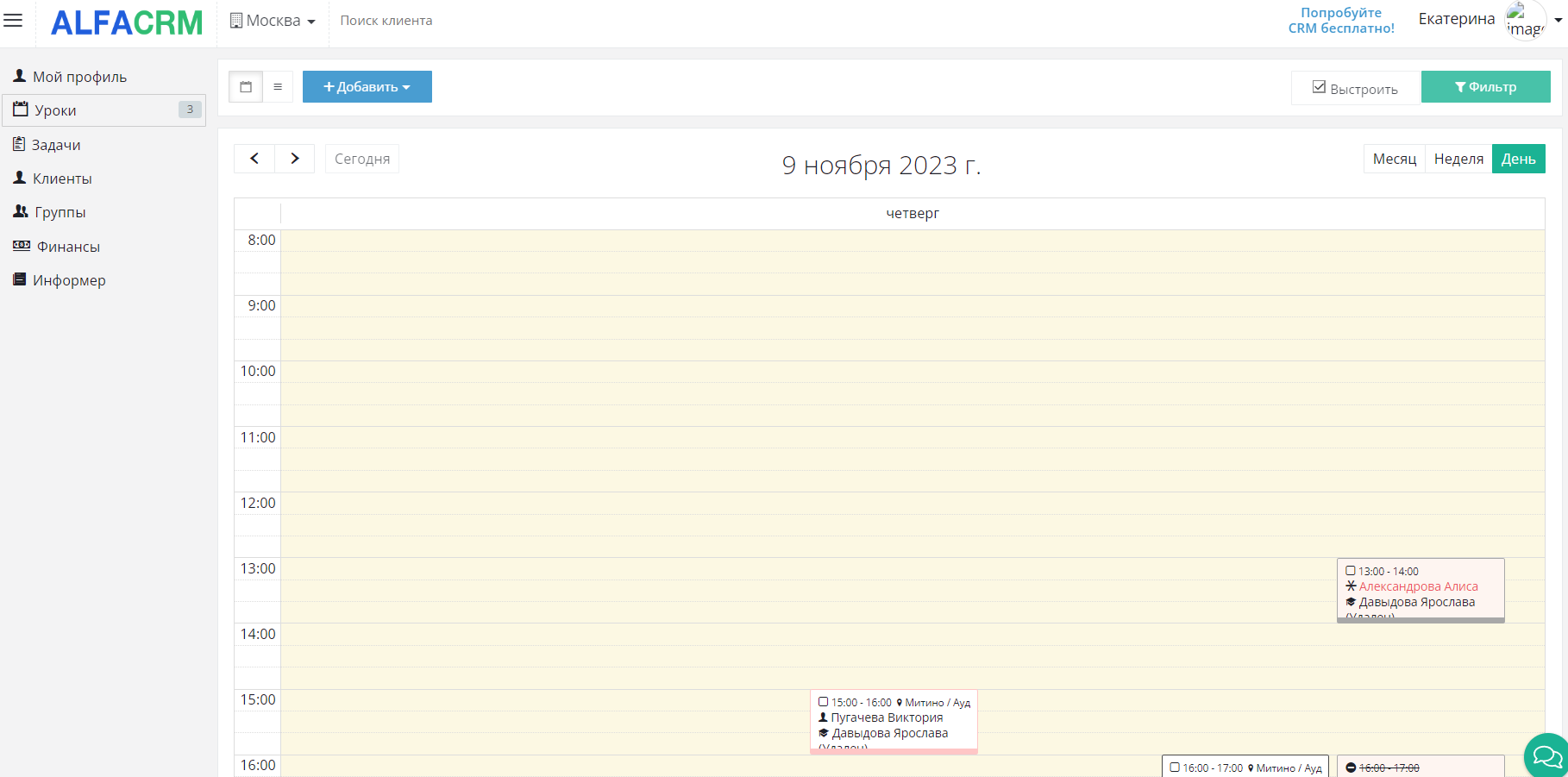
Несмотря на большое количество полезных функций, нет возможности автоматического создания расписания. Также для преподавателей представление отчета возможно только в табличном виде, без графиков.

**Другие функциональные возможности (личный кабинет, корзина, отзывы и др.).** Среди дополнительных функциональных возможностей следует выделить наличие личного кабинета для каждого пользователя, предоставляющего доступ к персональным настройкам и данным. Кроме того, в системе имеется функциональность для распространения сообщений и уведомлений о различных событиях. Также пользователи могут общаться с помощью различных мессенджеров внутри системы, например, WhatsApp. Пользователи могут также обращаться к ссылкам и ресурсам, связанным с образовательными приложениями и дополнительными образовательными ресурсами.

«ALFACRM» [3] – управление учебным центром, языковой, музыкальной, спортивной школой. Система автоматизирует основные процессы учебного центра, а также развивает бизнес. Пример интерфейса данной системы представлен на рисунках 1.6 – 1.7.



**Рисунок 1.6 – Интерфейс аналитики «ALFACRM»**



**Рисунок 1.7 – Фрагмент интерфейса расписания «ALFACRM»**

Проанализируем интерфейс данной системы.

**Общая оценка интерфейса (первое впечатление).** Первое впечатление от пользовательского интерфейса является положительным. Он отличается отсутствием избыточной информации, что способствует легкому восприятию и обеспечивает удобную навигацию между страницами.

**Цветовое решение интерфейса.** В дизайне интерфейса присутствуют акцентные цвета, такие как синий и зеленый, что является популярным и распространенным решением. Данное цветовое сочетание способствует созданию комфортного визуального восприятия, это особенно важно для пользователя в условиях длительного взаимодействия с системой. Кроме того, второстепенные элементы подчеркиваются использованием оттенков желтого цвета, его применение добавляет разнообразия в дизайн и подчеркивает важность этих элементов.

**Объём и структура представленной информации.** Интерфейс эффективно структурирован, предоставляя пользователю легкое восприятие информации без визуальной перегрузки. Данные организованы с учетом их функционального назначения, обеспечивая легкость восприятия. Например, задачи на текущий день представлены в формате канбан-доски, а расписание оформлено в виде таблицы, что упрощает структурирование информации.

**Наличие и структура меню.** В интерфейсе приложения предусмотрено вертикальное меню, при этом каждый раздел снабжен наглядной иконкой, отражающей его специфику. Дополнительно реализована функциональность скрытия меню, которой не было в предыдущих рассматриваемых системах.

**Удобство навигации.** Навигация по страницам в приложении реализована удобным образом. Пользователь осуществляет перемещение между разделами с использованием бокового меню. Кроме того, при переходе между разделами пользователю предоставляется информация о текущем местоположении, что обеспечивает ясность в структуре интерфейса и повышает удобство использования.

**Наличие и информативность подсказок (помощь).** В интерфейсе приложения предусмотрены разнообразные подсказки, направленные на содействие пользователю в процессе использования системы. Однако следует отметить отсутствие функциональной кнопки «Помощь», предназначенной для предоставления дополнительной поддержки и решения возможных трудностей, которые могут возникнуть в ходе взаимодействия с приложением. Это важный аспект, который следует учесть в рамках обновлений интерфейса для обеспечения максимальной поддержки пользователей при использовании системы.

**Удобство форм для ввода информации.** Формы для ввода данных в интерфейсе приложения сделаны достаточно удобно. Они снабжены понятными подсказками, а также оснащены механизмами проверки ввода, направленными на предотвращение опечаток и минимизацию возможных ошибок.

**Возможность поиска информации (технология организации поиска, удобство, понятность, релевантность).** В системе отсутствует возможность поиска информации. Но есть механизм сортировки списков, который может упростить задачу поиска информации в таблицах и различных списках.

**Основные функциональные задачи (сформулировать с использованием функций управления) в соответствии с назначением системы анализируемого типа.**

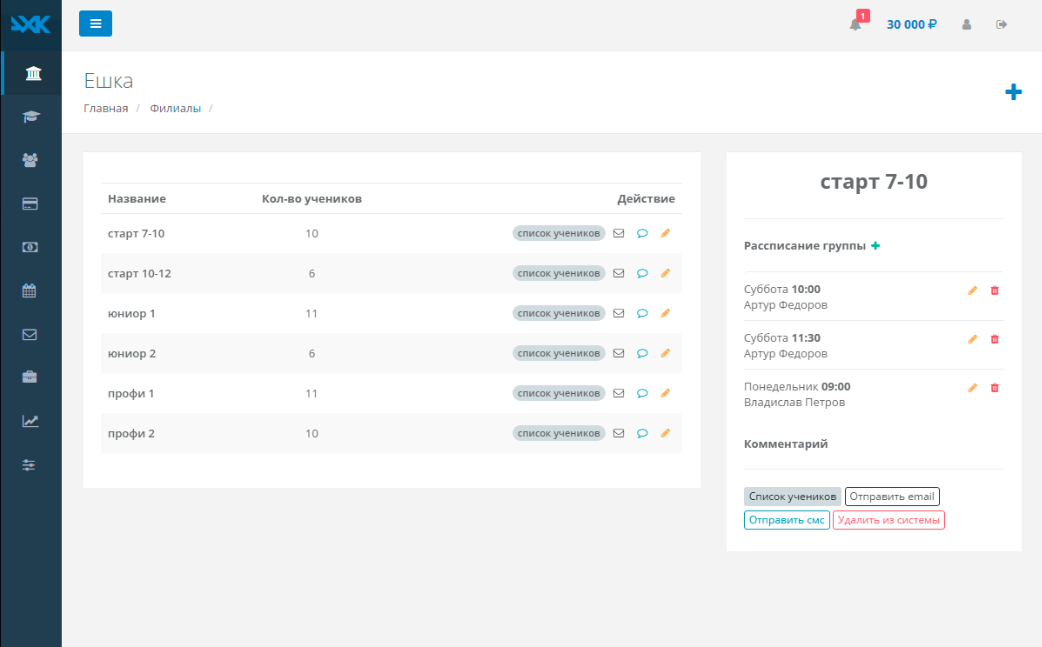
Основные функциональные возможности:

* автоматизация продаж;
* автоматическое составление отчетов;
* учебные бизнес-процессы;
* учет и анализ финансов;
* интеграция с различными системами.

Можно сделать вывод, что данная система преимущественно используется для ведения финансовых процессов в организации, а также для анализа различных процессов. В системе отсутствуют возможности учета посещаемости и успеваемости учеников, а также автоматического составления расписания.

**Другие функциональные возможности (личный кабинет, корзина, отзывы и др.).** Среди дополнительных функциональных возможностей следует выделить наличие личного кабинета для каждого пользователя, предоставляющего доступ к персональным настройкам и данным. Также приложения позволяет настроить систему под организацию.

«AXKED CRM» [4] – CRM-система для управления школами, кружками и студиями. Интерфейс данной системы изображен на рисунках 1.8-1.9.



**Рисунок 1.8 – Интерфейс учета групп «AXKED CRM» (1)**



**Рисунок 1.9 – Интерфейс аналитики «AXKED CRM» (2)**

Проанализируем интерфейс данной системы.

**Общая оценка интерфейса (первое впечатление).** Первое впечатление от интерфейса положительное. Он не перенасыщен информацией, что обеспечивает более легкое восприятие и способствует быстрой навигации по страницам.

**Цветовое решение интерфейса.** Преобладающей цветовой гаммой в пользовательском интерфейсе является гамма синего цвета. Синий цвет часто используется в веб-дизайне, поскольку ассоциируется со спокойствием, открытостью и ощущением безопасности. Он создает атмосферу комфорта, поддерживает восприятие системы с высоким уровнем доверия и ассоциируется с высоким профессионализмом. Учитывая ежедневное взаимодействие пользователей с системой, применение синего цвета способствует легкости восприятия интерфейса и снижает утомляемость глаз.

**Объём и структура представленной информации.** Информация в интерфейсе приложения хорошо структурирована и не создает впечатление избыточной нагрузки. Разнообразные данные представлены в виде таблиц, что способствует более легкому восприятию информации и обеспечивает ее удобное восприятие. Этот подход улучшает общую ясность и организацию данных, делая интерфейс приложения более привлекательным и удобным для пользователей.

**Наличие и структура меню.** В интерфейсе системы предусмотрено вертикальное меню, в котором разделы представлены в виде иконок. Однако это решение может вызвать определенные трудности для пользователей, так как они могут испытывать затруднения в идентификации значения каждой иконки. Этот подход может потребовать от пользователя дополнительных усилий, так как для перехода в нужный раздел ему придется сначала разгадать значение конкретной иконки, прежде чем принять решение о выборе.

**Удобство навигации.** Навигация по странице здесь также, как и в других системах реализуется с помощью бокового меню. В интерфейсе системы предусмотрено отображение заголовков разделов, что обеспечивает пользователю информацию о текущем местоположении. Такой подход способствует улучшению навигации, поскольку пользователи могут ясно определить, в каком разделе они находятся в текущий момент использования приложения.

**Наличие и информативность подсказок (помощь).** В интерфейсе приложения отсутствуют подсказки. Но есть раздел «Помощь», в который пользователь может перейти при возникновении трудностей.

**Удобство форм для ввода информации.** Формы для ввода данных в интерфейсе приложения достаточно удобные, но в полях ввода телефона и email-а отсутствуют маски, что может привезти к затруднениям.

**Возможность поиска информации (технология организации поиска, удобство, понятность, релевантность).** В системе отсутствует возможность поиска информации или сортировки данных, это может привезти к трудностям в использовании.

**Основные функциональные задачи (сформулировать с использованием функций управления) в соответствии с назначением системы анализируемого типа.**

Основные функциональные возможности:

* управление группами и расписанием;
* учет посещаемости;
* учет успеваемости;
* ведение финансов и счетов;
* добавление собственных уроков преподавателями;
* автоматическое составление отчетов;
* различные рассылки.

В системе реализовано много полезных функций, но также, как и в других, в ней отсутствует возможность составления автоматического расписания, а также ведение портфолио ученика.

**Другие функциональные возможности (личный кабинет, корзина, отзывы и др.).** Среди дополнительных функциональных возможностей следует выделить наличие личного кабинета для каждого пользователя, предоставляющего доступ к персональным настройкам и данным.

Подведем итоги анализа с помощью сравнительной таблицы. Результаты представлены в таблице 1.1.

Таблица 1.1 – Итоговое сравнение систем

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Критерий** | «Дневник.ру» | «Мой класс» | «ALFA CRM» | «AXKED CRM» |
| Общая оценка интерфейса | 3 | 2 | 5 | 5 |
| Цветовое решение интерфейса | 5 | 4 | 5 | 5 |
| Объём и структура представленной информации | 4 | 2 | 5 | 5 |
| Наличие и структура меню. | 5 | 5 | 5 | 5 |
| Удобство форм для ввода информации | 4 | 5 | 5 | 3 |
| Возможность поиска информации | 2 | 5 | 2 | 0 |
| Основные функциональные задачи в соответствии с назначением системы анализируемого типа | 5 | 5 | 4 | 5 |
| Другие функциональные возможности | 4 | 5 | 5 | 4 |
| **Итого** | 32 | 33 | 36 | 32 |

В итоге можно заключить, что существует потребность в создании унифицированной информационной системы, способной объединить следующие функциональности:

1. Для учителя:

* автоматическая генерация расписания;
* ведение учета посещаемости и успеваемости;
* удобный анализ данных о посещаемости и успеваемости.

1. Для директора школы:

* учет сотрудников;
* контроль оплаты обучения;
* аналитика процессов.

1. Для учеников:

* ведение портфолио с возможностью добавления заметок;
* онлайн-запись на интересующие курсы;
* оплата выбранных курсов.

Таким образом, создание информационной системы для художественной школы, объединяющей вышеописанную функциональность, представляется крайне актуальным и полезным шагом. Это обеспечит эффективное взаимодействие учителей, директора и учеников, оптимизируя управление и обеспечивая удобство в использовании для всех участников образовательного процесса.

# Процессы AS-IS и TO-BE

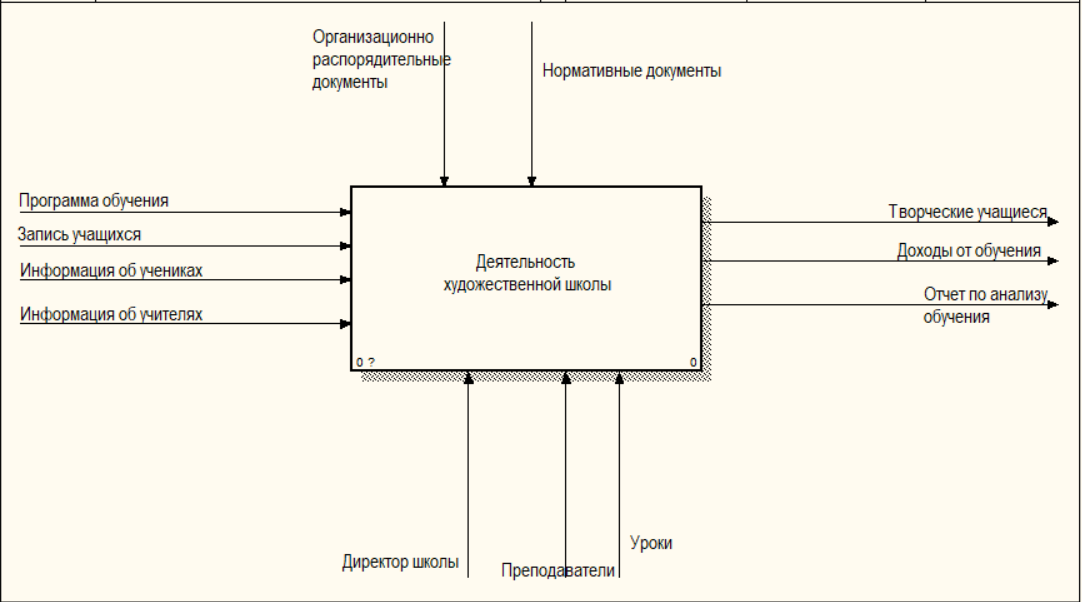
Для того, чтобы понять, как функционирует организация, выявить области для улучшения и оптимизации, необходимо описать основные бизнес-процессы школы.

В ходе анализа были выявлены следующие бизнес-процессы:

* прием документов учащегося;
* составление расписания;
* проведение занятий;
* учет успеваемости;
* учет посещаемости;
* анализ результатов обучения.

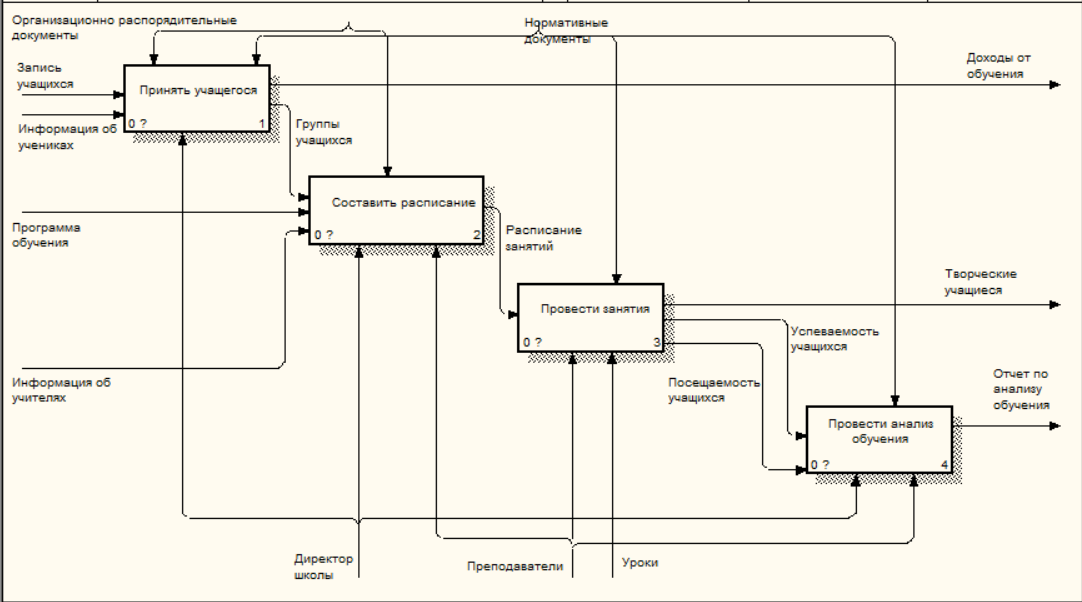
Для отображения структуры, последовательности и связи данных бизнес-процессов были разработаны диаграммы IDEF0. Они отражают реальное исполнение бизнес-процессов, которые можно представить в виде схематической модели AS IS.

Сначала построим контекстную диаграмму верхнего уровня, где рассмотрим объект «Деятельность художественной школы» (см. рисунок 1.10). Стрелки в этой диаграмме представляют полный комплект внешних интерфейсов объекта. Здесь определена область моделирования и ее границы.



**Рисунок 1.10 – Деятельность художественной школы (уровень A-0) (AS IS)**

Далее объект был декомпозирован на основные бизнес-процессы, осуществляемые в школе (см. рисунок 1.11). На диаграмме показано, как они связаны между собой.



**Рисунок 1.11 – Декомпозиция деятельности художественной школы (уровень A0) (AS IS)**

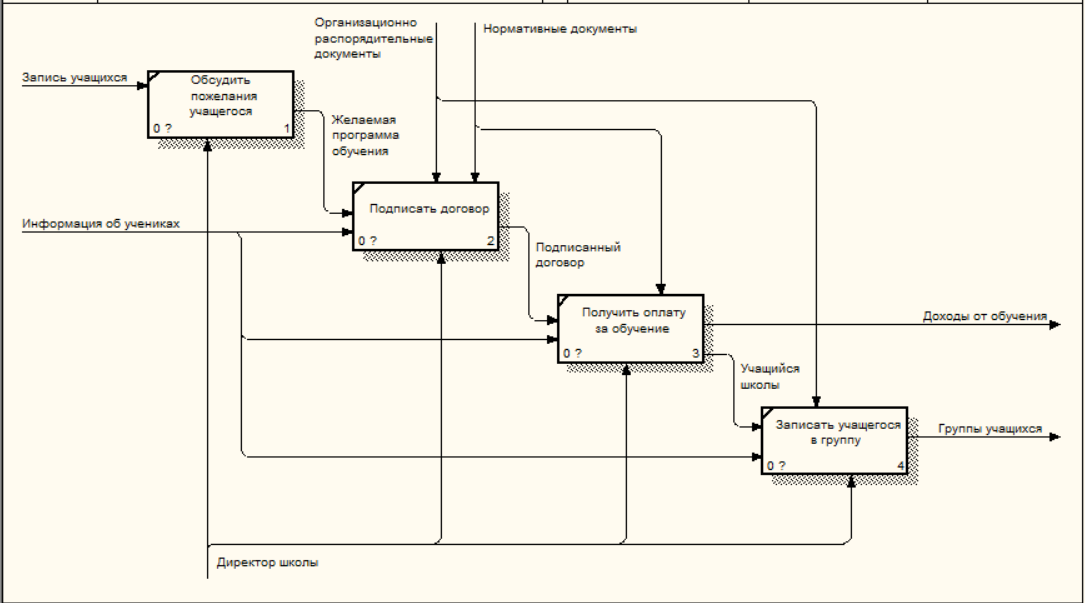
Теперь необходимо декомпозировать каждый из этих процессов, чтобы более точно определить, как они протекают в данной организации.

Декомпозиция процесса «Принять учащегося», как изображено на рисунке 1.12, демонстрирует, что данный процесс оказывается крайне трудоемким для каждой из сторон, участвующих в нем. Это обусловлено тем, что каждый желающий записаться в данное учебное заведение вынужден являться лично в организацию с целью подписания необходимых документов и совершения первоначального платежа за обучение. Со стороны директора, в свою очередь, требуется индивидуальное сопровождение каждого желающего, что значительно увеличивает временны́е затраты.

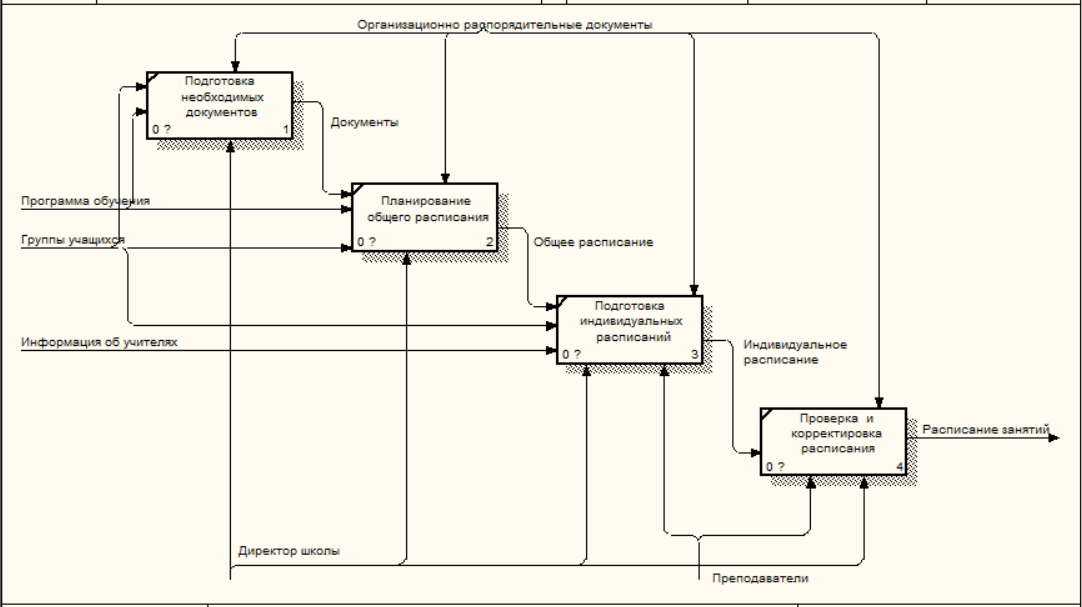
Таким образом, данный процесс характеризуется высокой степенью рутинности и потребностью во множестве личных визитов, что также приводит к замедлению его прохождения.

Следующим рассмотрим декомпозицию процесса «Составить расписание» (см. рисунок 1.13). Следует отметить, что в текущий момент расписание разрабатывается исключительно вручную. Это подразумевает значительные временны́е затраты, поскольку каждая корректировка требует пересмотра всего графика занятий. Также необходимо учитывать, что с ростом числа учащихся и сотрудников данная задача будет становиться все более сложной и объемной.

В связи с вышесказанным, возникает необходимость в автоматизации процесса составления расписания с целью повышения эффективности этого процесса и уменьшения вероятности ошибок, особенно в условиях динамичной и расширяющейся структуры.

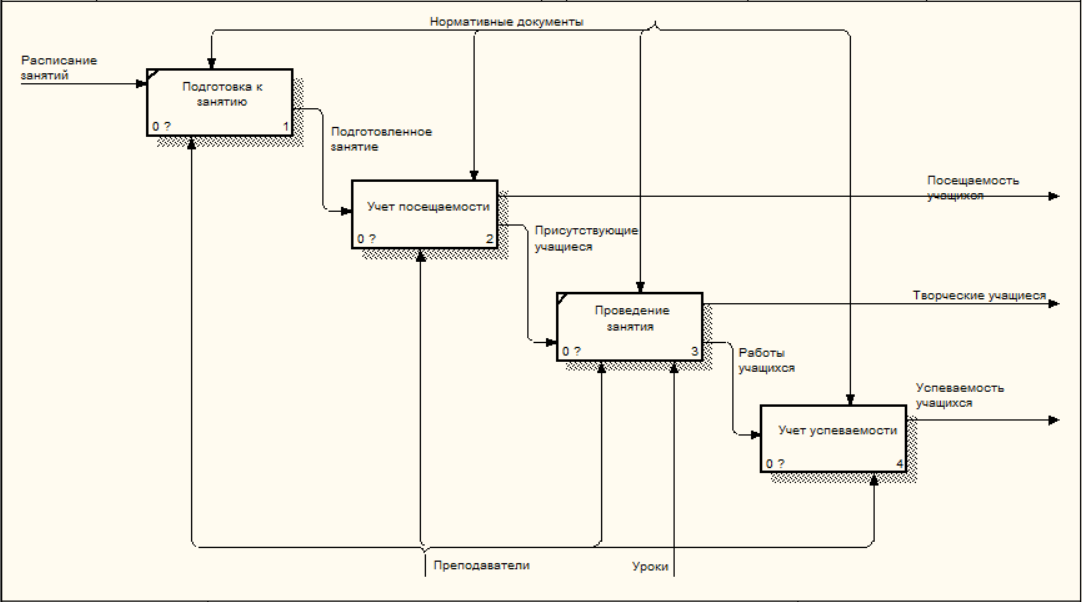


**Рисунок 1.12 – Декомпозиция процесса «Принять учащегося» (AS IS)**



**Рисунок 1.13 – Декомпозиция процесса «Составить расписание» (AS IS)**

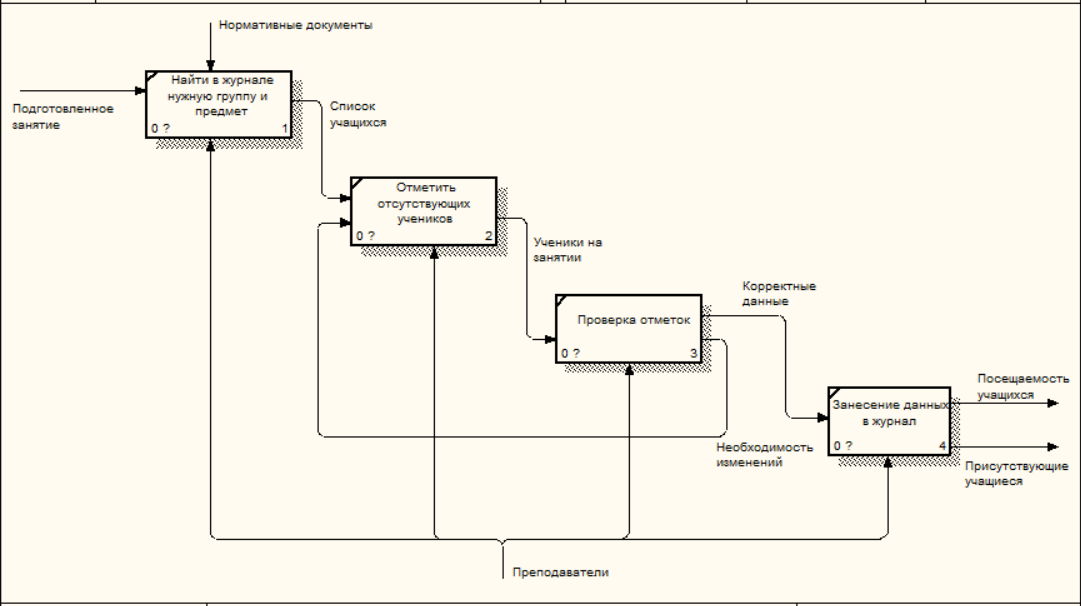
Следующим рассмотрим процесс «Провести занятия» (см. рисунок 1.14). Данный процесс является неотъемлемой и обыденной частью деятельности учебного учреждения и, в целом, не предполагает возможности значительной автоматизации. Однако следует обратить внимание на процессы учета посещаемости и учета успеваемости, которые являются важной частью процедур управления.



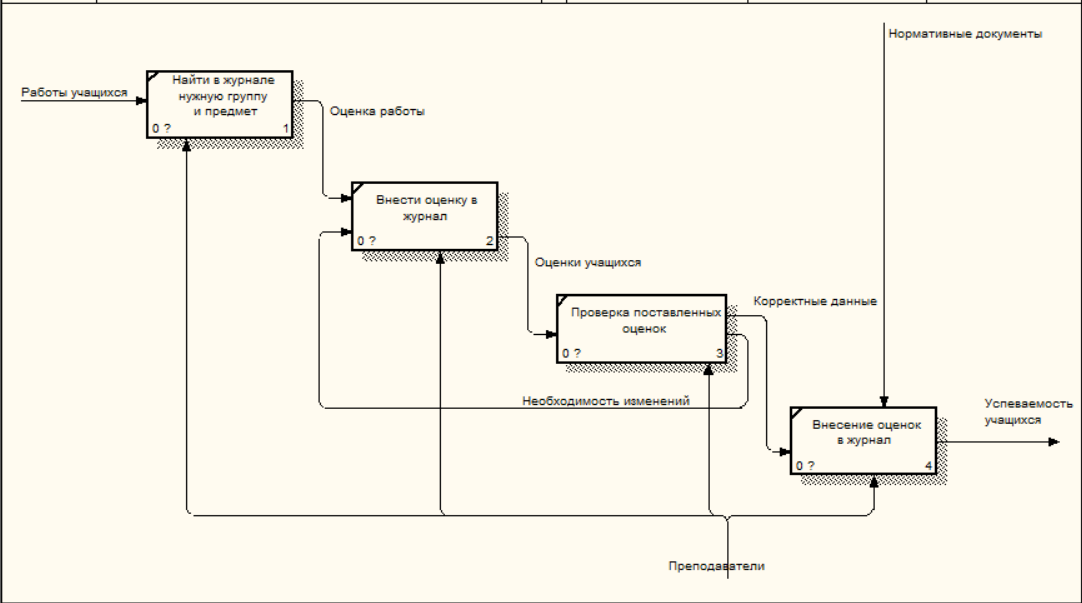
**Рисунок 1.14 – Декомпозиция процесса «Провести занятия» (AS IS)**

Рассмотрим декомпозицию двух важных процессов, а именно «Учет посещаемости» (см. рисунок 1.15) и «Учет успеваемости» (см. рисунок 1.16). В контексте учета посещаемости начальные записи фиксируются в соответствующих журналах преподавателей. Однако такая методика представляет собой неэффективный подход, поскольку бумажные журналы требуют постоянного ношения с собой, а также неудобны для последующего анализа и статистики. Дополнительно, возможны помарки в процессе учета, что может усложнить интерпретацию информации. Аналогичная проблема возникает и в случае учета успеваемости студентов.

С учетом перехода к цифровому формату учета посещаемости, преподаватели смогут вести записи более эффективно, используя приложение. Это обеспечит удобство внесения данных и создаст базу для дальнейшего удобного анализа. Также будет легко избежать потери данных.



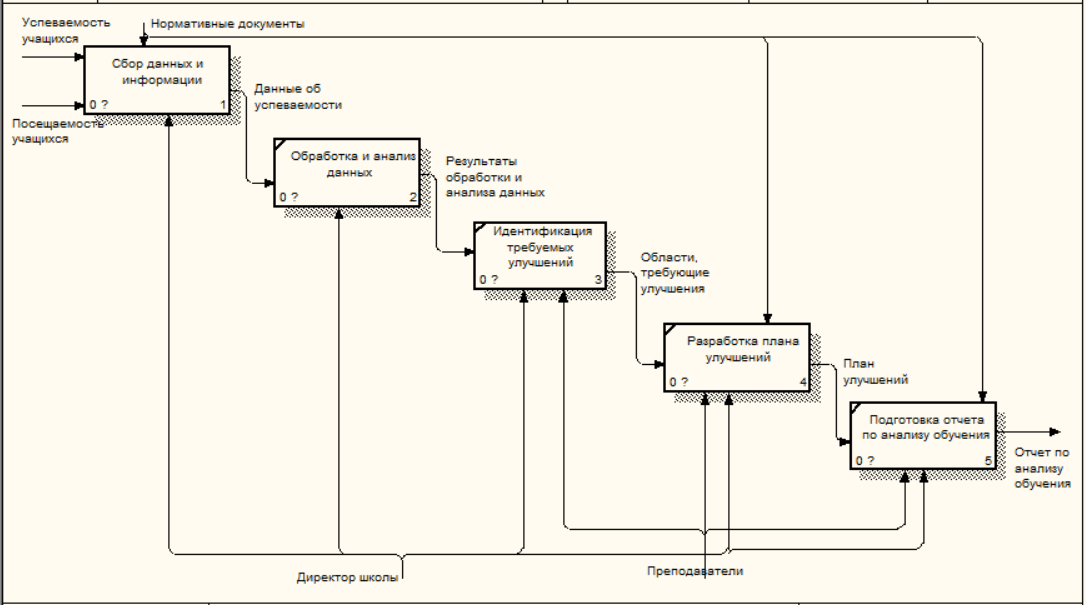
**Рисунок 1.15 – Декомпозиция процесса «Учет посещаемости» (AS IS)**



**Рисунок 1.16 – Декомпозиция процесса «Учет успеваемости» (AS IS)**

Декомпозиция процесса «Провести анализ обучения» (см. рисунок 1.17) выявляет ряд значительных трудностей. Данный процесс характеризуется необходимостью сбора и обработки данных вручную, что требует значительных временных и умственных ресурсов. В процессе сбора информации необходимо внимательно переносить данные из бумажных журналов в Excel, что может привести к возможным ошибкам и потере ценного времени.

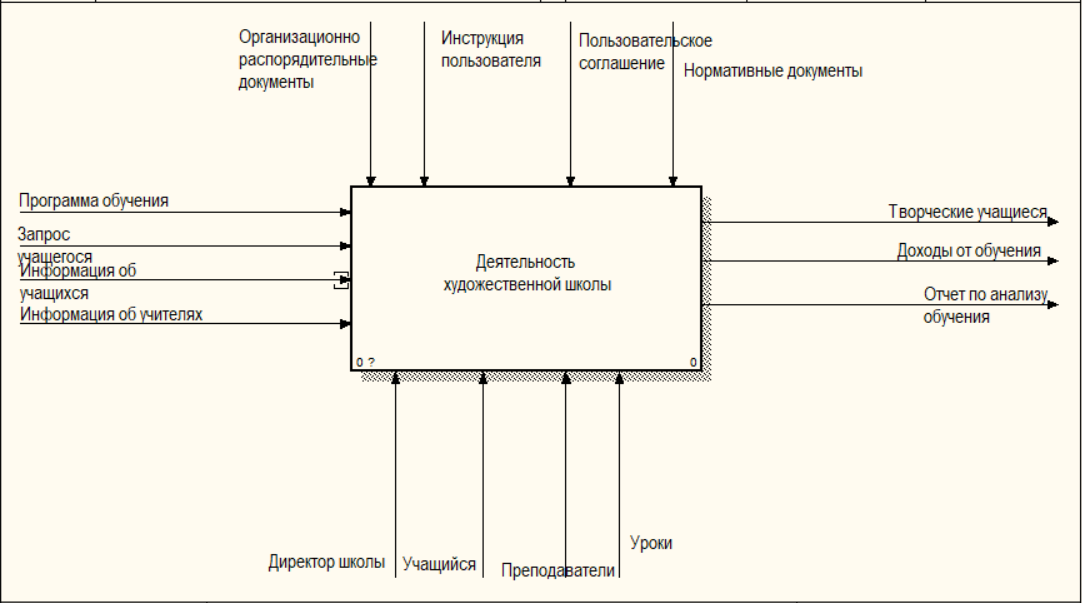
Анализ информации об обучении важен для оценки эффективности учебных программ и выявления областей, требующих улучшения. Однако сложность и рутинность процесса сбора и анализа данных вручную ограничивает возможности получения полного и актуального представления об образовательных процессах.



**Рисунок 1.17 – Декомпозиция процесса «Провести анализ обучения» (AS IS)**

Ранее мы рассмотрели текущий порядок осуществления основных бизнес-процессов в школе. Теперь необходимо представить процессы, какими они будут в автоматизированной информационной системе для школы. Для этого также составим IDEF0 диаграммы.

В контекстной диаграмме верхнего уровня (см. рисунок 1.18) появились новые элементы управления – это пользовательское соглашение и инструкция пользователя, а также в механизмах управления появились учащиеся школы.



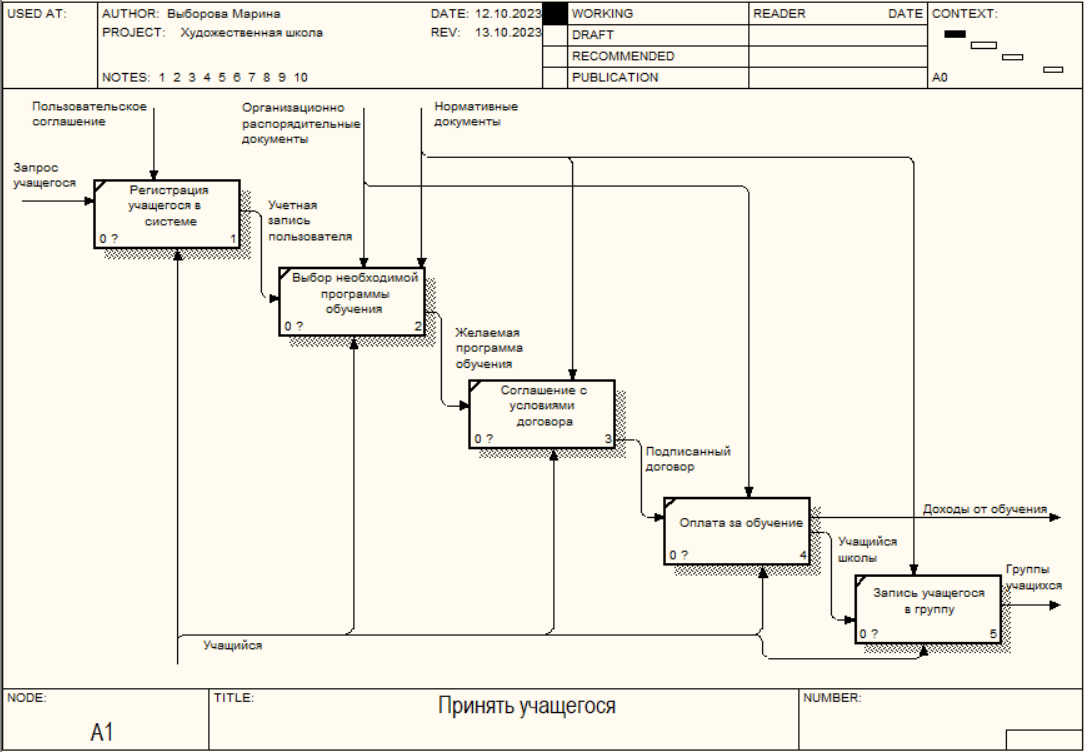
**Рисунок 1.18 – Деятельность художественной школы (уровень A-0) (TO BE)**

Основные процессы и взаимосвязи между ними остались неизменными, аналогично тому, как это было представлено в модели AS-IS (см. рисунок 1.19).



**Рисунок 1.19 – Основные процессы художественной школы (TO BE)**

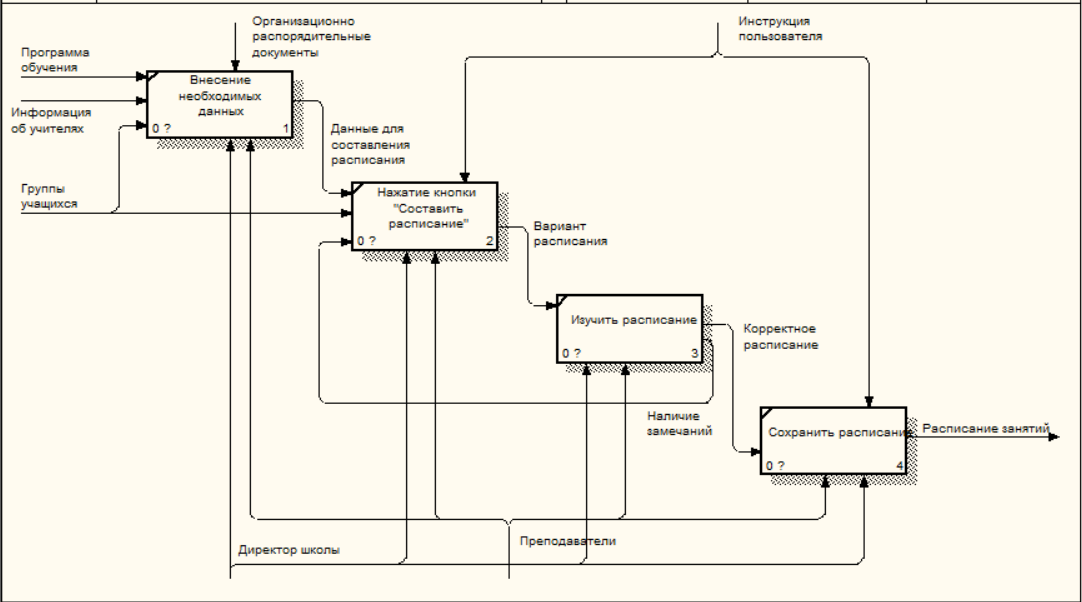
Рассмотрим декомпозицию процесса «Принять учащегося» (см. рисунок 1.20). Теперь потенциальным ученикам школы предоставляется удобная и автоматизированная процедура записи на обучение, что значительно упрощает взаимодействие с учебным учреждением. Претенденты на обучение могут осуществить регистрацию в специализированной системе онлайн, выбрать соответствующий курс обучения, и получить договорное соглашение в электронном формате. Данный процесс призван обеспечить более гибкий и удобный механизм для потенциальных учеников, позволяя им провести этап регистрации и оплаты в удобное для себя время и месте.



**Рисунок 1.20 – Декомпозиция процесса «Принять учащегося» (TO BE)**

Одновременно с этим внедрение автоматизации существенно сократило необходимость в личных встречах потенциальных учеников с директором для подписания договоров. Теперь директору не требуется тратить время на индивидуальные встречи с каждым кандидатом, что облегчает процесс приема учащихся и повышает его эффективность. Эти изменения содействуют оптимизации процесса приема учащихся и создают более удобные условия для будущих учеников.

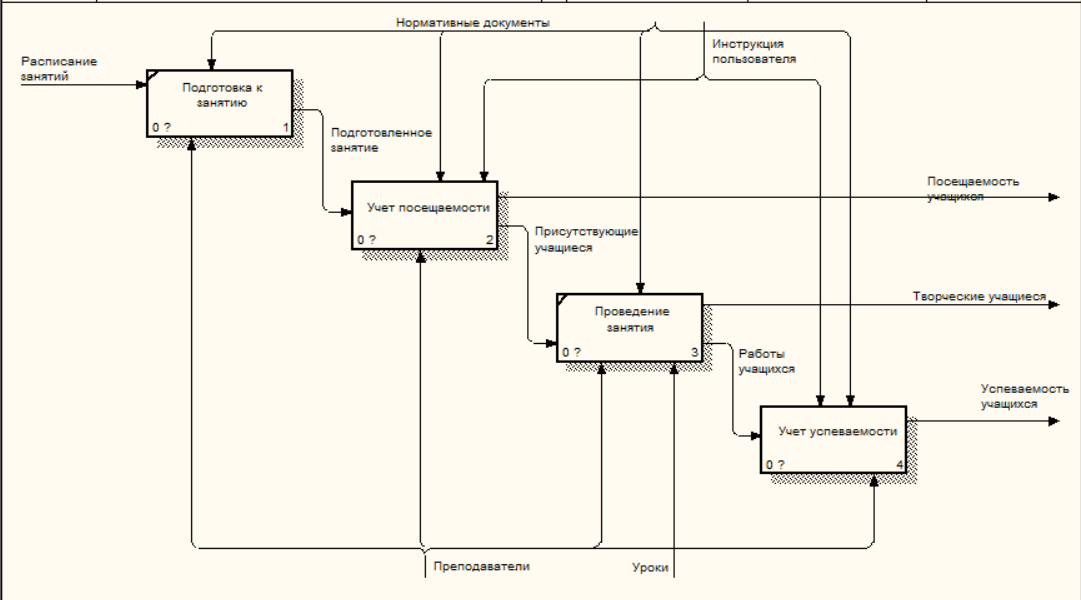
Далее рассмотрим декомпозицию процесса «Составить расписание» (см. рисунок 1.21). В рамках автоматизированной системы пользователю требуется всего лишь внести необходимые параметры, такие как нагрузка на преподавателей, количество учебных часов для каждого предмета, доступное время для проведения занятий и другие сведения. После внесения данных пользователь может нажать на соответствующую кнопку, и система автоматически создаст расписание, учитывая указанные параметры.



**Рисунок 1.21 – Декомпозиция процесса «Составить расписание» (TO BE)**

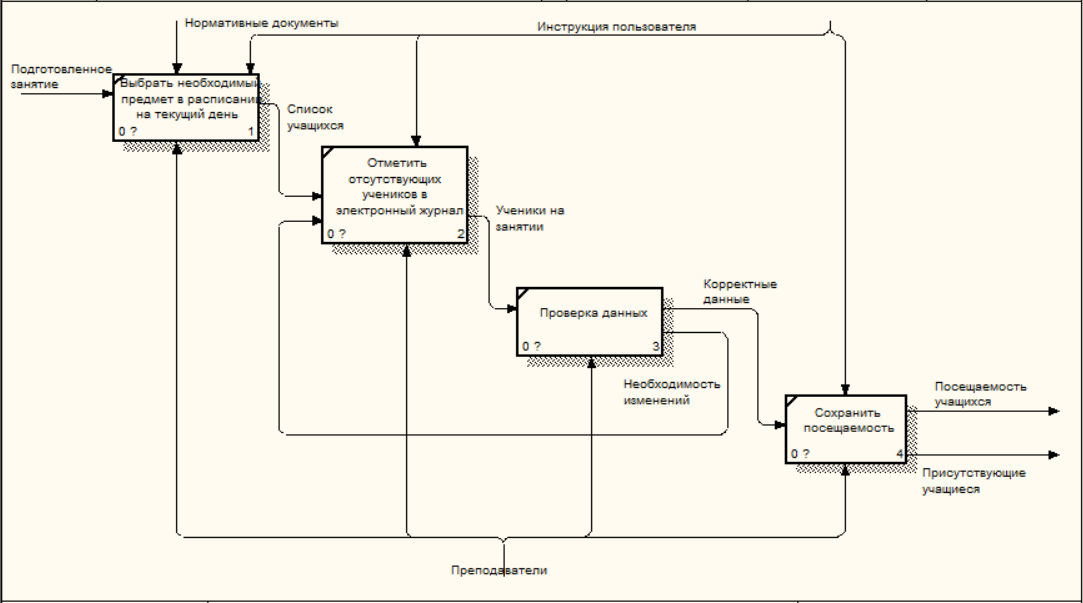
Также важно отметить, что в случае, если полученное расписание не удовлетворяет потребностям пользователя, он имеет возможность генерировать расписание повторно до получения желаемого результата. Этот подход значительно экономит время, которое ранее затрачивалось на ручное составление расписания.

Процесс «Провести занятия» остался неизменным, он представлен на рисунке 1.22.

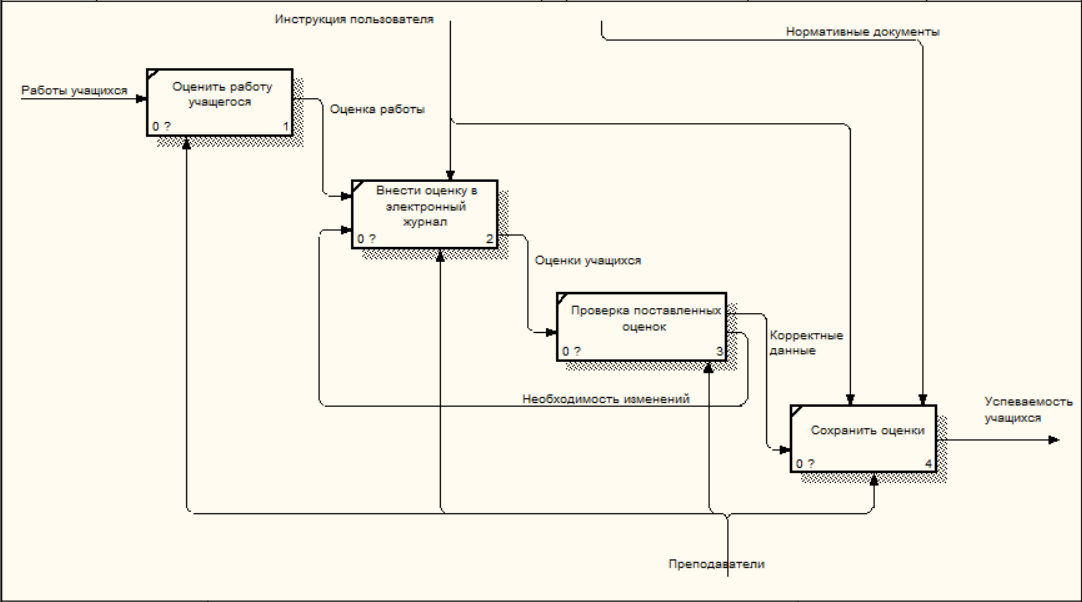


**Рисунок 1.22 – Декомпозиция процесса «Провести занятия» (TO BE)**

Процессы «Учет посещаемости» (см. рисунок 1.23) и «Учет успеваемости» (см. рисунок 1.24) будут осуществляться в электронном дневнике. Преподавателям предоставляется удобная возможность внесения изменений по необходимости. Эта система позволяет преподавателям производить корректировки в данных в любое удобное время, что уменьшает вероятность появления ошибок и обеспечивает надежность данных.



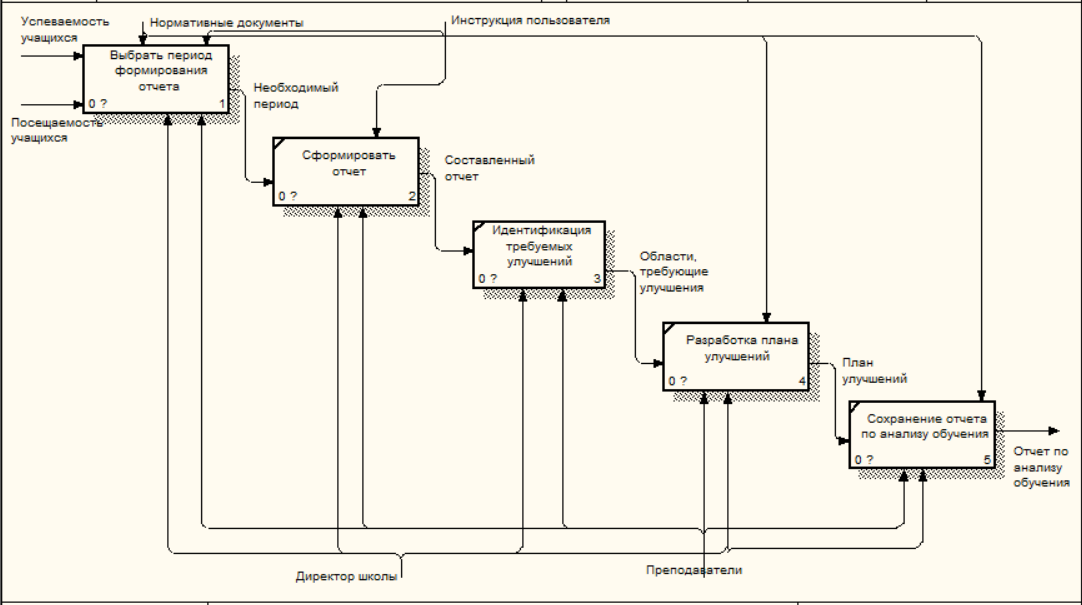
**Рисунок 1.23 – Декомпозиция процесса «Учет посещаемости» (TO BE)**



**Рисунок 1.24 – Декомпозиция процесса «Учет успеваемости» (TO BE)**

Внесенные изменения сразу же регистрируются в базе данных, что обеспечивает актуальность информации и готовность данных для их последующего использования при анализе учебного процесса. Это значительно повышает точность данных, сокращает временные затраты на внесение корректировок и делает весь процесс учета более прозрачным и управляемым.

Рассмотрим процесс «Провести анализ обучения» (см. рисунок 1.25). Теперь для создания отчетов не требуется ручного сбора и внесения данных в электронные таблицы. Данные о посещаемости и успеваемости автоматически регистрируются и хранятся в базе данных. На их основе генерируются отчеты в соответствии с выбранным пользователем периодом.



**Рисунок 1.25 – Декомпозиция процесса «Провести анализ обучения»**

**(TO BE)**

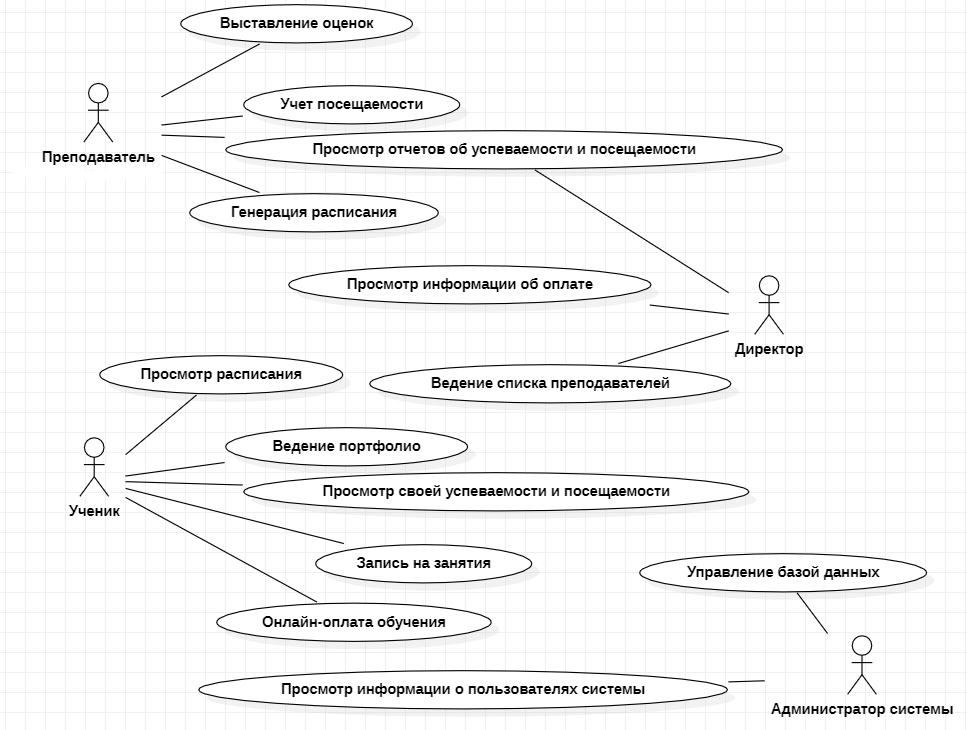
Для получения отчета пользователю достаточно указать необходимый временно́й интервал и активировать режим формирования отчета. В случае необходимости, отчет можно сохранить для последующего использования. Эта новая система значительно снижает трудоемкость процесса анализа обучения, делает его более быстрым и удобным, а также повышает точность и надежность данных, так как они автоматически регистрируются и обрабатываются.

# Описание вариантов использования

В разрабатываемой системе предусмотрены следующие роли:

* администратор системы – специалист, имеющий возможность корректировки информации в БД, проведения профилактических мероприятий для поддержания работы системы;
* преподаватель – имеют возможность внесения данных (оценки, посещаемость учеников), просмотр отчетов о посещаемости и успеваемости учеников, просмотр расписания занятий;
* ученик – имеют возможность просмотра своей успеваемости и посещаемости, добавления своих работ в портфолио, просмотра расписания занятий, возможность записи на занятие и онлайн-оплаты за обучение;
* директор – имеет возможность просмотра информации о посещаемости, успеваемости учеников, оплаты за обучение в системе, ведение списка преподавателей, а также составления различного рода отчетности.

На рисунке 1.26 представлена диаграмма использования для проектируемой системы.



**Рисунок 1.26 – Диаграмма использования**

Опишем более подробно данные варианты использования.

**Спецификация варианта использования «Выставление оценок»**

**Цель:** выставить оценки.

**Активные субъекты:** учитель.

**Краткое описание:** выставление оценок учащимся путем выбора необходимого значения.

**Основной поток событий:**

1. Активный субъект инициирует процесс выставления оценок.
2. Система выдает необходимую страницу. На экран выводится 2 формы ввода даты и таблица с присутствующими учениками, полученными из базы данных.
3. Пользователь выбирает необходимую ячейку в таблице для выставления оценки.
4. Система показывает список оценок, которые возможно поставить, из базы данных.
5. Пользователь выбирает необходимую оценку.
6. Система добавляет эту оценку в ячейку.
7. После выставления всех оценок пользователь инициирует процесс сохранения этих значений.
8. Система заносит полученные оценки в базу данных.

**Специальные требования:** нет.

**Предусловия:** перед активизацией варианта использования должен быть выполнен процесс учета учеников на занятии.

**Спецификация варианта использования «Учет учеников на занятии»**

**Цель:** учесть учеников на занятии.

**Активные субъекты:** учитель.

**Краткое описание:** произвести учет посещаемости учащихся путем выбора необходимого значения.

**Основной поток событий:**

1. Активный субъект инициирует процесс учета учеников на занятии.
2. Система выдает необходимую страницу. На экран выводится 2 формы ввода даты и таблица с учениками, полученными из базы данных.
3. Пользователь выбирает необходимую ячейку в таблице для выставления посещаемости.
4. Система показывает список вариантов посещаемости («был», «не был», «болеет», «другое») из базы данных.
5. Пользователь выбирает необходимый вариант.
6. Система добавляет этот вариант в ячейку.
7. После выставления всей посещаемости пользователь инициирует процесс сохранения этих значений.
8. Система заносит полученную посещаемость в базу данных.

**Специальные требования:** нет.

**Предусловия:** нет.

**Спецификация варианта использования «Ведение списка преподавателей»**

**Цель:** вести список преподавателей.

**Активные субъекты:** директор.

**Начальное состояние:** ведение списка преподавателей путем внесения необходимых данных в таблицу «Преподаватели»

**Основной поток событий:**

1. Активный субъект инициирует процесс ведения списка преподавателей.
2. Система выдает необходимую страницу. На экран выводятся необходимые поля для внесения преподавателя в таблицу и таблица с действующими преподавателями в школе.
3. Пользователь заполняет поля данными о преподавателе и инициирует процесс внесения данных
4. Система вносит информацию в базу данных и отображает новую запись в таблице.

**Специальные требования:** нет.

**Предусловия:** нет.

**Спецификация варианта использования «Ведение портфолио»**

**Цель:** вести портфолио.

**Активные субъекты:** ученик.

**Начальное состояние:** ведение портфолио путем добавления фотографий работ в галерею.

**Основной поток событий:**

1. Активный субъект инициирует процесс ведения портфолио.
2. Система выдает необходимую страницу. На экран выводится галерея с работами пользователя.
3. Пользователь инициирует процесс добавления фотографии.
4. Система добавляет фотографию в галерею пользователя и вносит информацию о фото в базу данных.

**Специальные требования:** нет.

**Предусловия:** перед активизацией варианта использования должен быть выполнен вариант использования «Открыть портфолио».

**Спецификация варианта использования «Запись на занятие»**

**Цель:** записаться на занятие.

**Активные субъекты:** ученик.

**Начальное состояние:** запись на занятие путем выбора и оплаты понравившегося курса.

**Основной поток событий:**

1. Активный субъект инициирует процесс записи на занятие.
2. Система выдает необходимую страницу. На экран выводится список доступных курсов.
3. Пользователь выбирает понравившееся занятие.
4. Система выдает электронный договор, необходимый для записи на обучение.
5. Пользователь соглашается с условием договора.
6. Система выдает форму для оплаты обучения.
7. Пользователь вводит необходимые данные для оплаты и инициирует процесс онлайн-оплаты.
8. Система вносит пользователя в группу желаемого курса и добавляет запись в базу данных

**Специальные требования:** нет.

**Предусловия:** нет.

# Выработка требований

Информационная система для художественной школы должна соответствовать следующим основным требованиям:

* возможность вести учет посещаемости и успеваемости учащихся;
* возможность автоматической генерации расписания и его просмотр;
* просмотр и составление отчетов об успеваемости и посещаемости;
* просмотр информации об оплате для директора школы;
* ведение списка преподавателей для директора;
* ведение портфолио для учащихся;
* просмотр успеваемости и посещаемости для учащегося;
* возможность онлайн-записи на занятие;
* онлайн-оплата за обучение.

# Проектирование

# Выбор и обоснование средств проектирования и реализации

# Средства проектирования

Для проектирования системы использовались следующие инструменты:

* **DBDesigner** выбран как инструмент для проектирования баз данных благодаря своему интуитивному интерфейсу, широкому набору функций, включая генерацию скриптов для различных СУБД, и статусу свободного программного обеспечения с открытым исходным кодом, что обеспечивает гибкость и доступность для разработчиков.
* **Figma** был выбран из-за его высокой гибкости и универсальности. Это интуитивно понятное средство, которое обеспечивает не только дизайн интерфейсов, но и создание детальных прототипов. А также возможность работать в браузере без необходимости установки дополнительного программного обеспечения повлияла на выбор в пользу этого инструмента.
* **StarUML** был выбран в качестве средства проектирования из-за его многофункциональности и удобства использования. Этот инструмент предоставляет широкий спектр возможностей для моделирования, включая диаграммы классов, вариантов использования и многие другие. Интерфейс StarUML интуитивно понятен, что упрощает процесс создания и редактирования диаграмм.

# Средства реализации

Выбор использования технологий React, Node.js, Express.js и PostgreSQL для реализации проекта был обоснован рядом причин.

Во-первых, React обеспечивает создание приложения с использованием компонентов, которые могут быть многократно использованы. В случае автоматизированной системы для художественной школы, это может включать компоненты для управления расписанием занятий, регистрацией студентов, отслеживания прогресса и других функций. Компоненты облегчают поддержку и модификацию системы в будущем.

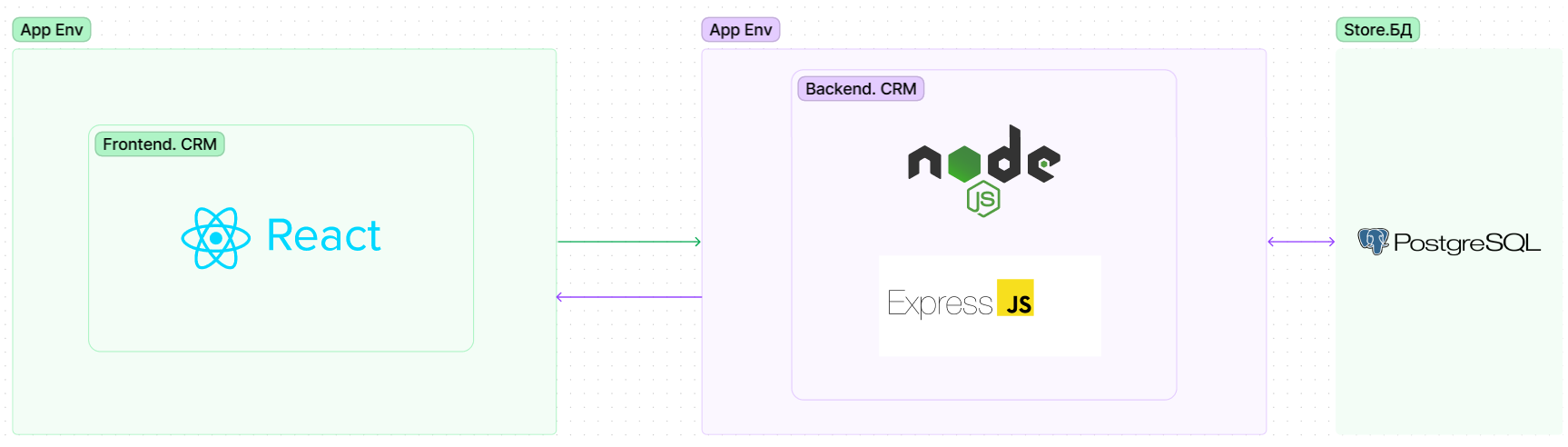
Во-вторых, выбор Node.js и Express.js для серверной части приложения обусловлен их преимуществами в области разработки веб-приложений. Node.js является средой выполнения JavaScript на стороне сервера, что позволяет использовать единый язык программирования как на клиентской, так и на серверной стороне. Express.js, в свою очередь, является легковесным и гибким фреймворком для создания веб-приложений на основе Node.js. Он предоставляет удобные инструменты для маршрутизации, обработки запросов и других задач, необходимых для разработки серверной части приложения.

PostgreSQL была выбрана в качестве базы данных для хранения информации и выполнения SQL-запросов. PostgreSQL является одной из самых популярных и широко используемых реляционных баз данных. Она обладает простым в использовании интерфейсом, хорошей производительностью и поддерживает полный набор функций SQL. В контексте проекта позволяет эффективно хранить и организовывать данные.

Таким образом, выбор технологий React, Node.js, Express.js и PostgreSQL для реализации данного проекта обусловлен их преимуществами и соответствием требованиям проекта, включая гибкость, эффективность и удобство использования в контексте создания визуальной новеллы для обучения SQL-запросам.

# Проектирование архитектуры приложения

Для приложения была выбрана клиент-серверная архитектура, которая представлена на рисунке 2.1.



**Рисунок 2.10 – Архитектура приложения**

Клиент-серверная архитектура представляет собой структурный подход к организации и взаимодействию программных систем. В этой архитектуре функции приложения распределены между клиентской (пользовательской) и серверной (центральной) частями. На клиентской стороне располагается интерфейс пользователя, обеспечивающий визуальное взаимодействие. Кроме того, часть логики приложения может быть реализована на клиенте для повышения отзывчивости.

Серверная часть, в свою очередь, содержит основную бизнес-логику, обработку данных и принятие решений. Здесь также осуществляется управление данными, и их хранение может быть реализовано через работу с базой данных или другими хранилищами данных. Сервер принимает запросы от клиента, обрабатывает их и возвращает результаты выполнения запросов.

Взаимодействие между клиентом и сервером осуществляется через клиент-серверное взаимодействие по сети. Это может быть реализовано с использованием различных сетевых технологий, таких как HTTP, WebSocket и другие. Ключевое в этой архитектуре — асинхронность взаимодействия, что способствует эффективности приложения.

Данная архитектура была выбрана, потому что она обеспечивает модульность приложения, позволяя разделять бизнес-логику и пользовательский интерфейс. Это содействует масштабируемости и облегчает управление доступом к данным.

# Проектирование хранилища данных

# Основные сущности

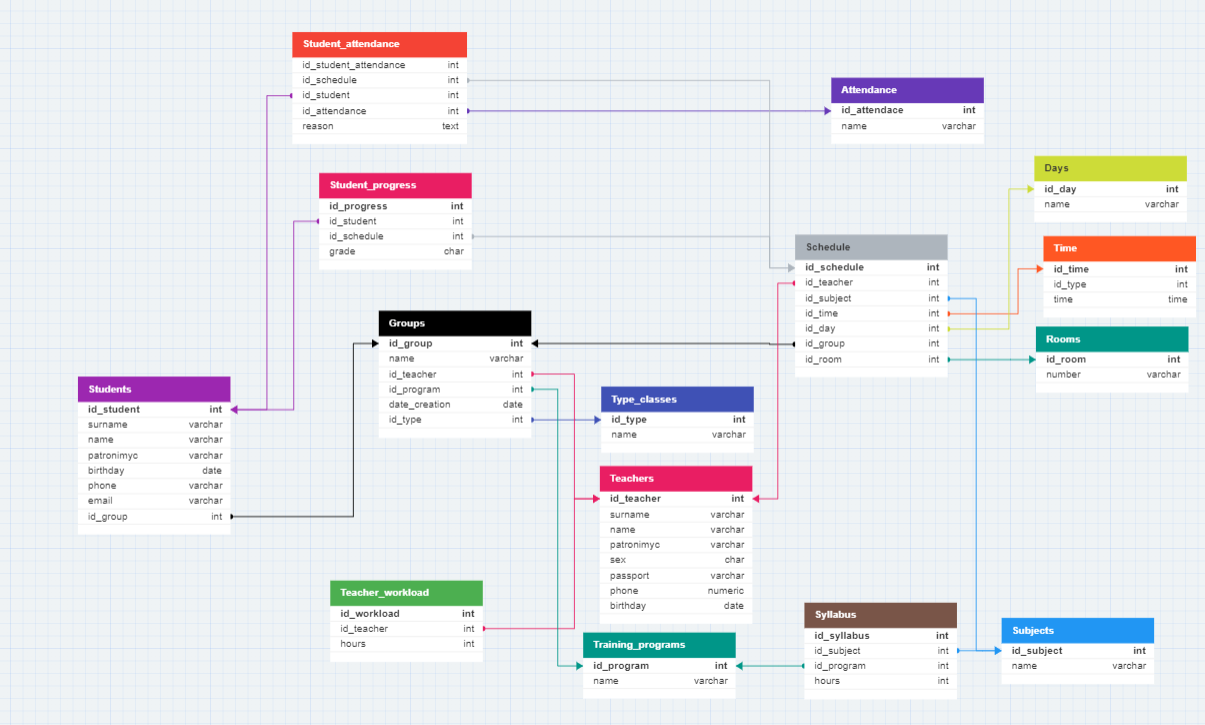
Система имеет следующие основные сущности, они представлены в таблице 2.1.

Таблица 2.1 – Описание сущностей

|  |  |
| --- | --- |
| **Сущность** | **Атрибуты** |
| Преподаватель | ФИО преподавателя, пол, паспорт, телефон, дата рождения |
| Группа | Название группы, ID преподавателя, ID учебной программы |
| Ученик | ФИО ученика, дата рождения, телефон, почта, ID группы |
| Расписание | ID преподавателя, ID предмета, ID времени, ID дня, ID группы, ID кабинета |
| Посещаемость | ID расписания, ID ученика, присутствие, причина |
| Успеваемость | ID расписания, ID ученика, оценка |

# Логическая модель данных

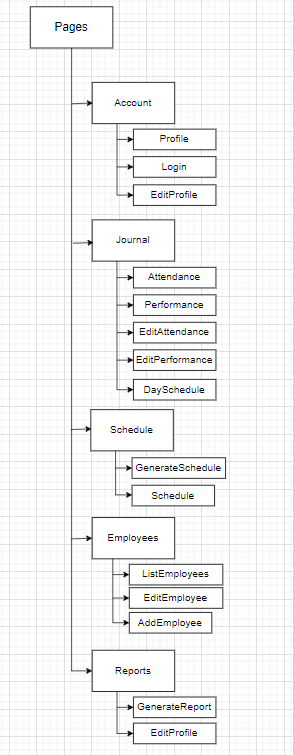
На рисунке 2.2 представлена логическая модель данных.



**Рисунок 2.11 – Логическая модель данных**

# Проектирование пользовательского интерфейса

На рисунке 2.3 представлена иерархия экранов.



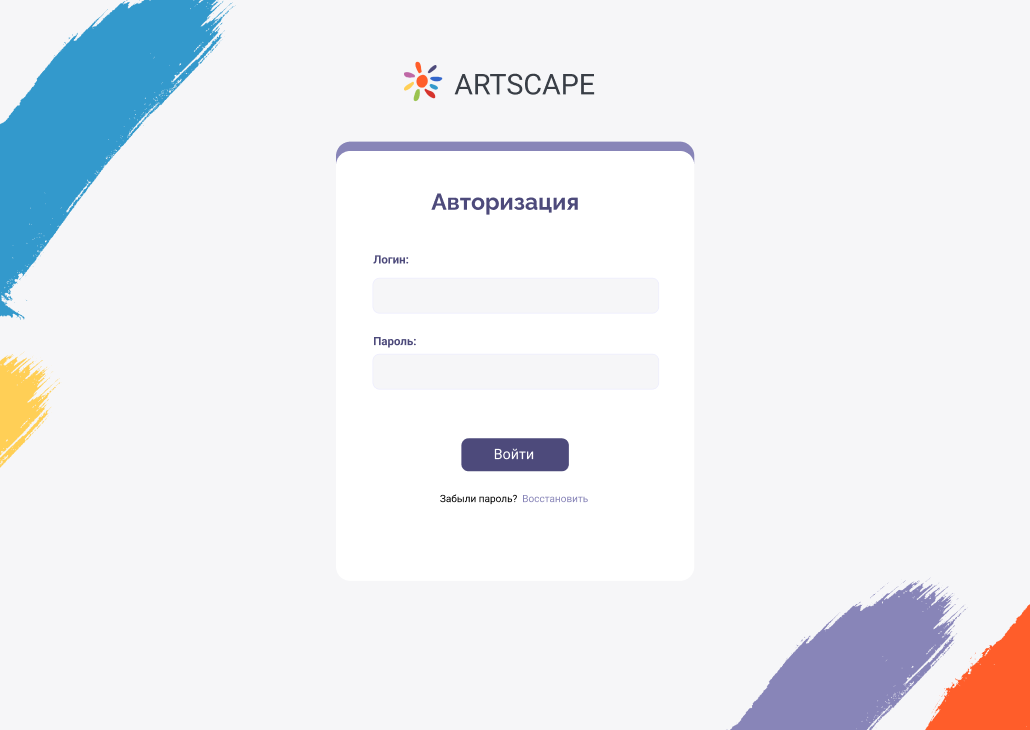
**Рисунок 2.3 – Иерархия экранов**

В таблице 2.2 приведено описание экранов системы.

Таблица 2.2 – Описание экранов системы

|  |  |  |
| --- | --- | --- |
| **Номер экрана** | **Краткое название** | **Описание экрана и его поведение** |
| **1** | **Account** |  |
| 1.1 | Profile | Личный кабинет пользователя, где представлены его персональные данные и фотография |
| 1.2 | Login | Форма авторизации, в которой необходимо ввести логин и пароль. При вводе некорректных данных пользователю выдается соответствующее сообщение. |
| 1.3 | EditProfile | Страница, на которой представлена возможность редактирования личной информации. Она содержит поля, в которых введены текущие данные. При вводе некорректных значений, например, вводе цифры в поле ввода ФИО пользователю выдается сообщение. Также, если пользователь пытается сохранить данные с пустыми полями, которые обязательны для заполнения, то система тоже выдают соответствующее сообщение |
| **2** | **Journal** |  |
| 2.1 | Attendance | На данной странице представлена таблица с учащимися соответствующей группы. В полях таблицы содержится выпадающий список со следующими значениями: «Был», «Не был», «Болеет», «Другое». Справа в нижнем углу представлена кнопка «Сохранить», при нажатии на которую происходит запись данных в БД |
| 2.2 | Performance | На данной странице представлена таблица с учащимися соответствующей группы. В полях таблицы содержится выпадающий список со следующими значениями: «5», «4», «3», «2». Также в ячейке таблицы есть иконка изображения, при нажатии на которую будет выведено модальное окно с оцениваемым рисунком. Справа в нижнем углу представлена кнопка «Сохранить», при нажатии на которую происходит запись данных в БД |
| 2.3 | DaySchedule | Страница, которая содержит расписание преподавателя на текущий день. После нажатия на определенное занятие будет совершен переход на страницу «Attendance» |
| **3** | **Schedule** |  |
| 3.1 | Schedule | Страница с расписанием на неделю. Показывает такие данные, как название предмета, аудитория, время и имя преподавателя |
| **4** | **Reports** |  |
| 4.1 | GenerateReport | Страница, содержащая поля ввода периода, на основе которого необходимо сформировать отчет. Также справа от них расположена кнопка «Показать», при нажатии на которую будет сформирован отчет в виде диаграмм и графиков |

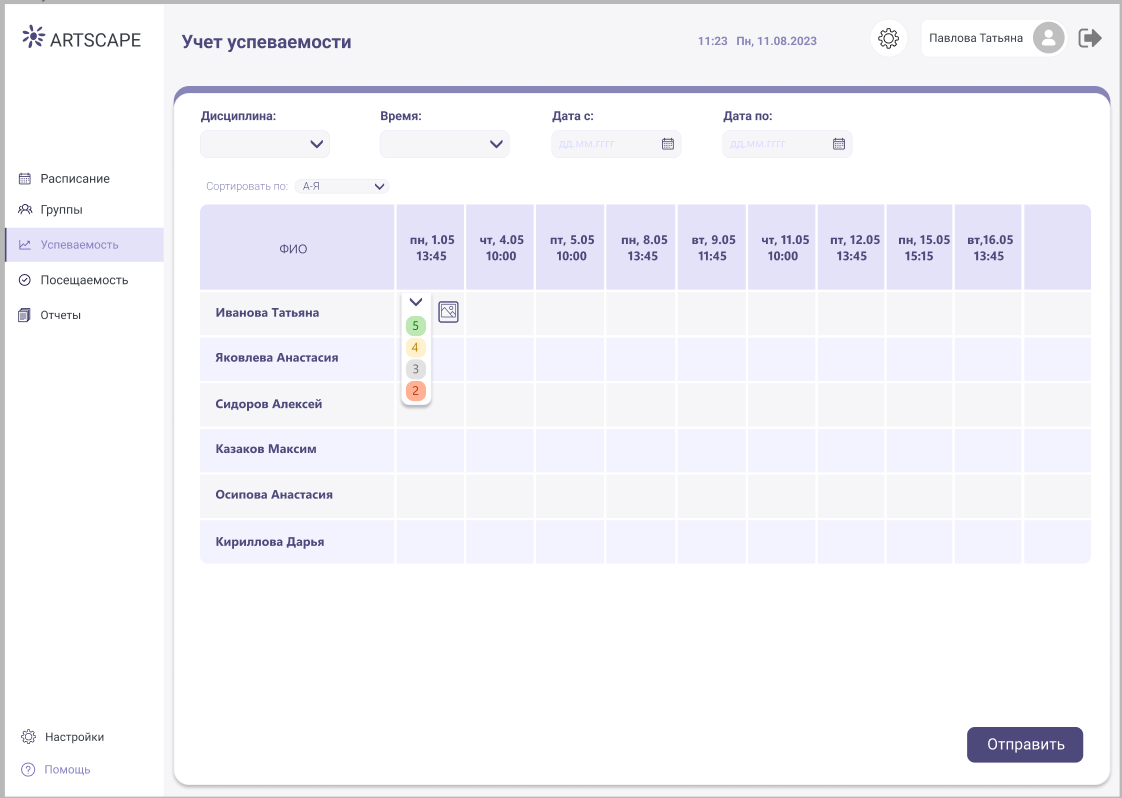
На рисунке 2.4 представлен макет страницы авторизации в системе.



**Рисунок 2.4 – Макет страницы авторизации**

Представленный на рисунке 2.5 макет экрана предназначен для заполнения документа в рамках задачи учета успеваемости с использованием учетной записи преподавателя. Навигационное меню находится слева для обеспечения удобства работы с системой. Для упрощения процесса заполнения документа в правой верхней части экрана отображается текущий день недели и время. Основная функциональность задачи «Учет успеваемости» расположен в центральной части экрана.

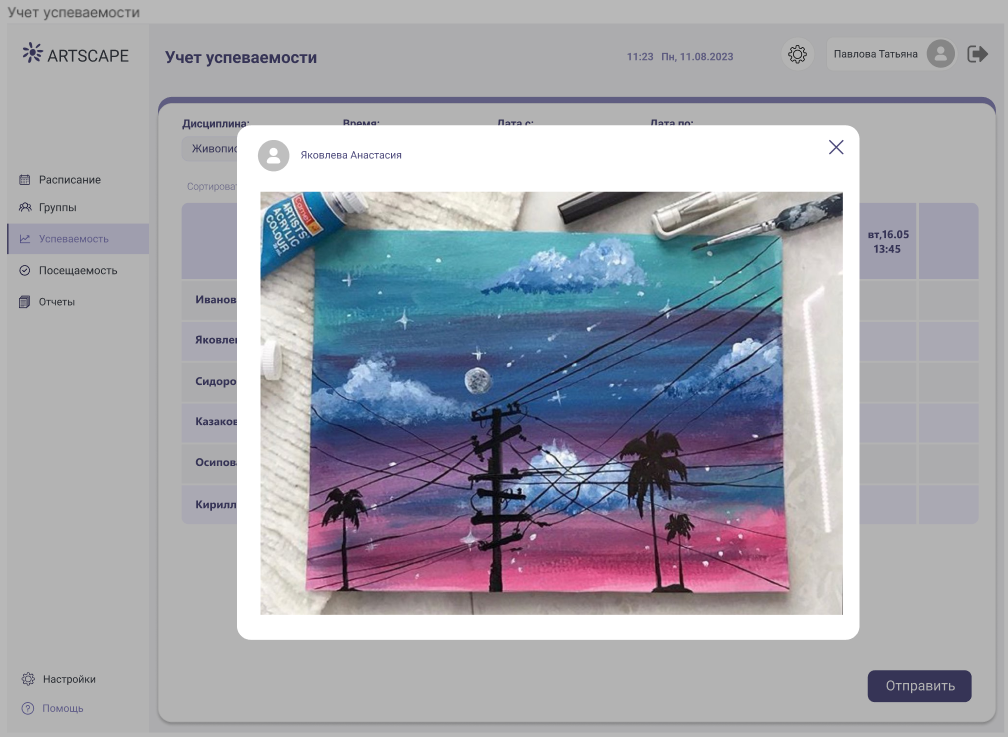
Для упрощения работы с документом в правой верхней части экрана выводится системная информация: текущий день недели и время, а также используются эргономические особенности работы с документом в цветовом решении, такие как выделение информационной части документа. Возможные оценки учеников отображаются в документе при помощи цветовой индикации в зависимости от высоты оценки. Таким образом, оценка «5» будет выделяться зеленым цветом, «4» – желтым, «3» – серым и «2» – красным, как самая низкая оценка.



**Рисунок 2.5 – Макет экрана «Учет успеваемости»**

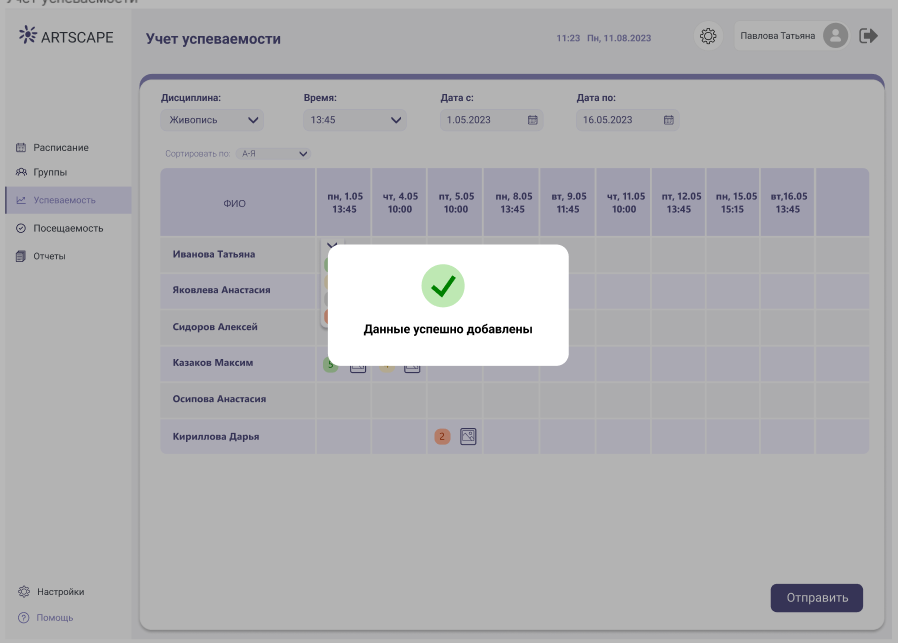
Для того чтобы предотвратить возможные ошибки пользователя при выборе дат, необходимо ввести ограничения при использовании календаря. Конкретно, при выборе даты начала периода «Дата с», необходимо заблокировать доступ к датам, которые находятся в будущем, т.е. даты, которые больше текущей даты. При выборе даты окончания периода «Дата по», необходимо заблокировать доступ к датам, которые меньше даты начала периода «Дата с» и датам, которые больше текущей даты. Таким образом, ограничения позволят пользователю выбрать только допустимые даты, что снизит вероятность возникновения ошибок при заполнении документа.

После нажатия на кнопку изображения будет выведено модальное окно с именем ученика и фотографией его работы (см. рисунок 2.6). Преподаватель может его просмотреть и проставить оценку.



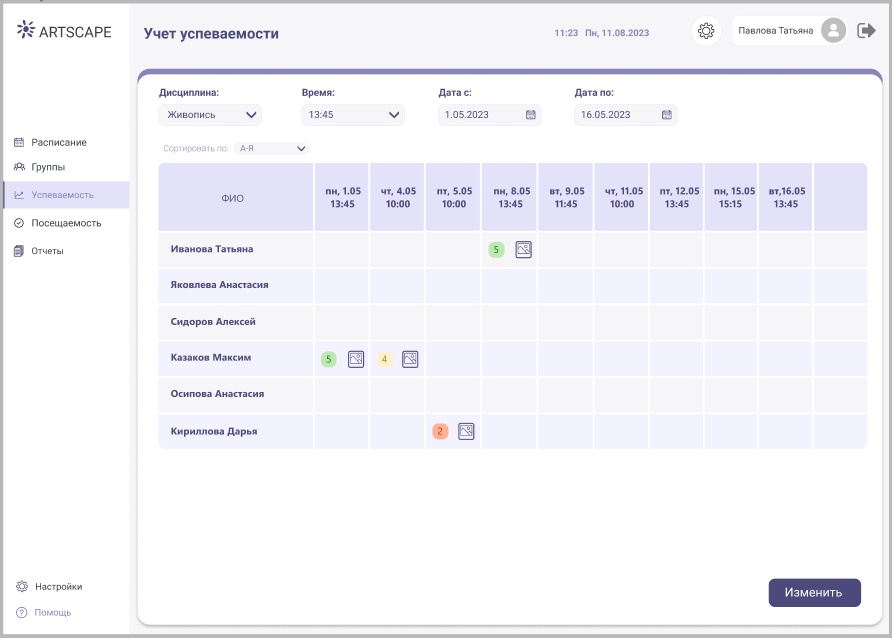
**Рисунок 2.6 – Окно с рисунком ученика**

После нажатия на кнопку «Отправить» данные будут сохранены в базе данных. Если запрос успешно выполнен, то пользователю будет выведено уведомление об успешной операции (см. рисунок 2.7) в соответствующем окне.



**Рисунок 2.7 – Успешная запись успеваемости в БД**

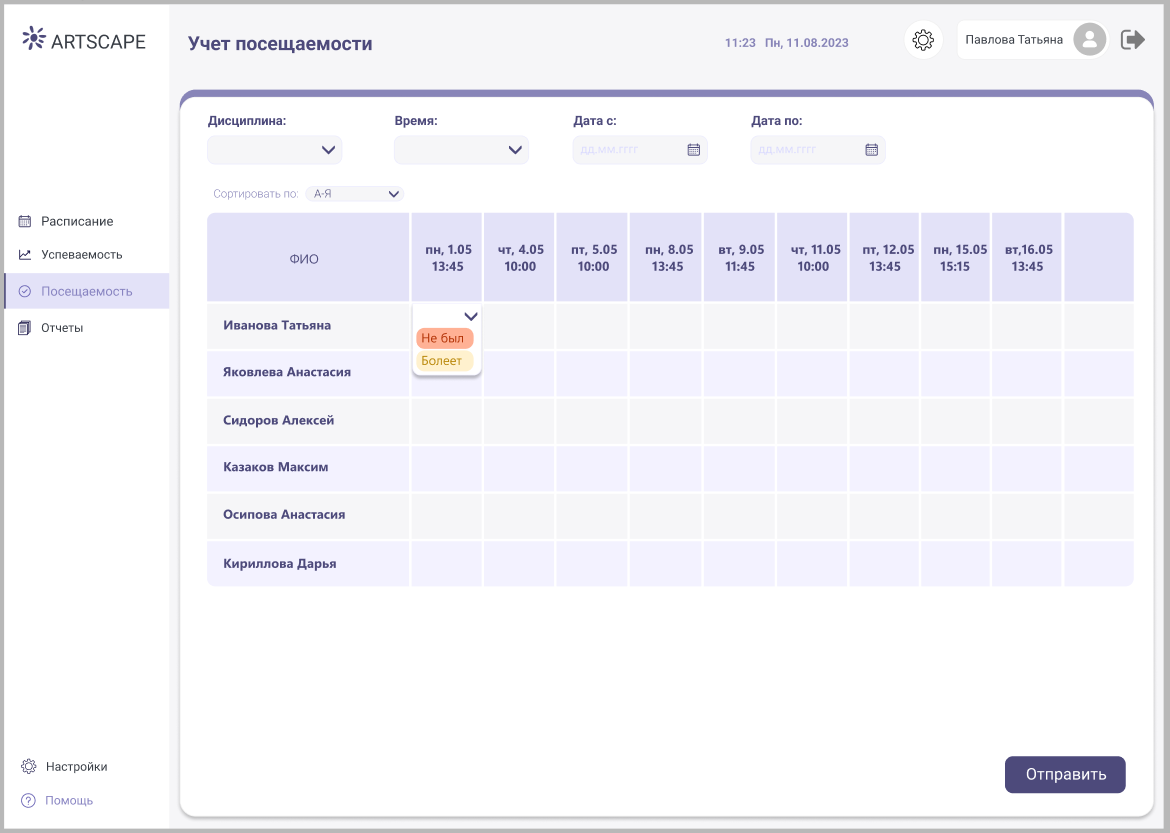
После завершения процесса добавления записи, кнопка «Добавить» преобразуется в кнопку «Изменить» (см. рисунок 2.8), и пользователю становится недоступно внесение изменений в текущую таблицу. Доступ к функции внесения данных будет восстановлен только после того, как пользователь нажмет на кнопку «Изменить», чтобы исправить ошибку, если таковая обнаружится после отправки успеваемости.



**Рисунок 2.8 – Состояние системы после закрытия окна при успешном сохранении данных об успеваемости**

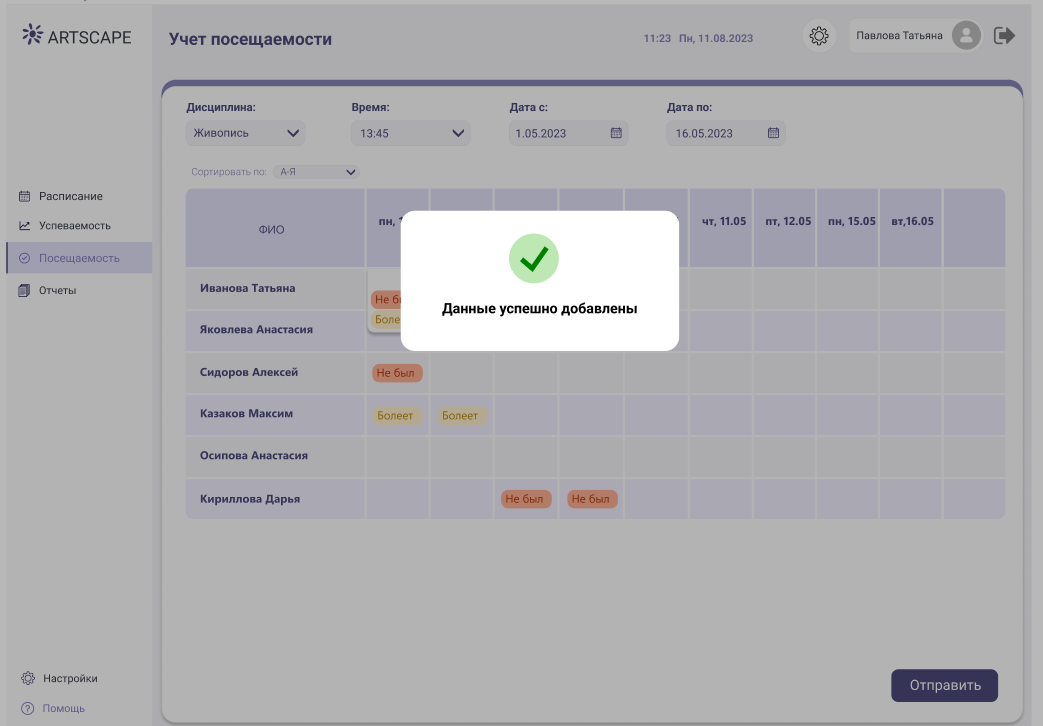
После выполнения действия «Изменить», кнопка с меткой «Изменить» возвращается в исходное состояние, т.е. метка на ней снова становится «Отправить». При этом пользователь может повторно внести необходимые изменения в таблицу и отправить их на сохранение в БД. Для этого необходимо выполнить те же действия, которые описаны выше для действия «Отправить».

На рисунке 2.9 представлен макет экрана, предназначенный для учета посещаемости с использованием учетной записи преподавателя. Навигационное меню находится слева для обеспечения удобства работы с системой. Для упрощения процесса заполнения документа в правой верхней части экрана отображается текущий день недели и время. Основная функциональность задачи «Учет посещаемости» расположена в центральной части экрана.



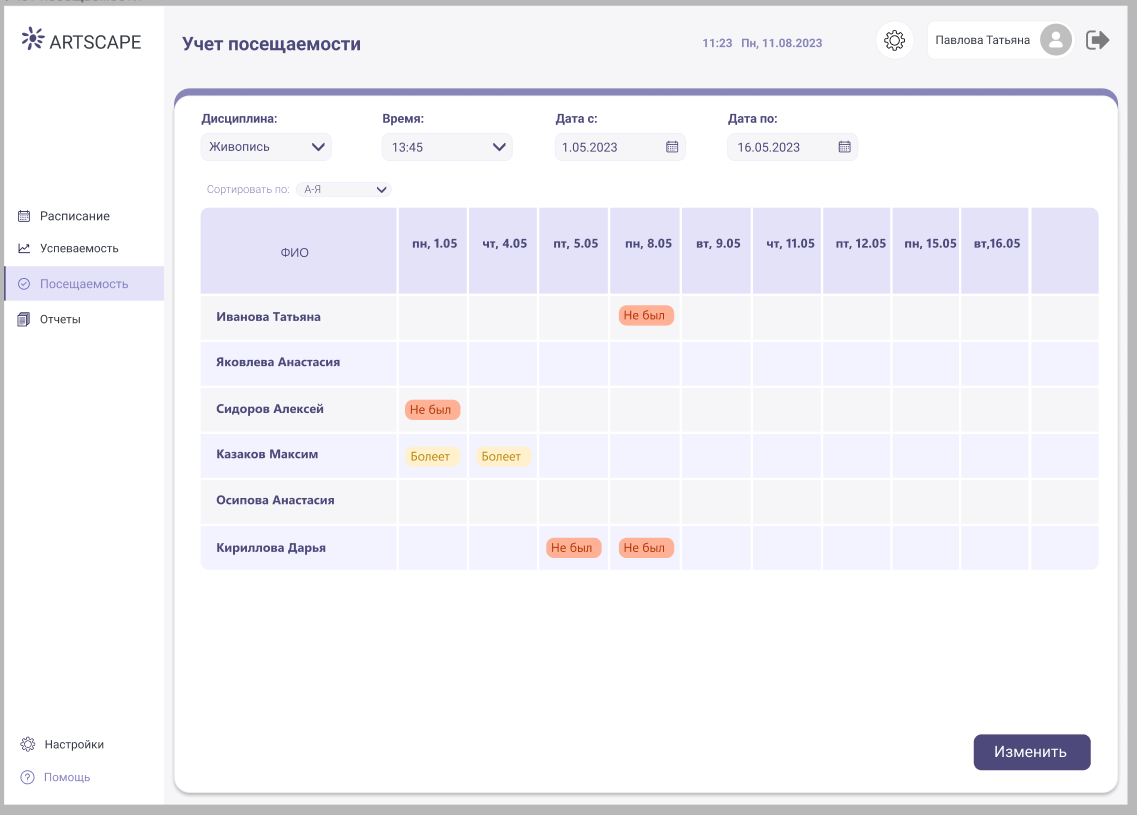
**Рисунок 2.9 – Макет экрана «Учет посещаемости»**

После нажатия на кнопку «Отправить» данные будут сохранены в базе данных. Если запрос успешно выполнен, то пользователю будет выведено уведомление об успешной операции (см. рисунок 2.10) в соответствующем окне.



**Рисунок 2.10 – Успешная запись посещаемости в БД**

После завершения процесса добавления записи, кнопка «Добавить» преобразуется в кнопку «Изменить» (см. рисунок 2.11), и пользователю становится недоступно внесение изменений в текущую таблицу. Доступ к функции внесения данных будет восстановлен только после того, как пользователь нажмет на кнопку «Изменить», чтобы исправить ошибку, если таковая обнаружится после отправки посещаемости.

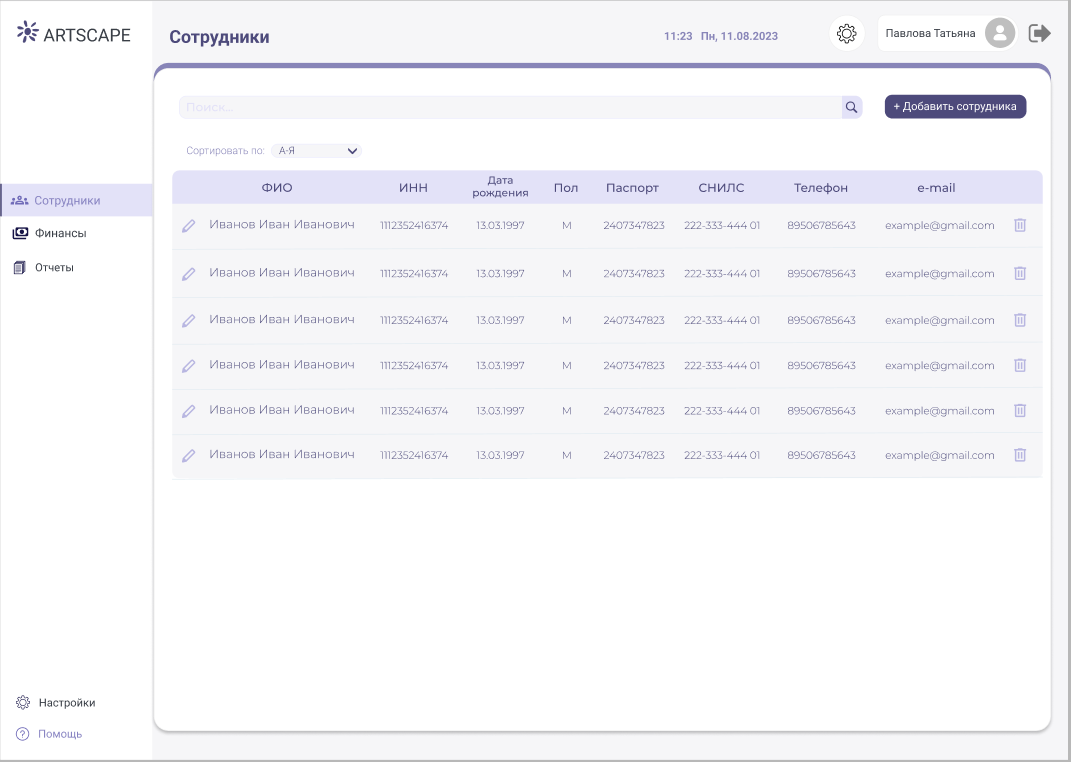


**Рисунок 2.11 – Состояние системы после закрытия окна при успешном сохранении данных о посещаемости**

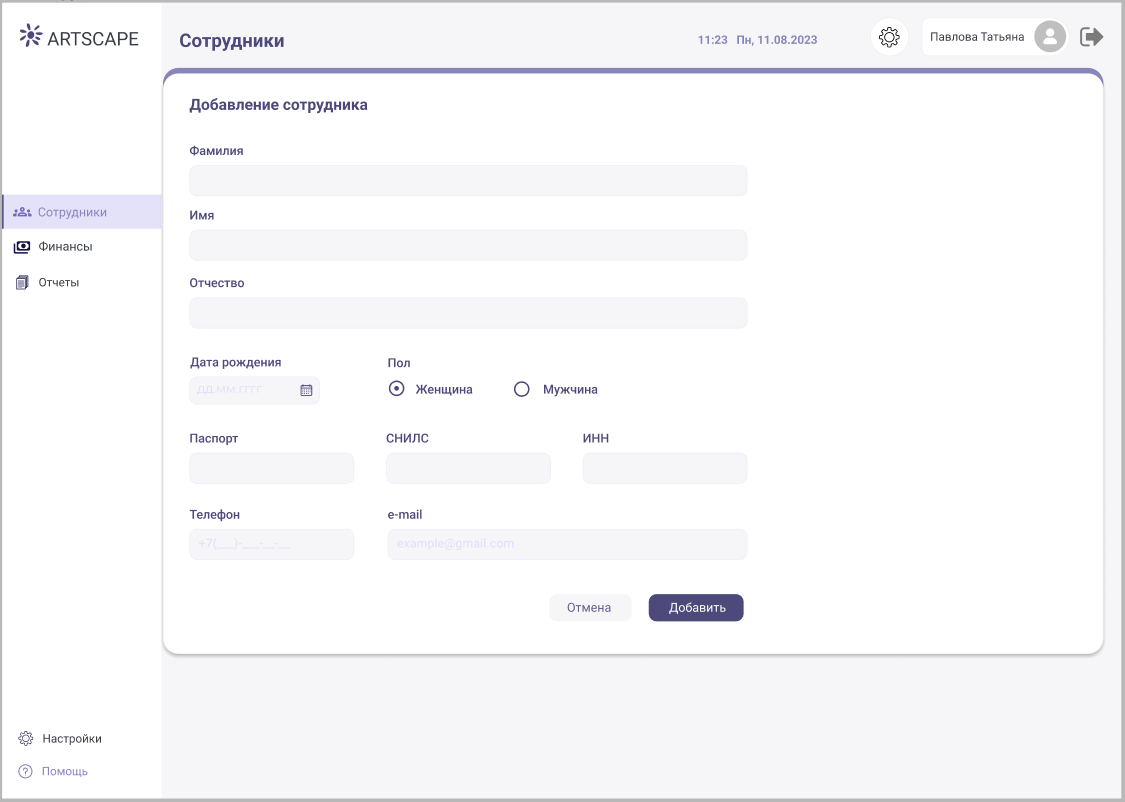
После выполнения действия «Изменить», кнопка с меткой «Изменить» возвращается в исходное состояние, т.е. метка на ней снова становится «Отправить». При этом пользователь может повторно внести необходимые изменения в таблицу и отправить их на сохранение в БД. Для этого необходимо выполнить те же действия, которые описаны выше для действия «Отправить».

На рисунке 2.12 представлен макет экрана, предназначенный для учета преподавателей со стороны директора.

При активации кнопки «Добавить сотрудника» (см. рисунок 2.13), откроется окно с формой, где необходимо внести соответствующие данные о новом преподавателе.



**Рисунок 2.12 – Макет страницы «Учет преподавателей»**



**Рисунок 2.13 – Макет экрана формы заполнения информации**

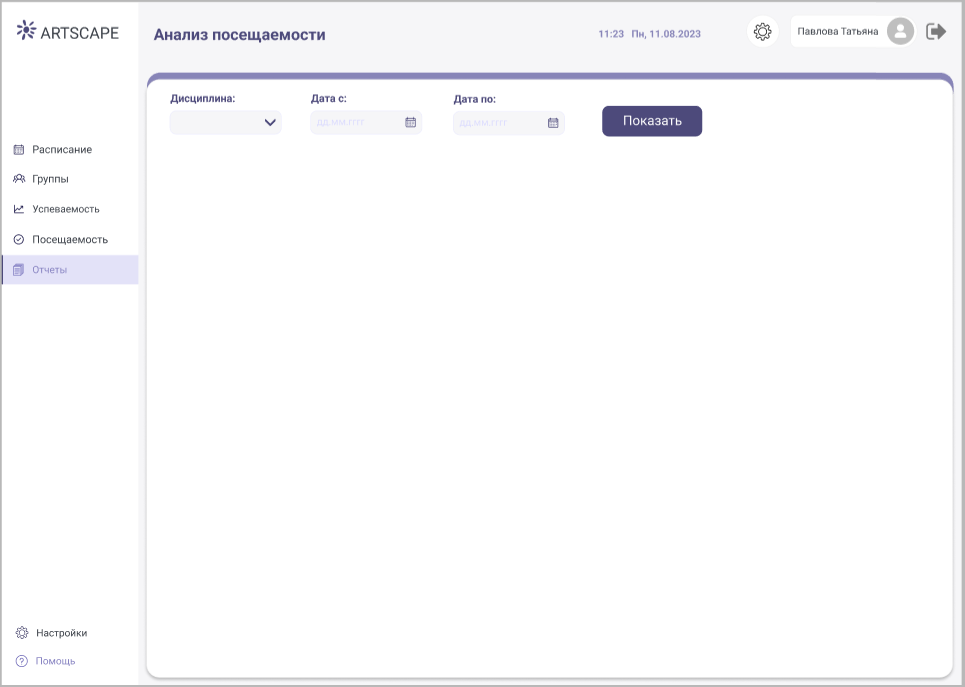
**о преподавателе**

После заполнения формы система оповестит об успешном добавлении сотрудника, а также автоматически сгенерирует логин и временный пароль для нового пользователя (см. рисунок 2.14).



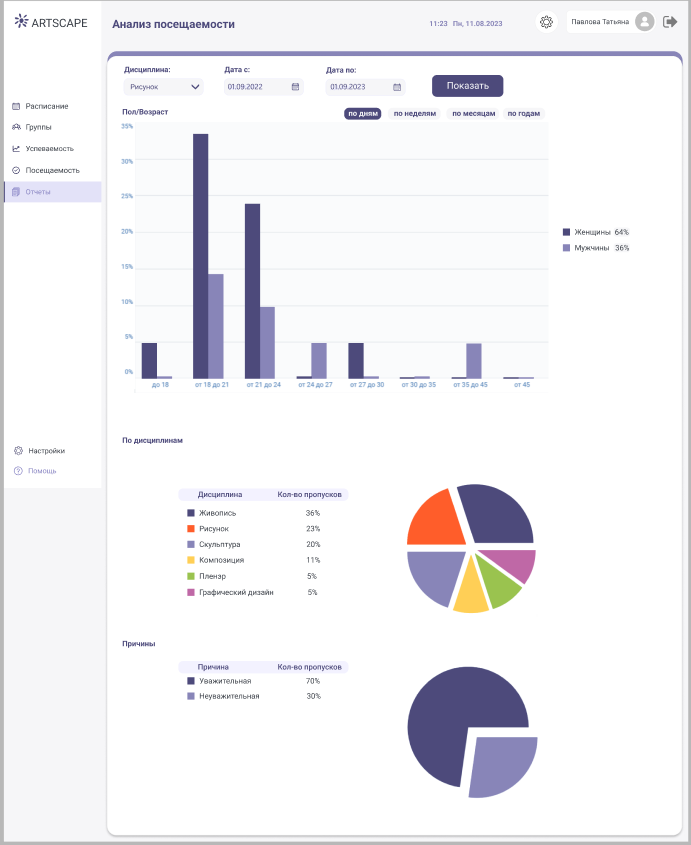
**Рисунок 2.14 – Макет экрана после успешного добавления преподавателя**

Для выполнения задачи «Анализ посещаемости» требуется ввести определенные данные, необходимые для формирования отчета. В макете экрана, который показан на рисунке 2.15, представлены поля для ввода этой информации.

****

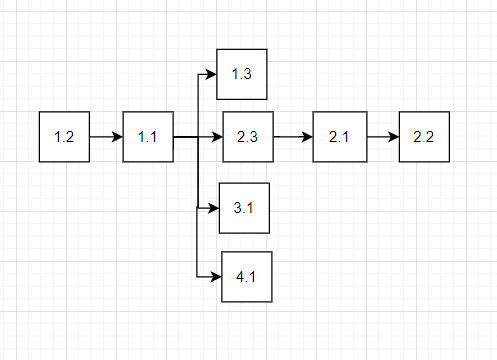
**Рисунок 2.15 – Макет экрана «Анализ посещаемости»**

После нажатия на кнопку «Показать» система формирует отчет (см. рисунок 2.16) согласно заданным условиям и отображает его в графическом виде, обеспечивая более удобный способ анализа полученных данных.



**Рисунок 2.16 – Сформированный отчет**

На рисунке 2.17 представлена карта экранов приложения.

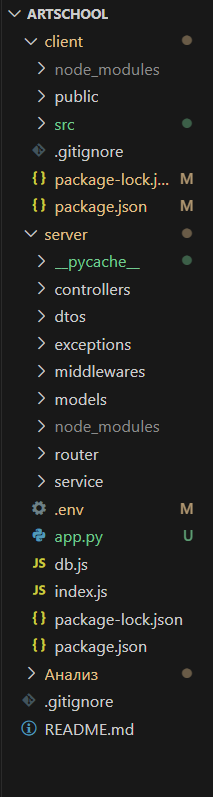


**Рисунок 2.17 – Карта экранов**

# Разработка

# Структура проекта

Проект структурирован на две основные части: клиентскую (frontend) и серверную (backend) (см. рисунок 3.1). Эта организация файлов и директорий обеспечивает четкое разделение между пользовательским интерфейсом и логикой обработки данных.



**Рисунок 3.1 – Структура проекта**

# Frontend

Для верстки страниц были выделены и организованы следующие компоненты, которые обеспечивают функциональность и визуальное представление интерфейса (см. рисунок 3.2).



**Рисунок 3.2 – Компоненты**

Рассмотрим некоторые компоненты и их код.

Компонент InputDate изображен на рисунке 3.3.



**Рисунок 3.3 – Компонент InputDate**

Код InputDate:

const InputDate = (props) => {

  const { label, id, value, onChange, ...inputProps } = props;

  const handleChange = (event) => {

    const newValue = event.target.value;

    onChange(newValue);

  };

  return (

    <div className="form\_\_input">

      <label>{label}</label>

      <input

        type="date"

        {...inputProps}

        value={value}

        onChange={handleChange}

        className="my\_\_input-date"

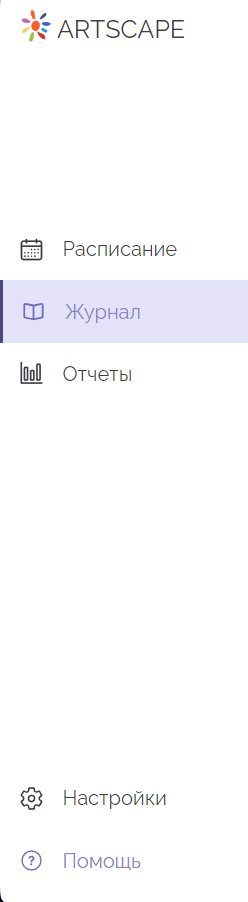
      ></input>

    </div>

  );

};

Компонент Sidebar (см. рисунок 3.4).



**Рисунок 3.4 – Компонент Sidebar**

Код Sidebar:

const Sidebar = () => {

  return (

    <div className="sidebar">

      <div className="logo">

        <img src={Logo} />

        <span>ARTSCAPE</span>

      </div>

      <ul className="sidebar\_list">

        {SidebarData.map((val, key) => {

          return (

            <li

              key={key}

              className="list\_element"

              id={window.location.pathname === val.link ? "active" : ""}

              onClick={() => {

                window.location.pathname = val.link;

              }}

            >

              <div id="icon">{val.icon}</div>

              <div id="title">{val.title}</div>

            </li>

          );

        })}

      </ul>

      <ul className="service\_list">

        {ServiceData.map((val, key) => {

          return (

            <li

              key={key}

              className="list\_element"

              id={window.location.pathname === val.link ? "active" : ""}

              onClick={() => {

                window.location.pathname = val.link;

              }}

            >

              <div id="icon" name={val.name}>

                {val.icon}

              </div>

              <div id="title" name={val.name}>

                {val.title}

              </div>

            </li>

          );

        })}

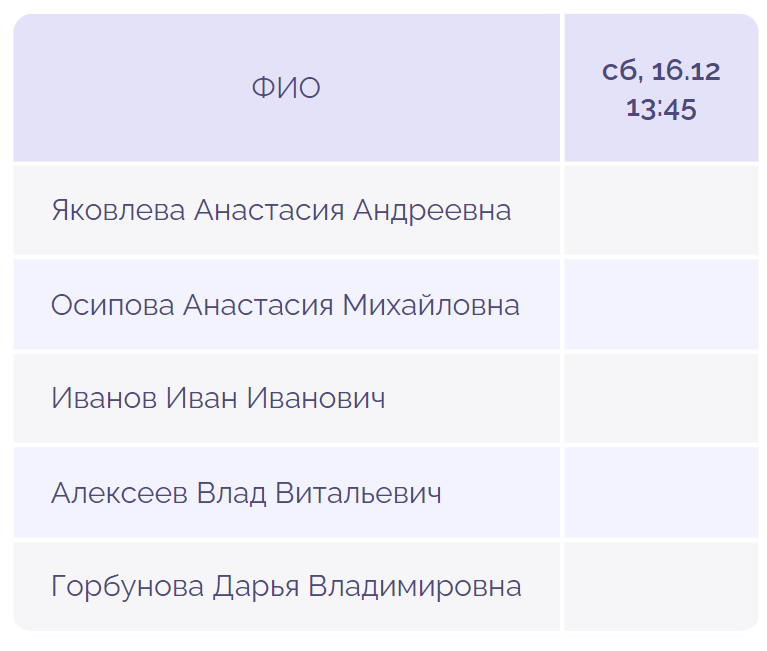
      </ul>

    </div>

  );

};

На рисунке 3.5 изображен компонент Table.



**Рисунок 3.5 – Компонент Table**

Код Table:

const Table = observer((props) => {

  const name = "attendance";

  const { journal } = useContext(Context);

  const { id } = useParams();

  const [showDropDownList, setShowDropDownList] = useState(true);

  const [selectedValue, setSelectedValue] = useState(null);

  const { buttonText, selectedValues, onSelect } = props;

  const getCurrentDayAndDate = () => {

    const daysOfWeek = ["вс", "пн", "вт", "ср", "чт", "пт", "сб"];

    const currentDate = new Date();

    const dayOfWeek = daysOfWeek[currentDate.getDay()];

    const date = currentDate.getDate();

    const month = currentDate.getMonth() + 1; // Месяцы в JavaScript начинаются с 0

    return `${dayOfWeek}, ${date}.${month}`;

  };

  useEffect(() => {

    fetchOneClass(id).then((data) => journal.setPeople(data));

  }, []);

  return (

    <table className="my-table">

      <thead>

        <tr>

          <th>ФИО</th>

          <th>

            <p>{getCurrentDayAndDate()}</p>

            <p>13:45</p>

          </th>

        </tr>

      </thead>

      <tbody>

        {journal.people.map((item) => (

          <tr key={item.id}>

            <td>{item.surname + " " + item.name + " " + item.patronymic}</td>

            <td>

              <DropDownList

                name="attendance"

                buttonText={buttonText}

                isDropdownVisible={buttonText === "Сохранить"}

                selectedValue={selectedValue}

                onSelect={onSelect}

              />

            </td>

          </tr>

        ))}

      </tbody>

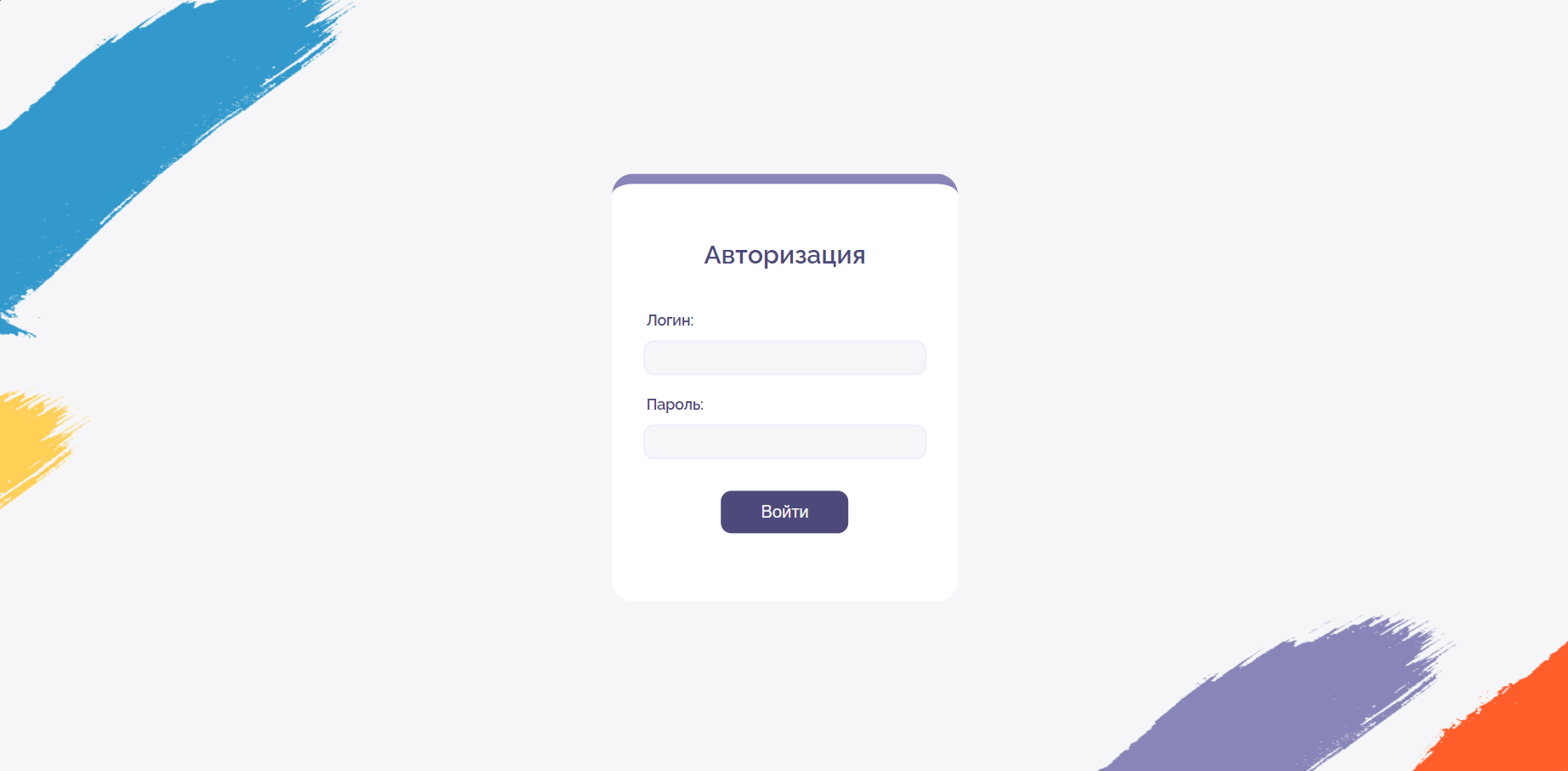
    </table>

  );

});

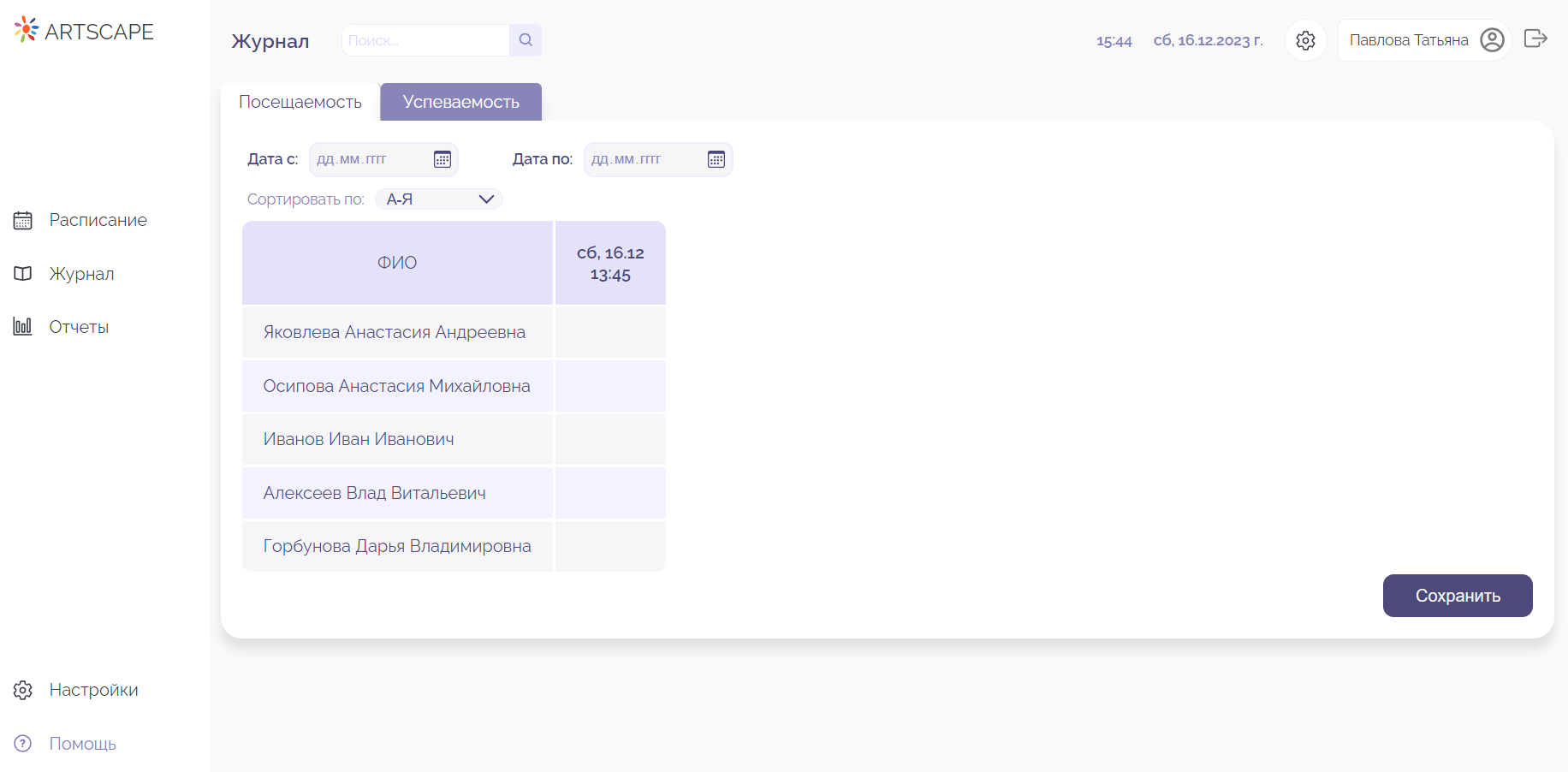
С помощью данных компонентов были реализованы следующие страницы: авторизации, расписания, учета успеваемости и посещаемости, отчетов. Рассмотрим каждую из них.

Страница авторизации представлена на рисунке 3.3.



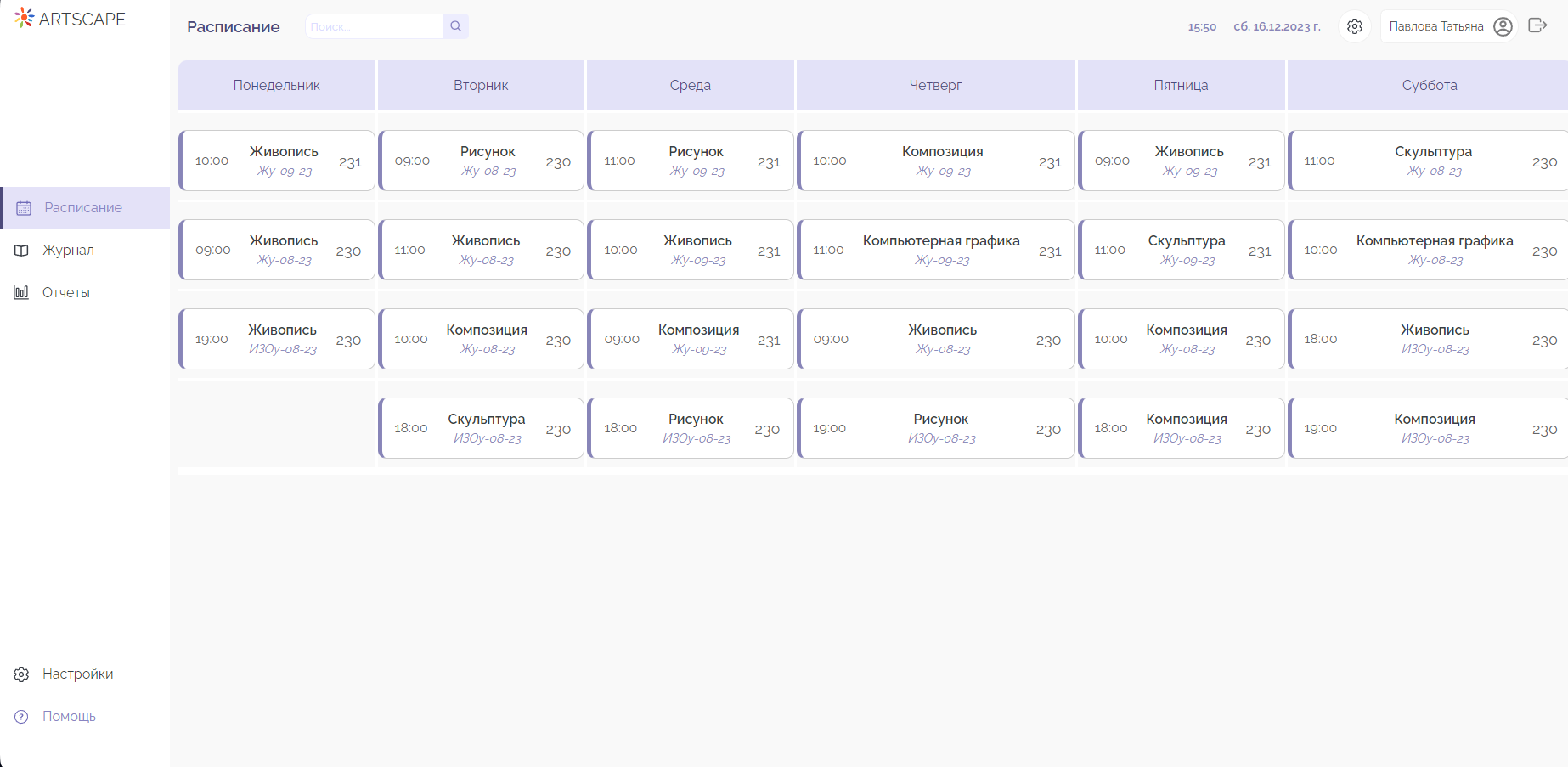
**Рисунок 3.3 – Страница авторизации**

На рисунке 3.4 изображена страница для учета посещаемости и успеваемости.



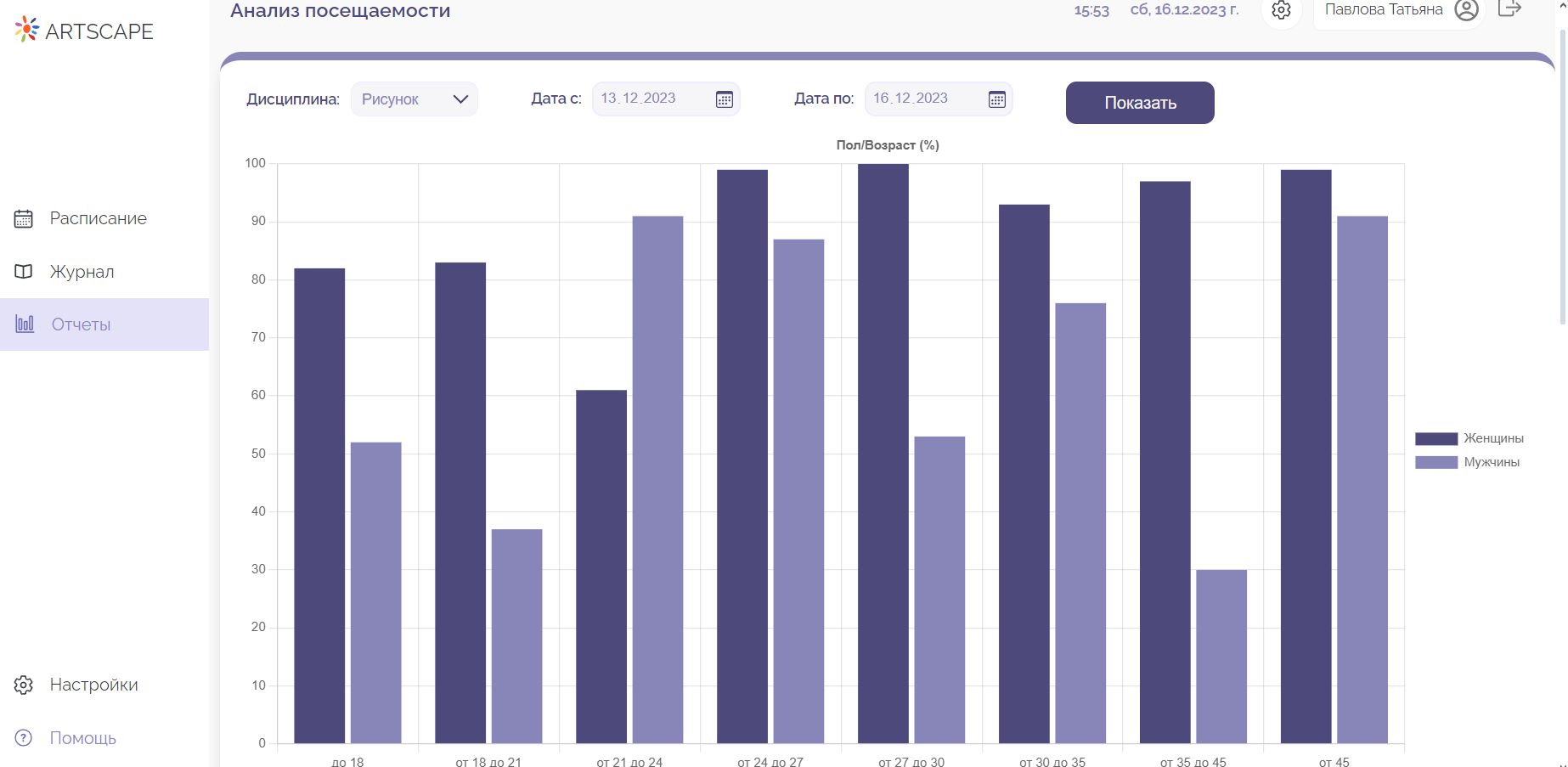
**Рисунок 3.4 – Страница учета посещаемости и успеваемости**

Страница с расписанием изображена на рисунке 3.5.



**Рисунок 3.5 – Страница с расписанием**

На рисунке 3.6 представлена страница с сформированным отчетом.



**Рисунок 3.6 – Страница с сформированным отчетом**

# Backend

Ниже представлены различные фрагменты кода, которые были реализованы в серверной части системы.

Генерация расписания:

def generate\_schedule():

        try:

            with app.app\_context():

                conn = psycopg2.connect(dbname='artschool', user='postgres', password='NASLEDNIKI', host='localhost')

                cursor = conn.cursor()

                cursor.execute("SELECT \* FROM groups")

                groups = cursor.fetchall()

                cursor.execute("SELECT \* FROM teachers")

                teachers = cursor.fetchall()

                cursor.execute("SELECT teacher\_id, hours FROM teacherworkload")

                teachers\_workload = cursor.fetchall()

                cursor.execute("SELECT number FROM rooms")

                rooms = cursor.fetchall()

                cursor.execute("SELECT \* FROM syllabus")

                syllabus = cursor.fetchall()

                cursor.execute("SELECT \* FROM subjects")

                subjects = cursor.fetchall()

                cursor.execute("SELECT name FROM days")

                days = cursor.fetchall()

                cursor.execute("SELECT hours FROM time WHERE type\_id = 1")

                morning\_time\_strings = cursor.fetchall()

                cursor.execute("SELECT hours FROM time WHERE type\_id = 2")

                evening\_time\_strings = cursor.fetchall()

                morning\_time = [t[0].strftime("%H:%M") for t in morning\_time\_strings]

                evening\_time = [t[0].strftime("%H:%M") for t in evening\_time\_strings]

                cursor.execute("SELECT \* FROM training\_programs")

                training\_programs = cursor.fetchall()

                # Инициализация расписания

                schedule = []

                # Максимальное количество уроков в день для учителя

                total\_hours\_per\_day = 4

                # Перебор всех учителей

                for teacher in teachers:

                    # Получение нагрузки учителя из базы данных

                    sql\_query = "SELECT hours FROM teacherworkload WHERE teacher\_id = %s"

                    cursor.execute(sql\_query, (teacher[0],))

                    result = cursor.fetchall()

                    # Преобразование списка кортежей в массив чисел

                    teacher\_workload = [row[0] for row in result]

                    # Получение групп, преподаваемых учителем

                    sql\_query = "SELECT \* FROM groups WHERE teacher\_id = %s"

                    cursor.execute(sql\_query, (teacher[0],))

                    teacher\_groups = cursor.fetchall()

                    # Перебор групп учителя

                    for group in teacher\_groups:

                        # Получение предметов и их нагрузки для группы

                        sql\_query = "SELECT subject\_id, hours FROM syllabus WHERE program\_id = %s"

                        cursor.execute(sql\_query, (group[3],))

                        group\_classes = cursor.fetchall()

                        # Переменная для отслеживания предыдущей аудитории

                        previous\_classroom = None

                        # Перебор предметов в группе

                        for class\_info in group\_classes:

                            subject\_id, hours\_per\_week = class\_info

                            subject = next((subj for subj in subjects if subj[0] == subject\_id), None)

                            if not subject:

                                continue

                            # Перебор часов по предмету в неделю

                            for \_ in range(hours\_per\_week):

                                # Поиск доступных аудиторий

                                available\_classrooms = [c for c in rooms if is\_available(c, morning\_time, schedule)]

                                if not available\_classrooms:

                                    available\_classrooms = [c for c in rooms if is\_available(c, evening\_time, schedule)]

                                # Прерывание цикла, если нет доступных аудиторий

                                if not available\_classrooms:

                                    break

                                # Выбор аудитории (если она совпадает с предыдущей, используем её)

                                if previous\_classroom in available\_classrooms:

                                    classroom = previous\_classroom

                                else:

                                    classroom = random.choice(available\_classrooms)

                                    previous\_classroom = classroom

                                # Выбор времени занятия в зависимости от типа группы

                                if group[5] == 1:

                                    time\_slot = random.choice(morning\_time)

                                else:

                                    time\_slot = random.choice(evening\_time)

                                # Выбор дня недели

                                day = random.choice(days)

                                # Проверка, свободен ли учитель в выбранное время

                                while not is\_teacher\_available(teacher[1], (day, time\_slot), schedule):

                                    if group[5] == 1:

                                        time\_slot = random.choice(morning\_time)

                                    else:

                                        time\_slot = random.choice(evening\_time)

                                    day = random.choice(days)

                                # Формирование урока и добавление в расписание

                                lesson = {

                                    "teacher": teacher[1],

                                    "subject": subject[1],

                                    "classroom": classroom,

                                    "time\_slot": (day, time\_slot),

                                    "group": group[1]

                                }

                                schedule.append(lesson)

                                # Уменьшение нагрузки учителя

                                teacher\_workload[0] -= 1

            return jsonify(schedule)

        except Exception as e:

            print(e)

            traceback.print\_exc()

            return jsonify({"error": str(e)})

        finally:

            if cursor:

                cursor.close()

            if conn:

                conn.close()

Подключение к базе данных:

module.exports = new Sequelize(

  process.env.DB\_DATABASE,

  process.env.DB\_USER,

  process.env.DB\_PASSWORD,

  {

    dialect: "postgres",

    host: process.env.DB\_HOST,

    port: process.env.DB\_PORT,

  }

);

Получение учащихся к конкретной группе:

  async getOne(req, res, next) {

    try {

      const { id } = req.params;

      const classToday = await Schedule.findOne({ where: { id } });

      const group\_id = classToday.group\_id;

      const people = await Student.findAll({ where: { group\_id } });

      return res.json(people);

    } catch (e) {

      next(e);

    }

  }

Функция авторизации:

  async login(req, res, next) {

    try {

      const { login, password } = req.body;

      const user = await User.findOne({ where: { login } });

      if (!user) {

        return next(ApiError.internal("Пользователь не найден"));

      }

      let comparePassword = bcrypt.compareSync(password, user.password);

      if (!comparePassword) {

        return next(ApiError.internal("Указан неверный пароль"));

      }

      const token = generateJwt(user.id, user.login, user.role\_id);

      return res.json({ token });

    } catch (e) {

      next(e);

    }

  }

# Заключение

В заключении курсового проекта можно подчеркнуть важность предпроектного обследования в контексте внедрения информационных систем в образовательные учреждения. Анализ текущих процессов и потребностей художественной школы позволил выявить ключевые аспекты, которые подлежат автоматизации, и определить оптимальные пути их решения.

Проектирование серверной части информационной системы для художественной школы представляет собой неотъемлемый этап в создании современной и эффективной образовательной среды. Решения, разработанные в ходе проекта, направлены на улучшение управленческих и педагогических процессов, содействуя более эффективному взаимодействию между сотрудниками и обучающимися.

В рамках курсового проекта были разработаны макеты страниц авторизации, учета посещаемости и успеваемости, расписания и отчетов, а также соответствующие им функции.

# Список использованных источников

1. Дневник.ру [Электронный ресурс]. [2023]. URL: <https://dnevnik.ru/about> (дата обращения: 23.10.2023).
2. Мой класс [Электронный ресурс]. [2023]. URL: <https://moyklass.com/crm>

(дата обращения: 24.10.2023)

1. ALFACRM [Электронный ресурс]. [2023]. URL: <https://alfacrm.pro/#googtrans(ru|null)> (дата обращения: 24.10.2023)
2. AXKED CRM [Электронный ресурс]. [2023]. URL: <https://axked.ru/login> (дата обращения: 24.10.2023)
3. Проектирование информационных систем : учебник и практикум для академического бакалавриата / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общ. ред. Д. В. Чистова. – М. : Издательство Юрайт, 2016.
4. Node.js. Разработка серверных веб-приложений в JavaScript / Хэррон Д. А. А. – М.: ДМК Пресс, 2012. – 144 с.: ил.